# Interactions between hierarchical learning and visual system modeling: image classification on small datasets

Thalita F Drumond

université de BORDEAUX

Inria LaBRI

Thèse

présentée pour obtenir le grade de

**DOCTEUR DE L'UNIVERSITÉ DE BORDEAUX**

École Doctorale de Mathématiques et d'Informatique

Spécialité : Informatique

# Interactions between hierarchical learning and visual system modeling

## Image classification on small datasets


par

Thalita FIRMO DRUMOND

Sous la direction de M. Frédéric ALEXANDRE et M. Thierry VIÉVILLE


Soutenue le 3 décembre 2020

Membres du jury :

| | | | |
|---|---|---|---|
| Mme. SAUZÉON, Helène | Pr. | Univ. de Bordeaux | Présidente |
| M. CARRÉ, Philippe | Pr. | Univ. de Poitiers | Rapporteur |
| Mme. ESCOBAR, Maria Jose | Asc. Pr. | Univ. Técnica Federico Santa María, Chile | Rapporteure |
| M. JOUFFRAIS, Christophe | DR | CNRS IPAL IRL2955, Singapour | Examinateur |
| M. VON ZUBEN, Fernando José | Full Pr. | Univ. Estadual de Campinas, Brésil | Examinateur |

**Title:** Interactions between hierarchical learning and visual system modeling: Image classification on small datasets

**Abstract:** Deep convolutional neural networks (DCNN) have recently protagonized a revolution in large-scale object recognition. They have changed the usual computer vision practices of hand-engineered features, with their ability to hierarchically learn representative features from data with a pertinent classifier. Together with hardware advances, they have made it possible to effectively exploit the ever-growing amounts of image data gathered online. However, in specific domains like healthcare and industrial applications, data is much less abundant, and expert labeling costs higher than those of general purpose image datasets. This scarcity scenario leads to this thesis' core question: can these limited-data domains profit from the advantages of DCNNs for image classification? This question has been addressed throughout this work, based on an extensive study of literature, divided in two main parts, followed by the proposal of original models and mechanisms.

The first part reviews object recognition from an interdisciplinary double-viewpoint. First, it resorts to understanding the function of vision from a biological stance, comparing and contrasting to DCNN models in terms of structure, function and capabilities. Second, a state-of-the-art review is established aiming to identify the main architectural categories and innovations in modern day DCNNs. This interdisciplinary basis fosters the identification of potential mechanisms — inspired both from biological and artificial structures — that could improve image recognition under difficult situations. Recurrent processing is a clear example: while not completely absent from the "deep vision" literature, it has mostly been applied to videos — due to their inherently sequential nature. From biology however it is clear such processing plays a role in refining our perception of a still scene. This theme is further explored through a dedicated literature review focused on recurrent convolutional architectures used in image classification.

The second part carries on in the spirit of improving DCNNs, this time focusing more specifically on our central question: deep learning over small datasets. First, the work proposes a more detailed and precise discussion of the small sample problem and its relation to learning hierarchical features with deep models. This discussion is followed up by a structured view of the field, organizing and discussing the different possible paths towards adapting deep models to limited data settings. Rather than a raw listing, this review work aims to make sense out of the myriad of approaches in the field, grouping methods with similar intent or mechanism of action, in order to guide the development of custom solutions for small-data applications. Second, this study is complemented by an experimental analysis, exploring small data learning with the proposition of original models and mechanisms (previously published as a journal paper).

In conclusion, it is possible to apply deep learning to small datasets and obtain good results, if done in a thoughtful fashion. On the data path, one shall try gather more information from additional related data sources if available. On the complexity path, architecture and training methods can be calibrated in order to profit the most from any available domain-specific side-information. Proposals concerning both of these paths get discussed in detail throughout this document. Overall, while there are multiple ways of reducing the complexity of deep learning with small data samples, there is no universal solution. Each method has its own drawbacks and practical difficulties and needs to be tailored specifically to the target perceptual task at hand.

**Keywords:** deep learning, image classification, small data learning, convolutional neural networks, transfer learning.

**Titre :** Apports croisés entre l'apprentissage hierarchique et la modélisation du système visuel: Catégorisation d'images sur des petits corpus de données

**Résumé :** Les réseaux neuronaux convolutifs profonds ("deep convolutional neural networks" ou DCNN) ont récemment révolutionné la reconnaissance d'objets à grande échelle, modifiant les pratiques en vision par ordinateur, consistant à définir des caractéristiques représentatives "à la main", désormais apprises de façon hiérarchique à partir des données, tout en les classifiant. Fort de la progression des performances matérielles, on exploite efficacement des quantités toujours croissantes d'images recueillies en ligne. Mais, dans des domaines spécifiques, comme en santé ou pour certaines applications, les données sont moins abondantes, et les coûts d'étiquetage par des experts sont plus élevés. Cette rareté conduit à la question centrale de cette thèse : Ces domaines à données limitées peuvent-ils bénéficier des avantages des DCNN pour la classification des images ? Ce travail repose sur une étude approfondie de la littérature, divisée en deux parties principales, avant de proposer des modèles et des mécanismes originaux, expérimentés.

La première partie couvre la reconnaissance des objets d'un double point de vue. Tout d'abord, la fonction visuelle biologique, est comparée et contrastée avec la structure, la fonction et les capacités des modèles DCNN. Puis, une revue de l'état-de-l'art identifie les principales catégories d'architectures et les innovations dans les DCNN récents. Cette base interdisciplinaire favorise l'identification des mécanismes — biologiquement et artificiellement inspirés — qui améliorent la reconnaissance d'images dans des situations difficiles. Le traitement récurrent en est un exemple clair : peu présent au niveau de la vision profonde, sauf le traitement aux vidéos — en raison du caractère naturellement séquentiel. Mais la biologie montre clairement qu'un tel traitement joue aussi un rôle dans l'affinement de notre perception d'une scène fixe. Ce thème est approfondi à travers une revue de la littérature consacrée aux architectures convolutionnelles récurrentes utilisées en catégorisation d'images.

La deuxième partie se concentre sur notre question centrale : l'apprentissage profond sur de petits corpus de données. Tout d'abord, le travail propose une discussion plus précise et détaillée de ce problème et de sa relation avec l'apprentissage hiérarchique des caractéristiques réalisé par des modèles profonds. Cette discussion est suivie d'une revue structurée du domaine, organisant et discutant les différentes voies possibles vers l'adaptation des modèles profonds à des données limitées. Plus qu'une simple liste, ce travail vise à trouver du sens dans la myriade d'approches du domaine, en regroupant les méthodes ayant un objectif ou un mécanisme d'action similaire, pour guider le développement d'application particulières, à petits corpus. Cette étude est complétée par une analyse expérimentale, explorant l'apprentissage de petits jeux de données avec des modèles et mécanismes originaux (précédemment publié comme papier de journal).

En conclusion, l'apprentissage profond sur des petits corpus de données peut donner de bons résultats, si cela se fait de manière réfléchie. Au niveau des données, il faut essayer de recueillir plus d'informations à partir de sources de données supplémentaires connexes. Au niveau de la complexité, l'architecture et les méthodes d'entraînement peuvent être calibrées afin de tirer le meilleur parti de toute connaissance spécifique au domaine. Des propositions sont discutées en détail au fil du document. Il existe de multiples façons de réduire la complexité de l'apprentissage profond avec de petits échantillons de données, mais il n'y a pas de solution universelle. Chaque méthode a ses propres inconvénients et difficultés pratiques, devant toujours être adaptée spécifiquement à l'application, c'est-à-dire à la tâche perceptive à accomplir.

**Mots-clés :** apprentissage profond, catégorisation d'images, apprentissage sur petit corpus de données, réseaux de neurones convolutifs, apprentissage par transfert.

## Unité de recherche

Laboratoire Bordelais de Recherche en Informatique (UMR 5800)
Domaine universitaire
351 cours de la Libération
33405 Talence

INRIA Bordeaux Sud-Ouest
Équipe-projet MNEMOSYNE
200 Avenue de la Vieille Tour
33405 Talence

**My thesis in a glimpse** The field of artificial intelligence has made many advances in the last decade, especially with deep learning for image recognition. Despite being biologically inspired, these deep neural networks function in a quite different way from our natural vision. Essentially, it is a mathematical object with numerical parameters that can be adjusted automatically, using large sets of previously labeled images. Only after "seeing" thousands of cats, dogs, cars, trees, people, etc., will the network be able to recognize these elements in new images (without understanding their meaning).

Having access to large databases of labeled images is unfortunately not possible in all fields of application. On specific industrial problems or in medical imaging for example, it can be difficult or even impossible to obtain hundreds of different images of the same condition, patient, etc... In addition, image labeling is more expensive since it requires an expert opinion.

This motivates the central question of this thesis: How can we take advantage of deep neural networks on small image datasets? This work moves a step towards this answer through an extensive literature review, complemented by an experimental study including the proposition of original models and mechanisms.

**Ma thèse en un clin d'œil** Le domaine de l'intelligence artificielle a connu diverses avancées dans la dernière décennie, en particulier avec le "deep learning" pour la reconnaissance d'images. Malgré une inspiration biologique, ces réseaux neuronaux profonds ont finalement un fonctionnement bien différent de notre vision naturelle. Essentiellement, il s'agit d'un objet mathématique avec des paramètres numériques qu'on peut ajuster de façon automatique, ayant recours à des larges corpus d'images pré-étiquetées. Ce n'est donc qu'après "avoir vu" des milliers de chats, chiens, voitures, arbres, personnes, etc, que le réseau sera capable de reconnaître ces éléments sur des nouvelles images (sans en comprendre le sens).

Avoir accès à des grandes bases d'images étiquetées n'est malheureusement pas possible sur tous les domaines d'application. Pour des problématiques industrielles ou en imagerie médicale, par exemple, il peut être difficile voire impossible d'obtenir des centaines d'images variées d'un même problème, patient, etc. De plus, l'étiquetage de ces images est coûteux car il demande un avis expert.

C'est là où réside la question centrale de cette thèse : comment peut-on profiter des avantages des réseaux de neurones profonds sur de petits corpus d'images ? Ce travail fait un pas vers cette réponse via une étude bibliographique étendue, complémenté par une étude expérimentale comprenant des propositions de modèles et mécanismes originaux.

*Fred et Thierry, un énorme immensurable merci à tous les deux.*
*Merci de m'avoir accepté dans votre équipe.*
*Merci de m'avoir guidé dans ce voyage de ouf.*
*Merci de m'avoir traité comme une collègue, une collaboratrice.*
*Merci de vote confiance et encouragement à tous les moments.*
*Bref, sans vous, ça ne se serait pas passé aussi bien.*

*Big thanks to all my mnemo-friends, old an new.*
*You all make this team a great environment to work, a safe-place to share ideas and be creative.*
*Thank you Mamie ICK for taking such good care of all of us.*
*Thank you Mr B our zen master.*
*Thank you Astroboy for being a constant reminder of what it is to be young and passionate by your research. And for all the board games :)*
*Thank you Mme SP, our official event hostess, for keeping us together.*
*Thank you Miss RS a.k.a. DA BOSS xD for your encouragements and random hugs of support.*

*Obrigada a minha família querida, que torce por mim, de longe, mesmo sem entender muito bem o que eu faço.*

*Obrigada ao meu companheiro de vida, minha nova família, por sempre estar aqui, por todo o suporte, de longe e de perto, sempre.*

# Table of Contents

**List of Figures** **11**

**List of Tables** **14**

**Acronyms** **15**

**1 Introduction** **17**
1.1 Context and motivation . . . 17
1.1.1 The revolution of deep learning . . . 17
1.1.2 Limits of bio-inspiration . . . 17
1.1.3 The problem of small datasets . . . 18
1.1.4 Deep learning literature . . . 19
1.2 Related topics . . . 20
1.2.1 Object recognition tasks . . . 20
1.2.2 Other relevant deep architectures . . . 21
1.3 Thesis focus and contributions . . . 23

**I Object recognition with deep convolutional neural networks** **26**

**2 Convolutional neural networks and visual system modeling** **27**
2.1 Introduction . . . 27
2.2 Convolutional neural networks . . . 29
2.2.1 Convolutional layers . . . 29
2.2.2 Pooling or subsampling . . . 30
2.2.3 Activation functions . . . 31
2.3 Visual system: a schematic overview . . . 32
2.3.1 A hierarchical system . . . 32
2.3.2 The schematic two-stream model . . . 33
2.4 The object identification pathway in relation to CNNs . . . 34
2.4.1 Core object recognition . . . 34
2.4.2 Early vision: from retina to V1 cortex . . . 37
2.4.2.1 Retinotopy . . . 37
2.4.2.2 Receptive fields . . . 38
2.4.3 Mid-level visual areas . . . 39
2.4.4 Infero-temporal cortex . . . 39
2.5 Discussion . . . 40

2.5.1 Comparisons of cortical representations and CNN representations . . 40
2.5.2 Main differences between biology and basic feedforward CNNs . . . . 42
2.5.2.1 Processing in retina and LGN . . . . . 43
2.5.2.2 Recurrent processing . . . . . 44
2.5.2.3 Interaction between dorsal and ventral streams . . . . . 45
2.6 Conclusion . . . . . 45

**3 Feedforward CNN architectures for object recognition 47**
3.1 Introduction . . . . . 47
3.2 Single pathway models . . . . . 49
3.2.1 First modern architectures . . . . . 49
3.2.1.1 Alexnet . . . . . 49
3.2.1.2 Zeiler and Fergus . . . . . 50
3.2.1.3 OverFeat . . . . . 51
3.2.1.4 VGG . . . . . 51
3.2.2 Parameter-saving architectures . . . . . 52
3.3 Parallel pathway archs . . . . . 52
3.3.1 Sequence of parallel blocks: the Inception family . . . . . 53
3.3.1.1 Role of auxiliary losses . . . . . 54
3.3.1.2 Variations and derivations . . . . . 54
3.3.2 Architectures featuring forward shortcuts . . . . . 55
3.3.2.1 Residual connections . . . . . 55
3.3.2.2 Gated shortcuts . . . . . 56
3.3.2.3 Multiple forward shortcuts . . . . . 57
3.4 Discussion . . . . . 59
3.4.1 Relation between forward shortcuts and recurrent networks . . . . . 59
3.4.2 Attempts at understanding the success of residual shortcuts . . . . . 60
3.4.3 Feature reuse by concatenation vs memory efficiency . . . . . 61
3.4.4 Width vs depth on ResNets . . . . . 61
3.5 Conclusion . . . . . 62

**4 Recurrent and feedback CNN architectures for object recognition 63**
4.1 Introduction . . . . . 63
4.2 Functionalities brought by recurrent processing . . . . . 64
4.2.1 Spatial context integration . . . . . 64
4.2.2 Iterative processing . . . . . 66
4.2.2.1 Iterative spatial attention . . . . . 67
4.2.2.2 Handling occlusion and image clutter . . . . . 68
4.2.3 Response modulation . . . . . 68
4.3 Architectural trends . . . . . 69
4.3.1 Recurrence within layers . . . . . 69
4.3.1.1 Intra-layer temporal recurrence . . . . . 69
4.3.1.2 Intra-layer spatio-temporal recurrence . . . . . 73
4.3.2 Feedback between layers . . . . . 74
4.3.2.1 Explicit biological inspiration . . . . . 75

4.3.2.2 Ladder networks . . . 76
4.3.3 Hybrid models with added recurrent layers . . . 77
4.4 Discussion . . . 79
4.4.1 Feedback vs recurrent feedforward . . . 79
4.4.2 Internal representations on feedback networks . . . 79
4.4.3 Training considerations . . . 80
4.4.3.1 Temporal recurrence depth . . . 80
4.4.3.2 Computational requirements . . . 81
4.4.4 Biological plausibility . . . 81
4.5 Conclusion . . . 83

**II Image classification on small datasets 84**

**5 A review of strategies to use deep learning under limited data 85**
5.1 Introduction . . . 85
5.2 Problem statement . . . 86
5.2.1 Small data scenarios . . . 86
5.2.2 Related problems . . . 87
5.2.3 Implicit and explicit knowledge . . . 88
5.2.4 Knowledge transfer . . . 89
5.3 Axis I: Get more data . . . 90
5.3.1 Synthetic data . . . 91
5.3.1.1 Augmentation at the input space . . . 91
5.3.1.2 Augmentation in the feature space . . . 92
5.3.1.3 Augmentation through generative modeling . . . 92
5.3.2 Unlabeled data . . . 94
5.3.2.1 Semi-supervised learning . . . 94
5.3.2.2 Weak supervision . . . 95
5.3.2.3 Active learning . . . 97
5.3.3 Related datasets . . . 98
5.3.3.1 Pre-training and fine-tuning . . . 98
5.3.3.2 Zero-shot learning . . . 101
5.4 Axis II: Reduce model complexity . . . 102
5.4.1 Explicit regularization . . . 103
5.4.2 Architecture adaptation . . . 106
5.4.2.1 Intra-layer structure . . . 106
5.4.2.2 Inter-layer structure . . . 107
5.4.3 Training strategies . . . 108
5.4.3.1 Meta-learning: learning to learn . . . 108
5.4.3.2 Curriculum learning . . . 109
5.4.3.3 Multi-task learning . . . 111
5.5 Conclusion . . . 112

**6 Analysis of DCNN applied to small sample learning using data prototypes 113**
6.1 Introduction . . . 113
6.1.1 The data requirement challenge . . . 113
6.1.2 The interpretability issue . . . 114
6.1.3 On network architecture . . . 115
6.1.4 Hyperparameter dependence . . . 116
6.1.5 The present contribution . . . 116
6.2 Related works . . . 117
6.2.1 Prototypes in literature . . . 117
6.2.2 Few-shot learning and learning to learn . . . 118
6.2.3 Interpretability . . . 120
6.3 Model proposed . . . 121
6.3.1 Model architecture . . . 121
6.3.2 Model specification . . . 122
6.3.2.1 Direct use of the prototypes . . . 123
6.3.2.2 Combining prototype-encoded and original features . . . 124
6.3.2.3 Relation between prototypes and category information . . . 125
6.4 Experiments and results . . . 125
6.4.1 Datasets . . . 125
6.4.2 Studying the model under fixed features . . . 126
6.4.2.1 Study of a sample episode . . . 127
6.4.2.2 Interpretability of the model . . . 131
6.4.2.3 Comparison over multiple episodes . . . 132
6.4.3 Comparison over different training sets with varying sizes . . . 132
6.4.4 Experiments in the meta-learning paradigm . . . 137
6.5 Discussion . . . 138
6.5.1 Performances and use of the algorithm . . . 139
6.5.2 Perspectives and future work . . . 140
6.6 Conclusion . . . 141

**7 Conclusion 142**
7.1 Summary of contributions . . . 142
7.1.1 Understanding object recognition from an interdisciplinary stance . . . 142
7.1.2 Image classification over small datasets with deep models . . . 144
7.2 Further developments . . . 145
7.3 Open perspectives . . . 147

**Appendix 150**

**A Prototype model derivation and formal interpretation 151**
A.1 Sample to prototype association . . . 151
A.2 Reminder on softmax function . . . 151

A.3 Deriving the model variational equations . . . 152
A.4 Analysis of the k-means extended metric . . . 152
A.5 Probabilistic interpretation of representing samples by prototypes . . . 153
A.6 Duality between partition, prototypes and metric . . . 154
A.7 Relation between softmax and prototypes . . . 155

**B Methodology for hyperparameter analysis on prototype model with fixed input features 157**

**C Relevant benchmark datasets in object recognition 159**
C.1 MNIST . . . 159
C.2 Omniglot . . . 160
C.3 CIFAR 10 and 100 . . . 160
C.4 ImageNet classification 2012 . . . 160
C.4.1 mini-Imagenet . . . 161

**D Résumé étendu en français 162**
D.1 Contexte et motivation . . . 162
D.1.1 La revolution du « deep learning » . . . 162
D.1.2 Limites de la bio-inspiration . . . 163
D.1.3 Le problème des petits corpus de données . . . 164
D.1.4 Litterature sur le « deep learning » . . . 165
D.2 Positionnement du travail et plan de la thèse . . . 166
D.3 Résumé des contributions . . . 167
D.3.1 Étude et compréhension de la reconnaissance d'objets d'un point de vue interdisciplinaire . . . 167
D.3.2 Classification d'images sur des corpus restreints avec des réseaux profonds . . . 169

**Index 171**

**Bibliography 172**

# List of Figures


1.1 Thesis organization diagram. . . . . 24

2.1 Typical convolutional neural network (CNN). Credits: Aphex34 CC BY-SA 4.0 28
2.2 A simplified anatomic schematic of the visual neural pathways in the brain, from retina to the primary visual cortex (V1). Credits: Miquel Perello Nieto CC BY-SA 4.0 . . . . . 33
2.4 After reaching the primary cortex in the occipital lobe, visual processing is roughly divided in two streams or patwhays: the dorsal pathway goes up to the parietal lobe, while the ventral pathway goes lower towards the inferior temporal lobe. Credits: OpenStax College CC BY 3.0 . . . . . 34
2.3 Anatomical hierarchy of brain regions implicated in visual processing, according to Felleman and Van Essen (1991). The ventral pathway associated to "core object recognition" is highlighted in red. Credits: Cortes (2012). . . . . 36

3.1 Proposed taxonomy of feedforward CNN architectures. Each family is represented by an archetypal small module of 3-4 layers. Top and bottom circles stand for module's input and output, respectively. . . . . 47
3.2 Diagram representing an "influence genealogy" of some important CNN models for computer vision. Dates correspond to arXiv first release or date of publication, whichever is the earliest. . . . . 48

4.1 Proposed taxonomy of recurrent and feedback CNN architectures. Each family is represented by an archetypal small module of 3-4 layers. Top and bottom circles stand for module's input and output, respectively. . . . . 64
4.2 Schematic representation of the recurrent structure of different spatio-temporal recurrent models. . . . . 73
4.3 Feedback nets predict an output at every time-step, connecting the loss to intermediate hidden states and changing how features are learned. Adapted from Zamir et al (2017). . . . . 79

5.1 Illustrative diagram differentiating the different small data scenarios ranked in the text. Rectangles represent data matrices (with samples as rows and features as columns). Different colors mark different types of data (see legend in the top right corner) and dashed lines delimit distinct datasets. . . . . 87
5.2 Knowledge sources . . . . . 89
5.3 Strategies for deep learning under small data. This diagram illustrates the point of view presented in this chapter. . . . . 90

5.4 Main architectural schemes for zero-shot learning. . . . . 102

6.1 The three layer model. The input data is fed to a convolution neural network, transforming the original image into a set of feature vectors. Then the information in this high dimensional feature space is summarized via a set of prototypes, in order to understand the landscape. Finally, the relation between a given data point and the label to infer is calculated through their proximity to the prototypes. . . . . 121
6.2 Left: Sub-sample of 10 classes from the Omniglot dataset, showing 5 samples per class (total is 20 per class). Right: T-SNE visualization of the Inception v3 features for the 200 samples of this subset, see text for details. . . . . 127
6.3 Mean accuracy (with standard deviation bars) obtained during grid search for the inverse regularization parameter C and the number of clusters, under L2 regularization. Top: direct model. Bottom: combined model. . . . . 129
6.4 Average training loss on the k-mean algorithm as a function of the number of prototypes. The mean square distance between samples and prototypes decreases until a plateau (sometimes called elbow) and then either re-increases, as visible in this example, or do not significantly re-decreases as shown in Fig. 6.9 for other datasets. . . . . 130
6.5 Verification that the introduction of a priori information on the prototype category has no significant impact on the performances, as soon as the number for prototypes is high enough. Graphs show mean accuracy (with standard deviation bars) obtained during grid search under the best regularization parameter found. Left: under L1 regularization, $C = 10^4$. Right: under L2 regularization, $C = 10^2$. . . . . 131
6.6 The top-4 closest training samples visually illustrate what is being represented by each prototype. For each prototype `p1`, `p2`, .., its related class `c2`, `c5`, .. is written, this "main" class being estimated as the most probable class proposed by our algorithm. The number of samples corresponding to the related cluster is given. . . . . 132
6.7 T-SNE visualizations of dataset points (as Inception v3 features) together with learned prototypes, indicated by black markers. Each class is identified by color, and the predicted class for each prototype is annotated. Misclassified points are over-marked by an "x" of the predicted class color. Top: visualization of the training set; Bottom: visualization of the test set. . . . . 133
6.8 Box plots for test accuracies over 100 episodes. Boxes extend between 1st and 3rd quartiles ($IQR$), with median value noted at the bottom and marked by a green line. Bars mark a range of $\pm 1.5\ IQR$ from quartiles, with outliers as circles. Our models are marked with a `*`. . . . . 134
6.9 Performances on the k-means algorithm as a function of the number of prototypes, for training subsets with 500 samples from MNIST and CIFAR10 datasets. In these and contrary to Fig. 6.4, the optimal value corresponds to an elbow in both cases. . . . . 136

6.10 Mean accuracies over standard test sets for MNIST and CIFAR10 datasets, for increasing training set sizes. Results are averaged over 10 different training subsets. Our models are marked with a *. . . . . 137

7.1 Architectural taxonomy of CNNs: feedforward and feedback. Each family is represented by an archetypal small module of 3-4 layers. Top and bottom circles stand for module's input and output, respectively. . . . . 143

# List of Tables


4.1 Recurrent convolutional architectures from a functional point of view . . . . . 70
4.2 Summary of convolutional recurrent architectures. . . . . 78

6.1 Statistical significance for all pairs of classifiers tested on Omniglot. “++” and “+” mark all pairs which presented a significant difference (for $p < 0.01$ and $p < 0.05$ respectively), while "o"s mark those which did not. Non-reported pairs were found to be significantly different with $p < 0.01$ (except for direct L1 vs. direct L2 at 30 classes, for which $p < 0.05$). . . . . 135
6.2 Average test accuracy (%) for 5-shot learning on Omniglot over 1000 episodes. We compare to results reported by two other works: Matching networks (Vinyals et al, 2016) and Prototypical networks (Snell et al, 2017). . . . . 138

C.1 Information on datasets mentioned throughout this thesis. . . . . 159

# Acronyms

**aIT** anterior infero-temporal. 82

**ANN** artificial neural network. 42

**BN** batch normalization. 75

**BPTT** backpropagation through time. 80

**CGRU** convolutional gated recurrent unit (GRU). 71

**CIFAR** Canadian Institute for Advanced Research. 82

**cIT** central infero-temporal. 82

**CNN** convolutional neural network. 11, 17, 18, 20–22, 28–32, 42, 43, 45, 47, 49, 62–64, 81, 82, 85, 86, 89, 92, 95, 98, 99, 105, 107, 108, 149, 161–163, 165

**ConvRNN** convolutional recurrent neural network. 75, 81, 82

**CV** computer vision. 49

**DAE** denoising autoencoder. 23

**DCNN** deep convolutional neural network. 28, 46, 49, 63, 85, 86

**DNN** deep neural network. 18, 20, 43, 46, 83, 88, 102, 103, 147, 149, 163

**GAN** generative adversarial network. 22, 23, 93

**GPU** graphical processing unit. 17, 49, 147, 162

**GRU** gated recurrent unit. 15, 22, 69, 71

**HoG** histograms of Gaussians. 27

**ILSVRC** ImageNet large scale visual recognition challenge. 17, 161, 162

**IT** infero-temporal. 18, 33, 39, 40, 75, 163

**LGN** lateral geniculate nucleus. 33, 42, 75

## Acronyms

**LOC** lateral occipital cortex. 41

**LSTM** long-short term memory. 22, 69, 71–74

**MLP** multi-layer perceptron. 29, 30

**PCN** predictive coding network. 67

**pIT** posterior infero-temporal. 82

**RBM** restricted Boltzmann machine. 22, 67

**ReLU** rectified linear unit. 31, 75

**ResNet** residual network. 31, 72

**RNN** recurrent neural network. 22, 71, 75

**SAE** sparse autoencoder. 23

**SGD** stochastic gradient descent. 49, 110

**SIFT** scale-invariant feature transform. 27

**SRN** simple recurrent network. 72, 74

**SVHN** Street-view house numbers. 82

**SVM** support vector machine. 27

**TPE** tree-structured Parzen estimator. 75

**V1** primary visual cortex. 33

**VAE** variational autoencoder. 23, 93

Chapter 1

# Introduction

## 1.1 Context and motivation

### 1.1.1 The revolution of deep learning

Image recognition has undergone a revolution in the past decade, majorly led by the successful performances obtained using deep neural networks. Even though these models had been around since the late 80s – early 90s (LeCun et al, 1990a), their successful application had been limited by both data and hardware requirements. Recent advances in hardware — with the possibility of treating large matrix operations in parallel with consumer graphical processing units (GPUs) — and in data availability — with the widespread use of digital cameras and online photo sharing platforms — have been decisive in changing this scenario. Quite emblematic of this revolution was the 12% accuracy improvement in the 2012 edition of the ImageNet large scale visual recognition challenge (ILSVRC), which ended up to be a hallmark of the successful use of convolutional neural networks (CNNs) for general purpose object recognition.

These machine learning models have not only achieved unprecedented accuracy at object recognition, but have also brought a new paradigm of jointly learning feature extraction and classification, within a single model. Indeed, this end-to-end training paradigm, mapping images to classes directly, has become a *de facto* standard in most computer vision works. It is often argued that this manner of proceeding — learning from examples — is closer to how humans see and learn to recognize objects in nature. However, such similarities between computer and biological neural vision exist on a rather superficial level, while our neurons and cortical regions operate on a much more complex fashion, implementing functions absent from typical CNNs.

### 1.1.2 Limits of bio-inspiration

Neural networks are made of successive interconnected layers of artificial neurons, which are loosely inspired by the functioning of natural neuronal cells. More specifically, they model the thresholded information propagation performed by neurons, when they selectively pass

along an electric action potential. One artificial neuron is a composed mathematical function, that first computes a linear combination of its inputs, then puts the resulting output through a non-linearity, which emulates the biological thresholding mechanism. This modeling is highly simplified and neglects any temporal aspects relevant to the functioning of real neurons.

Bio-inspiration is also present at a higher level, in the way neurons are organized and the patterns they learn. Deep neural networks (DNNs) take their naming predicate from their archetypal architecture consisting in a stack of multiple neuron layers. Each layer takes in outputs of its previous counterpart, conducting one further step of information treatment before feeding it to the subsequent one. This incremental processing structure, when trained on natural image datasets, has been shown to function in a hierarchical fashion, similar to the functional organization of the different cortical regions involved in our own biological vision. Early layers tend to learn to detect low level image features such as parts of borders in different orientations — in a functional analogy to our primary visual cortex — while later layers get specialized in detecting more abstract concepts and ultimately the image class — in an analogous role to that of the infero-temporal (IT) cortex.

While this functional similarity seems to be present, there is much left unaccounted for in typical feedforward CNNs. Shortcut connections between regions, local recurrent processing, long-range feed back connections, attentional modulation — these are all features absent from these models and reputed to play important roles in the processing of a visual scene. Recurrent processing for instance is believed to act in recognizing objects under visual clutter, a task in which CNNs may have some difficulty. One may wonder how the incorporation of such functionalities could help the performance of CNNs, both in quantitative and qualitative fashion. Furthermore, one may wonder which of these improvements would be the most relevant to any particularly difficult scenario.

While a detailed study of this problem is not the core objective of this thesis (the reader may refer to (Medathati et al, 2016) for a detailed review), this work is careful when discussing bio-inspiration and plausibility of reviewed and developed architectures. Without attachment to any particular level of biological plausibility, biological function and structure have served as a transversal reference for both the experimental developments and literature review, being addressed at multiple occasions throughout this work.

### 1.1.3 The problem of small datasets

All this progress however is linked to the widespread availability of large-scale datasets, with thousands of samples per image category, amounting to millions of images in the full dataset. As large neural networks have a huge number of learnable parameters, a great number of examples is necessary for the model to learn a pertinent feature space representation and achieve generalization in its predictions. On smaller datasets, such large models are likely to overfit training data and not generalize to the target data. Many if not most domains are

actually more likely to produce small labeled datasets, for a myriad of reasons. For instance, in medical imaging diagnostics, trials with each single patient are limited, some conditions are rarely encountered, and labeling is costly as it demands one or multiple experts opinion. In industry, the context of images taken from a production plant, a particular machine, or a pipeline step, may be specific enough that a dedicated dataset would need to be collected onsite. This same dataset would later require an expert technician to label whichever faulty or deviant condition they may be trying to automatically identify. The fact is that on most organizations not directly working with data collection, but simply willing to automate a certain internal process, composing a dedicated dataset is not straightforward, neither in terms of cost nor logistics. Therefore, in most of these cases, one may have to deal with challenging conditions in terms of data availability for the application of any machine learning method, and deep learning in particular.

In many such cases, the first reflex of a well-trained machine learning engineer may be to recur to less complex models — like decision trees for instance — and in particular to ensemble versions of them — like random forests or XGboost. Then again, let us not forget our interest in exploiting image data. The data-oriented features learned through deep models seem to play a fundamental role in the recent breakthroughs of image recognition. Previous feature engineering methods have not yielded a comparable level of success in image classification. Moreover, these data-specialized features may be all the more invaluable to domain-specific recognition, in which general visual features may not be as discriminant as would be a set of task or domain-related specificities. This conflicting will of modeling small datasets in a data-oriented fashion without overfitting is the main motivation topic behind the present work. Throughout this work we seek to understand the hierarchical feature learning performed by deep neural networks — in particular convolutional architectures — while mapping out the circumstances under which they can be successfully exploited under limited training data availability. A detailed exploration of this problem with an analysis of its different composing parts constitutes a major part of this thesis' work.

### 1.1.4 Deep learning literature

Overall, research in the machine learning community has heavily shifted towards deep networks-based work recently. Indeed, conferences specifically related to the topic (NeurIPS, ICML, IJCNN, among others) have seen an exponential increase in the number of submissions and overall participation. Additionally, the pre-publishing of articles on open online platforms, such as ArXiv, has also set and increased pace in the field, with interactions, paper citations and derivative works all happening much faster than a typical peer-reviewed publication cycle. It is a difficult and time consuming matter on itself to navigate through this ever-growing literature corpus, trying to discern promising trends from ephemeral hacks.

When trying to focus on the specific problem of small data learning, one will eventually re-

alize it brings on a few more particular difficulties. First, this is not one, but multiple problems. Several different practical settings can be referred to as being some form of small data learning. Secondly, not that many works have tried to directly measure and address the difficulty of training CNNs on small datasets, rather addressing similar difficulties encountered in other sub-fields (e.g. fine-grained classification, metric learning, transfer learning, meta-learning, to name a few). These two aspects combined lead to a third general observation: the current state of knowledge in this area is not readily available to anyone with an incipient interest in this problem. Instead, it takes reading works from multiple related areas in order to map out the challenges and levers of action in face of any particular form of data scarcity. It is therefore paramount to revisit these different areas, structuring and linking apparently disconnected studies covering these multiple aspects of the subject. This knowledge organization effort is one of the major aspects of this thesis' work.

## 1.2 Related topics

Given the context of this work, some notions parallel to the core topics of CNNs and image classification will be occasionally mentioned in the course of the following chapters. Namely, it is important that the reader has a basic familiarity with other closely related object recognition tasks, as well as a minimal notion of other contemporary DNN models developed (or popularized) over the past decade. The following sections thus provide brief explanations on these topics, pointing towards specific references for further information. The reader familiar with these concepts may skip the current section and proceed to reading section 1.3.

### 1.2.1 Object recognition tasks

Besides image classification, object recognition may concern other more specific tasks that aim to provide more detailed information than simply a global image label. The following paragraphs describe some relevant object recognition tasks, particularly focusing on those performed on still images. Object recognition on videos is considered out of the scope of this thesis.

**Image classification** In plain image classification, each sample gets an image level label, regardless of the presence of objects corresponding to other labels. In some datasets, the ideal labels are quite clear for whomever is carrying the annotation process, because only one object is present per image. This is the case with the handwritten digits dataset MNIST and the multi-alphabet character dataset Omniglot (see more on these datasets in appendix C). For datasets containing realistic images, like ImageNet, MS-COCO and others, there is naturally more ambiguity in identifying the target class for an image.

**Object localization and detection** In object detection and localization, models have

to predict an object's presence and position within the input image. Both tasks are similar and their names are occasionally interchanged. Usually, the name *object localization* is associated with image-level classification. In this case, a single label is predicted, and the corresponding object within the image is contoured. *Object detection* is more general and assumes the model will try to recognize and localize multiple objects within the image —- which implies in possible multiple labels per image. Object detection is thus a multi-label classification problem, in which the presence or absence for each category is predicted (often on a probability scale).

In both cases, it is necessary to predict image coordinates (a double regression problem) for the recognized object category (or for each instance of all detected categories). Most often the final goal is to predict a bounding box enclosing each occurrence of the detected objects. It is evident these tasks requires even more costly annotations, composed of label plus bounding box coordinates.

The ImageNet challenge also provided specific datasets for both these tasks, derived from the original classification one, but having undergone additional annotation. Winner models also followed the CNN trend, with additional machinery dedicated to bounding box prediction. For more on the advances obtained in these tasks through CNNs, the reader may refer to the surveys by Agarwal et al (2018) and Zou et al (2019).

**Semantic segmentation** Dividing an image into its composing parts, providing a label for each of them: that is the goal of semantic segmentation. More than simply predicting bounding boxes, this task involves predicting more precise boundaries for each recognized element. When all elements of a scene need to be identified, the task is also referred to as *scene labeling*. When only some instances of particular objects are to be detected, the task is also known as *object segmentation* or *instance segmentation*.

This task is performed thanks to specific datasets which are annotated with contours dividing the different labeled parts. Typically, these annotations get converted into a target image that is pixel-wise labeled by applying the annotation masks over the original images, attributing to each pixel the label corresponding to its containing segment. Given this target, the predicting network has to output a single-class prediction for each input image pixel. Difficulties particular to performing this task with CNNs include the need for heavy pixel-wise labeling, the challenge of identifying precise borders and mapping up the predicted segmentation (often sub-sampled) to the input image. A review of recent advances on this front can be read in Garcia-Garcia et al (2017).

### 1.2.2 Other relevant deep architectures

Despite the protagonist role of convolutional models in the popularization of deep networks, they were not the sole architecture family taking part in the latest advances. Other kinds of deep models have equally evolved along the past few years, achieving similar widespread status in their respective application fields. Here we cover briefly some of these models, which

get mentioned throughout this thesis, providing references for further detailed information.

**Recurrent neural networks** The first Recurrent neural networks (RNNs) have been proposed not long after the first feedforward models (e.g. Rumelhart and McClelland, 1987; Elman, 1990). These networks incorporate a time dimension, such that a recurrent unit's output will be dynamic. At any instant $t$, its output $h[t]$ will not only depend on the input $x[t]$ but also on the previous output value $h[t-1]$. Currently used models have improved on this simple recurrent structure, mainly trying to circumvent particular training difficulties of RNNs (related to vanishing or exploding gradients, refer to Bengio et al, 1994). Long-short term memory (LSTM) networks are one successful example. First proposed by Hochreiter and Schmidhuber (1997), they are more capable of learning long-term dependencies, making them appropriate to sequence modeling applications (refer to Goodfellow et al, 2016g, for more details this type of task).

Different models may feature other memory or gating structures and incorporate additional internal states in the output computation (refer to Yu et al, 2019, for a review). Gates can be composed of extra neurons within a recurrent unit, with memory cells corresponding to the current result of gating computations. Along with LSTM units, gated recurrent units (GRU), first proposed by Cho et al (2014), are also widely used today. An LSTM unit has a separate memory cell, along with *input*, *output* and *forget* (memory cell overwriting) *gates*. *Forget* and *input gates* control which information should be kept ( from the current input and from the past), being used to calculate the next memory cell state. The *output gate* computes the next output value, combining this newly computed memory cell state with the previous output and the current input. GRUs are simpler, with no explicit memory cell. They still count with an *update gate* — deciding which input information to keep/ignore for the next output —- and a *reset gate* — controlling how much past information influences the next output.

In general, recurrent network models are widely used for language modeling (e.g. Mikolov et al, 2013), language translation (e.g. Wu et al, 2016) and also for audio processing and speech recognition (e.g. Sundermeyer et al, 2015). In image processing, they are mainly applied to tasks involving video, such as video captioning (Shetty and Laaksonen, 2015), action detection (Xu et al, 2019), video segmentation (Pavel et al, 2015; Siam et al, 2017), video super-resolution (Shetty and Laaksonen, 2015), to cite a few. Recurrent CNN models applied to image classification will be reviewed in chapter 4.

**Generative models** Besides discriminative models — like most classification and regression predictive models — another approach in statistical learning concerns generative models. While discriminative models estimate the probability of an outcome — a discrete label or real value — conditioned on data samples, i.e. $p(y|x)$, generative models also model data samples directly, either on their own i.e. $p(x)$ or jointly with labels, i.e. $p(x, y)$. Deep generative models (refer to Goodfellow et al, 2016d, for an overview) include a variety of autoencoders, stochastic models (like the restricted Boltzmann machine (RBM)), and generative adversarial

networks (GANs).

Autoencoder architectures, as the name suggests, are trained to predict the input sample, encoding it through a mirrored architecture (refer to Goodfellow et al, 2016a, for a textbook introduction to autoencoders). First, the encoding part maps an input (say, an image) to a middle layer of neurons. Then, the decoding part maps the value of these neurons back to an input-like sample (say, another image). There are several variations of this basic architectural principles, such as sparse autoencoders (SAEs) (e.g. Makhzani and Frey, 2014) denoising autoencoders (DAEs) (first proposed in Vincent et al, 2008) and variational autoencoders (VAEs) (first proposed in Kingma and Welling, 2014).

Another important group are the generative adversarial networks, first proposed by Goodfellow et al (2014) (see Creswell et al, 2017, for an overview). These architectures have two sub-models — the *generator* and the *discriminator* — interacting in a zero-sum game[1], one trying to "fool" the other. On one side, the generator takes a random seed as input and maps it out to an image. On the other side, the discriminator takes this image as input and tries to predict whether it is real or artificially generated. The objective function enforces the generator to produce images that fool the discriminator, while the latter is enforced to improve its detection of artificial images (refer to the tutorial by Goodfellow, 2016, for detailed explanations). Ever since their first proposal in 2014, a surprising high number of variations has been proposed in the literature[2], turning research on this type of architecture into a field in itself. GAN models have been successfully applied to realistic data generation or transformation in multiple domains, including not only image and video (e.g. with CycleGAN by Zhu et al, 2017a), but also audio (e.g. with WaveGAN by Donahue et al, 2019) and text (e.g. with MaksGAN by Fedus et al, 2018).

## 1.3 Thesis focus and contributions

In the first section of this introduction we have established the difficulty and general interest of applying deep learning to small datasets. Even while restricting the scope to object recognition, and particularly to image classification, there is still a large diversity of approaches in the domain. Given the rapid evolution of deep learning research and the large corpus of literature, it is indispensable to review the state-of-the art in order to place any novel proposals. Moreover, as the small data scenarios are multiple, it becomes necessary to clarify and distinct the different versions of these problems prior to addressing any particular instance of it. We have therefore addressed this question through two main methodological axes (see fig. 1.1):

- An extensive literature review work targeted at posing a clear and concise view of the

[1] Zero-sum game in the game-theory sense of the term.

[2] Occasionally referred to as the "GAN zoo". See `https://github.com/hindupuravinash/the-gan-zoo` for a frequently updated list

field and the problem to be treated (chapters 2 to 5);

- Experimental propositions exploring novel mechanisms within the reviewed framework (chapter 6).

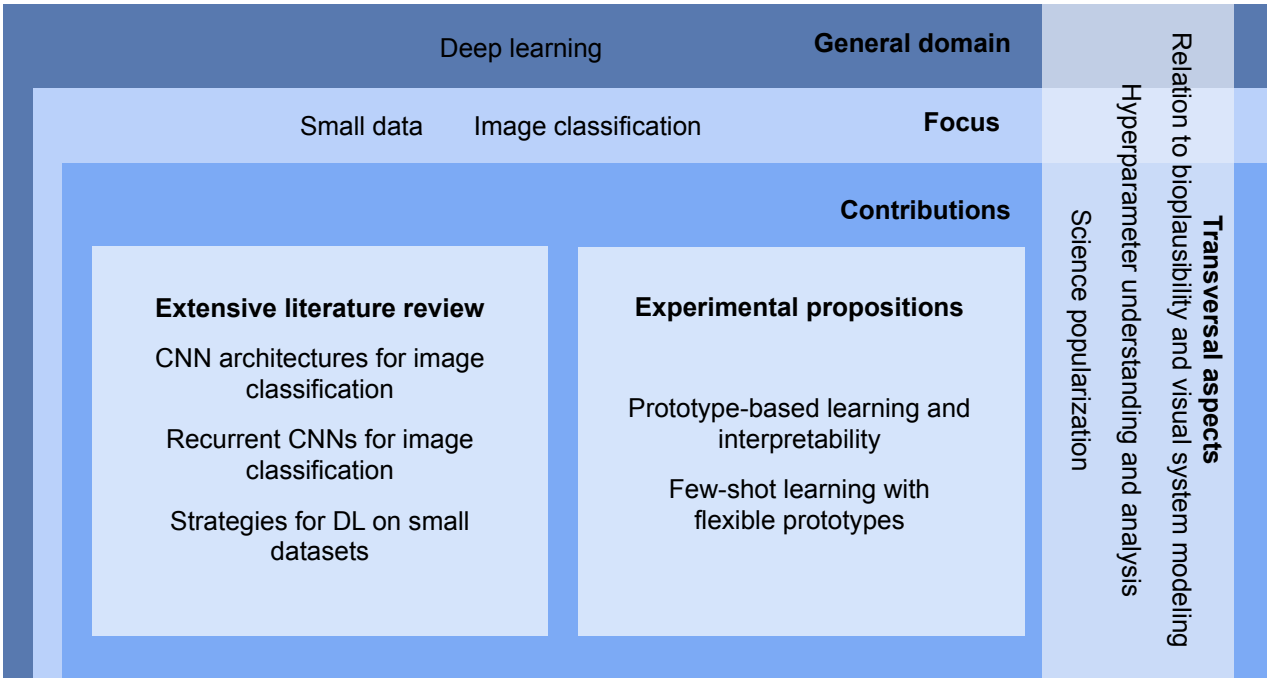



**Figure 1.1** Thesis organization diagram.

**Outline** From our discussion it is also clear that the problem of object recognition with CNNs on small datasets is in fact a meta-problem enclosing multiple sub-problems. As such, it is necessary to (i) identify its composing parts and (ii) decide which and how to address each of them. We have thus decided for a two-part organization of this work, first addressing object recognition (part I), then following with a specific treatment of the small data case (part II).

In part I we have approached object recognition from an interdisciplinary point of view, discussing artificial CNNs as well as their biologically inspired aspects:

- First of all, we present the basic functioning of object recognition, both in CNNs and in the brain (chapter 2). In particular, we review recent work comparing CNN and cortical representations, and summarize functional and architectural features not modeled by typical CNNs.

- As modern capabilities of object recognition have been achieved with feedforward CNNs, this category of models is reviewed and organized in chapter 3. While other reviews on the matter have been published in the course of this thesis work, they did not

necessarily frame the subject in a way pertinent to the work developed here. Therefore it was in any case relevant to revise the progress history of architectural principles and innovations, highlighting the trends pertinent to this present work.

- Complementing the previous review work, chapter 4 addresses recurrent and feedback CNN models for image classification. The chapter aims to organize the reviewed models according to their use of recurrence, from both functional and architectural perspectives. Besides proposing a literature survey with innovative focus and perspectives, the study of recurrent processing in CNNs for image classification further exemplifies the potential knowledge exchange between neuroscientific modeling and artificial neural networks.

Having posed our paradigm of object recognition, we proceed to addressing the conditions corresponding to small data learning in part II in two steps:

- A more detailed description of the problem is presented in chapter 5, including a review of the main strategies present in the deep learning literature. This chapter is an effort towards organizing existing knowledge in the field, while linking it to related tasks presenting similar difficulties.
- Ultimately, the work is completed by experimental propositions within the small data framework, including the proposition of novel mechanisms and perspectives, building upon existing deep models.

Eventually, a typical thesis work tends to open many more research questions than it closes. After an extensive review work, it is natural to identify spaces of possible contribution and interaction between fields that could help advance knowledge. As these could be the subject of many other thesis, these open perspectives will be exposed and discussed in the thesis conclusion.

# Part I

# Object recognition with deep convolutional neural networks


**2 Convolutional neural networks and visual system modeling** **27**
2.1 Introduction . . . 27
2.2 Convolutional neural networks . . . 29
2.3 Visual system: a schematic overview . . . 32
2.4 The object identification pathway in relation to CNNs . . . 34
2.5 Discussion . . . 40
2.6 Conclusion . . . 45

**3 Feedforward CNN architectures for object recognition** **47**
3.1 Introduction . . . 47
3.2 Single pathway models . . . 49
3.3 Parallel pathway archs . . . 52
3.4 Discussion . . . 59
3.5 Conclusion . . . 62

**4 Recurrent and feedback CNN architectures for object recognition** **63**
4.1 Introduction . . . 63
4.2 Functionalities brought by recurrent processing . . . 64
4.3 Architectural trends . . . 69
4.4 Discussion . . . 79
4.5 Conclusion . . . 83

Chapter 2

# Convolutional neural networks and visual system modeling

## 2.1 Introduction

Traditionally, object recognition was tackled from a pure computer vision framework, including a thoughtfully designed feature extraction algorithm, in conjunction with some robust machine learning model, for instance a support vector machine (SVM), to correctly classify the object category (Andreopoulos and Tsotsos, 2013). The key point is that these features were not learned from data, but demanded careful engineering from image processing specialists, leveraging knowledge of intrinsic image properties and/or image statistics (from natural or domain-specific images Torralba and Oliva, 2003; Hyvärinen et al, 2009), which were relevant to the problem at hand. The computed features were condensed into a feature vector, also referred to as an image descriptor.

Designing image descriptors was an important line of research of its own until the early 2010s (Zheng et al, 2018). Image descriptors are typically obtained after non-linear filtering operations, which detect basic image characteristics (such as color, texture or higher order statistics) — a process known as feature extraction. Features can be computed globally from the entire image, but since 2003 the study of local descriptors - such as scale-invariant feature transform (SIFT) (Lowe, 2004) and histograms of Gaussians (HoG) (Dalal and Triggs, 2005) — gained importance Zheng et al (2018). In that case, images are partitioned with features being computed for each part — object/background segmentation for instance. Then these local features can be aggregated (for instance, computing histograms) into a single feature vector.

Meanwhile, inspired by how text documents could be represented via a "bag-of-words" model, image descriptors started to be encoded according to how they represented the database at hand, before being directly used for classification or retrieval, Sivic and Zisserman (2003) being an early proponent of this approach. The idea was to create a dictionary of visual words — via clustering for example — that will correspond to different feature dimensions in the final representation vector. It is worth noticing this strategy has brought a learning component to the usual recognition pipeline, as the "bag-of-visual-words" is usually

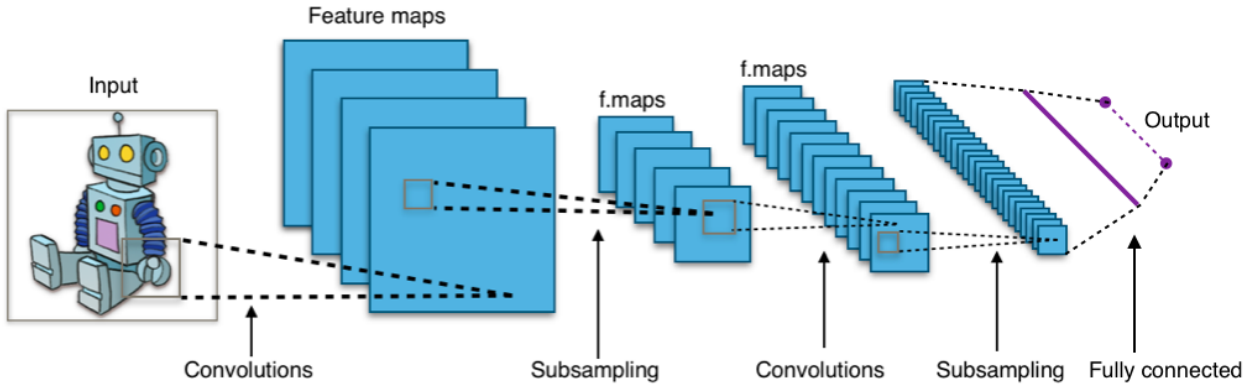



**Figure 2.1** Typical CNN. Credits: Aphex34 CC BY-SA 4.0

learned from the dataset. This combination of feature extraction and "bag-of-features" was extensively used in both image classification and retrieval, before the popularity rise of deep networks (Yang et al, 2007; Zheng et al, 2018).

It was with surprise, during the Large-Scale Image Recognition Challenge of 2012, that the community received the news that a deep convolutional neural network (DCNN) fed with raw pixels with little pre-processing had beaten all the traditional methods based on handcrafted features by a significant margin (Krizhevsky et al, 2012; Cardon et al, 2018). With inspirations from the neurobiology of the visual cortex, convolutional networks are named after the basic image processing operation they perform, cascaded with point-wise non-linearities and sub-sampling (pooling) (see fig. 2.1). This class of models has been around long before this technical breakthrough, but had not yet proved able to outperform well founded computer vision methods in practice. Availability of large datasets together with improvements in hardware and GPU programming tools were fundamental for the recent success of these large architectures.

With this result, an important paradigm shift intervened in the field of object recognition. Instead of concentrating efforts in feature engineering and the extraction of good descriptors, the use of deep neural networks can implicitly extract intermediate representations of the input data, achieving high accuracy in their classification. The hierarchical way in which visual features are detected by deep networks parallels the hierarchy of the early visual cortex, but considering only a feed-forward simplification of it. These relations will be further discussed in the course of this chapter (sections 2.4 and 2.5.2).

Early versions of hierarchical neural architectures with local receptive fields date back to the 1980s, with the Neocognitron architecture (Fukushima, 1980), based on functional and architectural principles observed in the early visual cortex (Hubel and Wiesel, 1962). This bio-inspired model sparkled the development of the first backpropagation-trained convolutional networks designed by LeCun et al (1990a), who successfully applied the model to handwritten digit recognition. After Krizhevsky et al in 2012, many architectural innovations were proposed

including parallel branches (Szegedy et al, 2015) and forward shortcut connections (He et al, 2016a). These recent advances will be discussed in more detail in chapter 3.

Along with the initial bio-inspiration, other functional similarities have been observed in CNNs trained on natural data and the way they hierarchically extract and integrate visual features. Conversely, much of the functionality of the biological visual system remains unaccounted for in these convolutional models. In the following sections, the basic functioning of modern convolutional networks is presented along with a description of its main building blocks. We then follow along to a brief description of the current knowledge about the visual system from neuroscience. Since our focus remains being artificial networks, this text has no intention of being a self contained introduction to visual system modeling. Both these introductory descriptions set base for the subsequent discussion on the relation between CNNs and visual cortex modeling, highlighting to which level and extent they mimic biological function, along with aspects that remain absent from mainstream models.

## 2.2 Convolutional neural networks

Convolutional neural networks (also known as CNNs or ConvNets) are typically constituted of at least two convolution layers. Since convolutions are linear operations, cascading such layers would be reducible to a single linear operator. On a ConvNet, however, as on a traditional multi-layer perceptron (MLP), this operation is followed by a non-linear operation, the activation function. Every few layers, a spatial subsampling (pooling) operation is performed, aggregating information from multiple neurons in the previous layer. These operations and their functions will be discussed below. The reader familiar with these concepts can skip to section 2.3.

Since the focus here is on object recognition from images, this description of convolutional neural networks refers typically to the case where 2D convolutions are applied to images. Nevertheless, CNNs are applicable to other domains such as audio (for instance Hershey et al, 2017) and even text processing (e.g. Kim, 2014), using typically 1D convolutions.

### 2.2.1 Convolutional layers

A convolution is a standard linear operation, widely used in signal processing as a filtering operation. In 2D, a convolution is the operation used to apply a filter (or kernel) to an image. Intuitively speaking, a convolution consists in reversing the kernel and sliding it over every position in the image, multiplying the overlapping pixels and adding them up to generate one pixel of the filtered image. More strictly, the operation typically performed in neural networks skips the filter reversing part, corresponding actually to a cross correlation operator. However, in the neural network community, it became standard to refer to this operation as convolution, and this is the definition used throughout this thesis. A mathematical expression

of the (discrete) convolution operation (in the neural network sense) can be given as follows (Goodfellow et al, 2016b, chapter 9):

$$\mathbf{S}(i,j) = (\mathbf{I} * \mathbf{K})(i,j) = \sum_{m}^{H} \sum_{n}^{W} \mathbf{I}(i+m, j+n)\mathbf{K}(m,n) \tag{2.1}$$

where $I$ is the input image matrix, $K$ is the filter kernel matrix and $H, W$ are the image's height and width, in number of pixels.

From a traditional neural networks point of view, this operation can be described as follows. Each pixel of an image correspond to one of the input neurons, which are arranged in a 2D grid. Each pixel of the output image corresponds to a hidden or output layer neuron. Connectivity between the two layers, however, will not be thorough but limited to local vicinities: each output neuron is connected to a sub-grid of the input layer. The synaptic weights of this local connectivity mask corresponds to the weights the convolution filter. For the correspondence to be complete, these weights should be shared among all output neurons. In summary, a convolution layer can be seen as a locally connected neural layer with shared weights.

These characteristics largely reduce the number of parameters in a CNN, if compared to a traditional fully-connected MLP. It is an interesting example of architectural domain adaptation to reduce model complexity, a theme further discussed in chapter 5. Another interesting characteristic of CNNs derived directly from this weight sharing (or sliding filter) is its equivariance to translations: a spatially translated object results in its activation being equivalently translated on later layers.

Variants of the basic convolution layer include strided convolution (where the step taken in the sliding process is larger than 1) (Springenberg et al, 2014), dilated (or *à trous*) convolution and deconvolution widely used for semantic segmentation (Chen et al, 2016; Wang et al, 2017b), tiled convolution (where instead of a single kernel matrix, a set of kernels is circled through as we slide over the input) (Le et al, 2010), among others.

As done in traditional image processing, convolutions with a particular filter work as pattern detectors. Early visualizations of filters learned by a CNN trained over the large scale ImageNet dataset (see appendix C) indeed demonstrated how those of the first layers resemble basic traditional detectors, such as Gabor filters (orientation) and color filters, particularly in the first layers (Zeiler and Fergus, 2014). As we rise in the hierarchy, it becomes more complicated to understand the patterns being detected, but the principle is the same.

### 2.2.2 Pooling or subsampling

Another operation present in the pipeline architecture of a ConvNet is spatial subsampling. In this operation the input divided in portions with a regular grid, with each part being summarized to a single value, reducing the dimension of the feature maps. This operation is

known as pooling or subsampling. Typical pooling operations are the max pooling (where the region is summarized by the largest value among them) and the average pooling (region replaced by its mean). When it is important to avoid loosing too much spatial resolution, a frequent alternative is to delay subsampling to the final layer, applying a global average pooling at the final feature maps and just before the output classifier layer — such as done in GoogleNet (Szegedy et al, 2015) or in residual networks (ResNets) (He et al, 2016a). More recently, an adaptive pooling strategy has been proposed for specific applications, using traditional segmentation algorithms to define an adaptive pooling mesh (Tsai et al, 2015). It has also been proposed to substitute pooling layers for strided convolutions, which also achieve spatial aggregation and dimensionality reduction, but with learnable parameters (Springenberg et al, 2014). Nevertheless, max-pooling has worked well in practice and remains the most popular subsampling operation in CNNs.

### 2.2.3 Activation functions

In traditional neural networks the neurons have an non-linear activation function. Originally a thresholding Heaviside function, common smooth substitutes are the hyperbolic tangent and the sigmoid function. These two activation functions are bounded, saturating for both positive and negative extremes. Additionally, they are differentiable, which is a necessary property for the backpropagation algorithm. They were most common when building shallow networks, but more recently, the rectified linear unit (ReLU) has proven to work better experimentally, specially when dealing with deeper networks (Glorot and Bengio, 2010; Jarrett et al, 2009). These functions are applied to each neuron independently, on a given layer. Using the ReLU means particularly that only positive values get through to the following layer. Although this function is not differentiable everywhere — since it has a discontinuity at 0 — it is so in a piece-wise manner — before and after 0. This property is used in order to derive an adapted version of gradient backpropagation through ReLU units.

An experimental analysis by Glorot and Bengio (2010), monitoring activations and gradients has brought some light on the reason of ReLU's success. While the saturation of sigmoid and tanh units is not a significant issue when using shallow networks, it can slow down (or even impede) convergence for deeper networks — a problem known as the *vanishing gradients*. These saturating activations have derivatives that tend to vanish with their corresponding backprogation updates on lower layers, due to successive multiplications by increasingly small values. On the other hand, ReLU activation does not saturate, and its derivative avoids the vanishing of lower layer gradients.

Some other activation functions have been explored, beginning with variants of the ReLU such as the leaky-ReLU, the exponential and scaled exponential linear units (ELU and SELU) (see a comparison by Pedamonti, 2018), but also extending to completely different approaches such as interpolating functions (Wang et al, 2018a).

## 2.3 Visual system: a schematic overview

To clarify to which extent CNNs are related to the functioning of our biological vision, it is important to understand how this process takes place in the brain and how it is modeled from a neuroscience point of view.

Before discussing similarities and differences between biology and CNNs, it is important to have in mind at least a rough notion of visual cortex organization and function (section 2.3). More detailed descriptions will follow while discussing the object recognition pathway and its similarities to feedforward convolutional networks (section 2.4) and particularly when discussing aspects missing from these basic feedforward models (section 2.5.2).

In order to build a global picture, this section presents a brief overview of two important aspects of the visual system: its hierarchical organization and the traditional division in dorsal and ventral streams. Readers familiar with these topics may skip to section 2.4.

### 2.3.1 A hierarchical system

The biological visual system is traditionally viewed a hierarchical succession of specialized brain regions. This idea goes back to Marr (1982), who first proposed a model in three levels, each computing increasingly complex properties from the visual stimuli. In his model, these hierarchical levels processed information sequentially, aggregating outputs from the previous level. First, the *primal sketch* level would detect light intensity changes and how they are geometrically organized. Pooling from the first level features, the $2\frac{1}{2}$*-D sketch* level should convey information associated with surfaces and their orientation in space, the perceived distance to the viewer, their reflectance and illumination, dealing with discontinuities in these quantities. The final *3D sketch* level, though keeping some surface information, would shift the perspective to the object, portraying a representation of its dimensional structure (Marr, 1982, chapter 1).

The general idea of a hierarchical organization was supported later by anatomical studies, in particular by Felleman and Van Essen (1991), who proposed a description of the visual cortex organized in a hierarchy of regions, with ascending and descending connections (see fig. 2.3). Later, several neurophysiological studies have observed how neurons of lower or upper regions are tuned to prefer more or less complex stimuli, resonating with the general idea of incremental perception of visual features (although not complying precisely with the geometrical view by Marr of 2.5D and 3D representations, as discussed further in section 2.4). In a nutshell, a more modern view of Marr's hierarchy would be comparing the primal sketch level to the early-vision front end, a second sketch level to mid-level areas extracting higher-level features, with finally a third sketch level corresponding to upper regions computing more abstract representations.

As far as this first rough description is concerned, visual perception starts at the retina, with photoreceptive cells stimulating ganglion cells that form the optic nerve. These optic fibers run to the center of the brain, with connections to the thalamus, mainly the lateral geniculate nucleus (LGN), but also the pulvinar (Pul) (Cortes, 2012). From the thalamus, optical fibers go to the occipital lobe – the back of the brain – to the primary visual cortex (V1) (see fig. 2.2). The primary visual cortex detects basic visual indices, such as orientation patterns, which are fed to upper regions in the visual system anatomical hierarchy. Even though a major sequentially hierarchical path can be identified (highlighted in red in fig. 2.3), including mainly areas V2, V3,V4 and IT, the interconnection pattern between regions is more complex and includes multiple parallel connections and shortcuts. A more detailed description, focusing on regions and mechanisms relevant for object recognition, will be done in the following sections.

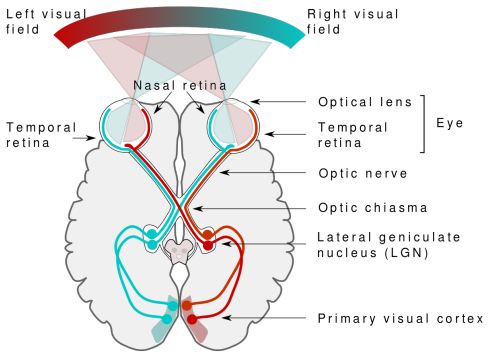



**Figure 2.2** A simplified anatomic schematic of the visual neural pathways in the brain, from retina to the primary visual cortex (V1). Credits: Miquel Perello Nieto CC BY-SA 4.0

### 2.3.2 The schematic two-stream model

Besides its remarkable hierarchical organization, another prevalent view of the visual system is its anatomical division in a dorsal and a ventral streams. Neurophysiological experiments (e.g. Mishkin et al, 1983) have led to a popular (although rather schematic) view of these two streams as functionally distinct (fig. 2.4):

- The dorsal pathway, also known as "Where" (or "Where-How") pathway, is implicated in processing of movement and localization of objects. This pathway interacts with motor areas, being also involved in the ability to manipulate an observed object.
- The ventral pathway, also referred to as the "What" pathway, mostly implicated in processing object identification and/or categorization.

Looking at a lower level, one can see this specialization and division of processing actually starts earlier. In the retina, photoreceptive cells feed into ganglion cells of different kinds, which in turn yield pathways with distinct functions. To mention the two most well known, there is the magnocellular-pathway which has a faster communication of spatially coarser luminance-based inputs, while the parvocellular-pathway conveys more contrast-based detailed spatial information (particularly from the central vision) at the expense of a lower time sensitivity. The fast magnocellular pathway is implicated in fast processing of movement, and concentrates towards the dorsal pathway, correlating with its function of motion processing. The slower

parvo-cellular pathway conveys detailed information mostly used in the ventral pathway for more precise object identification.

The reader may refer to specific literature for details, such as Ungerleider and Haxby (1994) for more on the two-stream view, or Goodale and Milner (1992) for more on the "Where/How" pathway.

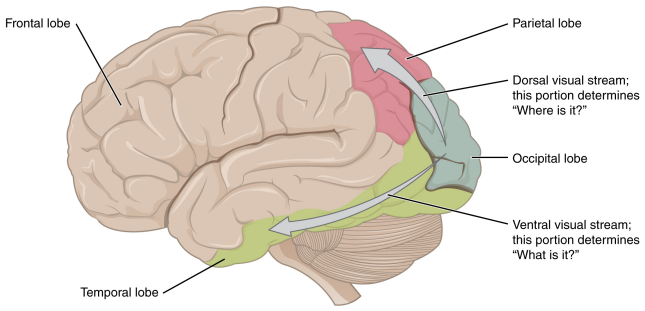



**Figure 2.4** After reaching the primary cortex in the occipital lobe, visual processing is roughly divided in two streams or patwhays: the dorsal pathway goes up to the parietal lobe, while the ventral pathway goes lower towards the inferior temporal lobe. Credits: OpenStax College CC BY 3.0

This schematic view of functionally distinct regions distributed in two independent streams is classic in the field, although rather imprecise. Indeed, under detailed scrutiny, the direct identification between anatomy and function is not completely accurate, as there are two-way connections linking non-adjacent regions as well as connecting ventral and dorsal streams. These and other aspects that portray a more complex view of the visual system will be discussed in section 2.5.2. For instance, Casagrande (1994) discuss the limits of this oversimplifying two-stream view, proposing a third pathway, which is closely related to non-standard retina cells (as discussed in Gollisch and Meister, 2010)[1]. This additional stream is related to fast reaction to perceived dangers, structuring a coarse and fast recognition tightly linked to sensory-motor action, thus beyond the scope of our study.

## 2.4 The object identification pathway in relation to CNNs

Overall, CNNs reflect the organization of the feedforward part of the ventral pathway, though in a very simplified manner. There are arguments to support this feedforward path from retina to IT is enough to explain our basic object recognition abilities (DiCarlo et al, 2012). However, more complex tasks such as tracking a moving object or identification under occlusion might demand more complex mechanisms currently ignored by standard CNNs (Roelfsema, 2006). This section dives into the "what" pathway and explains its main properties, while drawing a parallel with similar characteristics found in standard ConvNets.

### 2.4.1 Core object recognition

Primates are capable of precise identification or coarse categorization within less than 100 ms (Fabre-Thorpe et al, 1998). Additionally we are robust to a multitude of identity-

[1] Readers particularly interested in this topic may refer to Teftef et al (2013) for a review and modeling of these retinal cells, or to Carvajal (2014) for their relation with cortical visual system.

preserving transformations such as object position, size and pose, lighting conditions and background context. This cognitive ability is defined by DiCarlo et al (2012) as the problem of core object recognition. It is of course a very basic part of visual perception as a whole, and does not account for the totality of visual tasks primates can accomplish.

Experiments observing electrophysiological signals from neuronal populations in the cortex of primates (e.g. Fabre-Thorpe et al, 1998) support that core object recognition can be achieved within exposition intervals as short as 100 ms. This *fast-brain* processing mechanism proposed by Fabre-Thorpe et al has also been modeled computationally. In particular, Viéville and Crahay (2004a) have proposed an implementation training a shallow neural network along with solving a primal SVM classifier.

Furthermore, this time is sufficient for observation of related feed-forward activity up to the IT cortex, indicating that some form of recognition can be performed even before any top-down mechanism kicks in. While more recent work has highlighted the important role played by feedback loops and recurrences in vision as a whole (e.g. Roelfsema, 2006), feed-forward mechanisms have been shown sufficient to explain this restricted ability of fast "core object recognition" (DiCarlo et al, 2012). This *fast-brain* recognition is however limited to simpler cases, in which there is relatively little ambiguity about the object to be recognized.

**Building invariance** Computationally, to make a decision on the identity of an object, a subject should have a neuronal representation of the visual scene, with some set of neurons "reading out" this representation to provide or not a strong activation signal concerning the object category, following its presence or absence in a given part of the visual field (DiCarlo and Cox, 2007). Additionally, since primates are able to recognize objects across a multitude of transformations — position, pose, illumination and clutter — these representations would ideally be invariant to these identity-preserving transformations (DiCarlo et al, 2012).

Given the hierarchical aspect of the ventral stream, the search for an invariant representation translates to how the visual system can incrementally compute such invariants, obtaining approximately the same higher level activations for different initial retinal stimuli arising from the same object (or category). A geometrical intuitive description of this framework is to think about different object categories represented as entangled manifolds in the retinal space (DiCarlo and Cox, 2007). By separating dimensions related to each identity preserving transformation, the processing in each visual area takes a step towards untangling the object manifolds, allowing categorization to be achieved by a linear readout performed by a population of neurons.

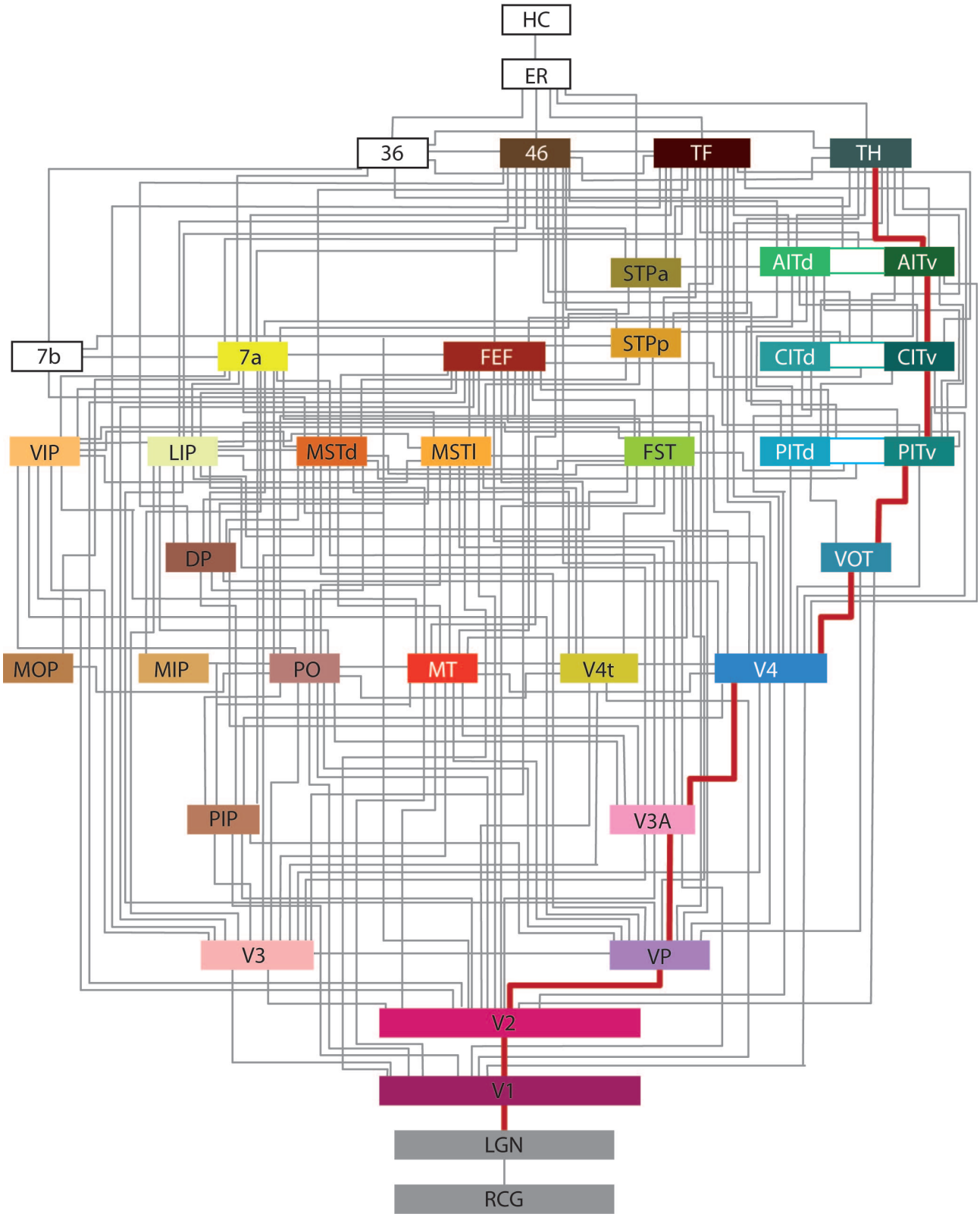



**Figure 2.3** Anatomical hierarchy of brain regions implicated in visual processing, according to Felleman and Van Essen (1991). The ventral pathway associated to "core object recognition" is highlighted in red. Credits: Cortes (2012).

### 2.4.2 Early vision: from retina to V1 cortex

Visual processing is not done exclusively by cortical areas. As the first luminous impulses fire up retinal ganglion cells, information processing is already taking place. Moreover, the way luminous signals are acquired and transmitted is intertwined to how later processing stages are organized. The central part of the visual field is captured through the fovea, a portion of the retina more densely packed with receptors than its surroundings. This variation in the concentration of receptors implies in a higher resolution central vision associated with a lower-resolution periphery vision. As a result, eye movements are essential in order to acquire a higher resolution perception of an entire scene. Most animals perform rapid eye movements (between 200 and 500 ms) (Land, 1999; DiCarlo et al, 2012), acquiring different "shots" of the same scene. As we are mainly concerned with recognition on still images, details of this acquisition mechanism are out of the scope of this review. For the following, descriptions concern the treatment of a unique visual frame, a single snapshot of the perceived scene

Some characteristics of the pre-cortical vision and early V1 processing are also tightly linked to the anatomy of image acquisition, such as the preservation of the spatial topology in the retina — a property known as retinotopy — and in consequence the spatially localized receptive fields. These characteristics have analogous counterparts in convolutional models, and will thus be further described in the following sections.

#### 2.4.2.1 Retinotopy

The visual field is not perceived uniformly by the retina, with the center of the visual field being over-represented in comparison to peripheral regions — due to the higher density of receptors present in the foveal region. Still, 2D spatial vicinities are retained, such that retinal perception implements a non-orthogonal transformation of the visual field. It has been observed, in experiments within cat's cortex (Hubel and Wiesel, 1962), that V1 retains this particular retinal topological organization, a property known as retinotopy. In primate visual cortex, retinotopic maps have been observed also in V2 and V3 regions, and the posterior region of the IT cortex (pIT) (DiCarlo et al, 2012). By analogy, it can be said that retinotopy is also a characteristic of convolution layers and, together with spatial pooling, helps to build up tolerance to small translations.

This 2D representation of the visual field present in these early regions raises interesting problems that have to be solved in the path towards object recognition. First, such spatial representation needs to give place to a more abstract representation, invariant (or at least tolerant) to spatial transformations that preserve object identity. This issue is partially addressed by CNNs, which achieve at later layers a more category-centered representation, being equivariant to translations. Second, the retinotopic representation being eye-referenced, achieving a pose invariant representation of an observed three-dimensional object may require

a shift in perspective to a different spatial frame of reference. This ability is not fully explained only by a feed forward pipeline, not present in standard CNNs.

#### 2.4.2.2 Receptive fields

Standard receptive fields in the retina are of center-surround type, as observed by Hartline (1940) and Kuffler (1953). Having an ON-OFF or OFF-ON response, they perform the computational function of filtering the "raw image" captured by the photo-receptors. At the functional level, such filters either implement contrast detection — detecting local spatial intensity variation — or motion detection — detecting temporal intensity variation. Later Hubel and Wiesel (1959, 1962) also recognized these concentric receptive fields on LGN, while in the primary visual cortex (V1), different elongated center-surround cells were observed.

Furthermore, in their study of cat's V1 cortex, they noticed the existence of more sophisticated receptive fields, which were classified as *simple*, *complex* and *hypercomplex cells*. Though all respond to bars and edges in particular orientations, simple cells are spatially spread detectors, while complex cells pool from the activation of simple cells, locally building tolerance towards spatial translation. A similar process goes on for hypercomplex cells, though they exhibit a clear non-linear behavior while complex cells are only linear.

Complex and hypercomplex cells are also selective to directions, while the latter are selective to lengths as well. It was later demonstrated that this separation in simple or complex receptive fields is not static, but actually highly adaptive depending on the visual input statistics (Fournier et al, 2011).

This pooling mechanism, together with the orientation selectivity, served as inspiration for many computational models of the visual cortex (e.g. by Riesenhuber and Poggio, 1999; Serre et al, 2005), and for bio-inspired computer vision models such as the Neocognitron (Fukushima, 1988). As described in the previous section, the same architectural feature has been carried over to modern CNNs. The successive aggregation of small receptive fields amounts to neurons responding to larger portions of the visual field, at the expense of a reduced spatial resolution. While in the brain there is another pathway specialized in spatial localization, standard CNNs used in object recognition simply lack spatial resolution at higher layers, with ad-hoc mechanisms being used to achieve tasks such as localization and segmentation (see chapter 3).

As of their learned receptive fields, visualizations of filters learned in the first layers of a CNN demonstrate how some of them correspond to similar basic detectors of orientation, center-surround, striped patterns and so on (Zeiler and Fergus, 2014; Springenberg et al, 2014). Nevertheless, detectors learned by upper layers are more complex not prone to such an intuitive observation, even though many efforts have been made towards producing semantically meaningful visualizations of their preferred stimuli (see Olah et al, 2017, for a review and interactive examples).

### 2.4.3 Mid-level visual areas

Although computations performed by the primary visual cortex and the infero-temporal cortex have been studied and understood to high detail, less is known about the computations performed in mid-level processing. Due to multiple phenomena such as clutter, occlusion, and 2D retinal representation, visual input representation is often impoverished, ambiguous, incomplete and noisy. Therefore obtaining an object-level representation invariant to identity preserving transformations demands not simply feature extraction, but actual feature inference from image statistics (Purves et al, 2014). There is evidence to support that mid-level areas such as V2 and V4 play an important role in attending to these demands.

While secondary visual cortex (V2) shares many sensitivity characteristics with the previous V1 area — including edges, color, orientation, spatial frequency — it is also sensitive to other cues such as angle and curvature (Kubilius et al, 2015). In addition, V2 neurons are sensitive to second-order edges, illusory contours inferred from differences in other visual cues such as texture patterns. This sensitivity has also been observed in V4, and includes inferring potential borders from discontinuities in orientation, motion and contrast (Kubilius et al, 2015). Another perception encoded by V2 cortex is border ownership, starting to give edges an object pertinence sense and thus defining contours (Perry and Fallah, 2014).

V4 neurons combine orientation responses from both V1 and V2 cortices, effectively encoding angles and curvature. It is no surprise then that neurons tuned to simple geometric shapes have also been identified in this area, although no extensive mapping is known (Perry and Fallah, 2014). V4 cortex has also been observed to be sensitive to complex curved fragments and three-dimensional parts of objects (Kubilius et al, 2015). This region also seems to be of importance to color vision, with center-surround receptive fields representing perceived color (as opposed to physical color), as well as hue-tuned cells invariant to luminance (Perry and Fallah, 2014).

### 2.4.4 Infero-temporal cortex

The IT cortex is usually referred to as the region responsible for object identification, and the idea of "grandmother neurons" — object-specific tuned neurons — often arises when comparing classifier CNNs to biology. While it may be an appealing view easy to explain, current knowledge portrays a more nuanced picture of which stimuli the IT cortex actually responds to and how it participates in object recognition.

A first striking observation is the difference in organization, when compared to the previous cortical regions. Although retinotopic maps are still observed in posterior IT, retinotopy is lost on central and anterior IT. Instead, a more semantic anatomical organization seems to have taken place. This view is supported by the existence of some task-specialized regions, particularly in face processing, identified in imaging studies (e.g. Freiwald et al, 2009).

Multiple experiments (e.g. Majaj et al (2015)) have demonstrated that object identity can

be inferred from IT neuronal activity, using a simple weighted sum scheme (i.e. a linear regression) . Individual IT neurons are activated by at least moderately complex combinations of visual features, and maintain their relative object selectivity over small to moderate changes in position, size, pose, illumination and background clutter (DiCarlo et al, 2012).

It was long thought that single neurons were highly selective towards a particular object while invariant to transformations in its position, pose, etc. This idea of object-specific neurons became known as “grandmother neurons”, and has been questioned ever since. In fact most IT neurons are not exclusively selective of one kind of object, but broadly tuned and responding to many different images and objects. Moreover, it has been observed that neurons with the highest shape selectivity are the least tolerant to these transformations, going against the idea of "grandmother neurons".

Indeed, a more contemporary view of IT portrays a distributed representation of visual semantics, with sub-populations of neurons being responsible for detecting particular concepts, instead of individual neurons. Furthermore, with such a distributed representation, individual neurons need not be *invariant* to any transformation, but *tolerant* to some small transformations for the visual stimuli. The aggregation of multiple tolerances to different transformations could then build sufficient invariances across a neuron population (DiCarlo et al, 2012).

This population coding view implies that the dimensionality of the IT cortex representation does not necessarily correspond directly to the number of neurons involved. In fact, the intrinsic dimensionality of this semantic space has been estimated in a neurophysiological study by Lehky et al (2014), based on stimuli-response recordings on 647 neurons in the macaque IT cortex. Their analysis has indicated an approximate dimension of 100 independent features, which is a surprisingly low number when compared to the $10^2$ to $10^3$-dimensional vectors typically present in typical CNNs’ upper layers. This fast comparison however neglects manifold hypothesis that typical data points expectedly lie near a manifold of lower intrinsic dimension, embedded in this high dimensional space. In this case, semantic representation in upper CNN layers may also have a much inferior intrinsic dimension.

## 2.5 Discussion

### 2.5.1 Comparisons of cortical representations and CNN representations

The success of CNNs in computer vision has also incited interest from the computational neuroscience community. Multiple studies have compared, under equal stimuli, registers of IT neuron populations — both electrophysiological and fMRI — to representations learned by standard CNNs — which had previously been supervisedly trained on large datasets of

natural images. The representations extracted from the CNNs, though not optimized to fit the neural recordings, were highly predictive of IT activity — both single and multi-unit (Yamins and DiCarlo, 2016).

Using DCNN models by Krizhevsky et al (2012), Zeiler and Fergus (2014) and Yamins et al (2014), Cadieu et al (2014) have compared their representations to V4 and IT cell recordings from monkeys. While the first two models were designed for the ImageNet recognition challenge, the third was trained on the same (smaller) dataset used to obtain IT neuronal recordings, attaining both high performance and high predictivity of IT neuronal activity. They have also included simpler bio-inspired models such as HMAX (Riesenhuber and Poggio, 1999), as well as some V1 and V2-like models in the analysis, though their categorization performance is near chance. On all DCNN models, Cadieu et al observed that the representation obtained from the penultimate layer can well predict IT multi-unit responses. Moreover, DCNNs perform comparably to IT multi-units on an object categorization task (and better than single units or than V4 representations).

Similarly, Khaligh-Razavi and Kriegeskorte (2014) have analyzed some unsupervised and supervised bio-inspired models, and the supervised DCNN by Krizhevsky et al (2012). They have observed that models which are not strongly supervised fail to explain IT data (monkey cell recordings and human fMRI), not correlating with its representational structure. On the contrary, well performing supervised models present a similar response structure to that of IT populations, including great clustering of representational patterns by category as well as within-category dissimilarity patterns.

In a study by Eickenberg et al (2016), the comparison was extended to different levels of both artificial and biological hierarchies. Feature maps from different layers were extracted from the OverFeat network (Sermanet et al, 2013) — also trained on ImageNet — and used in a linear model to predict fMRI activations. They observed that while lower layers of the network (1-2) predict better responses in v1 and v2, upper layers (3-5) predict better responses in upper areas (V3a, V3b and lateral occipital cortex (LOC)), with no clear preference for predicting v4 responses, which got predicted by all layers with equivalent accuracy. Overall, early layers correlate to early visual areas, while top layers correlate more with higher visual areas. The correspondence of intermediate levels however is not clear.

A large-scale study comparing CNNs, humans and monkeys was done by Rajalingham et al (2018). They were put perform the "same" visual recognition task, over the same dataset. Of course, the equivalence of the tasks between animals and CNNs is debatable, but in this case it simply meant that the network had to classify the same stimuli images presented to human and monkey subjects. Beyond global accuracy metrics, they have also observed accurate prediction of confusion patterns — at object-level — could be made by several state-of-the-art image classification CNNs (Krizhevsky et al, 2012; Zeiler and Fergus, 2014; Szegedy et al, 2015; Simonyan and Zisserman, 2015; Szegedy et al, 2016a; He et al, 2016a). However, when examining the discrimination performance on individual images, models were significantly

non-predictive of primate behavior, showing that current CNN models do not fully account for behavioral patterns of primates. The performed task consisted in binary discrimination after brief presentation (100ms) of a synthetic naturalistic image. Synthetic images were used to produce controlled multiple view points of 3D objects laid over an uncorrelated realistic background (avoiding classification by covariant background, common on natural images).

In a follow up effort to extensively and systematically benchmark how closely different models explain neuronal data, Schrimpf et al (2018) proposed Brain-score: an ensemble of both neural and behavioral benchmarks to score how similar an a artificial neural network (ANN) is to brain-like mechanisms of object recognition. ANNs with higher performance on ImageNet tend to have higher functional similarity to the ventral stream, scoring high on predicting V4, IT and behavioral data. However for the highest performing models (over 70% top-1 accuracy), this correlation is weaker and the variability between models is non-trivial, leading to reject the hypothesis of any direct correlation (and potential causality) between good ImageNet performance and high predictivity of brain activity. Moreover, considerable variability in activity (both neural and behavioral) remains unpredicted by any models. From the multiple models evaluated up to now, the top-3 brain-like models are the deep shortcut architectures DenseNet-169 (Huang et al, 2017) and ResNet-101 (He et al, 2016a), and also the shallower and more closely bio-inspired Cornet-S (Kubilius et al, 2018)). The project has an associated updatable platform for scoring and comparing models, released under `brain-score.org`, where both ANN models and neuronal or behavioral data can be submitted to augment the benchmark.

Overall, many comparisons have been made, revealing links between natural and artificial representations at different levels. It is worth noticing that feedforward CNNs have not provided accurate models of ventral neuronal activity when trained under easier conditions or on unsupervised tasks (Nayebi et al, 2018; Hong et al, 2016; Khaligh-Razavi and Kriegeskorte, 2014, more on section 2.5.1). This indicates current models need a sufficiently challenging task and/or supervision in order to learn representations that correlate with those along the ventral stream. Moreover, while low and top levels alone present clearer correlations to V1 and IT cortices, intermediate layers and mid-level areas do not correlate as clearly. Differences are in anyway expected, given that many elements of visual processing, including the participation of extra-cortical areas, are not at all modeled in these CNN architectures.

### 2.5.2 Main differences between biology and basic feedforward CNNs

The current understanding of the visual system goes well beyond the feed forwards fast ventral stream recognition. Notably neglected in most deep learning literature are the roles of extra-cortical structures and recurrent mechanisms, such as:

- Advanced processing and encoding done in retina and thalamic structures (LGN and Pulvinar);

- Interaction between dorsal and ventral streams — thus between recognition and localization;
- The role of recurrent processing and feedback connections.

These three topics are briefly discussed in the next few sections. Among these, the incorporation of recurrent processing is occasionally done when dealing with videos — that is, sequential images. Actually, beyond sequential treatment, recurrent processing has other functions related to a more refined treatment of a visual scene, functions which are starting to get more attention from the DNN community, as reviewed later in chapter 4. The other two topics however remain largely unexplored together with CNN models in image recognition.

#### 2.5.2.1 Processing in retina and LGN

Standard CNN models take as input a raw pixel-based image representation. Taking them as a model of biological vision carries an implicit assumption that luminous intensities are directly processed by the visual cortex without any pre-processing or encoding whatsoever. Of course this is an oversimplification that portrays as irrelevant any computations performed by pre-cortical areas. A more thorough study of the role of retinal and thalamic structures in visual processing makes clear their participation is not to be ignored.

Although often depicted as a passive sensor, the retina has a much more complex structure and function. Several cortex-like computations have been identified in the retina of several vertebrate species (see e.g. Kastner and Baccus (2013) for a review). In mammals, retina is formed by five layers of cells that form a complex recurrent network (with both feedback and lateral connections). It acts as a spatio-temporal encoder of the luminous stimuli, producing an adaptive, sparse, over-complete and temporally precise representation (Gollisch and Meister, 2010).

In higher mammals, each point in the visual field is sampled by about 20 different ganglion cells, with irregular receptive fields and varying spatio-temporal selectivity, thus outputting a variety of basic visual features (Masland, 2001). Retinal ganglion cells are capable of encoding complex features such as basic shapes (static or moving), also predicting changes in the perceived visual field and dynamically self-adapting to such temporal changes.

Overall, instead of providing pure luminance information, as does a camera sensor, the retina mostly encodes changes in the perceived environment, with lower activity when staring a static scene. When recording a video, a retina-like sensor would already output a spatially compressed stream, instead of a simple frame sequence. As detailed in Gollisch and Meister (2010) this corresponds to a "shallow" spatio-temporal processing (as modeled in Teftef et al (2013) for instance) that computes image contrast and temporal changes. Moreover, non-standard retinal cells also react to particular visual events having a characteristic spatio-temporal signature, also being involved in motion anticipation Berry et al (1999).

Similarly, there is evidence that thalamic participation in the visual process goes beyond being a simple relay between retina and the visual cortex, implementing a layer of temporal compression on top of the retinal spatial compression. Pattern motion selectivity and center-surround receptive fields have been identified in cat's pulvinar and monkey's LGN, respectively. Particularly the latter is under control of feedback information coming from the cortex, an example of top-down modulation returning to influence an early stage of vision (Medathati et al, 2016). Furthermore, this feedback to the thalamus completes a recurrent processing loop involved in detecting visual events and selecting visual targets. A more sophisticated modeling of the thalamic participation in vision hs been done by Carvajal et al (2013), who has modeled the thalamus as a distributed computational hub aggregating processing results from cortical and sub-cortical structures (such as the superior colliculus).

#### 2.5.2.2 Recurrent processing

Few models of object recognition include recurrent processing, even though it is effective in integrating information over large portions of the visual field. Recurrent processing allows top-down propagation of attentional modulation, contextual cues and other prior knowledge, integrating with low-level bottom-up information and affecting our final perception, beyond the first 100 ms (see Lamme et al, 1998, for a review).

This top-down modulation seems to play a relevant role in dealing with cluttered scenes, partial occlusion and grouping parts of objects, for example, which are relevant for more complex tasks such as semantic scene segmentation and object tracking (Herzog and Clarke, 2014; Medathati et al, 2016).

The ability to group forms and contours so as to segment a scene to its composing objects makes use of recurrent processing. Even though contrast and contour detection happens quickly at V1 and V2 areas, recognizing segments relevant to a specific task happens after a temporal delay involving feedback and lateral connections (Medathati et al, 2016).

When modeling the object recognition stream as feedforward pooling-based pipeline, detailed information is lost at every pooling stage. In this model, for a target object, at every stage only nearby clutter should harm recognition, and the more clutter the worse it should be. However some contradicting experimental evidence has been observed (e.g. Herzog and Clarke (2014)). In fact, even for a very basic recognition task, demanding only edge and direction discrimination (processed at V1 cortex), nearby distracting shapes can affect recognition in manners not predicted by the pooling model. For instance, adding more distractors may help performance (instead of hurting), while a rotation applied to the same distractors can degrade performance.

This suggests a more global processing of visual information (obtained at higher areas of the visual pipeline) can influence very low-level processing (at the level of V1), thus pointing to the action of feedback connections. Additionally, other experiments observed significant

influence of clutter in low-level processing only for exposition times higher than 160ms, but no effect below 120ms, arguing for the action of feedback connections in the extra exposition time (Herzog and Clarke, 2014).

#### 2.5.2.3 Interaction between dorsal and ventral streams

The classical view of ventral and dorsal streams as functionally independent and distinct, as interesting as it may be as a global first view of the visual system, is clearly over-simplified. Not only the are not independent, they also are not the only processing pipelines int he visual system. As exemplified below, there are multiple communications between these two pathways, integrating different types of features at different stages (Perry and Fallah, 2014). Additionally, at a lower level, there are cell types other than magno and parvo cells, that constitute other processing streams (Casagrande, 1994), all interacting among themselves.

Following the anatomical hierarchy, interactions between the magno and parvo pathways can be observed as early as in V1 cortex, where their signals get mixed and propagated to V2 and V3 areas. An influence of magnocellular signals over the ventral pathway could for instance explain why fast and coarse signals tune activity in the temporal cortex, influencing face recognition mechanisms (Giese and Poggio, 2003). At mid-level vision, exchange of color and motion information takes place between areas V4 (traditionally attributed to the ventral stream) and V5/MT (traditionally attributed to the dorsal stream). Interactions have also been observed as a strong influence of form signals – supposedly processed by the ventral stream – onto local motions analysis and motion integration – which would be the work of the dorsal stream. Such interactions occur at multiple stages in the anatomical hierarchy, playing an important role in resolving motion ambiguities, interpolating occluded information, segmenting the optical flow or in recovering an object's 3D structure (Medathati et al, 2016).

## 2.6 Conclusion

Our understanding of brain function and particularly of the visual system has evolved significantly over the years. Ever since Hubel and Wiesel (1959), there has been a great accumulation of experimental evidence, together with theoretical modeling and descriptive efforts, that have led to the current state of knowledge in the field. Hypothesis have been made at multiple levels — single neurons, populations, cortical and extra-cortical regions — and put to the test of experimental observations. While some mechanisms got to be pretty well-described — like the tuning of V1 and V2 neurons to particular low level visual cues — some other still long for a detailed understanding — such as the role of some inter-region connections or of extra-cortical regions involved in the visual.

Some architectural patterns and functional mechanisms have indeed worked as inspiration in the first CNN designs — namely the locally-connective fields and the hierarchical process-

ing. Their level of bio-plausibility remains though rather shallow, with purely feedforward architectures dominating among the most used models, as reviewed in the next chapter 3. Knowledge in the field has advanced greatly since the early 90s, with novel principles of recurrent treatment, attention modulation and long-range interactions having been observed as crucial for more refined visual treatments. The adoption of these new mechanisms by the DNN community has been done occasionally but is not widespread, as surveyed in the following chapter 4 . The study of these novel architectures may be able to inform neuroscience about the pertinence and effectiveness of these principles in accomplishing a visual recognition task, hopefully shedding light on computational mechanisms previously ignored or underestimated by the neuroscience community.

Being a descriptive science, results may occasionally contradict theoretical predictions or reveal unexpected mechanisms, leading to reformulate the state of what is currently known. One must keep this epistemic status of neuroscientific principles in mind, being open to their eventual reconsideration or even complete dismissal. Attempts at incorporating such principles to machine learning models should be aware of the context from which these principles have emerged and of the hypothesis considered in their formulation. On one hand, a naive accumulation of neuro-inspired principles may result in incompatible theories being brought into the same model, or simply be unhelpful as the operation conditions for a given mechanism may not be verified in the application domain. On the other hand, bringing in the right mechanism can account for a significant improvement, or even bring on novel hypothesis to the neuroscience modeling.

Reciprocally, besides modeling brain function, deep learning research could also take inspiration from neuroscientific methodology. While we don't have detailed theoretical descriptions of DCNNs "behavior", taking a descriptive approach, similar to that of neuroscience, is a potential path towards building knowledge about the higher-level functioning of these models. Overall, the well-informed interaction between the neuro and computer vision communities has the potential of being fruitful for both domains.

Chapter 3

# Feedforward CNN architectures for object recognition

## 3.1 Introduction

Even though feedforward architectures may seem limited in variability at a first glance, the last few years have seen a variety of architectural principles emerge around the basic single pipeline architecture. Multiple parallel pathways (section 3.3), shortcuts or skip-connections (section 3.3.2) and bottleneck structures have been proposed by different works, bringing improvements in accuracy, computational burden or other qualitative aspects of a classification model.

Given the centrality of the subject of object recognition to this thesis, it is important to review the main modern CNN architectures targeted at this aim. The field has been and continues to move fast, with almost simultaneous works sharing common influences and also directly influencing each other, as depicted by the diagram in fig. 3.2. The next sections lay down a portrait of the current state-of-the-art, focusing on feedforward architectures applied to the task of image classification. We propose to organize these propositions according to their usage of parallel proceesing patways within the architecture, as schematized in fig. 3.1. While other object recognition tasks may be mentioned occasionally, architectures dedicated to them are not of central concern. The goal is not to make an extensive list of architectures, rather focusing on the general principles that got widespread, and the works that have proposed or popularized them.

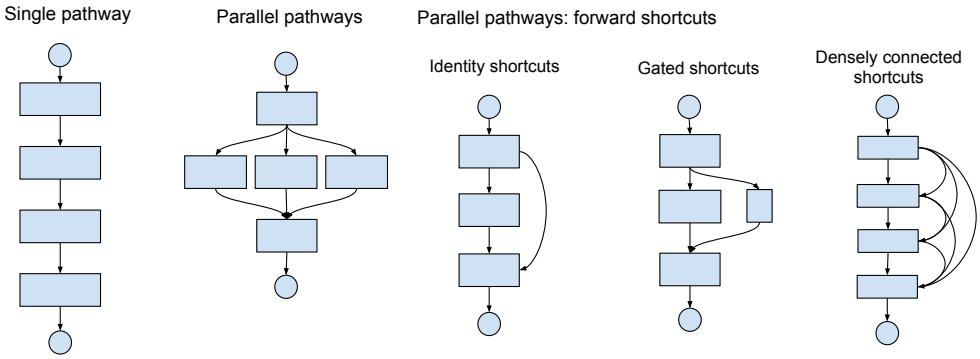



**Figure 3.1** Proposed taxonomy of feedforward CNN architectures. Each family is represented by an archetypal small module of 3-4 layers. Top and bottom circles stand for module's input and output, respectively.

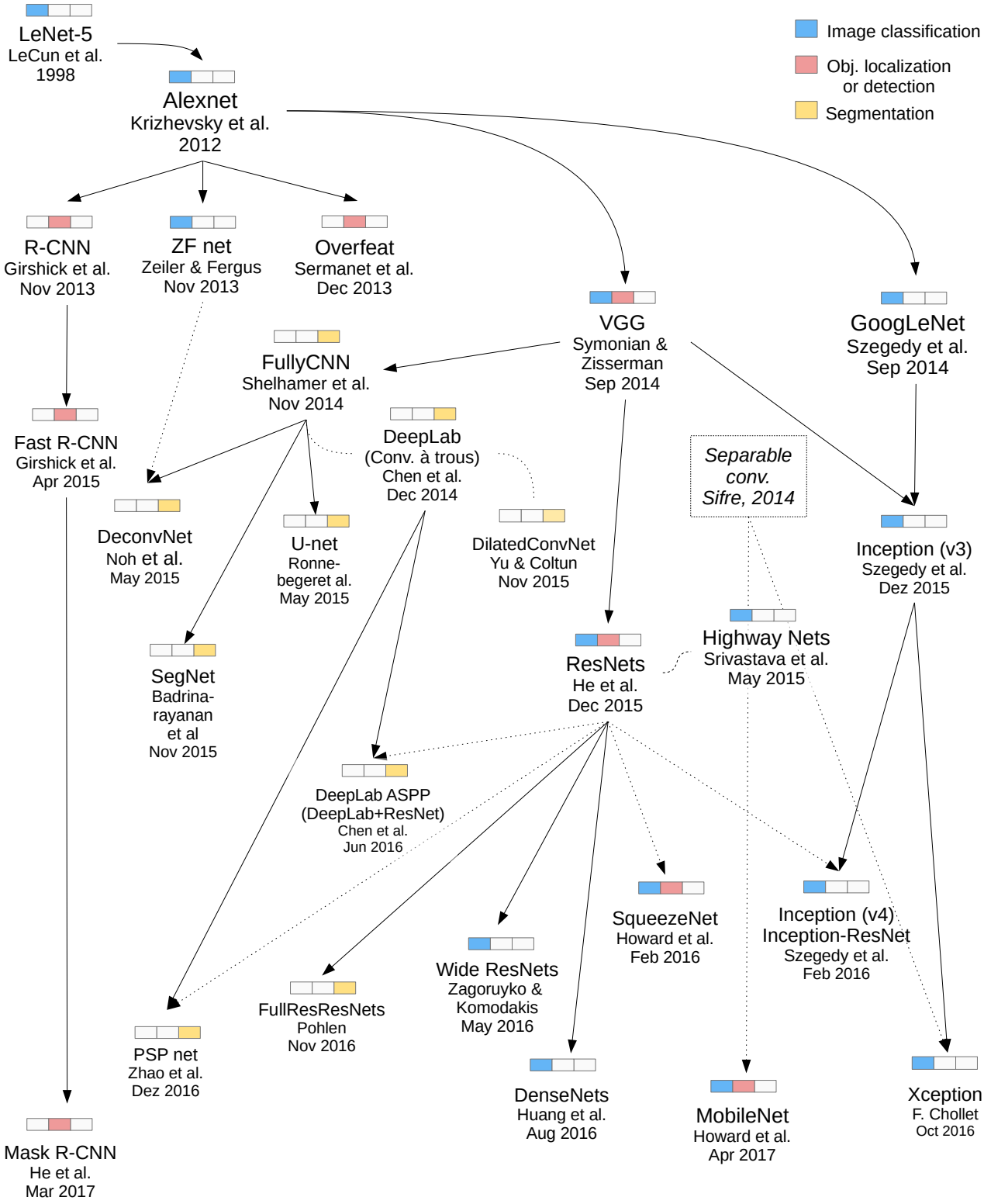



**Figure 3.2** Diagram representing an "influence genealogy" of some important CNN models for computer vision. Dates correspond to arXiv first release or date of publication, whichever is the earliest.

## 3.2 Single pathway models

The first major CNN architectures to achieve success in ImageNet classification were a single-pathway stack of convolutions, pooling and fully connected layers. Even though more recent architectures have better accuracy and/or are more economic in terms of parameter count, these first models have the pioneering merit, having set *de facto* standards for CNN design and training that remain relevant today. Besides, most of these models have been used as a starting point when attempting to tackle new practical problems with CNNs, and are still of interest for those trying to experimentally study and understand the computational "behavior" of a basic but functional CNN.

### 3.2.1 First modern architectures

#### 3.2.1.1 Alexnet

Alexnet, the first DCNN model to win the ImageNet challenge, in 2012, achieved Top-5 error of 18.2% and top-1 error of 40.7% (while the best previous approaches achieved next to 26% and 46%, respectively) (Krizhevsky et al, 2012). Even lower scores were achieved through network ensembles. Developed mainly by Alex Krizhevsky, a student in the group of G. Hinton, this work brought up many technical innovations that continued to be used by future architectures, including the use of ReLU activations for faster learning, and both data augmentation and dropout regularization as essential to avoid overfitting. They also observed slight reductions in error rates by using a local normalization and overlapping pooling windows.

Foreseeing future practices, this work also tried pre-training the networks on previous versions of the ImageNet dataset to achieve better results. Nowadays using ImageNet pre-trained CNNs is a standard starting point to tackle most computer vision (CV) problems. Training used stochastic gradient descent (SGD) with momentum and weight decay, the latter acting not only as a regularizer but also reducing the training error (thus improving optimization). A learning rate reduction policy was also used (based on plateaus of the validation error). The model with 60 million parameters needed to be split up in two consumer GPUs for 5 to 6 days, training for 90 epochs. Most of these training strategies remained on the models that followed. Overall, this work demonstrated the power of deep networks, settling that a highly over-parameterized deep CNN model, given enough data and clever engineering, can generalize, and that even better than traditional CV approaches.

#### 3.2.1.2 Zeiler and Fergus

Following Alexnet, many different research groups took the CNN path when proposing solutions to the ImageNet challenge. The winner of the classification challenge, Clarifai, was designed based on work by Zeiler and Fergus (2014), who proposed a novel method to allow visualization of patterns that maximize network activations. They coupled their CNN classifier to an auxiliary feature decoder architecture that tries to reverse the transformations induced by the main network, by applying "approximate inverse" transforms in the opposite order. In this way, the decoder net is a mirrored version of the main one, in which:

1. Max pooling is replaced by "unpooling", which up-samples the feature map (with zero padding). The location of the activation is chosen to be the same that was chosen at the corresponding pooling stage of the main network;
2. Convolutions are replaced by "deconvolutions" which are actually just convolutions with transposed filters;
3. ReLU is not replaced, to keep feature maps non-negative.

The entire model is trained on ImageNet classification, obtaining at the same time a classifier net and a feature decoder net. The visualized patterns are specific to each input image and represent parts of the image relevant to the activation of each feature map.

Provided this diagnostic tool, several experiments were performed, including invariance to translation, rotation and scale, sensitivity to partial occlusion of objects, and reusing features for other datasets. Overall, they observed that features were not random but presented many desirable aspects, which increase with network depth, such as compositionality, robustness to small transformations (except rotations) and class predictability. Early layers converged faster and detected basic patterns, while deeper layers took longer to converge and could detect more abstract patterns such as wheels, eyes or faces. Occlusion studies allowed to verify for some images that semantically meaningful image parts were indeed implicated in classification (although no misclassification example was discussed). Ablation studies and testing different model sizes led them to argue for the importance of the deep hierarchy as opposed to any single layer and indicated points in the network that could benefit from more parameters without incurring in overfitting (namely middle convolutional layers instead of top fully connected layers). Finally, their network design has profited from the empirical knowledge gathered from this exploratory study.

The proposed architecture in itself is not extremely innovative (5 convolution layers + 2 fully connected), but their design methodology is remarkable for attempting to understand more about the networks functioning before proposing changes. For its new method of feature visualization, the work has also settled an important milestone for interpretability in DCNNs, even though this aspect is not strongly explored in the paper: visualizations were not explicitly used to explain predictions and no incorrect predictions were analyzed. Furthermore, their visualization method has inspired follow-up works in semantic segmentation,

where convolution-deconvolution architectures such as U-Net (Ronneberger et al, 2015) or SegNet(Badrinarayanan et al, 2017) became common practice.

#### 3.2.1.3 OverFeat

The winner of the Imagenet localization challenge in 2013, OverFeat, besides having advanced a bit lower in the error rates (around 39% top-1 and 17% top-5), proposed an integrated approach to solve object classification, localization and detection all with the same network (Sermanet et al, 2013). Overall there is not much difference in the architecture: both AlexNet and OverFeat are constituted of 5 convolutional layers and 2 regular fully connected layers on the top (plus the output classification layer), with max-pooling after the 1st, 2nd and 4th layers. Local normalization and overlapping pooling were dropped (augmenting the number of parameters to 144 million), and stride on first convolution layers reduced.

The main innovation is in the inference procedure. They process the input image at multiple scales to obtain multiple class distribution vectors combined by averaging. For localization, the output classification layer gives place to a set of regressors (one for each class), trained over features from the last convolutional layer of the network to predict bounding box coordinates for annotated objects. Then both regressor and classifier are used across all locations and scales. Class predictions are grouped within each predicted bounding box to provide both a class and the confidence of the classification, and bounding boxes are iteratively aggregated to provide a final prediction. A similar scheme is adopted for object detection, except that an extra "background" class is used to label regions where no object has been detected, and bounding box regressors are not used.

Although more efficient object detectors followed (for instance Region-proposing CNNs (Girshick et al, 2014)), OverFeat architecture has remained relevant particularly because it has been the first major model to be released together with its weights, allowing to use it as a feature extractor. Using pre-trained models of-the-shelf has become a standard procedure ever since, with CNN features often providing a strong baseline result for image classification (Razavian et al, 2014). Besides its use as a fixed feature extractor, OverFeat has also served as base architecture on multiple works, such as image classification and detection on MS-COCO dataset, semantic segmentation of outdoor scenes (Pinheiro and Collobert, 2015), and multiple object detection inside a storage shelf or bin (Schwarz et al, 2017)

#### 3.2.1.4 VGG

Oxford's Visual Geometry Group (VGG) innovated in 2014 proposing a deeper family of architectures (9 to 16 convolutional layers + 2 fully connected + output layer), but using smaller filters — 3x3 instead of 7x7 and 5x5 — to keep the total number of parameters manageable (between 133 and 144 million depending on the net's depth) (Simonyan and

Zisserman, 2015). The intuition behind stacking 3x3 filters is that a larger receptive field can be achieved. Indeed, as far as receptive field size is concerned, two (three) 3x3 convolutions are equivalent to a 5x5 (7x7) convolution, but with less parameters. Local response normalization was also dropped, as it demanded to much computations for no gain in performance. Their best single nets achieved 24% top-1 and 7% top-5 error rates on ImageNet 2014 classification task.

This is one of the first works that propose successful deeper networks together with a simple architectural principle, exploring multiple versions with different depths : VGG-11,13,16 and 19 layers. This architecture has been reused and adapted by multiple works (eg.Shelhamer et al (2017) or Badrinarayanan et al (2017)) to different applications including semantic segmentation, and it became a popular option of pre-trained architecture used for transfer learning.

### 3.2.2 Parameter-saving architectures

Besides these pioneer architectures, some others are worth noticing due to their particular interest in reaching good accuracy under less computational resources. Smaller models can be trained on less powerful hardware and need less memory. When deployed, a smaller trained network demands less bandwidth to be distributed/downloaded and is more suitable to be run in embedded devices with limited resources. A first example on this direction is SqueezeNet, an architecture reaching AlexNet's performance under 50 times less parameters (Iandola et al, 2016). On one hand, their main strategies are to reduce the size of most filters (3x3 to 1x1) while decreasing the number of input channels for those layers which remain 3x3 convolutions. On the other hand, down-sampling is delayed to upper layers, so that feature maps do not get too small to fast, severely hindering accuracy. Another small model family is MobileNet (Howard et al, 2017). While SqueezeNet focuses simply on model size, MobileNet is also concerned with inference speed. Their main strategy to reduce computations is the use of channel-wise (or depth-wise) separable convolutions, while two other relevant aspects are left as hyperparameters: a width multiplier — controlling the number of channels in each layer — and a resolution multiplier — to act on input image resolution. Together, all these allow to provide options for different accuracy vs. computational burden trade-offs, suiting different budgets.

## 3.3 Parallel pathway archs

Parallel to the development of single path DCNNs, other groups explored the effect of adding parallel paths to basic architectures, in simple or complex ways. One very simple way of implementing multiple pathways is to jointly train multiple nets and using their average prediction as output, an approach that has been tested by Ciresan et al (2012). Although simple

to explain, this approach produces a very large model highly demanding on computational resources. There are more subtle ways of adding parallelism to the computations performed in CNNs without incurring in such extreme burden, besides providing interesting insights into the effectiveness of deep networks. Different propositions of parallel architectures have in common that they allowed training of deeper networks. Some approaches allow reduction of parameters, others focus on memory usage or number of computations. In what follows, the main tendencies of parallel architectures together with their main representative works are reviewed. When suitable, further discussion is provided on the rationale behind certain design choices and why they were successful in practice and/or widely adopted by the deep learning community.

### 3.3.1 Sequence of parallel blocks: the Inception family

This family of models is characterized by a succession of parallel blocks — inception modules — combined by concatenation of the feature maps (Szegedy et al, 2015). The rationale behind the inception modules is that the scale of features detected on images can vary widely and thus fixing the size of the kernels (an thus the receptive fields) is insufficient and arbitrary. The solution proposed in the inception block is to have a multitude of kernel sizes applied in parallel, creating multiple paths.

Before or after each path, 1x1 convolutional layers reduce the computational burden via dimensionality reduction, allowing the stacking of more blocks without achieving a too large model to compute or too many parameters. It is worth noticing that a convolution by a 1x1 filter is equivalent to a linear combination of input feature maps/channels, without touching the spatial dimensions, effectively decoupling spatial filtering and channel-wise filtering (Chollet, 2016). Therefore they can be seen as a convolutional implementation of a channel-wise fully-connected layer, with each input channel connected to each output channel through a single weight. Finally, fully connected layers give place to a simple global average pooling prior to the output layer, cutting even further down on the parameter count. For instance, the first inception model, known as GoogleNet, has 22 layers and yet 9x fewer parameters than AlexNet (7 million versus 60 million (Szegedy et al, 2016b)), while being more accurate. It has also less computations than its contemporary the VGG model — 1.5 billion flops (Szegedy et al, 2015) versus 19.6 billion FLOPS (He et al, 2016a) — although the latter has the advantage of lower resolution loss and better single net accuracy.

In the interest of further reducing computations and increasing depth, two principles were adopted for Inception V2 and V3: replacing $5\times5$ by two $3\times3$ convolutions (equivalent receptive field as discussed for VGG net) and replacing $n \times n$ convolutions by 1xn and nx1 convolutions (implementing a factorized filter with 2 vectors instead of single filter matrix). Furthermore, instead of simply stacking these two asymmetric convolutions, each filter-vector is applied in parallel, to avoid an exaggerate dimension reduction. For inception V3 in particular, they

added another path with factorized 7x7 convolutions, amounting to a 42-layer model with only 2.5 times more computations than 23-layer Inception V1. The third version was trained on 50 GPU in parallel for 100 epochs, being one of the earliest to benefit from distributed training powered by the Tensorflow framework (Abadi et al, 2016).

#### 3.3.1.1 Role of auxiliary losses

As deeper networks are more prone to suffer from vanishing gradients during optimization, they propose using auxiliary losses trained on outputs obtained from 2 intermediate points of the network. Each loss is a standard classification cross-entropy loss, but computed over predictions obtained from different network paths. The total loss is then a weighted sum of the main loss (from the main output) and auxiliary losses (from the auxiliary paths).

These auxiliary losses are supposed to help making features of earlier layers more discriminative, help gradient backpropagation and act as a regularizer. However, even if this auxiliary loss strategy has been kept throughout the evolution of the basic inception architecture, the authors conjecture that they act more as a regularizer, as experimental evidence — auxiliary branches could be removed without much affecting performances or convergence — question their effective participation in the tuning of lower layers (Szegedy et al, 2016b).

#### 3.3.1.2 Variations and derivations

Further work on this family of architectures intended to simplify the design, making the inception blocks more uniform (Inception V4), also exploring the incorporation of residual connections (Inception ResNet) (Szegedy et al, 2016a). They observed increased training speed with residual connections, and a slightly improved accuracy when compared to similarly expensive Inception models (in number of computations, Inception V3 comparable to Inception ResNet V1 and Inception V4 comparable to Inception ResNet v2). Their results point towards the use of residual connections having a complementary effect to that of the inception blocks.

An interesting variation proposed by Chollet (2016), the Xception model seeks to make more effective use of the 22 million parameters of the Inception V3 model, while achieving better accuracy. The hypothesis is that channel and spatial processing can be completely decoupled, being more efficient to perform depth-wise separable convolutions — spatial convolutions applied independently to each channel. With nearly the same number of parameters, the improvement observed over Inception V3 is attributed to a supposed increase in the efficient use of the parameters. Furthermore, Xception converged faster on both datasets tested. The inclusion of residual connections improves validation accuracy significantly.

### 3.3.2 Architectures featuring forward shortcuts

Forward shortcuts or skip connections are bottom-up links that skip one or more layers. The output from the source layer will then be combined with the target layer, either by feature map concatenation or by some arithmetical operation such as summation. An early usage of skip connections is presented in an architecture for semantic segmentation by Shelhamer et al (2017), who use them to gather information from larger resolution feature maps from earlier layers (before pooling operations). Later, they were used mainly to allow training of very deep networks. The following paragraphs discuss these architectures, their common and innovative aspects, trying to summarize current knowledge on the reason for their success. Newer models generalizing the shortcuts idea in different manners are also reviewed.

#### 3.3.2.1 Residual connections

On deeper nets, training accuracy could be at least as good as that of a shallower counterpart. The intuition behind this claim is simple: by construction, a shallower model could be implemented by the deeper one by setting some of its layers to the identity function. So there has to be at least an equivalent solution with a deeper network, but optimization is not being able to find it. The shortcuts in ResNet (short for residual networks) come then as a mean to facilitate the implementation of these identity functions (He et al, 2016a). The original motivation stated in the paper was to deal with the degradation that intervenes when stacking too many layers. Processing of a given layer can be completely skipped by the identity shortcut path. Actually, since the identity shortcut output is summed to the output of the skipped layers, a residual mapping is achieved. That is, for a given input $\mathbf{x}$, the function implemented by the residual block — the output of layers and shortcut — will be $H(\mathbf{x})$, where $F(\mathbf{x}) := H(\mathbf{x}) - \mathbf{x}$ is a residual function implemented by the stacked layers bypassed by the shortcut.

A later reformulation by He et al (2016b) pre-pends ReLU and BN operations before the convolution layers inside the residual function, so that the new *pre-activated* residual network can be given by the recurrence relation

$$\mathbf{x}_{l+1} = \mathbf{x}_l = F(\mathbf{x}_l, \mathbf{W}_l), \tag{3.1}$$

which recursively yields that

$$\mathbf{x}_L = \mathbf{x}_l + \sum_{i}^{L-1} F(\mathbf{x}_i, \mathbf{W}_i), \tag{3.2}$$

for any $L$-th layer and $L > l$. By analyzing the error gradient expression for this network (see discussion by He et al (2016b)), one can observe that it decomposes into two additive terms: one linked to the weights of previous layers and one independent of them, directly linked to

a given $\mathbf{x}_L$, effectively acting as a reverse gradient shortcut. This analysis supports the claim that shortcuts help gradient flow during optimization, since they ensure the gradient of a layer does not vanish even if weights are really small.

#### a. Variations and derivations

Residual Networks have achieved outstanding record breaking results on image recognition, object localization and detection on benchmarks such as ImageNet (winner of the 2015 classification challenge), MS COCO, CIFAR-10 and 100 (He et al, 2016a,b). Their success led residual connections to become popular architectural feature to experiment on any model family. For instance, SqueezeNet's small but effective architecture can achieve even better accuracy if residual connections are added (Iandola et al, 2016). Residual connections also help Inception-type architectures, as demonstrated by Szegedy et al (2016a), and have also been experimented with on generative adversarial networks (e.g. Mehrotra and Dukkipati (2017)).

On another note, many direct variations of the initial model were proposed in follow-up works. A simple variation proposed by Shen et al (2016) adds a multiplicative weight $\lambda_l$ to the residual $F(\mathbf{x}_l, \mathbf{W}_l)$ that is initialized to zero, thus inducing all residuals to effectively start as zero. This variant reduces overfitting on very deep models on CIFAR10, with maximum test accuracy achieved by a 1192 layer model (versus a 112 layer model, for an equivalent plain ResNet).

In ResNext (Xie et al, 2017a), another deriving model, channels are sliced into groups before applying separated convolutions for each of them. The trick reduces parameters and computations allowing to compute more filters (more channels on each layer) at the same parameter and computational budget. (if you split channels in b groups, you have w*h*in/b *out/b *b parameters instead of w*h*in*out). With the increased number of filters, it works a bit better than plain ResNet.

### 3.3.2.2 Gated shortcuts

Through a slightly different reasoning, Srivastava et al (2015) concurrently proposed a similar shortcut architecture: highway networks. Also motivated by the difficulties of training very deep networks, they propose a principle inspired from an unrolled LSTM, adding gated connections towards the following layers. For a given input $\mathbf{x}$, the output of a (coupled) highway block is

$$\mathbf{y} = H(\mathbf{x}, \mathbf{W}_H) \cdot T(\mathbf{x}, \mathbf{W}_T) + \mathbf{x} \cdot (1 - T(\mathbf{x}, \mathbf{W}_T)),$$

where the transformation gate $T(\mathbf{x}, \mathbf{W}_T)$ learns to regulate the information flow between layer (first term) and shortcut(second term).

Both shortcut approaches achieved successful training by SGD of extremely deep networks — 1001 layers for ResNet (on ImageNet) and 900 for Highway nets (on MNIST and CIFAR-10

and 100). The hypothesis that very deep networks might work better by learning identity functions on some layers is reinforced through experimental observations in Srivastava et al (2015), where activation's throughout highway blocks often remain the same for a significant number of blocks.

Highway networks have worked better than plain ResNets on some natural language processing tasks. Greff et al (2016) propose a comparison between parameter matched versions of ResNet and Highway nets on a language modeling task. They observe that the learned gating is crucial for improving performance on the task, and hypothesize that in tasks where partial feature adaptive replacement or transformation is necessary, highway networks have an advantage over plain residual shortcuts.

#### 3.3.2.3 Multiple forward shortcuts

Many works taking inspiration from ResNets generalized the residual shortcut idea to some form of multiple forward shortcut architecture.

##### a. Multiple paths by self-similarity

Larsson et al (2016) proposed a multi-path architecture based on the repetition of a simple expansion rule. This generates FractalNets, an architecture with large nominal depth but still having shorter paths to ease gradient flow. This architectural principle also allows training of very deep networks (up to 160), and works better under drop-path regularization – a version of dropout proposed by the authors to avoid co-adaptation of sub-paths so that every path learns to make reliable predictions. Their experiments observed that although the fractal structure helps training, in inference time the deepest sub-net can be used alone, with little effect in accuracy. Overall, this model could be an interesting alternative to learn a complex model in training time that automatically provides a smaller and faster version at inference time, which could be useful for computationally restrained applications.

##### b. Multi-level ResNets

One line of thought consists in proposing some generalization of the forward shortcuts idea. Initially some works added other level of shortcuts on top of the existing ones. Building on top of the hypothesis that optimizing residual mappings is easier, Zhang et al (2016b) hypothesize that a residual mapping of residual mappings is even easier. They design then a multilevel model named ResNet of ResNets (RoR), constituted by adding another level of shortcuts every 2 residual blocks. Targ et al (2016) propose ResNet in ResNet (RiR), an architecture mixing residual and conventional CNNs by having two pathways — one *residual stream* with identity shortcuts and another *transient stream* without — that partially communicate inside each block. This generalized residual block has sufficient expressive power to learn mixes between plain CNNs and ResNets. Their RiR architecture consists then in replacing each convolutional

layers within a ResNet by one of such generalized residual blocks. Both approaches improve ResNet test accuracy on CIFAR10 and 100 datasets, with an arguable upside for RoR since it is simpler to implement, does not add to training time and achieves better results.

#### c. Generalized forward shortcuts

On a second moment, forward shortcuts were generalized to a more extreme extent, with input connections coming from all preceding layers. A first proposal in this direction was DenseNet (Huang et al, 2017), a streamlined architecture of stacked dense blocks. These blocks ar composed of multiple composite layers (BN - ReLU - 3x3 conv), featuring forward shortcuts to all following composite layers inside the block. In this way, an L-layer block has $\frac{L(L+1)}{2}$ shortcut connections. All incoming feature maps are combined by concatenation (and not addition like in ResNets). This makes them more comparable to Inception networks, that are similarly structured as a sequence of internally parallel blocks. 1x1 convolutions as bottleneck layers between dense blocks to limit computational burden

One motivation behind this dense blocks is to exploit potential feature reuse from previous layers. Indeed, analysis of the weights in a trained DenseNet indicate that deeper layers within a dense block do reuse features from earlier layers. Another benefit is to be more parameter efficient than plain ResNets. Experiments by Huang et al (2017) obtained models with less parameters and less computations than ResNet for same error rate (observed on ImageNet and CIFAR). For instance, a DenseNet-BC with 0.8M parameters achieves accuracy comparable to pre-activated ResNet-1001 with 10.2M parameters.

DeludgeNet is a second proposal on the same line, generalizing forward shortcuts within blocks that are stacked together (Kuen et al, 2016). Similarly to DenseNets, a block is formed by multiple composite layers, each layer receiving input from all previous layers within a block DeludgeNets use a pre-activated bottleneck composite layer (formed by 3xBN-ReLU-Conv). Within this layer, 1x1 convolutions compress channels at the beginning and expand at the end, reducing the computational burden of applying 3x3 convolutions directly on many concatenated feature maps. Shortcut inputs go through 1x1 cross-layer depth-wise convolutions prior to being concatenated and fed to the next composite layer. This operation consists of a channel-separable convolution, that summarizes each channel dimension $c_i$ from all previous layers $l-1, l-2, \ldots$ onto a single feature map, keeping the number of channels constant for the next layer $l$ (instead of having a growing number of channels as in a DenseNet block). This should allow to save-up in the number of parameters, when compared to DenseNets.

The architecture yielded better results on CIFAR 10/100 when compared to ResNets. Comparison to DenseNets is less evident, as models of similar parameter count have close results, although DeludgeNets achieve so with less computations (GFLOPS). A less pronounced improvement is also observed on ImageNet experiments. Comparing two models with similar GFLOPs and performance on ImageNet, DelugeNet has more parameters than its DenseNet

counterpart (no result of a DenseNet of similar parameter count is reported).

## 3.4 Discussion

### 3.4.1 Relation between forward shortcuts and recurrent networks

The success of shortcut architectures, in particular ResNets, led to further works exploring and experimenting with their behavior under shuffling and removal of inner block layers and even entire blocks (Veit et al, 2016). Surprisingly, a very limited effect on accuracy is observed when these networks are submitted to these "lesions". A more pronounced drop is observed for early layers or layers right after a change in dimensionality. At a first glance, these results challenge the hierarchical representational view of deep networks, with each layer building up on and detecting more abstract features than the previous ones. Under this hypothesis, shuffling the representational levels makes no sense and should severely degrade performance (which has indeed been observed for the one-stream VGG-15 by Veit et al (2016), in which layer removal could rise classification error up to 90%). These observations led Veit et al (2016) to postulate that ResNets work as an ensemble by averaging (exponentially) many sub-networks, each only using a subset of all layers, thus justifying the robustness to layer removal and shuffling.

Reconciling shortcut architectures and hierarchical representations, Greff et al (2016) propose to view them performing an *unrolled iterative estimation*. As discussed in their work, these architectures can be seen as a simplified recurrent network (for ResNets) or an LSTM (for Highway nets) "unrolled" over time. In each residual/highway block, the first layer learns a first good initial estimate that is only refined by subsequent layers. Therefore, all layers in a block are working on the same level of abstraction, the same features. The transitions from a set of blocks — a stage — of higher to another stage of lower dimensionality (through a change in the number of feature maps) would also mean transitions of abstraction levels. Under this view, it makes sense that layers within a block (except for the first) can be removed and interchanged (to some degree) with little impact on accuracy, since (1) they are merely fine-tuning the filters from previous layers and (2) they are working on the same reference input towards a common output.

Foreseeing the utility of forward skip connections, Liang and Hu (2015) note that their recurrent convolutional architecture when time-unfolded corresponds to a CNN with multiple paths from input to output layer, and hypothesize it may facilitate learning. Analogously, Liao and Poggio (2016) provide a similar perspective, under a bio-inspired framework. They observe that a residual block with shared weights can be reformulated as a recurrent system. They propose then a stacked recurrent model, with recurrent hidden layers modeling each stage of the ventral cortical hierarchy, including identity shortcuts between cortical regions. The

weight sharing condition seems not to hinder performance significantly, as their experiments obtained similar validation accuracy (on CIFAR-10) for both shared and un-shared weight versions. This model will be discussed in more detail with other recurrent convolutional models in chapter 4.

### 3.4.2 Attempts at understanding the success of residual shortcuts

While the intuition behind suing shortcuts seems reasonable on a first look, it remains important to try and understand from a theoretical point of view why and how residual shortcuts help the training of very deep neural networks. Several works have taken steps towards this goal, trying to formalize how the addition of shortcuts influences the optimization landscape and how it affects SGD-based training (e.g. Orhan and Pitkow, 2017; Philipp et al, 2018) One interesting discussion we will develop hereafter concerns attempts at understanding which would be the optimal shortcut length in order to obtain the optimization benefits of these architectures.

Experimental observations by He et al (2016a) indicate that using shortcuts every 2 layers (2-shortcut) works better in practice than 1-shortcut or 3-shortcut networks. While this empirical intuition is useful in practice, it remains important to understand from a theoretical point of view whether (and why) 2-shortcut is indeed an optimal choice. Hardt and Ma (2016) have demonstrated that loss landscape has no spurious local minima for arbitrarily deep linear 1-shortcut residual networks (given that all weight matrices have small Frobenius norms). Apparently contradicting previous empirical results — 1-shortcuts did not ease training in practice — actually this result simply exposes the fundamentally different behavior of the original non-linear shortcut-1 ResNet, indicating that studying it via a linear proxy may not be sufficient to understand this phenomena.

Building up on this work, Li et al (2017) have demonstrated theoretical support fo this empirical observation, this time including ReLU non-linearities. They analyzed the Hessian matrix of a simplified ResNet (only standard fully-connected layers) of arbitrary depth, and shortcut lengths. Both theoretical analysis by Hardt and Ma and experimental results by Li et al indicate that as ResNets go deeper, weights tend to converge near zero during training, motivating to analyze the Hessian matrix around this point (here noted $H_0$)[1].

They have reached the following conclusions:

- For shortcut-1 networks, $H_0$ has a block Toeplitz structure, which is known to have exploding condition numbers[2] for increasing network depth.

[1] In practice of course, since zero is a saddle point for this optimization problem, weights were initialized to small random values around zero (Xavier initialization multiplied by 0.01)

[2] The condition number of a matrix $A$ is given by: $cond(A) = \frac{\sigma_{max}}{\sigma_{min}}$ where $\sigma_{max}, \sigma_{min}$ are the maximum and minimum singular values of A. In case $A$ is normal, i.e. $AA^T = A^TA$, this can be simplified to $cond(A) = \frac{|\lambda(A)|_{max}}{|\lambda(A)|_{min}}$ where $|\lambda(A)|_{max}$ , $|\lambda(A)|_{min}$ are the maximal and minimal eigenvalues of $A$ in absolute value. (according to Li

- For shortcut-$n$ networks, $n \geq 2$, zero is a saddle point of order $n-1$, with $H_0$ having a zero diagonal. In particular, for $n \geq 3$, $H_0$ is a zero matrix.
- For the special shortcut-2 case, $H_0$ presents a particular eigenvalue structure[3] such that condition numbers are independent of network depth. Thus, optimization (around zero) should work similarly for either shallower or deeper shortcut-2 networks.

Large condition numbers can have a large impact on convergence of first order methods, justifying why shortcut-1 networks are no better than no-shortcut ones. Moreover, for shortcut-2 networks, zero is a *strict* saddle point, which have been proven to be easy to escape from. Together with depth independent condition numbers, this helps explaining why training is easier using shortcuts with length 2. Note that these theoretical results are not valid for weight initializations other than (near-)zero, (eg. Xavier or MRS) — with experimental evidence demonstrating different convergence behaviors for other initializations.

### 3.4.3 Feature reuse by concatenation vs memory efficiency

Architectures featuring shortcuts may allow reusing low level features at upper levels of the network. This influence of previous feature maps can be done directly, by concatenating them with those at the destination layer, as done in DenseNets. Another form is to combine source and destination through some linear combination, as is done with residual connections (combination by summation).

In the case of DenseNets and DelugeNets, a drawback of aggregating these densely connected shortcuts by concatenation is that outputs from each layer need to be retained for computations on the next layer. The same does not happen when aggregation by summation is used (which is the case of ResNets). The issue is more pronounced on DenseNets, since even under a constant number of feature maps $k$, dense blocks have a linear growth rate of the number of output feature maps: for an input with $k_o$ channels and a dense block with $L$ layers producing $k$ feature maps each, the output will have $k_0 + k(L-1)$. Therefore, there is an increase in memory demanded during both training and inference. As reported by Kuen et al (2016), both DeludgeNets and DenseNets require more GPU memory than ResNets, although the first is more memory-efficient than the second. Altogether, this increase in memory requirement (alongside small margins of accuracy improvement) may explain why ResNets remained most popular over these densely-connected alternatives.

### 3.4.4 Width vs depth on ResNets

As stated earlier, many layers in residual blocks have little participation in the final accuracy and can be changed or removed. Thus it can be hypothesized that, under sufficient depth,

et al, 2017, version 1, section 2).

[3] $k$ copies of $\pm\sqrt{\sigma(A^T A)}$ where $k$ equals the number of shortcut connections.

increasing the number of filters in each layer of a ResNet can be a more effective way to use parameters than simply augmenting layer depth. Besides, a wider and not-so-deep architecture is more computationally efficient for the same number of parameters. In fact, fitting more channels is more adapted to profit from GPU parallelism than adding sequential layers. With this in mind, Zagoruyko and Komodakis (2016) performed a series of experiments with wide residual networks, and demonstrate that thoughtful widening of ResNets is more effective to improve accuracy than increasing their depth. For instance, a wider 16-layer model that is as accurate on ImageNet as ResNet-1001, with roughly the same number of parameters and being much faster to train. Additionally, they could successfully train networks with more parameters than ResNet-1001 without incurring in overfitting or performance degradation. Finally, these results bring more complexity to the discussion of deep vs shallow networks, highlighting the role of width (number of channels) as potentially determinant when deciding whether to "go deeper" in the number of layers.

## 3.5 Conclusion

Despite focusing only on feedforward architectures, this review confirms the diversity of architectural devices that have been at play on modern CNN architectures. The field has seen an explosion of different techniques and methodologies, some of which having had more impact and longevity than others. In terms of architectural features, residual connections have been an important mark. After the fast evolution of architectures between 2012 and 2015, ResNet-based architectures have remained state-of-the-art. Many implicit regularization methods have been introduced together with particular architectures, like dropout with AlexNet, or batch norm with ResNets. Beyond simply training better networks, some analysis tools have also been developed along the way, such as the deconvolution-based visualization method proposed by Zeiler (2014), at the base of many later feature visualization works.

The success of residual networks has also called attention to their relationship to recurrent architectures, and how they implement a form of time-unrolled recurrence. This observation opens an interesting line of inquiry about how and when recurrent-like mechanisms could help training more accurate or faster-convergence models. Furthermore, it resonates with recent neuroscientific knowledge on the importance of recurrent processing in the visual pipeline. From a qualitative point of view, it can be promising to experiment with similar mechanisms associated to CNNs in order to asses whether analogous functionality — like dealing with visual clutter, occluded borders or context-dependent recognition — could be achieved. In this spirit, the next chapter reviews forms of recurrent and feedback connections implemented in current CNN architectures aimed at image classification.

Chapter 4

# Recurrent and feedback CNN architectures for object recognition

## 4.1 Introduction

Recurrent networks naturally come to mind when dealing with sequential data, such as video, audio and text, since recurrent processing is intrinsically linked with temporal or sequential processing. In computer vision, they are often on the table when tasks have an intrinsic time component, such as object tracking on videos (e.g. Ning et al, 2017), or when doing multi-modal tasks linking vision and text, like visual question answering or image/video caption generation (e.g. Donahue et al, 2017; Ballas et al, 2015).

However, recurrent processing is not only useful for sequential inputs: it can also iteratively help to refine processing over static inputs. In biological vision, there is evidence — both anatomical and physiological — that feedback connections and local recurrences play an important role in object recognition (Spoerer et al, 2017; Nayebi et al, 2018), and are likely involved in recognition under challenging conditions possibly due to obtaining more robust object representations (Wyatte et al, 2014). Even static input stimuli are capable of inducing dynamic neuronal response trajectories (Issa et al, 2018). Nevertheless, these functional aspects of recurrent processing remain scarcely explored in the context of DCNNs. Such works are the object of this review, which aims at covering the functional benefits of introducing recurrences and feedbacks to DCNNs (section 4.2), while also summarizing the main architectural variants exploited to this aim (section 4.3).

Earlier forms of CNNs with recurrent processing consisted mainly in weight sharing between subsequent layers, which is equivalent to a time-unrolled representation of a recurrent network (e.g. Pinheiro and Collobert, 2014). This type of model has been used to study the influence of different architectural aspects — namely number of layers, number of parameters and number of feature maps — on model's accuracy (Eigen et al, 2013). More recently, the idea of recurrence through parameter sharing could be encountered in the work of Savarese and Maire (2019), although in a more indirect and implicit fashion. They have constrained their convolutional layers to learn linear combinations of templates from a bank of filters. Therefore weights get indirectly shared through these templates, even though learned weights

may get to modulate their activation.

Even without any particular functionality as target, the addition of recurrent processing may improve the overall accuracy of the system. A few works have explored whether enabling a CNN with recurrent processing outperforms its equivalent feedforward counterpart (Wen et al, 2018; Spoerer et al, 2017; Liang and Hu, 2015). Even though they may not outperform larger recent models such as ResNets and their derivatives, in some cases they have achieved similar performances, with the advantage of doing so with less parameters (e.g. Nayebi et al, 2018; Liao and Poggio, 2016). More generally, recurrence mechanisms have proven to be useful to improve classification performance of traditional feedforward architectures, such as Inception-v3 (Vallet and Sakamoto, 2016) or Inception-v4 (Alom et al, 2017).

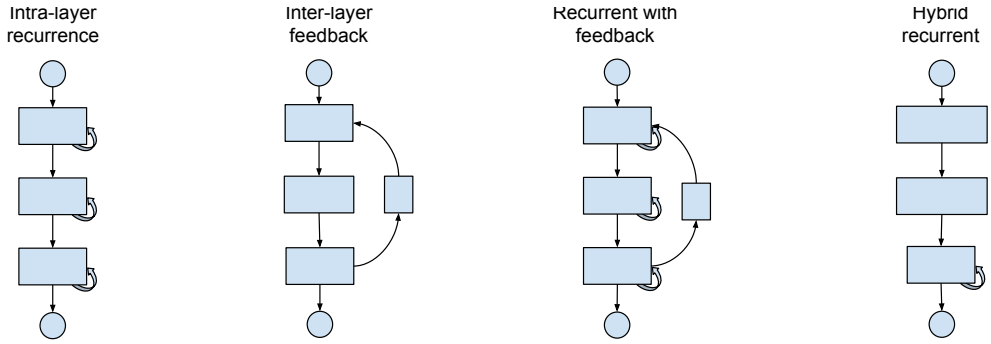



**Figure 4.1** Proposed taxonomy of recurrent and feedback CNN architectures. Each family is represented by an archetypal small module of 3-4 layers. Top and bottom circles stand for module's input and output, respectively.

## 4.2 Functionalities brought by recurrent processing

### 4.2.1 Spatial context integration

While recurrent processing quickly brings to mind its relation to time dynamics, local spatial recurrence is equally relevant in refining our visual perception of a scene. In the model by Liang and Hu (2015), each unit receives feedback from a local patch around it. Architecture is such that the size of this influencing neighborhood is increased at each time step, iteratively integrating more and more global contextual information.

The model proposed by Visin et al (2015), ReNet, intends to implement context integration via lateral connections, using bi-directional recurrent layers that integrate information in two steps — over columns then over rows. Visin et al argue these lateral connections should allow to remove redundant features at different locations of the image, helping to extract a more compact representation. Their model was applied to classification tasks on MNIST, CIFAR-10 and SVHN (see chapter C for a brief description of these datasets), with competitive quantitative results reported (among its contemporaries). Unfortunately a qualitative analysis

of the types of filters learned by this network and its function was left for future works.

Xie et al (2017b) propose a spatial integration scheme similar to ReNet, with an additional proposal of integrating their L-RNN module to pre-trained CNNs, interleaving convolutional and recurrent modules to model both local features and long range spatial dependencies, respectively. This strategy was particularly valuable for semantic segmentation tasks (on MS-COCO and PASCAL VOC 2012), which have limited amount of labeled data available (in relation to the large number of predictions that need to be done, since pixel-wise labeling is the target). For classification, it can also be a strategy to reduce model size: on CIFAR-10, accuracy results were comparable to those of a ResNet-164, while using only 15 layers and less parameters.

The spatially hierarchical recurrent model by Zuo et al (2016), where recurrence is done over a multi-scale pyramid of equivalent feature maps, also uses lateral connections for spatial context integration. Besides lateral connections between patches in a same spatial scale, patches on a lower hierarchical level influence corresponding patches in a higher level.

In fact, the use of recurrent models for spatial context integration has been further explored for the task of semantic segmentation, where the evident influence of a larger global context over local predictions clearly pushes towards modeling this relation more explicitly. It is the case of Visin et al (2015), who apply a variant of their ReNet model to this task, and also the case of Layer-RNN. For a broader review on this particular task, the reader may refer to the review by Garcia-Garcia et al (2017) (particularly section 4.2.5).

A similar consideration can be done for the object detection and/or localization task. For instance, the Inside-Outside Net model by Bell et al (2016) obtains contextual features with 4-directional spatial recurrent layers applied to convolutional feature maps of a VGG network (layer *conv5*). The spatial recurrence here is similar to the one presented by Visin et al (2015) and Xie et al (2017b), with lateral connections linking image patches horizontally and vertically. For each candidate region of interest, these contextual features are combined with intermediate features (from layers *conv3*, *4* and *5*) to feed a classification pipeline. Their architecture has reached state-of the art detection performance on PASCAL VOC 2012 and MS COCO datasets.

Contextual information may also come in the form of a segmentation mask, as in the model by Shrivastava and Gupta (2016), which augments a standard object detection architecture — Faster R-CNN (Ren et al, 2015) with feedback inputs to lower layers from an accessory semantic segmentation task. Here segmentation information is expected to guide the proposal of regions of interest (ROI) on which classification with a CNN is attempted in order to detect objects. It can be seen as a contextual priming process derived from global spatial information, including border assignment as it is an important product of semantic segmentation. While the combination of both tasks could indeed improve detection results (in terms of mean intersection-over-union or mean average precision), the applicability of their method is limited to datasets including segmentation labels, which are more expensive to obtain and

rarely available for custom real-life datasets.

**What-where interaction** Both recognition and localization are performed by the same model in the work by Cao et al (2015). It consists of a CNN enabled with feedback connections from predicted outputs, which help selecting relevant activations and suppress irrelevant responses. Using only image-level labels, the feedforward pass performs a weakly supervised localization proposing regions of interest (ROIs), which are further processed for final classification. When using a pre-trained VGG as base network, they achieved competitive localization performance when compared to another pixel-wise supervised VGG based method. Finally the spatial focus introduced by feedback signals could reduce the dependency on a heavily-labeled scheme.

Beyond classification, spatial context integration through recurrence has also been targeted in other contexts such as image generation (van den Oord et al, 2016) and salience detection (Liu and Han, 2018).

### 4.2.2 Iterative processing

When facing complex and ambiguous scenes, a longer time is needed by our brain to completely process and parse the scene's elements, iteratively refining its internal representation. This processing includes feedback propagation of top-down information and lateral interactions between same layers neurons. This functionality is referred hereafter as iterative processing. Analogously, in computer vision, iterative processing of images allows to refine judgment on particular locations or features. One should expect that incorporating such functionality to CNNs is likely to lead to more confident predictions.

An advantage of iterative processing is the possibility to have a first coarse answer in the fraction of the total inference time, further refining it in case of need (if the scene is complex or ambiguous, for instance). This notion of best output *so-far* is important for practical scenarios, for instance in robotic navigation, where a fast evaluation of the scene must be quickly available at all times (Zamir et al, 2017).

Another possibility is to allow for responses to different features of the stimuli at different levels of the recurrence. This "behavior" has been observed with the loopy CNN model by Caswell et al (2016), even though it was not explicitly coded for in the architecture. Cao et al (2015) achieve a similar functionality with a more explicit model that uses feedback signals to iteratively focus salient parts of the image, selectively ignoring irrelevant information and refining the final prediction. Using a pre-trained GoogleNet (ImageNet 2012) as basis, attempting re-classification with the feedback enabled version — that focuses attention on regions of interest — has improved top-5 and top-1 errors bey 1.29% and 1.79% respectively (Cao et al, 2015) .

In any case, iterative processing may as well allow intermediate representations to be refined so as to obtain more confident predictions. For instance, Wen et al (2018) propose a

bi-directional recurrent CNN (VGG-like) applied to image classification (on CIFAR, MNIST and SVHN). Inspired from predictive coding theory in neuroscience, this type of model processes the input information in a fundamentally different manner. Instead of computing and propagating representations forwards, each and every layer in a predictive coding network (PCN) targets to minimize the error between top-down prediction — actually the upper layer representation modulated though multiplicative weights — and bottom-up input — which in turn is the error computed by the previous layer. As a result, representations are dynamically adapted, progressively refining final classification predictions. A variant of this network, using only local recurrent processing (Han et al, 2018), yielded better results than the earlier version. It has also been tested on ImageNet, allowing improvements around 1% when compared to similar-size ResNets. Internal representations get updated to match top-down information, making classification decision sharper as time unfolds.

#### 4.2.2.1 Iterative spatial attention

Several works have proposed networks with a recurrent mechanism to take a sequence of crops of the original image, predicting regions of interest to be further processed. This mechanism can be regarded as a simulation of eye saccades, taking glimpses at different locations of the visualized scene.

Since Fu et al (2017) focus on fine-grained classification, they use this kind of mechanism to localize regions of the image that better help discriminate between classes, zooming into and extracting higher-resolution features from these regions.

Another network implementing spatial attention has been proposed by Ba et al (2015), but there the idea is to iteratively integrate information of "glimpses" from different regions, instead of zooming into the current image view for detailed perception. Another model which combines foveal glimpses, but using RBMs is the one by Larochelle and Hinton (2010). On the same line, Sønderby et al (2015) propose a recurrent version of the spatial transformer network [1] (He et al, 2015) that will sequentially focus on individual digits in a sequence.

By adding a recurrent block on top of a standard CNN,Vallet and Sakamoto (2016) tackle a multi-label classification problem with an image-to-sequence model, which implicitly processes successive locations in the image, outputting a sequence of corresponding labels.

Stretching beyond image classification, the same idea of using recurrent mechanisms to implement spatial attention has been applied on image captioning Xu et al (2015) and semantic segmentation Mnih et al (2014).

[1] Spatial transformer networks (Jaderberg et al, 2015) iteratively compute a transformed view on the input image to improve classification results.

#### 4.2.2.2 Handling occlusion and image clutter

Iterative processing can also serve the purposes of handling partially occluded objects, by iteratively assigning perceived borders to different visual elements and eventually inferring a likely completion of the occluded part. This is in line with the human visual system, capable of doing fast recognition under low-noise conditions, but takes longer to correctly process cluttered and partially occluded objects.

An early proposal of a feedback-enabled neural model, that can recognize and restore partially occluded patterns, has been presented by Fukushima (2005). The model tries to complete learned patterns using top down signals. Furthermore, it extrapolates and interpolates visible edges to complete unlearned patterns.

More recently, Spoerer et al (2017) adapted a basic CNN with either lateral connections (self-recurrences), top-down (feedback) connections between adjacent layers, or both at the same time. Their study aimed to evaluate whether and how much these connections help with recognition under occlusion, clutter and Gaussian additive noise. To isolate observation of these effects, they produced a dataset with images of single or multiple digits which could be partially occluded by fragments of other digits. While lateral connections were enough to recognize occluded single digits under light levels of debris, recognition of digits under higher levels of degradation started taking benefit from top-down connections. Feedforward networks however were slightly better at generalizing to un-occluded images. Recurrent networks were also better at identifying multiple cluttered digits on the same image, with lateral connections being crucial since the model with only top-down feedback was outperformed by an equivalent feedforward model. Similarly, recurrent models could better handle MNIST images under Gaussian noise, although here top-down connections were more effective than lateral connections at higher noise levels (lower SNR).

Cao et al (2015) did observe that with feedback connections, classification gets much improved on ImageNet images in which the target object occupies a small percentage of the total area. While overall improvement is of almost 2% (1%), when object is at most 20% of the image the improvement is of 5% (almost 4%) for top-1 (top-5) accuracy. This can be seen as a form of robustness to clutter, since the model is capable of ignoring the majority of the image to focus on the recognition target.

### 4.2.3 Response modulation

Modulatory feedback signals can be responsible for reinforcing features (either channel or spatial-wise) contributing positively for a correct prediction, masking out those which hinder it. In CNNs, those reinforcements can happen channel-wise, selecting feature maps, or spatially, selecting regions of interest on a group of feature maps — which can be seen as an implementation of spatial inhibition.

In general, spatial inhibition through modulatory feedback connections will also help

with cluttered scenes by focusing attention on relevant parts of the image. An example is the model by Cao et al (2015). They add to a base CNN feedback layers with binary control variables, that will selectively focus on locations of the feature maps which were relevant to the predictions, eliminating the influence of irrelevant features. During a feedback iteration, control variables are optimized to maximize currently predicted class score (under L1 sparsity inducing regularization). Salient regions highlighted by the feedback process directly indicate regions considered or dismissed when making a certain prediction, adding an inherent explainability aspect to this CNN model (that is otherwise achievable by *ad-hoc* methods, (see Guidotti et al, 2018, for a specific survey).

An explicit modulatory mechanism is also implemented by Li et al (2018), with top-down information flowing through modulatory masks that multiply convolutional feature maps, improving prediction confidence over iterations. In a less explicit way, the LoopyNet model by Caswell et al (2016) also implements modulatory feedback, particularly in the case where their loop layer has parameters and its output combined with the input by multiplication.

Stollenga et al (2014) propose and *ad-hoc* modulatory mechanism that is learned on top of a previously trained CNN. In fact, their system learns a policy to map the observed current activations to a corresponding modulatory action over these same activations. While the learning procedure is complex — it uses an evolutionary policy search strategy — this method has the advantage of being applicable to any existent pre-trained network.

## 4.3 Architectural trends

### 4.3.1 Recurrence within layers

#### 4.3.1.1 Intra-layer temporal recurrence

When thinking of recurrent CNNs it is straightforward to imagine layers that are recurrent and convolutional at the same time. Several works have experimented with this kind of architecture, reformulating different recurrent units, such as GRU or LSTM, to have a convolutional form (see 1.2 for a brief description of GRU and LSTM).

**Convolutional recurrent cell** The most simple form is to add a self temporal link, implementing a simple recurrent layer, as done by Liang and Hu (2015); Liao and Poggio (2016); Spoerer et al (2017). For a unit located at $(i, j)$ spatial coordinates, on th $k$-th feature map, its pre-activation $z_{ijk}[t]$ is given by:

$$z_{ijk}[t] = (\mathbf{w}_k^f)^T \mathbf{h}^{(i,j)}[t] + (\mathbf{w}_k^r)^T \mathbf{x}^{(i,j)}[t-1] + b_k \tag{4.1}$$

**Table 4.1** Recurrent convolutional architectures from a functional point of view

| | Model Name | Target tasks | Datasets | Functional goals |
|---|---|---|---|---|
| Visin et al (2015) | ReNet | Classification | MNIST, CIFAR-10, SVHN | Spatial context integration |
| Visin et al (2016) | ReSeg | Semantic Segmentation | | Spatial context integration |
| Liang and Hu (2015) | Recurrent CNN (RCNN) | | MNIST, CIFAR-10 and 100, SVHN | Spatial context integration |
| Xie et al (2017b) | Layer-RNN | Classification, Semantic Segmentation | CIFAR-10, Pascal VOC 2012 | Spatial context integration |
| Zuo et al (2016) | Hierarchical RNN (HRNN) | Object and scene classification | Places 205, SUN397, MIT indoor, ILSVRC 2012 | Spatial context integration |
| Ba et al (2015) | Deep Recurrent Attention Model (DRAM) | Sequential digit classification on SVHN | SVHN | Localize and recognize; Iterative spatial attention (multiple locations) |
| Fu et al (2017) | Recurrent attention convolutional network (RA-CNN) | Fine-grained classification | CUB Birds, Stanford Dogs, Stanford Cars | Iterative spatial attention (zoom in to a particular location) |
| Zamir et al (2017) | Feedback Networks | Classification and taxonomic prediction, Human pose estimation | CIFAR-100, Stanford Cars, MPII Human Pose | Iterative processing, on-the-fly predictions |
| Vallet and Sakamoto (2016) | Convolutional Recurrent Neural Network (CRNN) | Multi-label classification (multiple objects per image) | MS-COCO | Iterative processing |
| Stollenga et al (2014) | Deep Attention Selective network (dasNet) | Classification | CIFAR-10 and 100 | Activity modulation; Attention |
| Li et al (2018) | Learning with Rethinking (LR) | Classification | CIFAR-10 and 100, MNIST-background-image, ILSVRC 2012 | Activity modulation; feature selection |
| Caswell et al (2016) | Loopy Neural Networks (LoopyNet) | Classification | MNIST, CIFAR-10 | Modulatory feedback (not explitly stated) |
| Wen et al (2018) | Deep predictive coding networks (PCN) with global recurrent processing | Classification | MNIST, CIFAR-10 and 100, SVHN | Iterative processing |
| Han et al (2018) | Deep predictive coding networks (PCN) with local recurrent processing | Classification | CIFAR 10 and 100, SVHN, ImageNet 2012 | Iterative processing |
| Wang et al (2014) | Attentional Neural Network | Classification | MNIST, MNIST-background noise | Modulatory feedback (attentional mask) |
| Cao et al (2015) | Feedback CNN Look and think twice | Classification, Re-classification, weakly-supervised localization | Imagenet 2014 | Modulatory feedback (attentional mask) |

where $\mathbf{h}^{(i,j)}[t]$ and $\mathbf{x}^{(i,j)}[t-1]$ denote feedforward and recurrent inputs, respectively, corresponding to the 2D patch centered at $(i,j)$, both in vectorized form. The vectorized weights are given by $\mathbf{w}_k^f$ (feedforward) and $\mathbf{w}_k^r$ (feedback), with $b_k$ as bias.

**Convolutional LSTM** Alternatively, convolutions can be integrated with LSTM units, as done by Zamir et al (2017). For a layer $l$, taking $\mathbf{X}^l[t]$ as input and $\mathbf{H}^l$ as the corresponding hidden state, the LSTM equations become:

$$i^l[t] = \sigma\left(W_{x_i}^l(\mathbf{X}^{l-1}[t]) + W_{h_i}^l(\mathbf{H}^l[t-1])\right) \quad \textit{input gate} \tag{4.2}$$

$$f^l[t] = \sigma\left(W_{x_f}^l(\mathbf{X}^{l-1}[t]) + W_{h_f}^l(\mathbf{H}^l[t-1])\right) \quad \textit{forget gate} \tag{4.3}$$

$$\tilde{C}^l[t] = \tanh\left(W_{x_c}^l(\mathbf{X}^{l-1}[t]) + W_{h_c}^l(\mathbf{H}^l[t-1])\right) \tag{4.4}$$

$$\mathbf{C}^l = f^l[t] \circ \mathbf{C}^l[t-1] + i^l[t] \circ \tilde{C}^l[t] \quad \textit{cell gate} \tag{4.5}$$

$$o[t] = \sigma\left(W_{x_o}^l(\mathbf{X}^{l-1}[t]) + W_{h_o}^l(\mathbf{H}^l[t-1])\right) \quad \textit{output gate} \tag{4.6}$$

$$\mathbf{X}^l[t] = \mathbf{H}_H^l[t] = o^l[t] \circ \tanh\left(\mathbf{C}^l[t]\right) \tag{4.7}$$

where $W_{x_i}^l(\cdot)t$ is the convolution operator — single or multi-layer — with parameters $W_{x_i}^l$, $\sigma$ is the sigmoid function, and $\circ$ is the Hadamard (point-wise) product.

**Convolutional GRU** Naturally one can also implement a convolutional GRU unit as in the multi-label classification model by Vallet and Sakamoto (2016), or the semantic segmentation model by (Ballas et al, 2015). Following a similar notation, we have:

$$z^l[t] = \sigma\left(W_{x_z}^l(\mathbf{X}^{l-1}[t]) + W_{h_z}^l(\mathbf{H}^l[t-1])\right) \quad \textit{update gate} \tag{4.8}$$

$$r^l[t] = \sigma\left(W_{x_r}^l(\mathbf{X}^{l-1}[t]) + W_{h_r}^l(\mathbf{H}^l[t-1])\right) \quad \textit{reset gate} \tag{4.9}$$

$$\tilde{H}^l = \tanh\left(W_x^l(\mathbf{X}^{l-1}[t]) + W_h^l(r^l[t] \circ \mathbf{H}^l[t-1])\right) \tag{4.10}$$

$$\mathbf{H} = (1 - z[t]) \circ \mathbf{H}^l[t-1] + z[t] \circ \tilde{H}^l[t] \tag{4.11}$$

Vallet and Sakamoto (2016) observed higher accuracy results when using convolutional GRUs (CGRUs) instead of its non-convolutional version, in both resolutions tested [2]. Their model consists of a pre-trained Inception v3 network for feature map extraction, followed by a RNN block made of residual CGRU blocks and a classifier output layer [3]. Since it is an image-to-sequence model, more than correctly labeling elements in the image, it is important that it gets to detect most objects — high recall — while avoiding false detection — high precision. Both metrics were improved with convolutional GRUs, except in the case of their higher resolution model, which had slightly lower precision. This reduction is expected anyways, since this version of the model was fed with features from an earlier stage of Inception v3 — farther

[2] Resolution of the feature maps extracted from different points of Inception V3.

[3] Dropout and 1x1 convolutions are used prior to the RNN block, the latter performing dimensionality reduction, while global average pooling is done prior to the classification layer.

from the classification layers and thus less discriminative — than the ones that fed the lower resolution version.

**Convolutional reciprocal-gated cell** Targeting to provide a computational model of ventral activity, Nayebi et al (2018) test multiple variants of a convolutional recurrent model. Besides testing simple recurrent network (SRN) and LSTM recurrent cells, they propose a novel reciprocal gated cell, which has yielded more accurate and parameter efficient results in their ImageNet experiments. This particular choice of recurrent structure was crucial for model accuracy, as even after hyperparameter tuning both their model and the other standard baselines, their reciprocal-gated cell models achieved significant better accuracy. This module combines a hidden unit $h$ and a memory cell $c$ through the flowing update equations. For a certain layer $\ell$ at time instant $t$:

$$a_{t+1}^{\ell} = \left(1 - \sigma\left(W_{ch}^{\ell} * c_t^{\ell}\right)\right) \circ x_t^{\ell} + \left(1 - \sigma\left(W_{hh}^{\ell} * h_t^{\ell}\right)\right) \circ h_t^{\ell} \tag{4.12}$$

$$h_t^{\ell} = f\left(a_t^{\ell}\right) \tag{4.13}$$

and for the memory cell update:

$$\tilde{c}_{t+1}^{\ell} = \left(1 - \sigma\left(W_{hc}^{\ell} * h_t^{\ell}\right)\right) \circ x_t^{\ell} + \left(1 - \sigma\left(W_{cc}^{\ell} * c_t^{\ell}\right)\right) \circ c_t^{\ell} \tag{4.14}$$

$$c_t^{\ell} = f\left(\tilde{c}_t^{\ell}\right) \tag{4.15}$$

where $x_t^{\ell}$ is the hidden unit input, $a_t^{\ell}$ $h_t^{\ell}$are the pre-and post-activation output values of the hidden unit, and $c_t^{\ell}$ and $\tilde{c}_t^{\ell}$ are the pre and post-activation values of the memory cell.

The rationale behind this formulation is to obtain a recurrent cell that has both gated feedback signals — a improving gradient flow over time steps — and bypassing forward connections — improving gradient flow to over layer depth, as with identity shortcuts in ResNets. Overall, it is often the case that models will include some form of forward shortcuts — usually residual connections — in addition to recurrence links, such as in the model by Zamir et al (2017) which features residual skip connections every 2 layers (for mild improvements on CIFAR-100 dataset). Similarly, the CRNN model by Vallet and Sakamoto (2016) features a residual block including 3 CGRU units, while the bio-inspired model of Liao and Poggio (2016) uses recurrent shortcut-2 residual blocks in its composition. The rationale behind this usage is the same used for feedforward networks: these connections are expected let gradients flow further back thus improving convergence and overall performance.

**Temporal recurrence within a block of layers** Naturally, self-recurrent links can cover more than a single layer. That is the case with LoopyNets, where feedback signals skip 3 layers (Caswell et al, 2016), or the bio-inspired model by Liao and Poggio (2016), where it skips over 1 or 2 layers. Similarly, Feedback Nets (Zamir et al, 2017) experimented with feedback loops every 1 or 2 layers, going to the extreme case of a single feedback loop

skipping all layers. They empirically observed that often skipping over 2 or 3 layers led to better performances. A parallel can be made between this result and the theoretical and experimental analysis by Li et al (2017), supporting that residual shortcuts with length 2 specifically have particular properties leading to improved convergence rate (when initialized around zero, see section 3.4.2 for more details on this work).

#### 4.3.1.2 Intra-layer spatio-temporal recurrence

Many have proposed spatial recurrence layers, which are not explicitly convolutional, but can be viewed as a particular form of convolutional recurrent network. Most works presented in section 4.2.1 (namely Visin et al, 2015; Liang and Hu, 2015; Xie et al, 2017b; Zuo et al, 2016; Bell et al, 2016) use this type of architecture, which is hereafter discussed here in more detail. They all share a similar mechanism of 2D recurrence between image patches (see fig. 4.2), which has been strongly inspired by multidimensional recurrent networks (Graves and Schmidhuber, 2012) which had been applied successfully to handwritten digit recognition. However, the following models implement 1D recurrence relations between vectorized image patches — and not 2D between pixels.

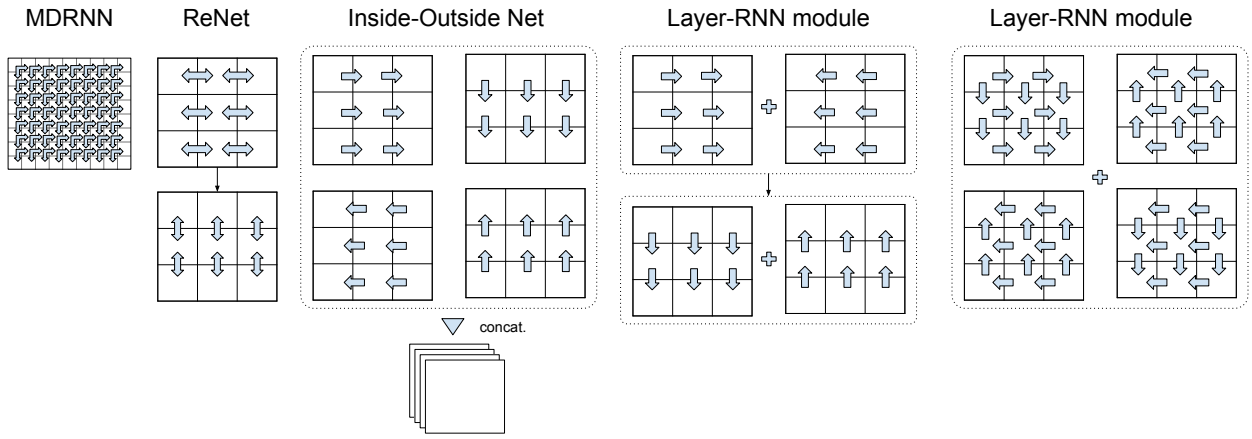



**Figure 4.2** Schematic representation of the recurrent structure of different spatio-temporal recurrent models.

For instance, Visin et al (2015) propose two-step bi-directional recurrent layers (with GRU or LSTM units). Images (or intermediate feature maps) are divided into patches in a grid pattern. The first recurrent layer connects patches in a column, doing a vertical sweep. This is repeated horizontally by the second layer. No explicit convolution is done, neither pooling is used, although authors argue their model can be seen as a variant of CNN where pooling has been replaced by lateral connections — which emulate the local competition among features induced by max pooling — and a non-overlapping convolution is performed.

The Layer-RNN model by Xie et al (2017b) features similar architectural principles, with spatial recurrent modules, named L-RNN, that scan along each row and column Instead of

using bi-directional RNNs, each of these modules has two layers of stacked 1D RNNs for each sweep direction (left-right and right-left for horizontal orientation, bottom-up, and top-down for vertical orientation), producing a set of feature maps that finally get combined by sum or concatenation. Almost the same recurrent structure is found in Inside-Outside Net (Bell et al, 2016), except that each direction is processed independently and the four different outputs get concatenated. Here again convolutional modules (residual blocks in their classification model) are replaced by their recurrent modules (L-RNN), although the spatial organization of L-RNN allows to decompose its computations into a convolutional part and a recurrent part. Furthermore, this rationale supports the addition of modified L-RNN modules to pre-trained CNNs, leaving the convolutional part to be computed with the pre-existing convolutional layers, only introducing spatial recurrence after each layer.

A similar kind of recurrence was proposed by Zuo et al (2016), in what they call a convolutional hierarchical recurrent network (C-HRNN). In these modules recurrent interactions take place, not only between neighbor tiles, but also corresponding tiles at different scales (from lower to higher resolution maps), incorporating spatial context over multiple scales. After extracting feature maps with some CNN layers, they get pooled at multiple scales and divided into spatial patches that get pre-processed by the so-called hierarchical recurrent layers. Both SRN and LSTM have been evaluated as the basic recurrence mechanism implemented by these layers, with LSTM often (but not always) yielding mild improvements over the SRN version. No clear preference has been established by their experiments, indicating this might be an application dependent choice. Particularly in the case of limited computational resources, it might be more interesting to keep it simple with SRNs since performance gains from LSTM were not extremely relevant.

### 4.3.2 Feedback between layers

Feedback between layers has been implemented in several works. For instance, the deep predictive coding network by Wen et al (2018) has feedback between all subsequent layers. Feedback Net, besides featuring ConvLSTM layers, has explicit feedback connections every 1, 2 or $n$ layers (Zamir et al, 2017) . Vallet and Sakamoto (2016) also include a single feedback link from output to the input of their RNN block, skipping over several residual CGRU blocks. The main idea here is to reuse information from representations of higher abstraction levels. Feedback links most of the time are simply propagated by an identity function, ultimately combined with the input of the receiving layer via addition, though some works experiment with multiplication or concatenation.

An exception to these identity feedbacks are architectures which aim to implement explicit modulatory mechanisms. In this case, the feedback information often goes through some neuron layers to generate modulatory maps — namely via feature masks to be applied over existing feature maps (by multiplication) thus implementing feature selection at spatial and/or

channel level. Most models presented under section 4.2.3 follow this type of architecture (namely Li et al, 2018; Stollenga et al, 2014; Cao et al, 2015).

Particularly, Cao et al (2015) explore both adding or multiplying the feedback signal, the first yielding better performances than the latter. Using a pre-trained GoogleNet (ImageNet 2012) as basis, attempting re-classification with the feedback enabled network version — that focuses attention on regions of interest — has improved top-5 and top-1 errors by 1.29% and 1.79% respectively.

#### 4.3.2.1 Explicit biological inspiration

Looking closer at the connection structure between the elements of the visual system, a computational model aiming to represent this architecture would necessarily need feedback connection between regions. Liao and Poggio (2016) have conjectured that a class of moderately deep RNNs is a biologically plausible model of the ventral stream in visual cortex. With this in mind, they have proposed an architecture where each implicated thalamo-cortical area — from LGN to IT — is modeled by a recurrent residual block — taking inspiration from residual networks (He et al, 2016a) — in which the recurrent link includes an identity path skipping over one layer. Besides self-block recurrences, there are connections in both directions — either between subsequent layers or, in a densely connected version, between each and every pair of layers. Since no feedback from retina to cortex has been observed, retina is modeled by a "pre-net" (1 to 3 3x3 convolution layers) which receives no feedback signals. Even though evidence points that recurrent processing is present among retinal cells (Gollisch and Meister, 2010), authors have remained with a simple feedforward modeling. There is also a "post-net" (batch normalization (BN), ReLU, global average pooling and fully connected layer) to output classification vectors from IT representation. Both time-invariant and time-variant (which is a form of residual net, see section 3.4.1) versions have been analyzed.

In a more extensive experimental analysis, Nayebi et al (2018) have studied several variants of convolutional recurrent neural networks (ConvRNNs) based on different forms of recurrent cells and feedback architectures. Over 5000 trials of hyperparameter search [4] have been performed prior to selecting a specific ConvRNN model. They have defined an architecture search space varying multiple aspects such as the path combination operation (add, tanh and multiply, sigmoid and multiply), the choice of convolutional kernel size and where/whether to use depth-separable convolutions. The large sample size allowed to observe some tendencies among the best performing models that support the reciprocal-recurrent local design: it was indeed better to have both gated feedback and forward shortcuts on the same cell. On a global scale, the search also pointed to long-range feedbacks between specific layers which would often boost (or reduce) accuracy.

While computationally expensive, this search allowed selecting particular connections

[4] Using Bayesian tree-structured Parzen estimator (TPE) algorithm (Bergstra and Bengio, 2012).

that helped performance, instead of having to learn weights for all possible connections during training (as implicitly done by Liao and Poggio). With the carefully selected model, a good predictive of primate neuronal activity and dynamic trajectories could be obtained (particularly for higher level areas V4 and IT), while at the same time achieving good object categorization accuracy. Their experiments have demonstrated that not only a well-designed recurrent model performing a challenging real-life-like task — ImageNet classification — can be more accurate than feedforward ones, not only on the task itself but equally as a dynamic model of primate object recognition under presentation of still images.

#### 4.3.2.2 Ladder networks

Ladder network is an architecture inspired from denoising autoencoders (DAE). It is not convolutional but it is worth mentioning due to its use of feedback and lateral connections. The first proposals of this architecture targeted unsupervised (Valpola, 2014) and semi-supervised (Rasmus et al, 2015) learning tasks. More recently, Prémont-Schwarz et al (2017) proposed a recurrent version, but mainly for temporal modeling purposes, although it has also proven useful for perceptual grouping on still images.

The architecture is composed of two feedforward encoding cascades of layers — one clean and another with additive noise, both sharing the same weights —- and a feedback decoding cascade of layers, each receiving lateral input from its corresponding layer on the noise bottom-up path. The combination of lateral connections and upper layer feedback signal is done via a linear combination plus a multiplicative term mixing both signals. A second instance of the same function, but this time going through a sigmoid non-linearity, complements the combination. The role of the feedback path is to denoise features from the noisy feedforward path, learning to do so by comparing its outputs to the corresponding features from the clean feedforward path. Here lies the main architectural difference regarding DAE, in which noise is only added to input which is the only information being denoised by the decoder. This denoising objective for each layer yields a non-supervised cost, that may be used alongside a classification cost on semi or fully supervised tasks.

Pezeshki et al (2015) conducted a detailed empirical study of how different design elements contribute to its performance, both on semi-supervised and fully supervised tasks. They have verified that the additive noise at every layer — an thus a corresponding reconstruction cost — acts as a regularizer, improving generalization. While it may be a way of leveraging information from unsupervised data, in the fully supervised scenario the reconstruction cost terms may become irrelevant, as observed in their experiments: hyperparameter tuning (grid or random search) yielded zero weighs for the reconstruction cost on all but the first layer. Moreover, lateral connections turned out to be a vital component of the architecture, being more important than additive noise. In particular, reconstruction costs were also irrelevant (zero weight) when lateral connections were removed. Regarding the combination function

of lateral and feedback connections, they observed the sigmoid term to be of little importance, while removing the multiplicative term led to a stronger degradation.

### 4.3.3 Hybrid models with added recurrent layers

Let us now consider models that integrated recurrence by adding some form of recurrent layer (simple, GRU, LSTM or others) to a basic CNN architecture, usually (but not always) between the last convolutional layer and the output layer. It is a strategy widely used on video processing (see for e.g. Donahue et al, 2017; Ng et al, 2015), where models typically extract features from video frames using a 2D CNN and feed them to a recurrent model in order to characterize the video's temporal structure (Ballas et al, 2015). In this section however the focus is on architectures that model other recurrence relations on still images.

A recurrent multi-scale architecture is used by Fu et al (2017), with one VGG-style feed-forward network for each scale. In addition, an attention layer predicts a region of the current scale's input to be cropped and fed to the next one, performing a select-and-zoom iterative process. After 3 iterations, features from all scales are used for predictions. While classification is achieved through a regular softmax cross-entropy loss, prediction of the next attended region comes from a inter-scale ranking loss that aims to always select a more discriminative region for the next scale. Both are optimized alternately.

Another kind of hybridization is proposed by Xie et al (2017b) when they add recurrent layers after each convolutional layer of a pre-trained network. An advantage of this method, as argued by the authors, is that it is possible to add recurrent layers to a pre-trained CNN and only fine-tune the parameters of the newly added layers, reducing total training time. In fact this approach improved performance in all scenarios tested in their work. More generally, this modularity advantage is not specific to their model, but is extendable to any hybrid CNN+RNN model, which can support resorting to these models when data is relatively scarce and using a pre-trained network is a necessity.

**Table 4.2** Summary of convolutional recurrent architectures.

| | Model Name | Architecture | | | | | Can be easily added to a pre-trained net? |
|---|---|---|---|---|---|---|---|
| | | Intra-layer recurrence | Explicit conv. recurrent layers | Inter-layer feedback | "External" feedback | Hybrid CNN+RNN | |
| Visin et al (2015) | ReNet | Yes, replaces conv. layers by spatial RNN | No | No | No | No | No |
| Visin et al (2016) | ReSeg | Yes, replaces conv. layers by spatial RNN | No | No | No | No | No |
| Liang and Hu (2015) | Recurrent CNN (RCNN) | Yes | Yes | No | No | No | No |
| Xie et al (2017b) | Layer-RNN | Yes, interleaves or replace conv. layers with L-RNN modules | No | No | No | Yes | Yes |
| Zuo et al (2016) | Hierarchical RNN (HRNN) | Yes | No | No | No | Yes, integrates CNN with HRNN on top | Yes |
| Ba et al (2015) | Deep Recurrent Attention Model (DRAM) | Yes, on top recurrent layers | No | No | Yes, indirect feedback via additional modules | Yes | Yes |
| Fu et al (2017) | Recurrent attention convolutional network (RA-CNN) | In the attention prediction network | No | No | Yes, Only to chose attention focus for next scale network | Yes | Yes |
| Zamir et al (2017) | Feedback Networks | Yes, recurrent convolutional blocks | Yes, convolutional GRU, LSTM and RNN | Yes | No | No | No |
| Vallet and Sakamoto (2016) | Convolutional Recurrent Neural Network (CRNN) | Yes | Yes, convolutional GRU | No | No | No | Yes |
| Stollenga et al (2014) | Deep Attention Selective network (dasNet) | No | No | Yes | No | No | Yes |
| Li et al (2018) | Learning with Rethinking (LR) | No | No | Yes | No | No | Yes |
| Caswell et al (2016) | Loopy Neural Networks (LoopyNet) | Yes, intra-block of conv layers | Yes | No | No | No | No |
| Wen et al (2018) | Deep predictive coding networks (PCN) with global recurrent processing | Yes | No | Yes | No | No | No |
| Wang et al (2014) | Attentional Neural Network | No | No | No | Yes | N/A | Yes |
| Cao et al (2015) | Feedback CNN Look and think twice | No | No | Yes | No | No | No |

## 4.4 Discussion

### 4.4.1 Feedback vs recurrent feedforward

An interesting discussion raised by Zamir et al (2017) is the difference between recurrent networks with inter-layer feedback — which they name recurrent feedback nets — and recurrent nets with only self-layer recurrences — which they name recurrent feedforward (see supplementary material from Zamir et al, 2017). With self-recurrences, there is no combination of different levels of abstraction. In particular, there is no direct influence of the network's current output over its activity. Conversely, once feedback is added, higher level information is dynamically propagated top-down at every iteration. Moreover, a prediction output has to be available at every timestep since it provides a feedback signal which allows dynamic adaptation, refining predictions at each iteration (see fig. 4.3). This forces the model to make meaningful predictions at every timestep, even if it might be changed in later iterations.

Zamir et al (2017) have studied their model with and without feedbacks (leaving only self-recurrences) and observed that removing feedbacks and leaving only recurrences changed the model's ability to make early and taxonomically correlated predictions (in which predicted labels are iteratively narrowed down to subcategories).

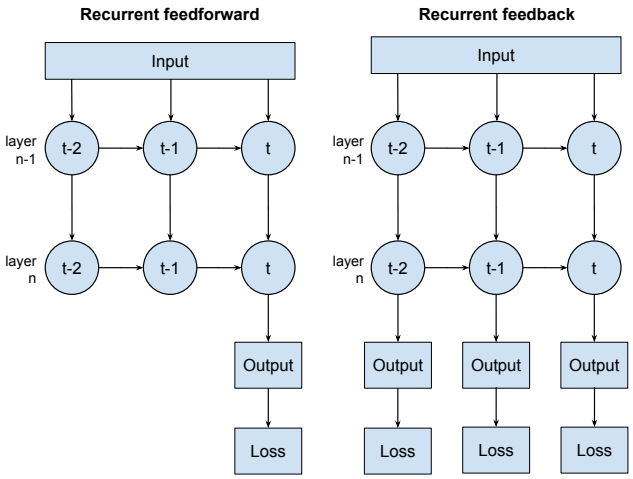



**Figure 4.3** Feedback nets predict an output at every time-step, connecting the loss to intermediate hidden states and changing how features are learned. Adapted from Zamir et al (2017).

### 4.4.2 Internal representations on feedback networks

In the Feedback net model by Zamir et al (2017), a recurrent feedback CNN, which generates an output at every time step is applied to image classification. They argue their model adjusts its representations on an abstract coarse to fine scale, contrasting it to the typical

hierarchical organization of features on depth of a network. This claim is partially supported by t-SNE visualizations of how the representation evolves through network depth (for a feed-forward net) or iterations (for a feedback net), even though it seems that these aspects – depth and time iterations — are not necessarily equivalent to base this comparison. Nevertheless, it remains that activation maps observed for layers of equivalent "virtual depth" (= depth $\times$ iterations) were qualitatively different from those presented by a purely feedforward or even a "recurrent feedforward" network (this idea is further discussed in the next section). Additionally, their model's predictions seem to conform to a hierarchical structure of the learned classes: as the "virtual depth" increases, predictions get narrowed, say from hamster to rabbit, to quote one example from the paper. This aspect has emerged naturally from the architecture, while the same has not been observed for feedforward models, even when trained with a an auxiliary loss considering both coarse and fine-grained labels.

### 4.4.3 Training considerations

#### 4.4.3.1 Temporal recurrence depth

Starting from static feedforward models, the inclusion of time dimension consequently adds some related extra hyperparameters. Some examples are the input stimulus duration (how long to keep input image presentation), temporal recurrence depth (how long to unroll the model during backpropagation through time (BPTT) training ), readout time (how many time iterations to run before using output as prediction, during test/deployment phase), among others. While not many works explore the effects of these hyperparameters on their model's function and accuracy, quite a few at least mention or experiment with the temporal recurrence depth. Most models do so to a moderate extent — from 2 to a few tenths of iterations.

Caswell et al (2016) for example unroll their network in time for 3 iterations. In a more thorough exploration, Zamir et al (2017) test training different network depths (12, 15, 18, 21) and time iterations(though their main experiments all used $T = 4$ iterations). They observed that for a given network depth, there is an ideal number of iterations, with performance degrading for higher or lower values. In particular for their model, training with $T = 4$ iterations was ideal for a 12-layer network and $T = 8$ for an 8-layer network. Additionally, having feedback (having $T > 1$) was always better than not having any. This issue was also explored briefly by Liao and Poggio (2016), but varying time iterations only during test. Having trained with $T = 10$, a similar effect has been observed: test error drops until $T = 10$ then re-increases if further iterations are done. However since in this case no analysis changing $T$ during training was done, this result could be due to overfitting to the specific time-depth condition.

Overall, it is natural to expect such limit, since more time iterations implicate repeated application of the same filters, which might at some point lead to strong information loss. From another point of view, a parallel can be made with degradation in very feedforward deep networks, but considering that a recurrent network has a larger *virtual depth* (time $\times$ depth) as $T$ increases.

The distinct nature of predictive coding models leads to expect a different behaviors regarding the number of recursive iterations. Wen et al (2018) tested $T = \{0, 1, \cdots, 6\}$ for their PCN model, and noted that larger $T$ tended to lead to faster convergence and better accuracy. Indeed throughout the iterations it is possible to see how the network refines its predictions and corrects itself, particularly for ambiguous images. It is natural to expect such convergence behavior given the objective of individual layers to predict next level predictions with minimal error.

#### 4.4.3.2 Computational requirements

With recurrent models, the need to represent internal states in the computing graph leads to a larger memory need during training. As observed by Vallet and Sakamoto (2016), it can be important to limit the input size of features feeding recurrent blocks. Using features extracted from different points of an Inception V3 network, they had either $8 \times 8 \times 2048$ (0.13 M parameters) or $17 \times 17 \times 768$ (0.22 M parameters) inputs. To limit memory usage and fit the model inside their available hardware, a 1x1 convolutional layer as used to reduce dimensionality to 256 channels, prior to their residual CGRU sub-network.

When processing still images, a recurrent model in general will require T iterations of feedforward and feedback passes, against a single feedforward pass of a plain CNN. Considering both top-down and bottom-up passes take $N$ FLOPS, a CRNN will take $2TN$FLOPS against only $N$ FLOPS for a CNN. However, if the input is a video, this gap is straightened, with CRNNs taking $2N$ per frame, against $N$ per frame on a CNN (as remarked by Wen et al, 2018).

On the bright side, recurrent models can be capable of achieving state-of-the-art results under a lower number of trainable parameters, thus reducing the computational impact of time-unrolling and storing hidden states. For instance, the median ConvRNN by Nayebi et al (2018) could achieve the ResNet-34's top-1 ImageNet performance (around 73% accuracy) under only 75% of the parameters and half the number of layers.

### 4.4.4 Biological plausibility

Feedforward CNNs have not provided accurate models of ventral neuronal activity when trained for simpler tasks or on unsupervised tasks (Hong et al, 2016; Khaligh-Razavi and Kriegeskorte, 2014, more on section 2.5.1). As reported by Khaligh-Razavi and Kriegeskorte

(2014) (more on section 2.5.1), CNNs trained on unsupervised tasks have not yielded accurate models of per-image (not average) neural responses, particularly in the higher visual cortex. Training for real complex tasks seems to be critical to obtain a highly predictive model of th ventral stream activity, including the case of recurrent models (Nayebi et al, 2018). In this sense, though many works make reference to biological inspiration, few actually try to model neuronal activity and perform well on realistic benchmarks such as ImageNet (see table 4.1). Mild improvements on simpler datasets (such as Canadian Institute for Advanced Research (CIFAR) and Street-view house numbers (SVHN)), observed by Liao and Poggio (2016) for instance, might be simply due to an extra number of parameters of the model rather than an actual contribution of recurrent processing.

The hypothesis of recurrent processing only being useful under challenging conditions has been studied and experimentally challenged by Kar et al (2019). By comparing performances of primates (humans and monkeys) to that of a feedforward CNN, Kar et al (2019) could identify challenging images which could not be well categorized by the non-recurrent CNN model. Investigating further IT response to such images, measurements indicated they would take longer to be processed, taking extra $\sim$30 ms on average to observe first activity signals on IT cortex. As expected, this delay was reflected in the non-recurrent model abilities: even though early IT responses could be well explained, late activity (around 100-150 ms) could not. In this latter case, very deep nets and shallow recurrent models were the best predictors, suggesting the need for further iterations of non-linear transformations for challenging inputs, be it in the form of more layers or temporal recurrence.

The model by Nayebi et al (2018) has proven to be a step towards a more bio-plausible ConvRNN model. In their work, extensive hyperparameter and architecture search has been performed prior to selecting a specific ConvRNN configuration. Out of more than 5000 model-samples, the median accuracy model was later fully trained on ImageNet and had its predictability of neural activity analyzed. Their fully-trained median ConvRNN was capable of correctly classify images that take longer to be decoded , which are challenging for feedforward models according to the study of Kar et al (2019). Comparisons between primate neuronal activity and model activity (under equal static visual stimuli) demonstrated how a particular form of recurrent structure is determinant for an accurate modeling of the activity of the ventral stream. Not only it could well-predict time-average values but also predicted dynamic neuronal response trajectories for single-images highly correlated to the neurological ground-truth. Representations learned by this ConvRNN were compared highly predictive of neuronal population recordings of higher visual areas — particularly linking upper layers 6 to 8/9 with areas V4, anterior infero-temporal (aIT) and central infero-temporal (cIT)/posterior infero-temporal (pIT), respectively. Furthermore, the complex connectivity of their ConvRNN appeared to be critical to explain neuronal dynamics, as a simpler baseline dynamic model could not achieve similarly accurate predictions.

## 4.5 Conclusion

There have been many studies trying to integrate recurrent mechanisms into CNNs, for purposes other than explicit image-sequence treatment. Some studies seek to understand whether these mechanisms can bring the functionalities they are reputed to perform in our brains — like the study of recognition under occlusion by Spoerer et al (2017). Reasonably, these studies often concern themselves with simplified synthetic datasets, with controllable variability, in order to allow deduction of more precise conclusions. Unfortunately, this also implicates in results that may not translate well in practice to more realistic images. Studying this type of mechanisms in application to real-world data remains a topic under-explored in experimental research.

Other works, more concerned with practical results, tackle directly more realistic problems, at the expense of a clearer understanding of how the inclusion of recurrent processing and/or feedback loops has brought any improvements. The complexity of the task and of the entire architecture fog the ability to conclude whether these additions were actually able to fulfill any of the functional "purposes" their analogous counterparts serve in our own natural visual system. Moreover, the inclusion of recurrence complicates the theoretical analysis of DNNs even more, further reducing the possibility of developing an intuitive comprehension of their inner functioning. It is therefore important to advance theoretical analysis of these architectures.

From a neurological point of view, it can be argued that the brain does not "stack up more layers" in order to process a complex input, it simply takes longer with recurrent loops over the same network structure (Wen et al, 2018). In that sense, the inclusion of recurrences also presents itself as an alternative to the trend of "going deeper" with the inclusion of ever more layers to feedforward models. The relationship between residual networks and recurrent layers points towards the possibility of running shallower models over multiple time-steps as lower parameter-count option.

Moreover, it is a potential method for having similar performances under many less parameters that deserves further experimental and theoretical exploration. In particular, their suitability to small data regimes remains to be studied. The lower dimension of this model space could be particularly important in situations of limited data availability. However, the relationship between model size and generalization is not that straightforward for deep networks and is closely related to how they are optimized, as discussed in the next chapter. This hypothesis could thus be strengthened by further experimental testing and theoretical developments.

# Part II

# Image classification on small datasets


**5 A review of strategies to use deep learning under limited data 85**
5.1 Introduction . . . 85
5.2 Problem statement . . . 86
5.3 Axis I: Get more data . . . 90
5.4 Axis II: Reduce model complexity . . . 102
5.5 Conclusion . . . 112

**6 Analysis of DCNN applied to small sample learning using data prototypes 113**
6.1 Introduction . . . 113
6.2 Related works . . . 117
6.3 Model proposed . . . 121
6.4 Experiments and results . . . 125
6.5 Discussion . . . 138
6.6 Conclusion . . . 141

Chapter 5

# A review of strategies to use deep learning under limited data

## 5.1 Introduction

Following the growth of Internet-based companies in the last decades, data collection from all aspects of everyday life has become commonplace to a level that other industries increasingly take interest in making profit from their own data as well. However, collecting and labeling large-scale datasets is not as ubiquitous as it may seem. Under certain application domains, building such dataset is expensive, if not practically impossible. Difficulties may arise from the limited occurrence of the objects under consideration — unique medical conditions, infrequent machine faults or rare animals in nature, for instance. Alternatively, data scarcity may simply be due to the novelty of the targeted object. In either case, the ability to infer patterns and learn to generalize predictions from few samples is paramount to successfully exploiting available data.

On the image domain, CNNs have proven invaluable in achieving a new level of prediction accuracy, when compared to traditional hand-engineered features, in diverse domains. The greatest breakthroughs, however, have intervened in the context of large-scale datasets. The concept of *large-scale* is of course relative to multiple factors. Intuitively, one may account the quantity of samples per category, the total number of such categories, the diversity of conditions images were taken, as factors weighting in for the appreciation of a given dataset as sufficiently large — or conversely, insufficiently small. Nevertheless, it remains that CNNs in general need a significant quantity of data in order to successfully learn meaningful features together with class boundaries.

From a theoretical perspective, the complexity of the model family composed by DCNNs is generally high, taking into consideration the typically very large number of learnable parameters. Generalization boundaries predicted by usual complexity theories (see discussion by Jakubovitz et al (2018) and first chapters of Shalev-Shwartz and Ben-David (2014)) provide relationships between error rate (in test, thus generalization capability), number of training samples, and the cardinality of the model family considered during learning. Even for modest sized CNNs, the cardinality of the model search space is so high that an impossibly high

number of training samples is necessary. Yet, CNNs generalize on large-scale-but-theoretically-insufficient-data.

Explanations for such counter-intuitive behavior are subjects of current theoretical and experimental research (Jakubovitz et al, 2018). Intuitively, it is hypothesized that even though model space is large, the way in which networks are trained has a high likelihood of exploring only a little portion of this space, which contains local minima of high generalization capability. Guiding learning towards these solutions seems more difficult under limited data, calling for a special attention to learning methodologies applied in this scenario.

Mapping out the existing strategies, methodologies and techniques which potentially increase the likelihood of learning over limited data are the main objectives of this review chapter. The following matters will discuss and categorize several works that manage to apply DCNNs successfully under some form of data availability constraint. When suitable, works relevant to other related topics (section 5.2.2) will also be reviewed in order to complement the selection of potentially useful strategies. We choose to address this matter from the dual perspective of training set size vs model complexity, which define the two main axes of this review (sections 5.3 and 5.4). The focus is divided according to the lever of actuation: either more data is to be used, or model complexity should be reduced — either directly through architectural constraints, or indirectly through particular training strategies aiming to guide the learning process. It is blatantly evident that the amount of research in the field of neural networks has expanded exponentially over the past few years. As such, this survey is not intended to be thorough on works concerning small sample deep learning and related fields, nor will it describe such works in detail, rather focusing on aspects pertinent to the structured view we propose.

## 5.2 Problem statement

### 5.2.1 Small data scenarios

The expressions “small sample learning” or “learning under limited data” are sufficiently broad to encompass multiple situations regarding the availability of labeled or unlabeled data. Different strategies may apply for each situation (schematically depicted in fig. 5.1):

A. There may be a considerable amount of labeled data, but it remains insufficient in regard of the total number of input features used for learning;
B. There may be limited labeled data, but a large amount of unlabeled data on the exact same domain, that can be leveraged by a semi-supervised or weakly-supervised strategy;
C. There may be limited labeled data on the target task, but there are related datasets from the same or other domains that can be leveraged by a transfer learning approach;

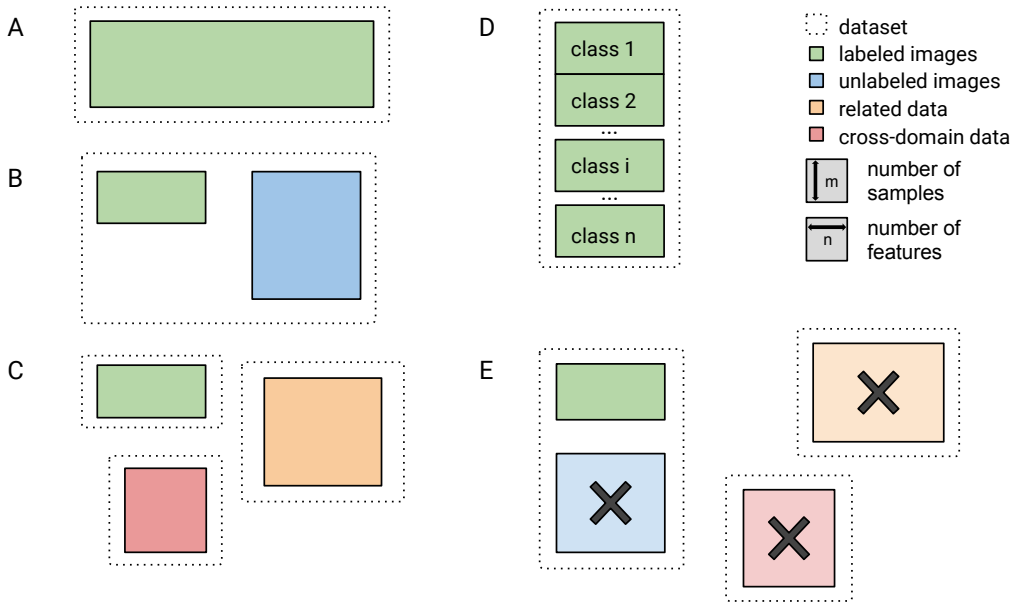



**Figure 5.1** Illustrative diagram differentiating the different small data scenarios ranked in the text. Rectangles represent data matrices (with samples as rows and features as columns). Different colors mark different types of data (see legend in the top right corner) and dashed lines delimit distinct datasets.

D. There may be limited labeled data on each category, but a large number of categories makes a larger dataset in total. Transfer learning might still be relevant in this scenario, which is closely related to fine-grained classification;
E. In a worst case scenario, there is little labeled data and eventual source domains are too different to have a positive influence in learning the target task. In that case, explicit knowledge about the target application and domain may be invaluable in achieving this rather impossible task.

### 5.2.2 Related problems

The previously described scenarios naturally link small sample learning to other subareas and frameworks studied within machine learning literature. Despite their common ground, these other areas differ in their main targeted issue as well as their usual methods. Nevertheless, similarities are sufficient to consider these subareas as an important source of methodological ideas relevant to learning on not-so-big datasets. Hereafter we briefly discuss some of these related problems.

**High-dimensional data** Regardless of the reason leading to a small dataset, high dimensional input spaces are a frequently related issue. Data from neuroimaging studies are a representative example on this matter. Due to constraints in the number of participants and access to equipment, usually few trials are made, leading to small datasets. If we are to use the raw data directly, each example has millions of voxels, leading to a poorly sampled input space,

regardless of how many volunteers participate in the study. In such situations dimensionality reduction and feature extraction methods are often necessary steps. More recently, some studies have managed to use DNNs for different sub-tasks linked to MRI imaging — such as image restoration, segmentation and classification — to varied degrees of success (see Lundervold and Lundervold, 2019, for a specialized review).

**Imbalanced classes and oversampling** In imbalanced datasets, some categories are under-represented with respect to the the remaining ones. These classes need to be learned on "small class-specific data", although other classes and the overall data set may have a large amount of samples. Class imbalance can be alleviated by oversampling techniques (for instance by sampling batches with a higher probability for the under-represented classes), but results are limited if the overall data is anyway small. It can be the case that only some particular classes are under-represented, leading to unbalanced classifiers. When the overall amount of images is sufficiently large, oversampling techniques like boosting can bring sufficient improvement. However, plain oversampling cannot increase the sampling coverage of the input space, which is desirable under small data constraints.

**Image retrieval** In image retrieval, the goal is to provide a ranked list of database images according to their similarity to a given query sample. By analogy between query and test samples, and between the database and the training set, one could frame classification in a similar fashion: rank training samples by similarity to the test ones, then compute class predictions from this ranking. This line of reasoning has been exploited for instance by Triantafillou et al (2017), for few-shot learning (discussed in section 5.4.3.1). Information retrieval in general is also concerned with larger databases and fast query treatments, which are not of interest for small data learning.

**Fine-grained classification** Fine-grained classification problems generally comprise specific sub-classes within a same object or scene category. Some examples are identifying particular car models (Krause et al, 2013) or bird species (Wah et al, 2011). In this type of problems, differences between classes are more subtle, which make learning distinctive features more difficult. Moreover, it is often the case that the number of fine-grained categories is so large that, even with tens of thousands of images, the number of samples per category is relatively small, approximating to a small sample learning scenario.

### 5.2.3 Implicit and explicit knowledge

Human memory capabilities can be categorized between procedural and declarative. While the later concerns knowledge one can enunciate and transmit explicitly in a given language code, the first is less evidently considered knowledge even though it is every much as important, if not more, than any explicit knowledge one may declare. Procedural memory concerns our acquired or innate abilities to act on the world under different situations. It is

our "know-how", our "savoir-faire", an implicit knowledge one may not be able to easily explain despite being able to perform accordingly.

Explicit knowledge can be very useful in tailoring a model architecture or loss function to a particular need. Quoting a widely used example, using specific computer vision knowledge on image filters was crucial when choosing to share weights across all locations in a locally connected layer — making it an actual convolutional layer. Such choice has a significant impact on model size and complexity, being primordial for the success of CNNs. Other examples of useful prior explicit knowledge will be given across this chapter.

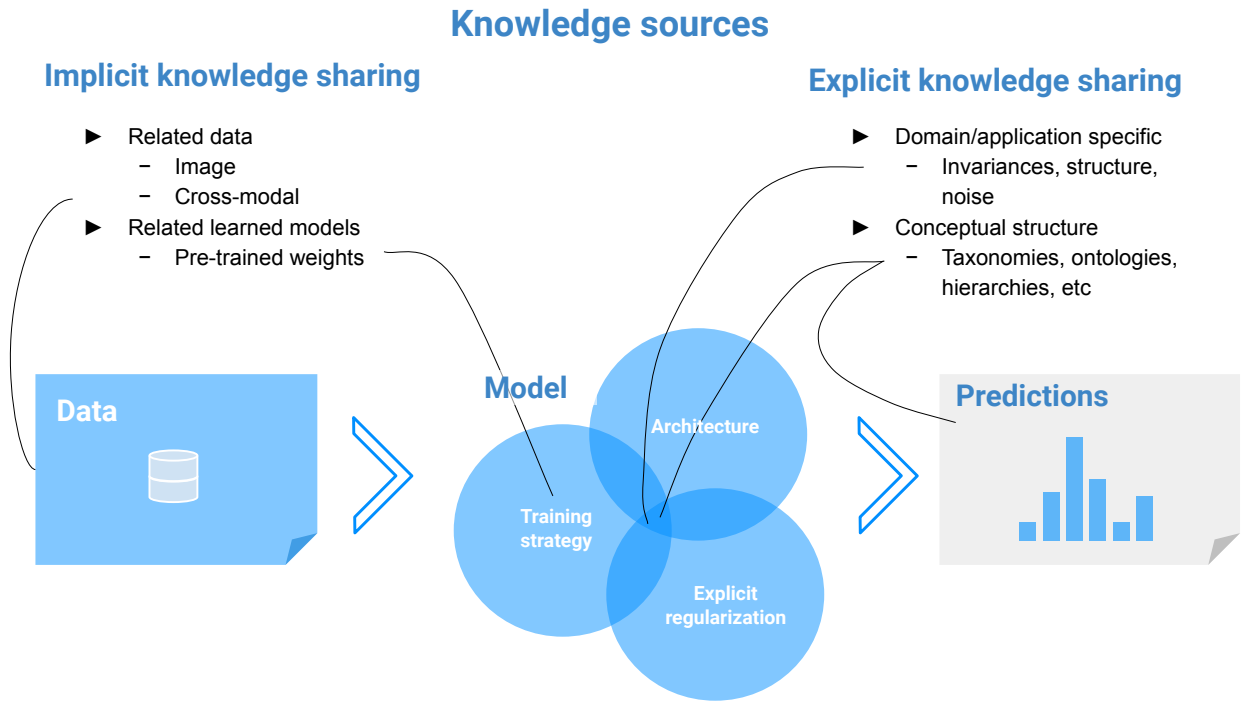



**Figure 5.2** Knowledge sources

### 5.2.4 Knowledge transfer

The idea of sharing information between tasks or domains to help solving a target task has a direct inspiration on human cognitive abilities. For instance, a guitar player should spend significantly less time when learning to play the bass, in comparison to a not-musically-trained individual, due to the fact that his previous knowledge is relevant to solving this new task. Analogously, one can imagine that knowledge gained by a machine learning model could help another in accomplishing a different but somehow related task.

As defined by Pan and Yang (2010) (and restated by Weiss et al, 2016; Ruder, 2017b), *knowledge transfer* or *transfer learning* aims to improve learning of a predictive function on a target task by making use of knowledge from a source task, given that source and target

domains and/or tasks are different. In traditional machine learning there is usually an implicit assumption that training and test are done on datasets of the same domain — that is, presenting identical distributions $P(X)$ and represented on the same feature space $(\mathcal{X})$. Such requirement is partially waived in this definition of transfer learning, since domains may differ. However, this definition is broad enough to include situations where domain is the same/similar, with the difference in tasks characterizing the knowledge transfer setting.

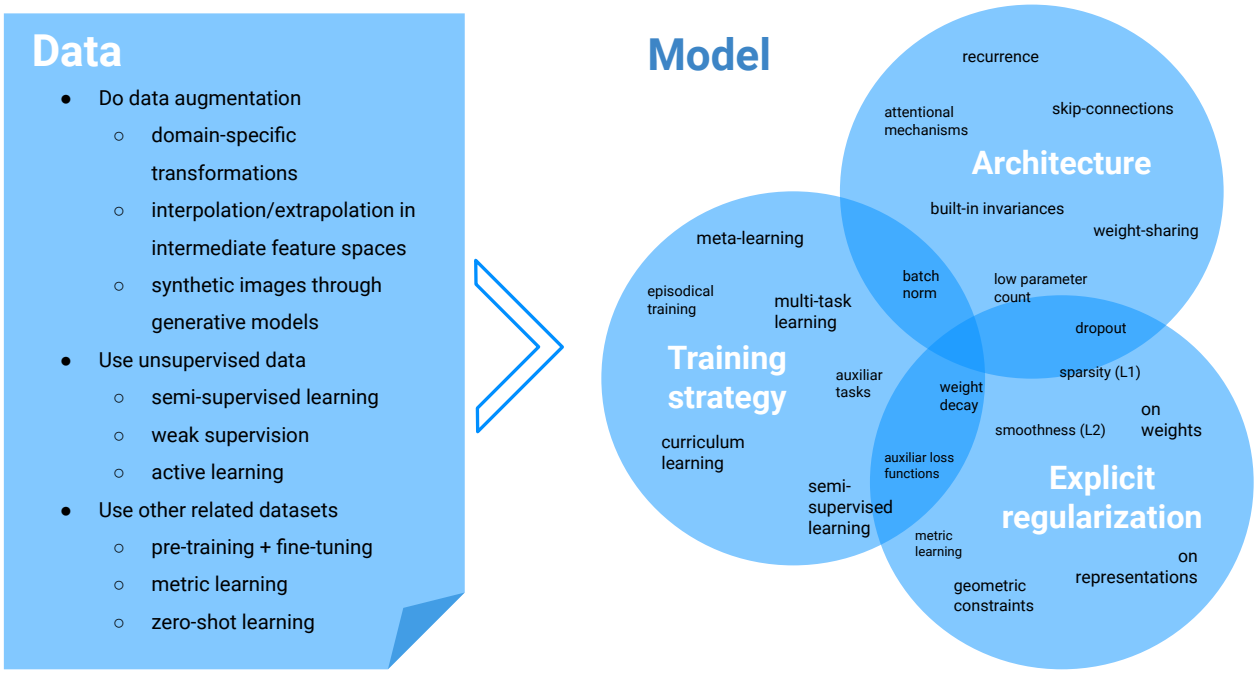



**Figure 5.3** Strategies for deep learning under small data. This diagram illustrates the point of view presented in this chapter.

## 5.3 Axis I: Get more data

It may seem contradictory to expect more data whilst discussing means for learning under limited data budgets. It is however a hard limitation (though difficultly quantified in practice) that a minimum amount of data variability — corresponding to a representative sampling of (likely regions on) the input space — is necessary for any statistical model — and for deep networks in particular — to infer invariants that will lead to a sufficiently correct mapping of the label space. If not enough variability is present, there are different strategies that can be used to increase a model's sampling of the image space, from basic random data augmentation to more complex generative modeling (section 5.3.1).

Alternatively, it is often the case that the limitation is mostly present in terms of labeling efforts than in amounts of raw unlabeled images. These occasions are adapted to unsupervised feature learning and semi-supervised strategies (section 5.3.2). If some extra labeling work can

be afforded, active learning strategies can help on making it more efficient (section 5.3.2.3).

Finally, one can also resort to other datasets, generic or domain specific, that can be used in either unsupervised or supervised pre-training (section 5.3.3.1). Related datasets for other modalities can also help in building a semantically meaningful representation, as exemplified by zero-shot learning approaches (section 5.3.3.2). These three aspects — synthetic data, unlabeled data and other datasets — guide the presentation of data-related strategies in the following subsections.

### 5.3.1 Synthetic data

Artificially augmenting training data has been widely done in the image recognition domain, from LeNet (Lecun et al, 1998) in 1998 until today. It sheds some light on how fundamental and widespread this practice is to notice that all the ImageNet challenge-winning models since 2012 have made use of some data augmentation strategy. Of course it is not restricted to deep networks and applicable to almost any statistical learning method.

Data augmentation is particularly beneficial when the ratio model complexity/dataset size is too high: more data samples reduce the risk of overfitting with complex models (like most deep networks). Even though the case of large deep networks exacerbates the need for extra data, feeding a shallow network with augmented data is not necessarily worthless: analogous boosts in performance can be obtained, although possibly less pronounced than those obtained with deep over-parameterized networks (as observed by Chatfield et al, 2014).

#### 5.3.1.1 Augmentation at the input space

The simplest augmentation method is to randomly sample training images to be corrupted with a small quantity of noise, additive or otherwise, while keeping the labels of the original images. On a more structured perspective, this "noise" can take the form of pre-defined distortions and transformations (Poggio and Vetter, 1992; Simard et al, 2003), often derived from domain or application-specific symmetries. This can be easily exemplified in the image domain: a generic image classifier ideally should output the same labels for a mirrored image or a slightly rotated one.

The most efficient augmentation methods in general will try to generate synthetic images that are both semantically valid — meaning a realistic image is generated — and diverse — meaning a large variety of transformations is applied aiming to cover most degrees of freedom to which the model should learn to be invariant (Xie et al, 2019).

Multiple image transforms can be applied in sequence and/or at random for a more diverse augmentation: image jittering or random crops, relevant rotations, changes in brightness and contrast, horizontal or vertical flipping. It remains nonetheless essential to consider which transformations could actually be harmful for a given target application. For instance in

medical imaging, horizontal mirroring transformations are at times avoided as they may produce anatomically implausible images (e.g. Liu et al, 2019).

Overall, the more variability, the better, as long as generated images are realistic considering the application scenario.

#### 5.3.1.2 Augmentation in the feature space

While several semantically valid transformations are intuitive for the image domain, finding such transformations can be less obvious in other domains. This difficulty motivates the design of augmentation procedures intervening at features space. In the case of neural networks, this translates to creating new data points directly at some intermediate layer. For instance, DeVries and Taylor (2017) proposed one such method which generates new samples directly on feature space by manipulating vector representations of an autoencoder. Of course such methods are also a possibility for image CNNs, alongside traditional image augmentation.

Implicitly, feature space augmentation obtains new samples through some combination of feature vectors computed from the original training images. In the work by DeVries and Taylor (2017), a sequence autoencoder was trained and then used to encode input samples into context vectors. Augmentation was produced by applying noise, interpolations or extrapolations to such vector representations, which can be used directly as feature vectors for classifier training. By extension, such generated vector codes can also be fed to the trained decoder to obtain corresponding input representations, which could in turn be used for training or simply to provide more clarity on the visual quality of the generated data.

Similar augmentation methods can be done over CNN features, without the need for a decoding process. For instance, the MixUp method proposed by Zhang et al (2018) uses convex combinations of feature vectors — and corresponding label indicator vectors — to generate virtual samples. It has been successfully applied to semi-supervised training CNNs for image classification in follow-up works (Berthelot et al, 2019; Verma et al, 2019; Liu et al, 2019). Together with the unsupervised augmentation method proposed by Xie et al (2019), these works support that a well-modeled augmentation noise — going beyond simple image transformations — can improve accuracy, particularly in semi-supervised tasks — that is under a limited number of labels.

#### 5.3.1.3 Augmentation through generative modeling

The previous example of feature space augmentation by DeVries and Taylor (2017) is also a case of augmentation through generative modeling — in case the generated codes get to be decoded into new input images. Overall, generating new synthetic images that go beyond simple geometric transformations requires an explicit and generative modeling of the data

distribution (see section 1.2.2 for some brief descriptions of deep generative models).

Most recent image data augmentation work has relied on GAN model variants — for instance DAGAN model by Antoniou et al (2018)— with a smaller number of works experimenting with VAE models (e.g. Pesteie et al, 2019) or even combinations of both (like Bao et al, 2017).

These models implicitly estimate the data manifold from the samples available, from which new images will be sampled. It is not clear whether new knowledge is truly obtained through this process. Intrinsically, using them to generate new images can be seen as a more advanced form of interpolating and extrapolating from known images. Therefore generative data augmentation cannot extrapolate to completely unseen categories (or parts thereof). In terms of internal representations, classes which are better modeled by disjoint clusters should have training samples representing each of them: generalization to a completely un-sampled cluster is impossible. As a result, these models are most likely to be useful when at least a basic pool of images covering all classes of interest is available.

While able to produce plausible example images, these generative methods are not fully reliable.

Output quality is difficult to automatically assess (due to lack of appropriate metrics, see Theis et al, 2016) and can remain quite irregular, resulting in a noisy dataset containing unrealistic images (in regards to the target application). Generally, synthesis quality is often highly sensitive to model hyper-parameters (and to the input noise vector in GANs). Thus it is prudent to check the overall quality and main modes of error (i.e. main cases of nonsense images generated) to eventually filter out unsuitable samples if necessary [1]. Some research works have carried out a sort of *visual Turing test*, in which a pool of images is randomly sampled from both the generator model and the original data, with human experts judging how realistic each sample is (for instance Baur et al, 2018). However, a manual validation procedure may be rather impractical and expensive in a production scenario.

Additionally, training these models also requires a substantial dataset. Many works achieve realistic generation by learning a class-agnostic data model (e.g. Baur et al, 2018), which implicitly increases the amount of training examples per class (since samples from all classes are combined). In this case, synthetic unlabeled data may be later incorporated into a semi-supervised strategy (section 5.3.2.1).

Class-aware generation may be more attainable with fine-grained datasets (e.g. Bao et al, 2017). Despite a relatively small sample for each class, images are more closely related among classes — since they all belong to a single "super-class" — making up a larger effective training set. Nevertheless, learning class-aware models over really small datasets remains a challenging task, subject to most of the same difficulties discussed throughout this chapter. Therefore it is no surprise to encounter works experimenting combinations of GAN and some of the

[1] It might be the case that such inadequate images do not compose a significant proportion within the artificial dataset thus not hindering overall classification accuracy significantly

strategies discussed here. To mention some examples:

- Soufi and Valdenegro-Toro (2019) use a generative model to translate related image data — symbolic traffic signs — into the target application domain — on-road sign recognition.
- If only a few classes are under-represented, it is possible to derive minority class examples by learning to transform from generic to specific images, as done by Koga et al (2018).
- Meta-learning: learn-to-learn a model that synthesize images from few examples, such as Clouâtre and Demers (2019); Hong et al (2020b,a) (see section 5.4.3.1 for more on meta-learning).

### 5.3.2 Unlabeled data

Labeling sometimes is the major bottleneck (due to time or human expertise availability constraints). In that case, if extra unlabeled data samples are available, there are multiple ways to profit from the information they carry. Evidently, unlabeled data can be used to generate synthetic data through some generative model, as discussed before. This section however focuses on more direct uses of unlabeled image sets.

Unlabeled data can be helpful in learning a more complete representation of the target dataset. They be included in the training process, side-to-side with labeled samples, in a semi-supervised learning algorithm, further discussed in section 5.3.2.1. In case labeling new samples is possible, efficient labeling is a concern of active learning, further discussed in section 5.3.2.3, which can also be associated with deep networks. If a detailed labeling is too costly, another alternative is to simplify the labeling task as much as possible, appealing to weaker forms of supervision (section 5.3.2.2. For instance, for a scene semantic segmentation task, if having a fully pixel-wise labeled image is too expensive, we can consider having only bounding boxes roughly circling the main objects of the scene.

#### 5.3.2.1 Semi-supervised learning

Many semi-supervised algorithms have been proposed over the years, although only a small number of them have been applied recently in the context of deep models, which are the focus of this chapter. For further information on general semi-supervised methods, the reader may refer, for instance, to the reference book by Chapelle et al (2006b).

Under a neural network framework, unlabeled data most often goes into calculating additional loss terms which enforce some *a priori* assumption about the data distribution (Chapelle et al, 2006a). First of all, it is implicitly assumed that these images bring novel information on the data distribution $p(x)$ that is relevant to the inference of the prediction function $p(x|y)$.

Otherwise, the additional information may not help or even hinder classification, introducing irrelevant or counter-productive biases into the learning process.

Besides this hypothesis, different types of semi-supervised loss terms derive from different assumptions. For instance, it can be assumed that the label-predicting function is smoother in high-density regions than in low-density regions, so that two close points in such a high-density region should have corresponding labels similarly close to each other. It implies that nearby points (say, in a cluster) should have close labels, which allows extending labels to nearby unlabeled examples. This hypothesis motivates strategies relying on weak-labeling or pseudo-labeling (see section 5.3.2.2), also serving as motivation for certain active learning methods (section 5.3.2.3).

Such notion of a homogeneous neighborhood motivates the clustering assumption: assuming nearby points tend to have similar labels implicates points in the same cluster are likely to belong to the same category. This also implies that the decision boundary between categories should lie in a low density region, as the opposite case implicates in a cluster which likely represents a single category. Multiple works have relied on this hypothesis, such as prototypical networks (Snell et al, 2017) which have been applied to semi-supervised few-shot learning scenarios (Boney and Ilin, 2017). Since classification on these models essentially relies on a nearest centroid classifier, it is evident that a coherent neighborhood — at the CNN feature space — is key for correct predictions.

Another related assumption is the manifold hypothesis, that is, that high-dimensional data tend to lie close to a low-dimensional manifold. This hypothesis being true, an algorithm can be designed to search for solutions lying close to one such manifold, reducing the effective dimensionality of the problem. This hypothesis has motivated semi-supervised learning works featuring low-dimensional manifold regularization prior to the revival of large neural networks (e.g. Belkin and Niyogi, 2004). A similar method has been used recently in the context of CNNs for image categorization, being easily extendable to incorporate unlabeled samples into the regularization term (Zhu et al, 2017b). This and other types of regularizations derived from a priori assumptions will be further discussed in section 5.4.1.

When labeled images are much fewer than unlabeled, a model is very likely to overfit to the small set of labeled samples if no care is taken. Thus in semi-supervised training it is important to dose the influence of the supervised loss term, which can be achieved by controlling the learning rate — increasing it during training according to some annealing function as done by Xie et al (2019) — or the balancing between supervised and unsupervised losses.

#### 5.3.2.2 Weak supervision

While gathering an expert labeled dataset may be expensive, it may be possible to gather lower quality labels for an existing pool of unlabeled images. Depending on the application

domain, multiple paths can be taken: having non-experts produce coarse or imprecise labels, gathering textual labels from the context in which the image was taken (say for web gathered data), using pre-trained classifiers (alone or in committee) to provide an automated pseudo-labeling.

As an example, let us discuss the case of automaticaly labeling web collected data — a method also known as *webly-supervision*. When categories of interest correspond to images which can be found on online search tools using the labels as queries, such images may be used to constitute an augmented dataset, or even an entire training set (which can be seen as a form of zero-shot learning). Methods for learning from automatically collected web images have been studied by several works across the years (e.g. Fergus et al, 2010; Chen and Gupta, 2015). Of course labels obtained in this fashion are not as trustworthy as careful expert manual annotations. However, if the signal-to-noise ratio is sufficient, this dataset may be a useful source of knowledge.

Using this kind of methodology, Kaur et al (2017) combined web-retrieved and manually curated data in order to improve food image classification. Similarly, Krause et al (2016) were able to achieve state of art accuracy on fine-grained classification benchmarks in four domains: dog breeds, bird species, Lepidoptera[2] species as well as aircraft models. High accuracy could already be reached using only web-collected noisy-labeled datasets, without any manually annotated training samples from the original benchmark datasets. As expected, combining both training sets yields even better performances.

Reasoning about the kinds of systematic label noise proper to the collection process can be useful to filter out part of the ambiguous or out-of-domain examples in a given sample. For instance Krause et al focused on reducing two types of noise: cross-domain — e.g. anything other than a bird retrieved with a bird query — and cross-category – e.g. a bird example incongruent with the query label used in retrieval. Their data analysis showed that, although difficult to remove without manual intervention, both types of noise could be managed. Overall, cross-domain noise was not too high (max 34.2%) and tended to reduce with the total quantity of retrieved images. With respect to cross-category noise, it could be reduced via a heuristic of eliminating ambiguous images — those appearing in more than one search result.

Of course this method is only applicable when the target domain is common enough to have a high number of images freely available on the web. Such a drawback makes this alternative highly infeasible in very specific applications, particularly within industry and healthcare.

[2]a biological taxonomic order including butterflies and moths

#### 5.3.2.3 Active learning

Much like semi-supervised strategies, active learning aims to make the most out of unlabeled data, although in a complementary fashion. While semi-supervised learning is interested in enriching model's knowledge on the data distribution, active learning is concerned with acquiring new knowledge on the label distribution in an efficient manner. This is then a strategy to be envisioned when a labeling expert has limited availability but can still contribute by labeling a few extra samples. Furthermore, it can be combined with semi-supervised labeling for possibly better results (see for instance Siméoni et al, 2019).

Due to the high level of expertise required in labeling, a common scenario for active learning is the medical imagery domain. A recent example is the work by Folmsbee et al (2018). This work is concerned with a particular health application — oral cavity cancer image-based diagnosis — which has intrinsically low quantities of data with a costly expert-dependent labeling process. Their pipeline of active learning has a specialist on the loop: it aims to semi-automate the labeling task by having an initial model provide guess labels for images. These weak labels are in turn qualitatively evaluated by the specialist who ranks them according to his perception of their accuracy. The 5 worst-classified get labeled and used as training set for another training session. This is repeated until either sufficient accuracy is achieved or no more unlabeled samples are left. Their results show this procedure — after 3 iterations — increases final accuracy by $\sim$3%.

The source of unlabeled examples may take multiple forms, from a simple pool of previously acquired data to an on-line stream of data. In either case, the main difference between different active learning proposals in general is the selection strategy applied over the unlabeled set at each active learning cycle (also referred to as *acquisition function*). Such a cycle alternates between model training and label acquisition, being repeated as long as needed or data is available (see Settles, 2010, for a review).

For instance, when using an *uncertainty sampling* strategy, after being trained over a (small) labeled dataset, model's confidence levels over its predictions are used to select samples for labeling, focusing on those with the least confident predictions. While being a popular class of acquisition functions, it depends on computing a reliable estimate of model uncertainty, which is not always straightforward to obtain for neural network models. Wang et al (2017a) have attempted using entropy of softmax predictions as an indicator of uncertainty, although it has been observed by other works (see for instance Papernot and McDaniel, 2018) that CNN softmax outputs can be very confidently wrong. An effective ensemble-based approach has been used by Beluch et al (2018), although being a computationally heavy option since it relies on training multiple CNN models. Less resource intensive selection schemes can be achieved by exploring different dropout paths inside the same network, as explored by Gal et al (2017).

Besides considering selection of individual examples, another concern in deep active

learning is to select an efficient query batch. In fact, most approaches have considered a traditional active learning scenario where a few samples are selected at every cycle, not being concerned with selecting sufficiently large sets forming a representative sample of model difficulties. In general, top-$k$ unreliable predictions are selected, regardless of them being very similar or representative of other unselected samples. Although it is a valid choice for selecting a few samples, better schemes can be imagined when selecting a larger query batch for labeling. Besides, since training CNNs is usually time demanding, requiring an expert to wait for each training cycle for only then labeling a few samples is impractical, time-inefficient and potentially costly. Some works exploring this issue are Ash et al (2019); Gissin and Shalev-Shwartz (2019); Kirsch et al (2019); Shui et al (2019); Yin et al (2017).

Despite all efforts into designing a good selection strategy, it is often the case that differences between different uncertainty selection functions is quite minor. Furthermore, experiments by Siméoni et al (2019) indicate these differences may be of low significance for image classification with CNNs, when compared to the gains obtained by unsupervised pre-training or semi-supervised learning based on label propagation (Iscen et al, 2019).

### 5.3.3 Related datasets

Datasets of similar nature, covering similar types of natural objects or scenes, can also be a knowledge source when the target dataset is too small. Such data may be used to help learning a useful feature representation space, in particular concerning lower level features, which have a larger chance of being common between related datasets. Besides related image data, other modalities can also be included during the representation learning process in order to reduce the dependence from image examples — an idea taken to its extreme in zero-shot learning (section 5.3.3.2).

#### 5.3.3.1 Pre-training and fine-tuning

In the image recognition domain, the popularization of CNN models and the ImageNet competition has lead to a situation where most mainstream models have open-access implementations (in one or multiple major deep learning frameworks) and have their trained weights available for download. Initially crafted for the 1000-classes prediction task, they started being reused for other tasks and datasets, with minor adaptations, leading to successful results (e.g. Razavian et al, 2014). This widespread re-use of supervised pre-trained models is one of the most common forms of transfer learning , where at least source and target tasks are different (since the source and target categories differ).

One way of profiting from a pre-trained model is to reuse the its learned weights, transferring at least those of the first layers of a network trained for a task $A$ to a network that will perform a task $B$. Learning task $B$ will train mostly the parameters of the final layers, maybe

fine-tuning the ones transferred from task $A$. This form of transfer learning can be seen as a way to start training on better initial conditions. A simpler form of knowledge transfer is to use the previously trained model as a feature extractor for a subsequent classifier trained on the target task, with the original weights not being adjusted at all during training (eg: OverFeat + SVM Razavian et al, 2014). In this case no feature learning takes place over the target dataset/task. Both methods work on the assumption that tasks $A$ and $B$ share some low-level features. This seems to be particularly true for visual tasks, where reusing part of a network trained with general image data has given surprising results in several domains, such as dermatology diagnostics(Esteva et al, 2017), lesion detection on CT scans (Shin et al, 2016) and industrial machinery diagnosis (Xiao et al, 2019), to cite a few.

**Unsupervised pre-training** On a similar rationale of re-using a previously learned feature representation space, it is also possible to learn a dedicated representation especially for the task at hand. Under restricted labeled data, unlabeled samples can eventually be used in this case. If the overall target dataset remains small, other related datasets may be necessary in order to have sufficient information to guide model training. Once the representation is learned, the model can be fine tuned or used as a feature extractor, in the same way supervisedly pre-trained models are used.

Generative models, like the ones discussed for data augmentation (section 5.3.1) can be used for representation learning, given enough unlabeled and/or related data. Unsupervised pre-training with autoencoders for instance, was commonly used in the early days of CNNs, but less so in more recent works (Goodfellow et al, 2016f). Considering big data scenarios, unsupervised pre-training — which ignores class information during early training — can induce an excessive bias in the representation learned by the CNN leading to underfitting in later supervised training. Under low-data regimes, however, this regularizing effect can be beneficial, since overfitting is a greater concern.

Some variants of unsupervised representation learning are referred to as self-supervised, in general when a proxy supervised task is designed such that supervision can be derived automatically from the task definition. For instance (Doersch et al, 2015), a proxy task can be to predict the spatial relationship between two image patches: is patch B above, below, left of or right of patch A? By building a patch cropping procedure that keeps this information, supervision is directly available from the data generation process itself. In this sense, the model is said to be self-supervised because even only unlabeled data is available, and supervision is implicit to the task the model is performing.

**Metric learning** Another form of learning a representation through a deep network is to use metric learning losses (see Kaya and Bilge, 2019, for a review). These models do not intend to model the data distribution directly. Instead, they target learning a representation in which distances get preserved. The exact type of implicit geometric prior depends on the choice of metric loss. A well-known example is Siamese networks, using contrastive loss

(e.g. Koch et al, 2015). This learning objective tries to minimize the distance between same class sample pairs, while maximizing distance under different class pairs. No higher order relationships (among more than 2 points) are modeled in this case. Note that in general, for a given set of samples, some form of supervision is necessary in order to decide between maximizing or minimizing the corresponding distance term in the loss function. Hence it is not a method capable of directly exploiting available unlabeled data, being more interesting to apply to related source datasets (in a transfer learning paradigm). After learning some form of geometric relationship among the input images, the learned mapping can be applied to source or target datasets as a feature extractor. This same geometric nature allows the use of a simple nearest-neighbors classifier, though more complex schemes can be envisioned.

**Fine-tuning weights and architecture** When considering neural network models, transfer learning usually comprises an architectural adaptation side, either because of a difference in the number of classes in the dataset, or because the prediction task is inherently different. In Zeiler and Fergus (2014), their CNN model was originally targeted at ImageNet, but they also tested it in other datasets by redesigning and retraining the final layers to match the new number of classes. In the case of a different task, changing might be more extreme. As an example let us consider semantic segmentation. In this task, where all pixels of an image have to be individually classified, it is common to reuse pre-trained architectures but changing the final fully connected layers to convolutional ones (Shelhamer et al, 2017). For such task, the multiple pooling layers have a major drawback of reducing resolution and limiting the precision of the boundaries arising from the final pixel-class prediction. With that in mind, different adaptations have been proposed such as skip-pooling connections (Shelhamer et al, 2017), dilated (à trous) convolution (Chen et al, 2016; Yu and Koltun, 2016) and even the use of a decoding architecture (Noh et al, 2016; Badrinarayanan et al, 2017).

Besides direct changes to the architecture, careful adjustment of hyperparameters for each layer is crucial to a successful fine-tuning. Experiments have observed more than doubled accuracy after proper calibration of learning rates (Dube et al, 2018). Particularly, having different learning rates for last and internal layers — or even a graduation following layer depth — can be significantly beneficial to final accuracy. The intuition is that upper levels of representation are more task specific thus needing more adjustment than lower ones. How far the transferred layers should be kept untouched or fine-tuned is related to multiple factors, including the size of the target dataset, as demonstrated by Soekhoe et al (2016). Their overall conclusion is: the smaller the target dataset is, the more rigid the weight transfer should be. Another important factor is the nature of the datasets and tasks involved and how similar they are between source and target domains. Fayek et al (2018) observed it to be more relevant than the choice of architecture regarding representation transferability between domains or tasks. Furthermore, this relationship is not necessarily symmetric: dataset A being relevant to dataset B has no implication on the relevance of B to A.

In general, fine-tuning a representation learned over different source data is a helpful form of transfer learning on small data. Obtaining good accuracy results is however not as straight-forward as it may seem at a first glance. While some intuitive relations between dataset/task similarity and strength of fine-tuning have been observed experimentally, there is no systematic method to measure the former and choose the latter accordingly. For each target task or dataset, only a case-based experimental analysis can guide decisions on how much to keep from the original learned parameters.

#### 5.3.3.2 Zero-shot learning

Zero-shot learning assumes a critical scenario where samples have to be classified into categories which may have presented no labeled examples during training. The original formulation of this problem assumes disjoint training and test label sets — i.e. none of the categories seen during training are present in test time. In practice however, using a training set as diverse as ImageNet ends up violating this assumption (since it may contain classes which overlap with those in the test set) (Xian et al, 2018). An alternative formulation, dubbed *generalized zero-shot learning*, assumes a broader test situation, where both seen and unseen categories can be predicted. This setting poses an additional difficulty since most zero-shot models tend to display a strong bias towards predicting seen training classes in detriment of the unseen test ones (Rahman et al, 2018). Naturally, performance of current models in this situation, which is closer to a real incremental learning, tends to be much lower than in simple zero-shot learning, pointing to the need for further research on this topic (Xian et al, 2018).

As previously discussed, the lack of information — in this case the complete absence of labeled training samples — has to be compensated with additional knowledge sources: some prior information on the unseen target classes has to be used. Notice that, even though no visual samples of test categories are available, zero-shot learning implicitly assumes that the set of possible test labels is known. In that sense, at least some textual information — the class label — can be used as a semantic target to gather information on the designated class. Additionally, unsupervised textual data on the subject can be compiled into a word vector representation, in order to leverage implicit knowledge from pre-trained word embeddings, such as word2vec (Mikolov et al, 2013). These word models are often used in zero-shot learning to help construct a mapping between category labels from seen and unseen classes. The implicit semantic relationships represented in these embeddings may be sufficiently aligned to distinctive visual features allowing to discern between classes. In this way, the word embedding allows a connection between training images and test labels, necessary for zero-shot prediction for unseen categories. Leveraging information from a semantic embedding requires some textual information to be available on all target categories (in order to link images to words), implying all possible categories need to known beforehand.

Overall, all zero-shot learning methods have to address this base problem of connecting

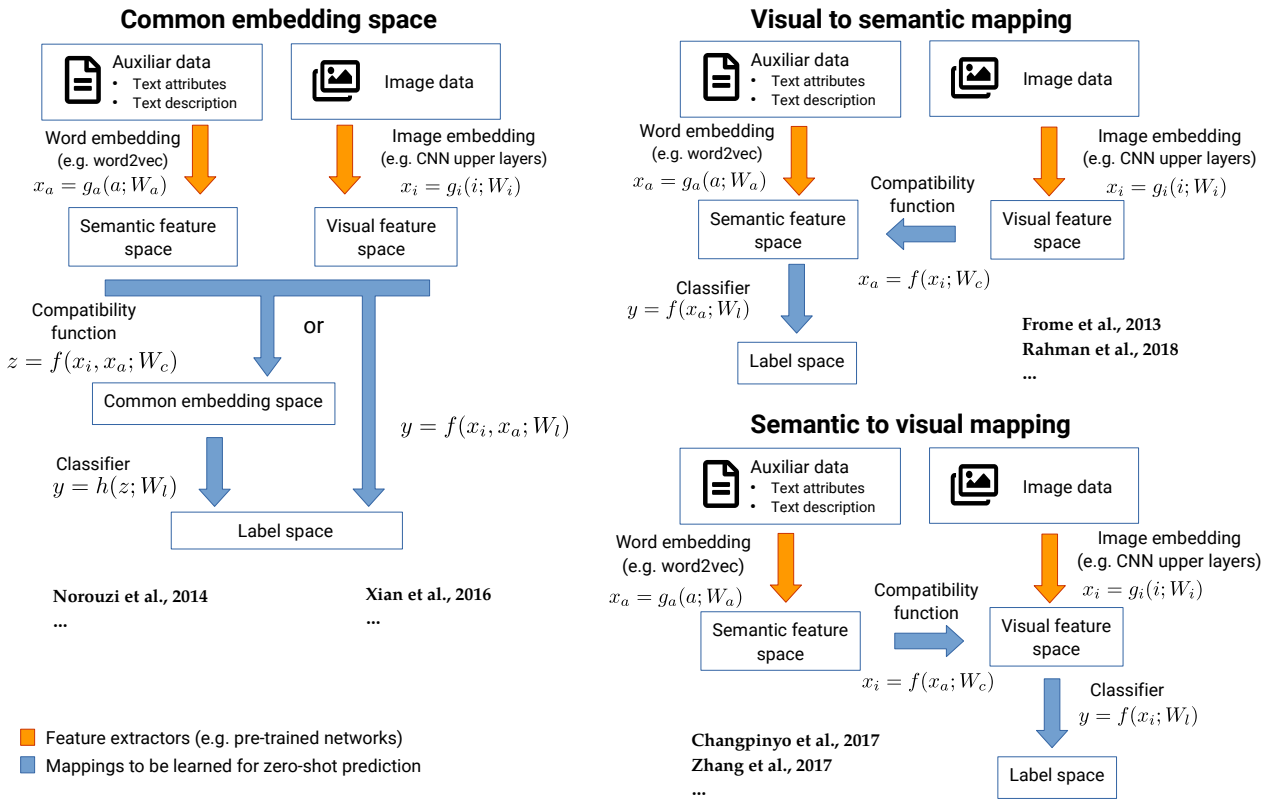



**Figure 5.4** Main architectural schemes for zero-shot learning.

observed and non-observed classes through some form of auxiliary information that is related to their visually distinguishing properties Xian et al (2018). This may be done by:

- joint learning of a visual-semantic class embedding;
- learning of a *compatibility function* mapping from visual to semantic space or
- mapping from semantic to visual space.

We schematized these methods in fig. 5.4, including some literature examples. For a more detailed review of zero-shot learning, refer to the survey by Xian et al (2018).

Being a very challenging task, accuracy in zero-shot learning is generally low, making it unsuitable for practical applications. Nonetheless the methodology brings cross-modal learning to the table, a path possibly worth exploring alongside direct learning over target image samples

## 5.4 Axis II: Reduce model complexity

Restricting the parameter space on large models, such as DNNs, is typically an important strategy to avoid overfitting, which is a major concern when dealing with small datasets. Conventional measures of complexity (e.g. Radamacher complexity or VC-dimension) score

high for such large models and thus predict low generalization capabilities, as well as extremely high bounds for the amount of training data necessary (for a given error rate). At the same time, the large parameter space of a deep model may be one reason behind its ability of learning complex patterns through SGD optimization. In fact, recent progress with large deep networks has also highlighted the fact that the relationship between their parameter count and effective model complexity is somewhat unclear (refer for instance to the survey by Jakubovitz et al, 2018).

Being highly over-parametrized, DNNs have proven capable of perfectly fitting training data, even on randomized labels (Zhang et al, 2016a). Despite this apparent over-fitting, they are capable of retaining good generalization capabilities on real test data. This behavior is counter-intuitive from a traditional statistical learning point of view, as it is not predicted by usual model complexity measures. It is hypothesized that there exists some form of complexity measure (and corresponding generalization bounds) that would score SGD-optimized DNNs low enough in order to explain their *perfect-fitting-yet-generalizing* capabilities (deriving such measure is the subject of current theoretical research around deep learning, see for instance Jakubovitz et al (2018); Neyshabur et al (2019)).

One heuristic explanation (Fan et al, 2019) is based on the limited power of SGD methods, which are used for network training. The intuition is that even though the space of possible functions is large, optimizing over real data has a tendency to explore only a sub-space of lower complexity functions, resulting in a low "effective complexity" model. It is as if the choice of optimization method itself acted as an implicit regularization mechanism. From this explanation, it stands out that controlling model complexity in deep networks is not simply connected to model size, but also to how the parameter-space is explored. Both aspects are discussed in the following sections.

### 5.4.1 Explicit regularization

Regularization methods are an integral part of machine learning and more importantly so under data restriction circumstances (Kukačka et al, 2017). Traditionally in machine learning literature, regularization is presented as a weighted penalty term added to the main loss function (usually related to the training error). These constraints impose limits to the model class being optimized, and are important for generalization, specially for convex optimization models.

Nonetheless, it has become mainstream to regard regularization as any modifications to the learning algorithm aimed at reducing generalization error, regardless of its influences to training error (as implied by the definitions of Goodfellow et al, 2016e; Kukačka et al, 2017). Under this broader perspective, traditional penalty-based methods are referred to as *explicit regularization*, while other more indirect methods are dubbed as *implicit regularization*. Hernández-García and König (2019) propose a more precise definition of this division,

considering explicit regularization as any methods designed specifically to constrain model capacity, without begin a fundamental part of the architecture. Any other methods having regularizing effects without bearing this intent fall into the implicit category.

This broader notion of regularization has opened space for multiple practices to be framed as forms of implicit regularization, leading to multiple intersections with other categories discussed in this review. For instance, referring to the survey by Kukačka et al (2017) on regularization, data augmentation and other data transformations as well as the choice of optimization methods, ensemble learning and other training strategies, are known for having an effective regularizing effect. Given the breadth of this umbrella term, this section is focused on explicit regularization forms. Relevant implicit methods are treated in their own specific sections.

**Regularizing weights** Weight $Lp$ -norm penalties are classical forms of explicit regularization which have been used in machine learning for years. In particular, smoothness-inducing $L2$-norm is one of the most commonly used. Less wide-spread in deep supervised models is the use of sparsity-inducing $L1$-norm Beyond $Lp$-norms, weight-decay (e.g. Simonyan and Zisserman, 2015; He et al, 2016a) and dropout variations (e.g. Srivastava et al, 2014; Wan et al, 2013) are the most common forms of explicit weight regularization. On one hand, weight decay is equivalent to L2 penalties (see for instance Simonyan and Zisserman, 2015; He et al, 2016a), also acting as a smoothness-inducing prior. On the other hand, dropout techniques have a more complex effect. These methods randomly set to zero a percentage of neuronal connections during each training iteration, preventing these synaptic weights from being updated during gradient back propagation. As a result, a sparsity-analogous effect is obtained: even though the overall weight matrix is not sparse, samples tend to be somewhat sparsely encoded, with fewer connections participating simultaneously for each single prediction.

In any case, explicit weight regularization methods often target a rather direct reduction of the effective number of parameters. As previously mentioned, this simplified link between parameter count and model complexity seems not to be a suitable theory for describing deep networks behavior. Other assumptions, not directly connected with reducing the number of parameters, could lead to designing more appropriate regularization mechanisms. Advances on this theoretical front may lead to better insights on if and how connections should (not) be constrained. Let us consider for instance the assumption that a well generalizing model should be robust to small input perturbations. By analogy with dynamic systems, such stability to inputs is characterized by matrices with low spectral norm. With this intuition in mind, Yoshida and Miyato (2017) have proposed this norm as a new form weight regularization. Their experiments on CIFAR-10 show smaller generalization gaps and higher test accuracy, although no results under smaller training (sub-)sets are provided.

**Regularizing internal representations** A general form of regularizing weights indirectly is to impose structural constraints to representations learned by particular layers. For

instance, the work by Zhu et al (2017b) has indeed analyzed how their proposed regularization influences generalization on small training sub-sets (of MNIST, CIFAR-10, SVHN). Using manifold regularization, they have observed improved generalization gap for decreasing training set sizes on all three datasets, when compared to weight decay or dropout regularization. This work is an example of regularizing internal representations so they meet pre-defined criteria, which in this case is to get data points to lie close to some low dimensional manifold. Coordinates on this ideal manifold are estimated as the CNN weight training progresses, in an alternate optimization loop[3]. In practice, regularization is implemented through a proximity term enforcing conformation of internal representation to the estimated manifold coordinates for each data point.

Also aiming for learning on small datasets, Belharbi et al (2017) propose a regularization that constrains hidden layers to learn class-wise invariant representations. Their regularization term is basically a sum of pairwise distances between points of the same class, for each class, thus enforcing same-class cluster to be formed. Similar functions involving pairwise distances — such as contrastive loss (Koch et al, 2015), N-pair (Sohn, 2016) or triplet loss (Schroff et al, 2015) — have been used for deep metric learning with CNNs in order to learn class-discriminant representations. On the same line, the regularization term proposed by Cheng et al (2018) also focuses on maximizing inter-class separation (and also minimizing intra-class scatter), with pairwise distances composed with a hinge function as regularizing loss term. In this work a discriminative-stacked autoencoder is trained in two stages — unsupervised metric learning then supervised with metric learning regularization — in order to obtain a classifier model.

Despite similarities, it is worth noting they target a slightly different task: the latter solves image set classification as opposed to its single image counterpart. Moreover, it applies the metric learning regularization on all hidden layers, while Belharbi et al (2017) have observed better results if the penalty is applied only on the latter layers. Notice however that they focus on similar pairs to enforce intra-class similarity , relying solely on the classification cross-entropy loss to learn inter-class distance. This latter loss term tends to have a greater influence on later layers and less so on earlier ones. Alternatively, Cheng et al (2018), also include dissimilar pairs in the metric learning loss, bringing both influences to all layers, which could explain the usefulness of their term on earlier layers.

Experiments by Belharbi et al (2017) have demonstrated significant improvements on reduced training subsets, when regularization is applied to the last hidden layer, although tests were only performed on MNIST data. On their turn, Cheng et al (2018) obtained successful results on four face recognition datasets as well as on the ETH-80 object recognition dataset. Overall, metric-based regularization seems promising, particularly considering the success of similar metric learning loss functions applied to fine-grained classification (e.g. Movshovitz-

[3] more specifically using alternating direction method of multipliers (ADMM) (Boyd et al, 2011)

Attias et al, 2017) and face recognition (Schroff et al, 2015).

**Regularizing network's output** Some output regularization methods are particularly useful in conjunction with data augmentation or semi-supervised learning (Berthelot et al, 2019). In fact unsupervised loss terms in semi-supervised settings (previously discussed in section 5.3.2.1) are also a form of regularizing networks output, as well as metric learning losses. Let us consider the example of consistency regularization. In this case the objective is to enforce an example and its augmented variants to have all the same label, with an unsupervised loss. This type of penalty may also intervene in any case of randomized treatment of the original samples, such as randomized transformations or different dropout trials. An example is the *transformation-stability* loss term proposed by Sajjadi et al (2016), which was applied to a few typical image benchmark datasets, including ImageNet. They simulated a small labeled sample setting by training with only a portion of available labels and could reduce the error rate for all the datasets tested.

## 5.4.2 Architecture adaptation

Architectural choices can have an important impact on the learning capability of the model and thus can be a lever for enhancing learning from small datasets. On one hand, simple architectural changes may directly result in reducing the total number of parameters. On the other hand, more structured adaptations may relocate computational resources in a manner more appropriate to the target task at hand. In any case, structure can be introduced both within and between layers as will be discussed bellow. Such modifications can be inspired from domain-specific expertise, after a thorough analysis of the functional elements involved in the accomplishment of a particular target task.

### 5.4.2.1 Intra-layer structure

At a lower level, structuring weights inside layers can be a means to reduce the total number of parameters and model complexity. A classical example is the weights of convolutional layers. As discussed in section 2.2.1, convolutional layers act as locally connected layers with shared weights, in comparison to fully connected deep architectures. These two characteristics — local connections and weight sharing — effectively reduce the number of parameters and can be considered in other applications that could benefit from the translation equivalence intrinsic to this connectivity pattern. Analogously, domain specificity may also motivate the design of particular layers and connectivity schemes which could be helpful in achieving better solutions for a data-restrained task.

Instead of assuming scalar values, the entries of a weight matrix can follow a family of suitable functions that will enforce certain invariances (e.g., weights as a function of an angle or a scale-factor in relation with rotation or zoom invariance) and symmetries (e.g.

isotropic diffusion, or linear combination of horizontal and vertical diffusion) specific to the application domain. It is the case for Gabor CNNs (Luan et al, 2018) which modulates kernel matrices with random pre-defined Gabor filters, in an aim to improve invariance to scale and orientation. Other examples of achitectural adaptations building rotation invariance are the transform-invariant-pooling layer (Laptev et al, 2016) — which focuses on the response of the main orientation within the feature map — and also Oriented Response Networks (ORN) (Follmann and Bottger, 2018) — which rotate kernels in multiple directions before convolutions. These models can achieve performances comparable or superior to state-of-the-art models, using a smaller number of parameters.

Another track is (by analogy with the kernel trick) to consider linear combination of predefined local operators, each operator being known as efficient to extract some cue. This includes also non-linear combination, as for instance, considering normalized gain control. The idea here would be to go toward something more generic thanks to these combinations and consequently less "expensive". An example is to constrain kernels to be spatially separable, as done in (Sironi et al, 2015), which reduces the number of free parameters. This technique has been applied to modern CNNs such as Xception (Chollet, 2016) and MobileNet (Howard et al, 2017).

Some works have tried to reduce the number of synaptic weights by iterating a network pruning step and a re-training step. This idea of optimizing network connections *a posteriori* is not new (LeCun et al, 1990b; Hassibi et al, 1993), but just recently has been applied to deep architectures. For instance, in the work of Han et al (2015), the first training pass aims at getting a coarse connectivity pattern, by thresholding out weak connections. Then they re-train to actually learn weights, iterating between pruning and re-training.

While the strategies mentioned in this section can reduce model complexity, their helpfulness under limited data has not yet been experimentally explored.

Eliminating weights or simplifying the connectivity pattern explicitly reduces the space of learnable hypothesis, and may have unexpected qualitative effects on the results of SGD optimization. Nonetheless they are expected to be helpful whenever the targeted invariances are relevant to the task at hand.

#### 5.4.2.2 Inter-layer structure

Besides introducing structure into the convolutional layers, layers and layer groups can be designed to a particular function, with dedicated modular inter-connections. Moreover, shortcuts or recurrent connections may lead to an eased optimization ( refer to sections 3.3.2 and 4.3, where several such examples have been discussed). We recall here some previously cited examples:

- Feedback connections can implement an iterative attention refinement mechanism as done by Fu et al (2017) for fine grained bird species recognition;

- Multiple glimpses of an object can be integrated with a dedicated attentional architecture (Ba et al, 2015);
- Adding feedback connections to standard models can improve fine-grained classification (Zamir et al, 2017).

Again, under limited data, over-complexifying the model should be avoided, and only resorted to if simpler architectures do not meet performance requirements. In that case, adaptations shall always be motivated by intrinsic properties of the domain or task at hand.

### 5.4.3 Training strategies

Training strategies may not explicitly reduce the number of parameters, but may alleviate the optimization burden while incorporating knowledge from different sources. This section covers such techniques which, while not acting directly on network architecture, can reduce the complexity of finding a good model within a given model class.

#### 5.4.3.1 Meta-learning: learning to learn

Meta-learning refers to techniques which try to accumulate "experience" from solving multiple source tasks in order to more effectively tackle a given target task. In other words, the designed system would be learning to learn, during a meta-training phase, in order to be a better learner on a specific task — during a meta-testing phase. In a broad sense, any method accumulating information from accessory tasks and data can be placed under this category, which could include for instance most techniques discussed in section 5.3. (see Vanschoren, 2018, for a review).

Regarding CNNs and image classification, different strategies of meta-learning have been applied. Beyond generically learning to learn, meta-learning methods may have several focuses, as identified by Shu et al (2018). For instance Finn et al (2017) method learns to learn a good initial state for further fine-tuning, in what can be seen as *learning to transfer learn*. Alternatively, some methods will meta-learn (part of) the optimization algorithm, in what can be seen as *learning to optimize.* An example is the work by Ravi and Larochelle (2017) which uses an LSTM meta-learner to learn how to provide a good learning rate sequence for network training. Proposals in this field have multiplied over the past few years, making it premature to define a strong taxonomy of meta-deep-learning approaches (refer to the survey of Vanschoren (2018) for a specific overview). Considering the intended scope of this review, the focus hereafter will be on one-shot and few-shot learning applications of meta-learning.

**One and Few-shot learning** In the recent context of deep networks, these methods are often applied in the context of limited data, to solving classifications tasks with only a few samples per class: few-shot and one-shot learning. Few-shot learning refers to classification

tasks where a model is to learn to discriminate between $N$ unseen classes given $k$ examples of each, with $k$ usually below 5 (Triantafillou et al, 2017). One-shot learning corresponds to the specific case where $k = 1$. This class of tasks is usually referred to in the form *$k$-shot $N$-way* tasks.

Most works on this track use similar data during both meta-training and meta-testing phases, with disjoint class sets (e.g. Vinyals et al, 2016; Ravi and Larochelle, 2017; Snell et al, 2017; Finn et al, 2017, to name a few). During meta-training, the classifier model is trained sequentially on similar *$k$-shot $N$-way* tasks, each time on a different set of $N$ classes. After accumulating experience in discriminating between $N$ classes with only $k$ samples for each, the model is finally trained on the target set of $N$ classes. This organization of the meta-training process is referred to as *episodic training* (first presented by Vinyals et al, 2016).

Applying this framework in practice thus requires such a similar (labeled) dataset, which may not be a realistic assumption in certain domains. Indeed the development of few-shot learning has mainly turned around benchmark datasets — mainly Omniglot (multi-alphabet character recognition) and mini-Imagenet ( subset of classes from main ImageNet) — which have a large quantity of images in total due to a very large number of classes. Therefore, by using all non-target classes during meta-training, learning ends being up still dependent on a significant amount of labeled data. As with any transfer learning approach, the closer the source data distribution is to the target data distribution, the more helpful the knowledge accumulated from source data should be. Again, in practice, available data may not be as similar as it is the case with these benchmarks.

Nevertheless, some works have successfully applied meta-learning strategies to very specific domains. For instance, Prabhu et al (2018) tackle a complicated problem of dermatological diagnosis. The class distribution in this task has challenging characteristics intrinsic to the data generation process. Firstly, it is heavily imbalanced due to some skin conditions being much more prevalent than others. Secondly, each class sample is highly heterogeneous, as a single diagnostic category may present itself under several visually different skin conditions. Starting from plain prototypical networks — a well known method in the few-shot learning community by Snell et al (2017) — they propose an adapted version that accommodates for the multi-modal nature of these classes. Out of 200 different classes, the largest 150 base classes are used as meta-training data, while the 50 remaining are targeted during meta-testing — as if they were "novel" classes. They could achieve an average precision around 30% on 5-shot and 50% on 10-shot tasks — the best results they achieved with pre-trained and fine-tuned networks stay around 20% and 40% , respectively.

#### 5.4.3.2 Curriculum learning

Considering the human ability to learn, it seems natural to carry learning from simpler to more complex concepts, building-up connections amongst them or defining derived notions

from such concepts. Indeed not only humans but also many animal species have been observed to learn better when examples are not randomly presented but organized in a meaningful order, focusing on the learner's zone of proximal development — presented in educational research as learning objectives attainable with some expert guidance (see for instance Matusov, 2001; Walker, 2010).

Designing such organization or curriculum for human learning is traditionally an object of study for educational research and cognitive sciences. Nevertheless, this concept has inspired works exploring curriculum learning in the context of machine learning (Bengio et al, 2009), including for deep networks(Hacohen and Weinshall, 2019). Translated to the machine learning context, the goal of curriculum learning is to propose an order on training samples according to their estimated difficulty so as to achieve better generalization and/or faster convergence during training (Hacohen and Weinshall, 2019). Indirectly the method is analogous to numerical continuation methods (Gulcehre et al, 2016), in the sense that it will start with a simpler model — due to the focus on easier parts of the data being modeled — that will slowly be deformed into a more complex one — as the more difficult data points are joined into the mini-batches.

In general, a curriculum learning policy requires the definition of two functions. First, a *scoring function* that estimates the difficulty level of a sample. Second, a *pacing function* that defines which and how many samples will be considered at each training iteration. In the context of SGD-based training, it resolves to a function sampling and proposing a sequence of mini-batches. Different proposals vary mainly on choices made in the design of these two functions.

For instance, Weinshall et al (2018) have obtained a scoring function via transfer learning. After training an SVM classifier on top of pre-trained CNN features, its margin-related prediction confidence was used as a difficulty score. Pacing was determined by a step-based increasing policy, either with fixed size or adaptive steps (considering the current training loss value). This step policy was responsible for increasing the probability of harder examples being chosen for a mini-batch as training progresses. Hacohen and Weinshall (2019) extend these experiments, additionally exploring a self-taught scoring function and different exponential decay pacing functions.

**Related methods** The core ideas of curriculum learning have inspired other related techniques such as curriculum dropout — an adaptation of the usual regularization technique that dynamically increases dropout rate during training (Morerio et al, 2017). The supporting rationale is that feature co-adaptation — which dropout aims to prevent — is unlikely to occur early during training. Instead, dropout would be needlessly complicating the task way too early. With curriculum dropout, this difficulty is gradually increased. In practice, small test accuracy improvements (at most +1%) have been observed across most image benchmarks considered by Morerio et al.

The general principle of increasing difficulties can also be applied to the decision procedure of a classifier, when a class hierarchy is available. One such example is the recurrent model by Zamir et al (2017) (discussed in chapter 4), which refines its predictions iteratively after each recurrence step. Since a valid output is provided at every iteration, it is possible to change target labels at each iteration, gradually increasing their specificity within the class hierarchy. Guiding learning from coarser to finer-grained targets is a means of enforcing a curriculum by leading the model to make easy-to-hard decisions (for instance, first decide for a vehicle, then for a bike, then for tandem bike).

Overall, curriculum learning has experimentally demonstrated being capable of speeding up convergence during early training and, on more difficult tasks (eg. finer grained distinctions), achieving local minima which improve generalization (Weinshall et al, 2018; Hacohen and Weinshall, 2019). On a simpler setting — that of a convex linear model — Weinshall et al have theoretically demonstrated this convergence property, which stems from the expected variance of gradients across samples with increasing difficulty. The higher the sample difficulty, the higher the gradient variance on that point, increasing the chances of steering off the local optima leading path. There may also be a link to numerical continuation, a method that can also lead to a generalization boost. Nevertheless, such improvements may only be noticed on harder tasks, in which class distinguishing features are not so clear. This may also be the case when learning over small datasets, although this perspective is yet to be experimentally validated.

#### 5.4.3.3 Multi-task learning

The general idea of multi-task learning is to leverage domain-specific information from related training tasks in order to improve generalization on the target task. While it remains unclear what defines a *related task* (Ruder, 2017a, reviews some definitions), in the context of DCNNs it is intuitive to think of tasks which share internal representation levels. Thus it is straightforward to think of multiple tasks sharing early layer parameters. Indeed, typical multi-task strategies applied to deep models involve some form of shared representation learning, either by hard parameter sharing (wired within the architecture) or constraint-enforced soft parameter sharing. A down-side of these approaches is that, besides using some domain-specific intuitions, there is no principled way of deciding which levels of representations to share between the two tasks. A less heavy bias is possible in soft sharing, particularly with architectures which dedicate resources (i.e. specific layers with learnable parameters) to *learn what to share*.

**Using auxiliary tasks** Beyond explicitly sharing features, multi-tasking can intervene directly as a training strategy, by the means of relevant auxiliary tasks. Again, what is considered "relevant" may vary widely among different domains. A series of examples are raised

by Ruder (2017a), of which a few are worth mentioning hereafter. When parts of the input are more difficult to be learned, an auxiliary task can be to focus specifically on these parts. It was the case of semantic segmentation for autonomous driving in the work of Caruana (1997) in which an extra task detecting only road lanes (which form a small portion of the typical images) helped them being more detectable by the main task. In cases where there are extra features (e.g. domain-related meta-data) which are not being used as input, an auxiliary task can be to predict these features from the input image.

When predicting fine-grained categories, an auxiliary task of making coarser predictions on super-classes can help guide the learned representation towards a more class-related structure. It can happen for instance in medical diagnosis, if the different conditions being predicted can be grouped into coarser groups according to a 3-stage risk scale (e.g. low, medium or high). Again, focusing on the representation structure, auxiliary tasks of autoencoding or metric learning can also be applied.

After all, the adequate auxiliary task is highly dependent on the available side-data and the domain of application. Reflecting on how the target task can be decomposed and which are its main difficulties can help identify points which could benefit from this extra bias. Another remaining design choice is how to balance between target and auxiliary tasks. Besides weighing between the two, one might imagine to dynamically increase the influence of the latter across training epochs, aiming at enforcing the introduced bias to avoid overfitting during late training.

## 5.5 Conclusion

Deep learning with small data samples is a complex task, that can be made possible if additional sources of knowledge — including domain specific expertise — get incorporated into the learning process. On one hand, novel information may come from incorporating more data into the mix. This includes data augmentation, use of unlabeled data, other related datasets or even cross-domain data. On the other hand, knowledge can impact choices that effectively reduce model complexity — either by regularizing its capacity or by guiding training process through particular strategies.

Overall, there are multiple ways of reducing the complexity of deep learning with small data samples. However, each of them has a limited impact and has its own drawbacks and difficulties. Besides, the impact of combined measures can sometimes be deleterious and has not been extensively studied. Without a universal solution, each target application needs to be analyzed on its strengths and difficulties, compared to other tasks studied in literature, in order to identify promising strategies.

Chapter 6

# Analysis of DCNN applied to small sample learning using data prototypes

*Note:* This chapter has been published as a journal paper:

Drumond et al, 2019. "Bio-inspired analysis of deep learning on not-so-big data using data-prototypes". *Frontiers in Computational Neuroscience*, 12, 100. ISSN 1662-5188. DOI: 10.3389/fncom.2018.00100.

## 6.1 Introduction

Deep neural networks have had a great success in many domains, especially in visual recognition tasks. From AlexNet in 2012 (Krizhevsky et al, 2012) to Residual Networks in 2015 (He et al, 2016a), this class of models has shown outstanding performances at the Large Scale Image Recognition Challenge (aka ImageNet), motivating the use of deep learning in multiple domains such as speech recognition and language processing (LeCun et al, 2015). The key idea is that, at least for threshold units with positive weights, reducing the number of layers induces an exponential complexity increase for the same input/output function (Håstad and Goldmann, 1991). On the reverse, it is a reasonable assumption, numerically verified, that increasing the number of layers yields an input/output function compact representation[1], as a hierarchical composition of local functions (Goodfellow et al, 2016c).

### 6.1.1 The data requirement challenge

However, deep learning remains very inefficient concerning data requirements. Most successful results are reported on large databases. For the simple handwritten digit recognition task, on the MNIST database we were already at 60K images for 10 classes (Lecun et al, 1998). For the complex ImageNet challenge, we reached more than 1M images for 1K classes (Krizhevsky et al, 2012). While large tech companies may have access to large amounts of labeled and unlabeled data, this is not the case in many domains, where the ability to

[1]Here compact means that adding a network layer allows a better input-output mapping approximation or a mapping considering a higher functional sub-space for a number of units which would have been exponentially higher with decreasing the number of layers.

generalize from only a few tenths of examples per class is required. This limitation not only concerns industrial application requirements but is also a strong limitation as far as modeling cognitive processes is concerned.

Elaborating over this necessity, Mouret (2016) argues for the three basic precepts to deal with small corpus of data, also called micro-data learning: (i) actively search about which is the most relevant data to consider (active learning), (ii) exploit every bit of information (detailed learning), (iii) use as much as possible prior knowledge. In fact, the quick learning ability of humans and animals is largely due to our prior knowledge about the world we interact with and is referred to as transfer learning (Weiss et al, 2016). This can be performed considering general knowledge, beyond ad-hoc application dependent choices. In practice, learning from limited data often requires embedding of prior knowledge at some stage, be it at the input data pre-processing, the architecture choice, cost functions and regularization rules, or even as smart training strategies. Following this track are works focused in few-shot learning (Dong et al, 2017; Rahman et al, 2018), which deals with very few learning samples (usually less than ten), when not only one sample, or zero (i.e., only a priori information).

We would like to study how to investigate a middle range of a few tenths of samples. We may call this situation "not-so-big-data". The rational for this is two-fold: On one hand it appears that learning entirely new concepts of forms in human (e.g., reading characters (Leroy, 1967)) does not require one or a very few number of samples, but more than ten. On the other hand, several industrial applications are able to provide ten to hundred samples for a given perceptual tasks, but not thousands. Such not-so-big-data learning is an extension of few-shot and micro-data learning, reusing several key features such as transfer learning, as reviewed in the section 6.2.2.

### 6.1.2 The interpretability issue

A step further, we argue that in order to be able to introduce prior knowledge at a general level (and not only using an ad-hoc mechanism limited to a single application), the lever is to provide an interpretable processing. Interpretable mechanisms can help to better tune the architecture with small data set, as discussed in this paper. Interpretability also requires a data description easy to explain to a lay audience, and means to provide an explanation for algorithmic decisions that may affect citizen life. We have such requirement in mind in this contribution, as already addressed in Drumond et al (2017b).

When considering such an issue in the present literature (e.g., in Kim et al (2016), or Zhao and Park (2016) it appears that interpretability is understood as "results are interpretable". However, if we really want to propose some easy to share and explainable mechanism, in order to be able to state that the process is transparent, we must not only look for interpretable results. We also must attempt to consider easily understood processes, i.e. interpretable mechanisms. This is the reason why in this paper we are going to take the risk to study to which extent

a "non trivial but easy to explain" mechanism, used with a small amount of data, can still provide performances close to the best state of the art performances.

### 6.1.3 On network architecture

One of the intuitions behind the success of deep learning is that stacking multiple layers induces learning of a hierarchical representation, a successive composition of local functions that ends up describing higher level concepts as reviewed in e.g., (Bengio and LeCun, 2007). This includes, for convolutional neural networks (CNN) with weight sharing, the capability to take global, e.g., translational invariant, features into account (Goodfellow et al, 2016b). That aspect of deep architectures has a strong parallel with our current knowledge of representations in the brain, especially in the visual pathway from retina until the infero-temporal cortex, where object identification is performed. This feed-forward pathway is indeed hierarchical and is responsible for our primary (first $\approx 100\,ms$) visual identification capabilities (Fabre-Thorpe et al, 1998). When considering, however, the complete visual system, a simple feed-forward pipeline cannot represent all the computations taking place in the brain. Identification is only part of our visual system, namely the What ventral pathway, associated to the Where-How dorsal pathway (Milner and Goodale, 1995). Furthermore, there are dynamic mechanisms, feedback modulations, shortcuts, lateral inhibition, to cite a few, that are often disregarded (Medathati et al, 2016).

Concerning the connectivity patterns within the visual system (Bullier, 2001), the capability to mix features from different levels of the hierarchy, as observed regarding the role of the pulvinar (Cortes, 2012), has to be considered. Integrating focus of attention within certain regions of the feature space (Saalmann et al, 2012), and other functionalities such as on-the-fly adaptation to incoming data, in link with the few-short learning paradigm (further discussed in section 6.2.2) is also important to take into account[2]. All these functionalities require rather general recurrent architectures (Medathati et al, 2016). The possibility to provide a framework in which not only feed-forward or specific recurrent architectures can be taken into account is addressed by other works (Viéville et al, 2017) and is beyond the scope of this paper. The present proposal is mainly inspired on how the infero-temporal cortex has a kind of prototypical representation, with neurons tuned to respond to particular categories (Viéville and Crahay, 2004b).

[2]Further aspects of feed-back functionality, such as on-line computations with feed-backs, will not be considered here.

### 6.1.4 Hyperparameter dependence

Automated machine learning[3] is "the process of automating the end-to-end process of machine learning", because the complexity of adjusting the hyperparameters, including selecting a suitable method, becomes easily time-consuming when not intractable. To this end, the general idea is to consider a standard machine-learning algorithm and to add on "top of it" another algorithmic mechanism, e.g., another machine learning algorithm dedicated to automatic hyperparameter adjustment, with the caveat of generating other hyperparameters for the meta-learning algorithm and without formal guaranty that this accumulation of mechanisms is optimal.

What corresponds to hyperparameters in the brain or for a fully autonomous system is managed in three ways: It is either modulated by other structures (such as the different cortico-thalamic pathways through the basal ganglia (McHaffie et al, 2005)) and at different time-scales (such as short-term versus long-term synaptic plasticity (Cooper et al, 2004)), or integrated as learning parameters, the system being able to learn new parameters and to adapt also the way it learns as a coherent process. A third track is to design robust processes in the sense that hyperparameters precise values are not critical. A key issue in a distributed system such as the brain, is that hyperparameters are usually global while still implemented via local processes (e.g., neuro-modulation). Some are also related to the network construction (e.g., the number of layers).

Here we are going to carefully study these aspects, in order to make explicit to which extents the proposed method is related to robust hyperparameters, thus without requirement about questionable value adjustment. If not, the hyperparameter adjustment is going to be managed using standard hyperparameter adjustment methods, and the interpretability of the such meta-parameter adjustment is also going to be made explicit.

### 6.1.5 The present contribution

Bio-inspired improvements of deep learning extend far beyond the three issues reviewed here, as discussed recently in Marcus (2018). The biological plausibility of deep-learning is another issue, see Bengio et al (2016) for a recent discussion. In a nutshell, there is a clear correlation between deep-learning CNN states and the primary visual cortex macroscopic activity as observed in Mensch et al (2017), while the contribution of deep-learning as a tool to better understand the brain is not obvious as explained in Medathati et al (2016). Nevertheless these issues are beyond the scope of this paper.

Here we would like to revisit the notion of prototypes, as a tool to improve deep learning on not-so-big data, considering some aspects of the brain architecture and addressing also

[3]The notions of "automated-machine-learning", "meta-learning", including "meta-optimization" and "hyperparameter" and the related "hyperparameter optimization", have precise meaning in machine-learning, and we assume this is known to the reader.

the hyperparameter dependence issue. There are multiple definitions and usages of what is called prototypes. Here prototypes are centers of a k-means procedure over training samples, acting as data-representatives in the feature space. The present work explores this notion of prototypes for interpretability and classification under not-so-big datasets, building up on the idea introduced in Drumond et al (2017b) with more comprehensive experiments on real datasets and including comparison with few-shot meta-learning methodologies. This work focuses on convolutional networks and applications to vision, although what is proposed in this paper is general and applicable to other deep learning frameworks.

Our main claim is that such data prototypes may help address the not-so-big data issue in an interpretable fashion, considering a few bio-inspired aspects made explicit in this introduction. Proposing a solution to both aforementioned issues — "not-so-big-data" learning together with data and process interpretability — is the main contribution of the paper. The key point of our proposal is the notion of data prototypes, as made explicit in the sequel. In order to introduce this notion, we need to discuss one aspect of network architecture and point out issues regarding the hyperparameters.

In the next section we are going to review the literature related to this subject, then formalize the model description, in order to experiment the idea in the subsequent section, allowing us to discuss based on these results how the issues quoted here can be addressed, while proposing several perspectives of this work.

## 6.2 Related works

### 6.2.1 Prototypes in literature

The term prototypes can be found in the literature under different meanings: a priori information, representation in clusters, quantification of space, as we discuss now.

Jetley et al (2015) proposes a prototypical priors layer, with a priori chosen prototype images of road signs, encoded using a HoG (histogram of oriented gradients) descriptor, and added as fixed units of the penultimate layer. Here, the network is ultimately trained to match the HoG representation of the prototypes but it is assumed to exist a standard representative image per class in the input space to be encoded as a prototype unit.

In prototypical networks (Snell et al, 2017), prototypes are defined as class centroids in the feature space spanned by the embedding CNN. In this simple approach, the nearest-prototype is used to classify a given sample. This proved to be effective on the considered datasets. However, this does not respond to the scenario of metric learning proposed in (Song et al, 2017). Also derived from this work, Gaussian prototypical networks (Fort, 2017) predict a covariance radius for each prototype, yielding some insight on the discriminating force of each of them.

Self-organizing maps, or the dictionary sparse representation methods (Rubinstein et al, 2010), perform a prototypical sampling on the data space (Hecht and Gepperth, 2016) and allow to represent what is known about a data distribution, using methods quoted in prototypical networks and known as optimizing statistical criteria (Banerjee et al, 2005).

Bio-inspired models also consider the notion of prototypes, as in, e.g., (Serre et al, 2007) which introduce a general framework for the recognition of complex visual scenes, that follows the organization of visual cortex building an increasingly complex and invariant feature representation and considering a redundant dictionary of features for object categorization. Furthermore, (Viéville and Crahay, 2004b) has related the notion of prototypes to a SVM model of the inferior temporal object recognition brain area.

### 6.2.2 Few-shot learning and learning to learn

The importance of learning new concepts from small data-sets motivated the study of few-shot learning tasks. Few-shot learning refers to the ability of learning to discriminate between $N$ unseen classes given $k$ examples of each, with $k$ usually below 5 (Triantafillou et al, 2017). This task is also referred to as k-shot N-way learning. One-shot learning is a special case of this setting, where the system must generalize from a single example for each class. Slightly different is zero-shot learning, where no example of a given class is available in the target domain (e.g., images) whereas information on the categories originates from other domains (e.g., textual descriptions) (Goodfellow et al, 2016f).

Learning a deep discriminative model for a large number of classes has high data requirement, and is prone to overfitting if applied directly in a few-shot data framework. For this reason, usually some form of transfer learning is used to tackle this problem: a model is trained on other classes before attacking the $N$ new ones. Since the model learned in this pre-training phase will have to be adapted to the target classes, few-shot classification can also be seen as a form of "learning how to learn", which is a very sensible framework when the goal is to learn whichever new categories to come. In this spirit, pre-training is a meta-training phase, where we learn a meta-model that can learn any set of $N$ classes given to it. Using the model on new k-shot N-way tasks corresponds to a meta-testing phase, where for each task some inner training may take place.

There are different ways of implementing this meta-learning setting [4], but any such should have a meta-training and meta-testing phase, with their respective class-wise disjoint datasets $D_{Mtrain}$ and $D_{Mtest}$. Details on each phase vary between propositions, but globally meta-training works like some sort of pre-training, where the model is trained on some discriminative task over the classes in $D_{Mtrain}$, while constructing a representation that will be useful to generalize to new classes in $D_{Mtest}$. In this sense, this methodology can still be seen as a form

[4] not to be confused with meta-learning in the sense of hyperparameter tuning and automatic machine learning

of transfer learning. Later, meta-testing phase will consist of actually performing the k-shot N-way task multiple times, for different subsets of $N$ classes drawn from $D_{Mtest}$. Each subset itself has to be divided into training and test splits, in order to have $kN$ reference samples for the new classes from one and evaluate performance on the other.

To give a clearer example, let us present a common organization of meta-learning: episodic training. First proposed by Vinyals et al (2016), it has continued to be adopted in other recent works (Santoro et al, 2016; Snell et al, 2017; Ravi and Larochelle, 2017; Fort, 2017; Ren et al, 2018). Arguing that approximating meta-training to meta-testing conditions could enhance learning, they proposed that, during each step in the meta-training phase, the model be trained on a different k-shot N-way task, with new $N$ classes drawn from $D_{Mtrain}$. Each such task is called an episode, and demands not only $kN$ examples as a training (or support) set but also some examples of the $N$ classes to serve as a test or query set. During this phase, the error over this query set can be used to adjust the model, while in meta-testing only the support set will be used as a reference to classify the query samples.

In line with the above discussion, many of the recent works focused on this problem, seek to learn an embedding space over the meta-training set that will hopefully generalize for unseen classes in meta-testing. This common feature space allows to compare test with training examples to decide on their class, usually with some kind of nearest neighbors algorithm, but sometimes resorting to more complicated models. Siamese nets (Koch et al, 2015) are an early example of such models, in which two identical networks are used to map a pair of examples into a learned metric space so that they can be compared via a distance function. The whole network is trained to predict whether an input pair belongs or not to the same class, this being repeated for multiple pairs. Matching networks (Vinyals et al, 2016) work on a one-shot setting, also relying on learning a network that will map support examples of each class to an embedding space, to be later matched to the query example. The embedding is composed by CNNs providing inputs to LSTMs, providing a context aware embedding that models dependence between the CNN feature vectors for each support point, and also between support points and query point. Prototypical nets (Snell et al, 2017) also learn an embedding based on a CNN, with the matching between support and query samples performed by a nearest class means classifier. The CNN is adjusted during meta-training, with a cross entropy loss function over the points in the query set, for multiple episodes.

This line of work is in close relation with metric learning, which aims to adapt a metric function over feature vectors for a given dataset (Bellet et al, 2013). If a significant metric is learned, distance-based classification becomes a relevant alternative, as exemplified by Mensink et al (2013). They consider metric learning methods for two distance-based classifiers, the k-nearest neighbor (k-NN) and nearest class mean (NCM), and propose new methods for NCM allowing to model more complex class distributions with multiples centers per class. This possible complexity in class distribution cannot be captured by local metric learning methods, as is the case with current deep metric learning. As discussed by Song et al (2017)

they are incapable of identifying scattered classes, with multiple clusters in the space. In response, they propose to learn an embedding function that directly maximizes a clustering metric (normalized mutual information).

Using memory augmented networks has also been explored in the context of few-shot learning (Santoro et al, 2016). The idea is to build on top of a Neural Turing Machine (Graves et al, 2014), an implementation of a content-based access memory for neural networks, adapting it to the one-shot learning task. This is yet another meta-learning approach, where the recurrent network is not trained to predict directly a specific set of classes, but tries to predict the right classification at each time-step based on the sample-class associations it could learn and keep in memory on the previous time steps. It still needs to see a large number (more than 10k) of episodes to make good predictions, incrementally making better predictions as it sees up to 10 examples of the same class. Compared to human memory, its functioning could be seen as a working memory, that can match new samples to recently seen examples but limited to a low amount of distinct categories.

### 6.2.3 Interpretability

Interest in the field of interpretable machine learning has raised in the last years, partly inspired by the ever-growing impact of machine learning systems in society. Nevertheless, interpretability is a broad term still lacking a precise definition, with open discussions on what it is and how to quantify it (Doshi-Velez and Kim, 2017; Lipton, 2016). One common understanding is to equate it to explainability — the ability to provide explanations to a model's predictions or, even better, the reasoning process behind its predictions.

Understanding the reasoning behind complex models such as deep neural networks is a difficult task. In visual recognition applications, one common effort is to try to visualize what types of features have been learned by certain units or layers of a convolutional network. This is usually achieved by optimizing network input to maximize the activations of the layer of interest. Since the first visualizations produced with DeconvNet (Zeiler and Fergus, 2014), there have been many propositions to improve the quality of the input reconstruction, including the use of regularization priors that enforce more "natural-looking" images (see Olah et al, 2017, for a review). Another way of producing such images is to search the training set for image patches maximizing the activations. This approach has the advantage of producing real examples, although not necessarily specifying which features in the image led to it to be put in a particular category.

In-between these feature visualization strategies is the problem of attribution, where the goal is to identify which regions of the input image were responsible for maximizing a chosen activation (Sundararajan et al, 2017; Bach et al, 2015). An example of attribution procedure is LIME, a method that locally approximates the model around the output prediction and goes back to the input image highlighting superpixels most responsible for its predicted class

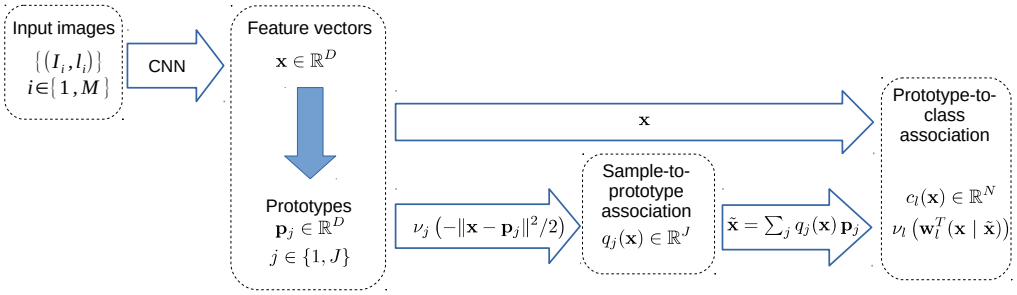



**Figure 6.1** The three layer model. The input data is fed to a convolution neural network, transforming the original image into a set of feature vectors. Then the information in this high dimensional feature space is summarized via a set of prototypes, in order to understand the landscape. Finally, the relation between a given data point and the label to infer is calculated through their proximity to the prototypes.

(Ribeiro et al, 2016). Other works are interested in generating some salience map information over the input image (Bach et al, 2015; Shrikumar et al, 2017). While feature visualization is still abstract and away from verbal human-level explanation, attribution or salience maps are grounded on real images and can provide some level of justification.

Even though these methods were developed having deep CNNs in mind, some of them can be generally applied to explain other classes of models. The explanation procedure itself can be seen as an explanation model, trained to provide justifications given the inputs and outputs of the black-box prediction model. Lundberg and Lee (2017) defines additive feature attribution methods, a class of local explanation models, unifying diverse approaches from literature (including Ribeiro et al, 2016; Shrikumar et al, 2017; Bach et al, 2015). Another way to use an accessory model as explanation is to distill the network into a class of allegedly interpretable models — such as decision trees — training the explanation model to mimic the networks predictions (Frosst and Hinton, 2017).

## 6.3 Model proposed

### 6.3.1 Model architecture

Here we introduce a prototype matching layer as proposed in Snell et al (2017), after the embedding provided by a CNN, defined by a set of prototypes sampling the input distribution in the feature space. This layer is a softmax taking both the sample features and the sample prototype proximity into account. This allows grouping examples with respect to the induced metric learned by the CNN layers, and taking the competition between prototypes into account. Differing from Snell et al (2017), this output feeds an additional final softmax classifier layer, that correlates the prototype's matching to class labels. This allows to introduce a soft but

sparse prototype-to-class label assignment: A given prototype is encouraged to represent a unique class, but in ambiguous situations this mapping is not hardwired. It also differs from alternative mechanisms such as support-vector machines, Kohonen-like maps, or radial basis functions (such as, e.g., in Lyu and Simoncelli (2009)).

Some specificity with respect to usual architectures using prototypes are taken into account here:

- The prototype layer is not the final layer but an intermediate upper-layer of the architecture, as a complementary information to perform the perceptual task. Here classification is the task, but this extends to other perceptual tasks.
- A prototype is a "data prototype" in the sense that its role is not only to reveal if a neighborhood of the feature space belongs to a given category of the classification task, but also to represent at a macroscopic level the feature space itself (e.g., which regions are to be taken into account, whether a category corresponds to a connected region or not, whether a region contains adversarial samples, etc). As a consequence, we neither hard-wire sample to prototype matching nor prototype to class label assignment, but let the estimation provide the best description of the data, given the classification task, in an interpretable way, as discussed below. This is quite different from the related works reviewed in the previous section.
- Given a dataset, the final task-related layer is fed with two alternative combined inputs: (i) The whole feature sample vector, and: (ii) The data in relation with the prototype space. With the former choice only, the prototype layer role is simply to represent the data in the feature space but is not involved in the classification task. With the latter choice only, the large dimensional feature sample (here of dimension 2048, see details in the next section) reduces to a very low dimensional space (less than 50 in our setup). This very likely introduces some bias, but may limit overfitting.

Let us now turn to the formal detailed description of the model.

### 6.3.2 Model specification

We consider[5] a set of labeled images $\{\cdots(\mathbf{I}_i, l_i)\cdots\}, i \in \{1, M\}$, labels being finite $l_i \in \{1, N\}$. This input space $\mathcal{I}$ is embedded in a feature space $\mathcal{X}$ of huge dimension $D$ through a

[5]**Notations:** Vectors and matrices are written in bold, the vector dot-product writes $\mathbf{x}^T\,\mathbf{x}'$, i.e., as a matrix product with the row vector transpose of the left column vector. The Heaviside function writes $H(u)$. Partial derivatives are written in compact form, e.g., $\partial_{x_n}\,\mathbf{f}(\mathbf{x})$ means $\frac{\partial \mathbf{f}(\mathbf{x})}{\partial x_n}$. The notation $\delta_{\mathcal{P}}$ stands for 1 if the property $\mathcal{P}$ is true and 0 otherwise (e.g., $\delta_{2>1} = 1$). The notation $\{a, b\}$ stands for an integer interval and $[a, b]$ for a real interval. When considering a vector $\mathbf{x}$ augmented by new dimension $\mathbf{c}$ we simply write $(\mathbf{x}, \mathbf{c})$ for the corresponding element in the Cartesian product of the corresponding spaces.

function $f : \mathcal{I} \to \mathcal{X}$, here defined by the CNN. We introduce the notion of prototype in order to discriminate between twisted features at the output of some CNN.

To this end, we propose to consider the estimation of a fixed-size set of prototypes in the feature space $\mathbf{p}_j$, $j \in \{1, J\}, J \geq N$. The fact this set is fixed is going to be discussed at the experimental stage, showing that the hyperparameter $J$ is robust.

We are in a supervised learning paradigm considering samples $\{\cdots (\mathbf{x}_i, l_i) \cdots\}$ providing examples of association between features and categories.

We proceed in two steps. First, $q_j(\mathbf{x}) = p(\mathbf{x} \in Q_j)$ is the probability for the feature vector $\mathbf{x}$ of a sample, to be associated to the $j$-th prototype. This association is written as $\mathbf{x} \in Q_j$, where $Q_j$ is the set of samples associated to the given prototype. Interestingly enough, our model does not implicitly assume that the $Q_j$ form a partition of the feature space, as discussed in Appendix A.5. These regions may overlap or uncover the whole space. Furthermore a sample may be sparsely represented, thanks to the soft-max criterion, by several prototypes as in a dictionary representation method (see e.g., Rubinstein et al, 2010). We approximate $q_j(\mathbf{x})$ as a function of the proximity $-\|\mathbf{x} - \mathbf{p}_j\|$ to the prototypes, writing:

$$q_j(\mathbf{x}) = \nu_j \left(-\|\mathbf{x} - \mathbf{p}_j\|^2/2\right), \tag{6.1}$$

where $\nu_j(\cdot)$ stands for the soft-max distribution, the related $D \times J$ weight matrix of this layer corresponding to prototypes coordinates in the feature space $\mathcal{X}$. See Appendix A1 for a detailed presentation of this equation.

Then, $c_l(\mathbf{x}) = p(l|\mathbf{x})$ is the probability for a sample to be associated to the label of index $l$. For the purpose of analyzing different aspects of the given architecture, we consider two alternatives: *direct* use of the prototypes and *combined* use of prototypes and features.

#### 6.3.2.1 DIRECT USE OF THE PROTOTYPES

The label of sample is estimated by another softmax layer:

$$c_l(\mathbf{x}) = \nu_l \left(\mathbf{w}_l^T \mathbf{q}(\mathbf{x})\right), \tag{6.2}$$

the related $J \times N$ weight matrix corresponding to the prototype-to-class association. This design choice introduces an important dimensional reduction (see e.g., Shalev-Shwartz and Ben-David, 2014, Chap 23) in the processing. It is a strong choice and it is going to be numerically studied against a vanilla alternative made explicit in the next subsection.

Considering an approximate cross-entropy criterion to adjust the parameters given sample features $\mathbf{x}_i$ and their label $l_i$ yields the following minimization:

$$\mathcal{C} = -\int_{\mathcal{X}} p(\mathbf{x}) \log\left(c(\mathbf{x})\right) + \frac{1}{C}|\mathbf{w}_l|_d \simeq -\frac{1}{M}\sum_i \log\left(c_{l_i}(\mathbf{x}_i)\right) + \frac{1}{C}|\mathbf{w}_l|_d, \tag{6.3}$$

where $d \in \{1, 2\}$ corresponds to the $\mathcal{L}^1$ or $\mathcal{L}^2$ norm, while $C$ is a hyperparameter weighting the regularization term.

A straightforward derivation (see Appendix A3 for the details) of the related normal equations leads to an estimation-minimization method:

- The $\partial_{\mathbf{p}_j}\mathcal{C} = 0$ equation yields a k-means estimation of the prototypes given the samples, weighted by the prototype proximity. This deep relation between divergence estimation and k-means algorithms is known (Banerjee et al, 2005) and will allow us to derive an efficient implementation, and to minimize the role of meta-parameters for this step. In our implementation we have experimented that considering only a standard k-means initialized by a k-means++ mechanism is numerically sufficient.

- $\partial_{\mathbf{w}_l}\mathcal{C} = \sum_i \left[\delta_{l=l_i} - c_l(\mathbf{x}_i)\right] \mathbf{q}(\mathbf{x}_i) + \mathcal{R}$, where $\mathcal{R}$ stands for regularization terms, yields a Hebbian-like interpretable 1st order rule for adjusting the prototype-to-class association, as a function of each training sample a priori class label.

#### 6.3.2.2 Combining prototype-encoded and original features

The combined model uses the probabilities for a given input sample to correspond to the prototypes in order to predict the class. The input sample is encoded via these numbers. It is obvious to decode such information, i.e., to reconstruct the predicted sample given this information:

$$\tilde{\mathbf{x}} \stackrel{\text{def}}{=} \sum_j q_j(\mathbf{x})\, \mathbf{p}_j, \tag{6.4}$$

where $\tilde{\mathbf{x}}$ is the simple Bayesian estimation of $\mathbf{x}$ given $p(\mathbf{x} \in Q_j)$, providing[6] a simplified form of linear autoencoder mechanism (Goodfellow et al, 2016a).

When $D \gg J$, it might be the case that the prototype information is insufficient to allow a good classification performance. Conversely, using a large $J$ may interfere with interpretability, a large number of prototypes being harder to analyses and eventually being less meaningful. To this end, we are also going to also consider

$$c_l(\mathbf{x}) = \nu_l \left(\mathbf{w}_l^T (\mathbf{x} \mid \tilde{\mathbf{x}})\right) = \nu_l \left(\mathbf{w}_l'^T \mathbf{x} \mid \tilde{\mathbf{w}}_l'^T \tilde{\mathbf{x}}\right), \tag{6.5}$$

in the upper classification layer, where $\mid$ is a vector concatenation operator and $\mathbf{w}_l'^T = (\mathbf{w}_l'^T \mid \tilde{\mathbf{w}}_l'^T)$. This provides the following layer with both features and prototype-encoded information to make a classification decision. Additionally, comparing $\mathbf{w}_l'^T$ with $\tilde{\mathbf{w}}_l'^T$ allows to numerically evaluate the contributions of both paths identifying to what extent the reduction of information from the whole data features to a prototype-encoded representation is relevant

[6] Obviously, if the classification task is performed on $\tilde{\mathbf{x}}$, i.e., if we consider $c_l(\mathbf{x}) = \nu_l \left(\mathbf{w}_l'^T \tilde{\mathbf{x}}\right)$, this is equivalent to the previous form $c_l(\mathbf{x}) = \nu_l \left(\mathbf{w}_l^T \mathbf{q}(\mathbf{x})\right)$, with $\mathbf{w}_l'^T \mathbf{p}_j = \mathbf{w}_l^T$, thus simply correspond to an over-parameterization of the classification layer using $D \times N$ weights, but with only $J \times N$ degrees of freedom.

to the classification decision.

#### 6.3.2.3 Relation between prototypes and category information

Another issue with respect to the state of the art is, as proposed in Rippel et al (2016), to consider the prototypes as a mixture between unsupervised information (sampling the feature space without any knowledge) and supervised hardwired information, each prototype being hardwired to a given category. In order to further understand the relationship between the data prototypes and the categorization, since in alternative approaches prototypes are often "hard-wired" to a given category, we have also investigated an extension of the k-means algorithm step on the learning set, taking also the learning sample category into account.

To this end the following extended metric for the k-means algorithm applied on learning samples is considered:

$$d\left((\mathbf{x}_i, l_i), (\mathbf{x}_{i'}, l_{i'})\right) = |\mathbf{x}_i - \mathbf{x}_{i'}|_2 + \beta \sum_j \delta_{l_i \neq l_{i'}} \tag{6.6}$$

In words we augment the feature space by new dimensions corresponding to each categories in order to move apart samples related to different categories. Clearly for $\beta = 0$ this reduces to our original setup, whereas for large values of $\beta$, we make explicit in Appendix A.4 how this allows to hard-wire prototypes to categories.

This complementary tool is not related to performances but to only better understand this aspect of the problem. Additionally, it is important to understand that there is a deep relationship between a softmax classification layer and a prototype representation, as discussed in details in Appendix A7.

## 6.4 Experiments and results

Here we perform a series of experiments to gain intuition about the proposed model, test its performance under a few-shot setting and demonstrate its potentialities as an interpretable alternative. In summary we used two experimental methodologies: first a k-means based implementation over fixed features, followed by a second end-to-end implementation used in a meta-learning setting. In both cases, we work under few-shot learning conditions, exploring the performance of these models for not-so-big data situations.

### 6.4.1 Datasets

A known benchmark dataset for few-shot learning is the Omniglot dataset (Lake et al, 2011). It consists of images of 1622 characters from 50 different alphabets, each drawn by 20 different people. The small number of samples per class and the large number of classes

make it a challenging problem to learn all the classes simultaneously. Therefore we will work on episodes: class subsets of $N$ classes randomly picked among the 1622 characters (see Fig. 6.2:Left for an example).

To study the effect of increasing training set sizes, we will use two larger standard benchmark datasets: MNIST[7] and CIFAR10[8]. Both datasets comprise 10 different classes. MNIST has 28x28 images of handwritten digits, 60K samples in the training set and 10K in the test set. CIFAR10 has 60K 32x32 color images of 10 object categories, with 50K in the training set and the remaining 10K on the test set. From these datasets, it will be possible to sample training subsets ranging from few-shot to many-shot situations.

### 6.4.2 Studying the model under fixed features

Contrary to usual few-shot learning paradigms we do not consider here meta-learning as discussed in section 6.2.2, but a transfer learning paradigm: by using a general purpose pre-trained CNN, we skip any meta-training routines that allow learning CNN filters on a particular few-shot dataset. We use fixed features extracted with a standard CNN architecture pre-trained on ImageNet data, namely Inception v3 (Szegedy et al, 2015), that has successfully been used in transfer learning settings (for instance Esteva et al, 2017; Shin et al, 2016). Features are normalized in the range $[0, 1]$. Based on features extracted from the penultimate layer of this network, our model has to learn to predict from very few samples (see Fig 6.2:Right for a visualization example of this feature space for a sample episode of Omniglot dataset). Fixing the embedding network allows to isolate and assess performance variations due to the final layers of the model, under equal input features. Assessing whether the tested models allow learning of a more appropriate representation is addressed in section 6.4.4.

Except for experiments with increasing training set sizes (section 6.4.3), all experiments in this section are performed on Omniglot dataset. In comparison to meta-learning methodologies, we are skipping the meta-training phase, where the embedded representation is learned, and going directly to meta-testing, where for each independent episode, the model is actually only learning to predict from few-shot data, for a certain quantity $N$ of classes. Thus we operate direct learning on each episode, with a training data set for parameter estimation and a test set for checking generalization. Each episode is split in half, with $10N$ samples for training and $10N$ samples for testing. When hyperparameter tuning is needed, 5x2-fold cross-validation over the training set is used, leaving $5N$ samples to train and $5N$ to validate. Once the hyperparameters are chosen, the model is re-trained on the entire training set. To see how the model performs over the entire dataset, observations are repeated for a large number of episodes.

With those two simplified design choices (using fixed features and performing direct

[7] http://yann.lecun.com/exdb/mnist
[8] https://www.cs.toronto.edu/ kriz/cifar.html

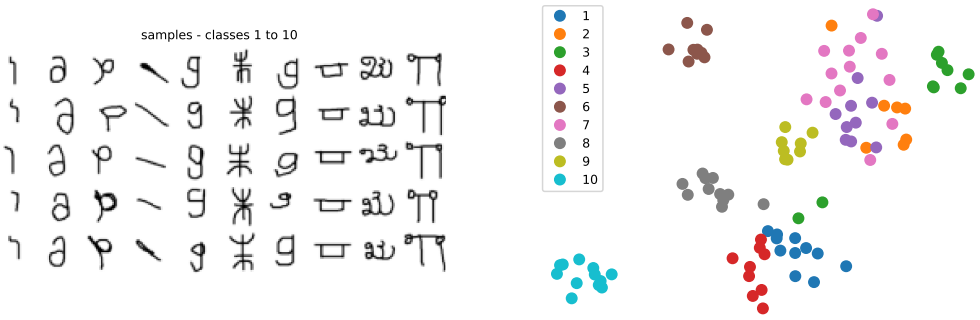



**Figure 6.2** Left: Sub-sample of 10 classes from the Omniglot dataset, showing 5 samples per class (total is 20 per class). Right: T-SNE visualization of the Inception v3 features for the 200 samples of this subset, see text for details.

learning), these results cannot be fairly compared to other few-shot meta-learning approaches (this will be addressed in section 6.4.4). More precisely, we withdraw from learning the CNN layers, which are not adapted to the target data to give a specific representation, relying on transfer learning instead. Moreover, using the direct model only, it drastically restrains the classification number of degrees of freedom from thousands to a few tenths. This is done on purpose in order to study to what extents this compromises the result. Nevertheless, we also know how to also guaranty performances, using the so-called combined model. In both cases, we compare our propositions to other classical classifiers tested in the same setting, namely: softmax, SVM and nearest centroids.

To visualize how data samples and prototypes are distributed in space, we obtain a 2D non-linear projection using T-SNE (Maaten and Hinton, 2008) over the data samples feature vectors together with the prototypes coordinates $p_j$ in the feature space, as shown in Fig. 6.2:Right. Such mechanism preserves local neighborhood of closed samples, whereas long range distances can not be interpreted since skewed by the non-linear projection: The map only reflects the similarities between the high-dimensional inputs in a stochastic, but not metric way.

Code and further analysis for this section and section 6.4.3 are available at `https://gitlab.inria.fr/mnemosyne/data_prototypes`.

#### 6.4.2.1 Study of a sample episode

Hyperparameter adjustment is a key factor in the success of any machine learning model. This first analysis focuses on gaining intuition on the behavior of the model on a sample episode with $N = 10$ classes (shown in Fig. 6.2). The aim is to study its hyperparameters, experiment with visualization of prototypes and data samples in a common space and try

to understand the relation between prototypes and their attributed class. In this section we consider only the hyperparameters introduced by the formulation presented in section 6.3.2. More details on these and other hyperparameters can be found in Appendix B.

In the current experimental setting, our base model has two hyperparameters: the inverse regularization strength $(C)$ and the number of clusters, thus of prototypes $(J)$. We explored 10 values of $C \in [10^{-3}, 10^{8}]$ and 10 values of $J \in [2N, 5N]$, where $N = 10$ is the number of classes (see Fig. 6.3). For the direct model under L2 regularization, we can identify that average performance is stable for $C \leq 10^{-1}$, and more sensitive around larger C values in $(10^{2}, 10^{6}]$, raising around $C \leq 10^{2}$. For for L1 larger values are preferred, presenting a stable score as soon as $C \leq 10^{2}$. For the combined model, we have a behavior similar to direct-L1 under both regularizations, with any values above a certain threshold being equivalently good ($10^{-2}$ for L2 and $10^{1}$ for L1). Since no clear preference could be stated between L1 and L2 regularization, both regularizations keep being analyzed in the following (and are included in the comparative analysis in section 6.4.2.3).

Regarding the number of prototypes $J$, as visible in Fig. 6.3, a higher number of prototypes is preferred by the direct model, while the combined model does not seem to depend on it. However, for the direct model, using a too large number of prototypes would ultimately amount to a nearest-neighbors-like algorithm and possibly counteract on their interpretability purpose (too many prototypes representing too many base concepts to support a simple explanation), while overfitting and poor generalization performances are expected. With the combined model, we can avoid this issue by choosing a lower number of prototypes while keeping high performances.

Nevertheless, a minimal number of prototypes can be important for interpretability, but can be ignored when looking only at accuracies in cross-validation. This motivates using another method for choosing the appropriate number of prototypes. As far as data representation is concerned, we can simply rely on usual k-mean prototype count adjustment methods. This is usually done by sweeping some range of values for the number of clusters and keeping the one minimizing its loss or, in case of saturation as $J$ grows, keeping the lowest value that reaches close to minimal loss. We exemplify this procedure in Fig. 6.4 for an Omniglot 10-shot episode, which suggests a number of prototypes around 20. Statistical significance of the observed elbow curve has been verified. More precisely, considering a least-squares adjustment of a parametric elbow model against a constant value model, the related Fisher ratio test probability threshold is $p = 0.2$ on the non-normalized values. It is only $p = 0.3$ on the normalized values, and it is more than $p = 0.001$ (non-normalized) and $p = 0.2$ (normalized) on the MINST data set reported on Fig. 6.9 and only $p = 0.3$ for the CIFAR 10 data set. We have tested for several parametric models (cubic, piece-wise linear, exponential) and obtained the same results. We hypothesize this is due to due to very small set of data used (10 samples by data point). However, as far as interpretability is concerned it is still interesting to have an estimation of a reasonable order of magnitude of the number of prototypes. We also are

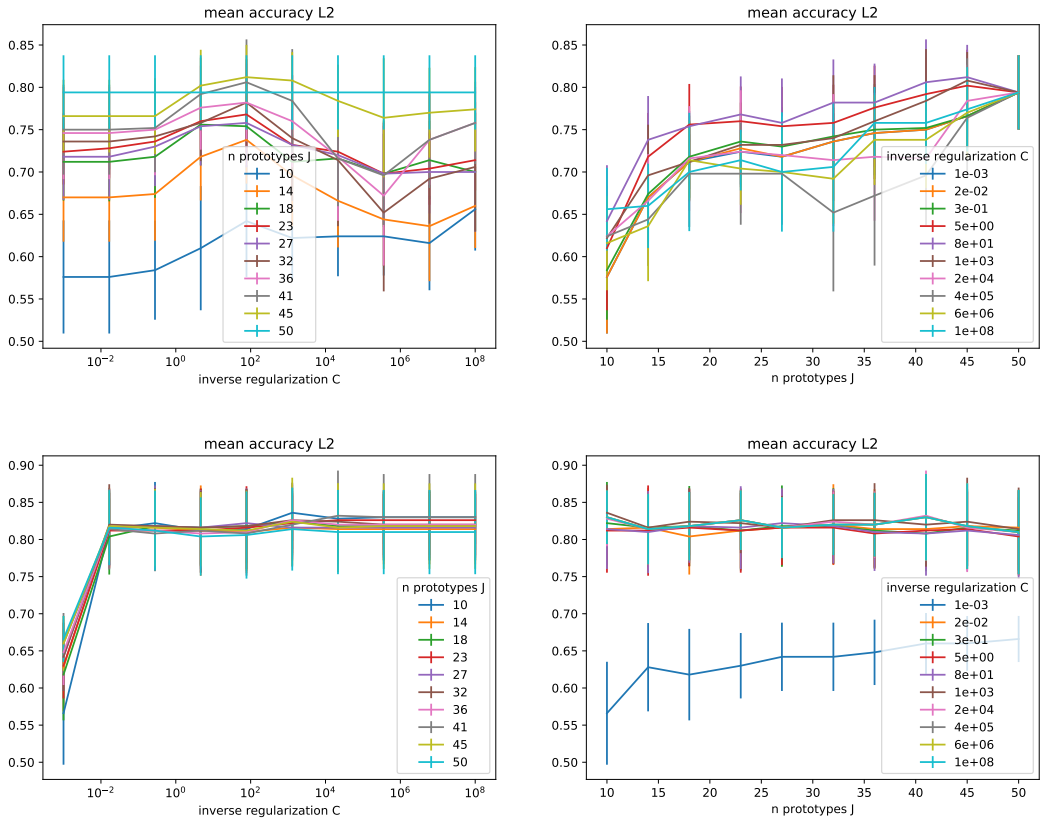



**Figure 6.3** Mean accuracy (with standard deviation bars) obtained during grid search for the inverse regularization parameter C and the number of clusters, under L2 regularization. Top: direct model. Bottom: combined model.

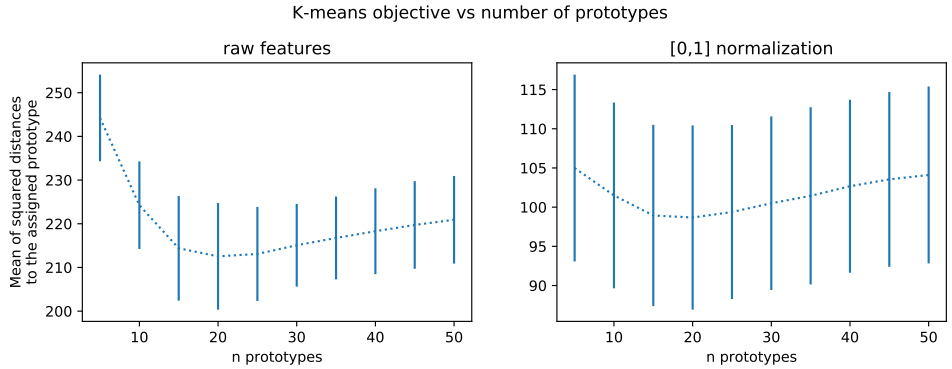



**Figure 6.4** Average training loss on the k-mean algorithm as a function of the number of prototypes. The mean square distance between samples and prototypes decreases until a plateau (sometimes called elbow) and then either re-increases, as visible in this example, or do not significantly re-decreases as shown in Fig. 6.9 for other datasets.

going to observe in the sequel that this number is not critical regarding performances, so that this rough estimate is sufficient.

In summary, we can choose the number of prototypes in the clustering phase based directly on its unsupervised loss, optimizing the number of prototypes for data representation. Altogether, the key points here are that this hyperparameter is to be considered more as a way to optimize the data interpretability than the classification performances, with the possibility of being automatically adjusted.

When using the extended metric described in section 6.3.2.3, we have an extra hyperparameter $\beta$ weighting the added class-aware term. Based on the previous analysis for $C$ and $J$, we decided to constrain the search of $C \in [10^2, 10^6]$ and reduce the number of parameters searched to 3. For $J$, we still search on the same range but limit the number of parameters to 5. Fig. 6.5 shows accuracy results on cross validation for different pairs of $\beta$ and $J$ (with $C$ fixed on the best value found by grid search). The contribution of this term for a fixed value of $J$ is not clear, at this stage, but it is interesting to notice that when $J = N = 10$ there is an improvement until a certain threshold, followed by a plateau ( $\beta$ lower than $\approx 7.8$ under both L1 and L2 regularizations). This can indicate that for large enough $\beta$ the class-aware term has dominated the positioning of each cluster undermining the influence of the first distance-to-prototypes term. It could also be that for small numbers of prototypes this additional term (that encourages prototypes to cluster samples corresponding to a unique category) has a positive influence on performances, whereas it is no more the case when the number of prototypes is high enough. We consider this result as a good indication that the compromise between a good data representation good classification performances seems possible with our so called data-prototypes.

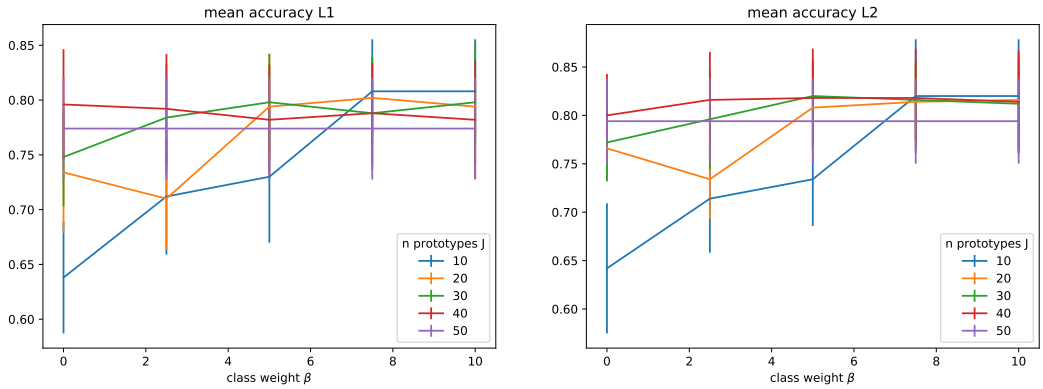



**Figure 6.5** Verification that the introduction of a priori information on the prototype category has no significant impact on the performances, as soon as the number for prototypes is high enough. Graphs show mean accuracy (with standard deviation bars) obtained during grid search under the best regularization parameter found. Left: under L1 regularization, $C = 10^4$. Right: under L2 regularization, $C = 10^2$.

### 6.4.2.2 Interpretability of the model

As previously discussed, prototypes in our model lie in the same feature space as the data and correspond to virtual examples. In this section we demonstrate how this fact can support explanations about the model's decisions. For such, we provide a visualization for each prototype by looking at its closest training samples, in Fig. 6.6. To observe prototypes in the feature space, we use T-SNE visualizations of training and test samples in the feature space together with prototypes (figure 6.7). Visualizations in this section correspond to our main model combining features and prototypes, under L2 regularization. With this model, since the performance is not sensitive to the number of prototypes, $J = 10$ prototypes was the value chosen by cross validation.

We observe the representative images obtained are coherent with the prototype classes, except for prototype 7, which was classified as class 2 but had a closest sample coming from class 7 (Fig. 6.6). This could be due to the fact explained by the fact that both classes correspond to very similar characters (both look like a lowercase "g" under different rotations, see figure 6.2), and indeed the T-SNE visualization shows these classes are very mixed in the feature space. Additionally, class 5 is also mixed in and corresponds to a similar character. Indeed, the top-2 most likely classes for `p7` predicted with close probability were `c2` with 87% `c7` with 7% (see figure 6.7), indicating `p7` lies in a slightly more ambiguous region of the feature space. In a scenario with an adaptive number of prototypes, this prototype-to-class association information could serve as an index for adding more prototypes in the region.

Representation by the closest training sample illustrates how each prototype can be represented by a corresponding input, having the advantage of being a hyperparameter-free method yielding real images. Of course feature visualization methods based on input optimization could also be applied, but appropriately tuning such models is fundamental for

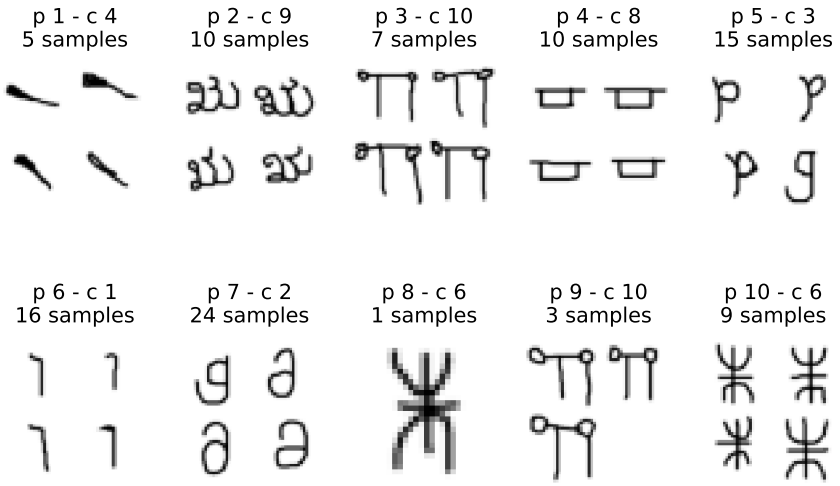



**Figure 6.6** The top-4 closest training samples visually illustrate what is being represented by each prototype. For each prototype `p1`, `p2`, .., its related class `c2`, `c5`, .. is written, this "main" class being estimated as the most probable class proposed by our algorithm. The number of samples corresponding to the related cluster is given.

obtaining realistic image representations and out of the scope of this work.

#### 6.4.2.3 Comparison over multiple episodes

Extending this analysis, we now compare test accuracies over 100 random episodes for different model variants, in order to obtain results of statistical significance. As reference methods we have a softmax regression layer (multinomial regression), an SVM and a nearest class centroids classifier (like the classifier layer used by Snell et al (2017)). From our proposal we test 4 variants: both combined and direct models, using L1 or L2 regularization (as defined in 6.3.2). Test accuracy results for 100 episodes are displayed in Figure 6.8.

Statistical significance of the results are verified by a non-parametric Friedman test with post-hoc Nemenyi tests (with $p \leq 0.05$, see table 6.1) (Demšar, 2006). No significant differences between both regularizations could be observed for the direct model, while L2 is significantly superior for the combined model. Between direct and combined, combined is significantly better in both cases. Regarding the baseline comparisons, we highlight that the combined model with L2 regularization differs significantly from a SVM for $N = 20, 30$, although no significant difference to softmax could be observed. Additionally, all models differ significantly from nearest-centroids.

### 6.4.3 Comparison over different training sets with varying sizes

In order to confirm our results we have considered two different datasets: MNIST and CIFAR10. These datasets have larger training sets, allowing to subsample training sets of size larger than $20N$ (which was the case for Omniglot). Additionally, they have a large standard

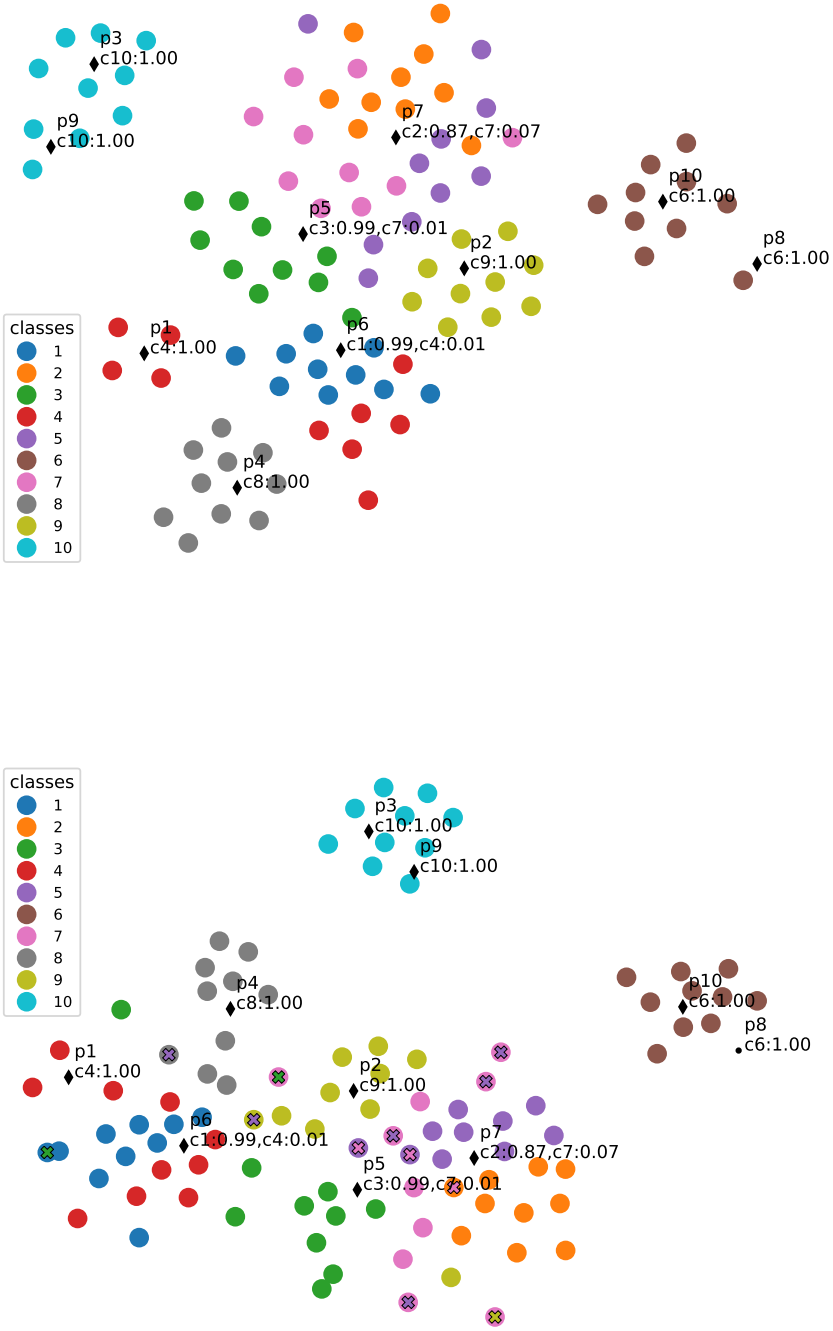



**Figure 6.7** T-SNE visualizations of dataset points (as Inception v3 features) together with learned prototypes, indicated by black markers. Each class is identified by color, and the predicted class for each prototype is annotated. Misclassified points are over-marked by an "x" of the predicted class color. Top: visualization of the training set; Bottom: visualization of the test set.

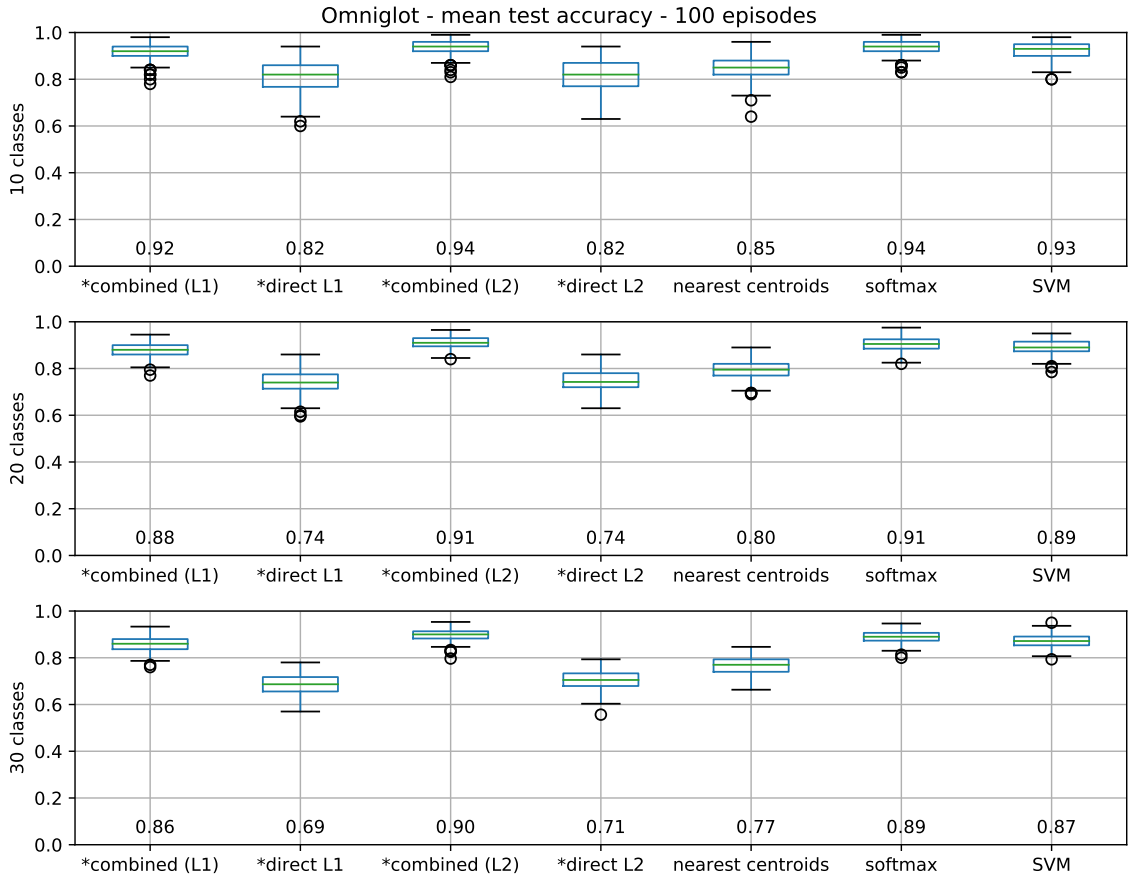



**Figure 6.8** Box plots for test accuracies over 100 episodes. Boxes extend between 1st and 3rd quartiles ($IQR$), with median value noted at the bottom and marked by a green line. Bars mark a range of $\pm 1.5$ $IQR$ from quartiles, with outliers as circles. Our models are marked with a $*$.

**Table 6.1** Statistical significance for all pairs of classifiers tested on Omniglot. "++" and "+" mark all pairs which presented a significant difference (for $p < 0.01$ and $p < 0.05$ respectively), while "o"s mark those which did not. Non-reported pairs were found to be significantly different with $p < 0.01$ (except for direct L1 vs. direct L2 at 30 classes, for which $p < 0.05$).

| | | | 10 classes | | | 20 classes | | | 30 classes | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| Models | L2 | combined L1 | combined L2 | softmax | SVM | softmax | SVM | L2 | softmax | SVM | L2 |
| L1 | o | ++ | ++ | ++ | ++ | ++ | ++ | o | ++ | ++ | o |
| combined L1 | ++ | - | ++ | ++ | o | ++ | + | ++ | ++ | ++ | ++ |
| combined L2 | ++ | ++ | - | o | o | o | ++ | ++ | o | ++ | ++ |
| SVM | ++ | o | o | ++ | - | ++ | - | ++ | ++ | - | ++ |

test set (10K samples) over which to evaluate the performances. Class distribution in these datasets is approximately balanced, and the distribution is preserved when sampling the training subsets. We explored training subsets of size 100, 200, 300, 500, 750, 1000 (that is, 10, 20, 30, 40, 50, 75, 100 samples per class on average). The idea is to evaluate performances ranging from few-shot data (here 10 samples per class) to not-so-big data (here a few tenths of samples per class). For hyperparameter selection we proceed exactly as done for Omniglot, by 5x2 cross-validation. All models are retrained over the whole training set using the best validation parameters. Statistical significance is also evaluated in the same way (Friedman non-parametric test + Nemenyi post-test). We report the mean test accuracy, averaged over 10 different random training subsets, all evaluated on the standard test partition for each dataset.

We also have numerically studied for these new datasets whether the k-means prototype count automatic adjustment is still numerically valid, as reported in Fig. 6.9, after Fig. 6.4. We thus have now observations for Omniglot (sample 10-shot episode), and MNIST and CIFAR10 (training subset with 500 samples). Such a method indicates 20 clusters for Omniglot and CIFAR10, but a larger number for MNIST (between 30 and 50 depending on a saturation tolerance). The optimal value does not correspond to a minimum but is still detectable as an elbow corresponding to the occurrence of a plateau. This result is not always statistically significant. It is the case for MNIST but not for CIFAR10.

A first evaluation of the means (and standard deviations, see figure 6.10) allows to observe that our combined models perform best or similar to a softmax or an SVM classifier, all performing better than nearest centroids or our direct models, which is congruent with results obtained on Omniglot (but here on a larger test set). Particularly for MNIST, the nearest centroids classifier is considerably lower than the top performing methods.

Considering statistical significance of the comparisons, on MNIST our combined model with L2 regularization is consistently better than nearest centroids, although the same can be said about softmax. In most cases, for both datasets, combined models, softmax and SVM are significantly better than either of the direct models, but comparison between them is inconclusive. For CIFAR10 the significance results are less conclusive, but we do observe some significance in the superiority of the combined model (L1 regularization) over nearest-

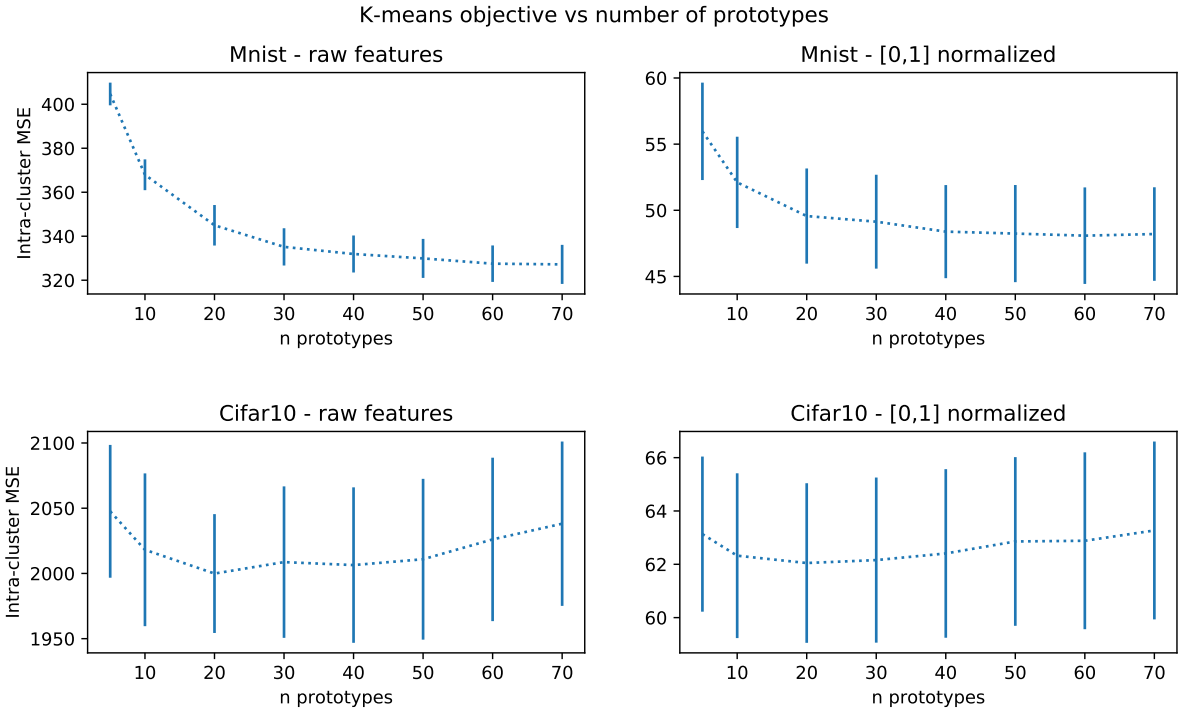



**Figure 6.9** Performances on the k-means algorithm as a function of the number of prototypes, for training subsets with 500 samples from MNIST and CIFAR10 datasets. In these and contrary to Fig. 6.4, the optimal value corresponds to an elbow in both cases.

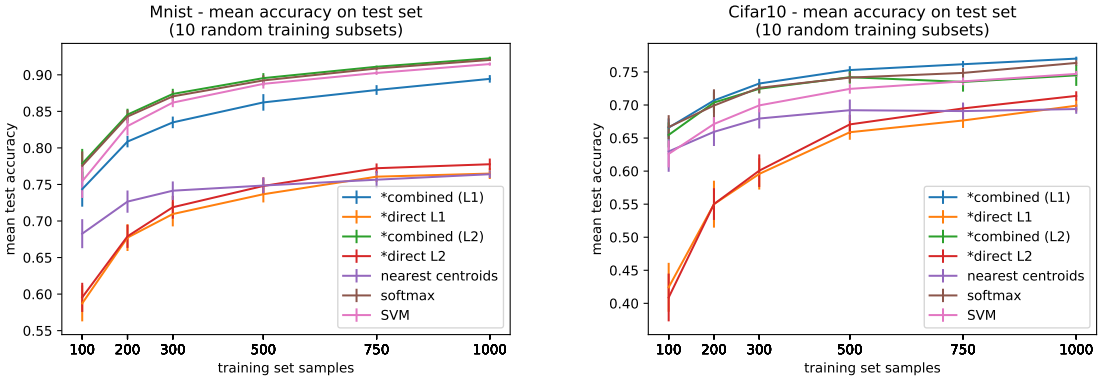



**Figure 6.10** Mean accuracies over standard test sets for MNIST and CIFAR10 datasets, for increasing training set sizes. Results are averaged over 10 different training subsets. Our models are marked with a $\star$.

centroids (between 200 - 1000 training samples). Detailed results for statistical testing of all classifier pairs for all training set sizes can be consulted in the code repository for this paper.

### 6.4.4 Experiments in the meta-learning paradigm

To allow a fair comparison to usual few-shot learning paradigms, we carried on end-to-end training of the model under meta-learning conditions. We continue to use Omniglot dataset described above, with 4 rotations per character used for data augmentation (0, 90, 270, 360 degrees), and use episodical training, following the methodology of Snell et al (2017). We use the same CNN architecture (4 layers, 3x3 kernels, resulting in a 64 dimension embedding) and experimental setup, based on their publicly available implementation[9].

In summary, the task here is to prepare a meta-model which will be able to learn to classify N classes based on k training samples, for different subsets of N classes. For this, the full architecture is meta-learned over a some episodes sampled from a meta-training set containing 1200 characters, with episodes from a meta-validation set containing 172 characters being used to evaluate and early-stop the meta-training, while the remaining 423 characters are used to generate meta-testing episodes (subsets of N classes not seen in the meta-training phase).

More specifically, the meta-training phase is carried on 100 randomly chosen 5-shot 60-way episodes. Another 100 5-shot 5-way episodes are sampled to form a meta-validation set. Meta-training is stopped after 200 epochs with no improvement on the meta-validation set, keeping the best model so far. At meta-test time, we apply the model to solve 5, 10, 20 and 30-shot learning tasks on 1000 episodes sampled from previously unseen classes.

Since these experiments are more computationally intensive, using cross-validation to choose hyperparameters becomes prohibitive. We fixed $J = 2N$, i.e. 2 prototypes per class and

[9]Code for prototypical networks available at `https://github.com/jakesnell/prototypical-networks`, under MIT license. Our fork is available at ]urlhttps://github.com/thalitadru/prototypical-networks

L2 regularization of $10^{-4}$ (via weight decay). Optimization of the complete model, including CNN weights, is done with an Adam optimizer with initial learning rate of $10^{-2}$ (and default moment estimation parameters). For each episode the top two layers of model need to be fitted on the new classes using the training (or support) k-shot examples. This optimization was run for 200 iterations, with an initial learning rate of 0.1 (also using Adam). The test (or target) examples serve to evaluate the model and, when in meta-training phase, calculate the loss and gradients that will adjust the bottom CNN weights.

We compare our accuracy results over 1000 test episodes to two other propositions in the literature: Matching networks (Vinyals et al, 2016) and Prototypical networks (Snell et al, 2017) (see table 6.2). These are seminal works in the field of few-shot learning and remain competitive compared to more recent proposals (for instance Garcia and Estrach (2018) or Finn et al (2017)). For prototypical networks, we re-ran the models using the code released by the authors to obtain results for more values of $N$ classes. Slightly different results are normal, since we did not do the final retraining step , where the model is retrained in the full meta-training set (including the portion separated for meta-validation).

Overall, we can observe that learning the CNN weights through this meta-training approach yields better performances in comparison to using fixed features, as expected. Considering that no hyperparameter tuning was done, we still obtain good results, a little lower than the state-of-the-art, even though they degrade faster as the number of classes goes up. Some discussion on this result is presented in the next section. Between combined and direct model, we also observe that the former performs better at a higher number of classes $N = 20, 30$, while the second is better for $N = 5, 10$.

**Table 6.2** Average test accuracy (%) for 5-shot learning on Omniglot over 1000 episodes. We compare to results reported by two other works: Matching networks (Vinyals et al, 2016) and Prototypical networks (Snell et al, 2017).

| Method | 5 classes | 10 classes | 20 classes | 30 classes |
|---|---|---|---|---|
| Matching Networks | 98.9 | - | 98.5 | - |
| Matching Networks - fine tuning | 98.7 | - | 98.7 | - |
| Prototypical Networks | 99.7 | - | 98.9 | - |
| Prototypical Networks (re-run) | 99.54±0.07 | 99.17±0.06 | 98.57±0.06 | 98.01±0.06 |
| Direct - 2N prototypes | 98.4 ± 0.2 | 97.2 ± 0.2 | 94.8 ± 0.3 | 92.6 ±0.4 |
| Combined - 2N prototypes | 97.1 ± 0.2 | 96.3 ± 0.2 | 96.2 ± 0.1 | 95.9 ±0.1 |

## 6.5 Discussion

### 6.5.1 Performances and use of the algorithm

We have proposed a simple three-step model, applied to classification tasks on visual data under very small (few-shot on Omniglot) to not-so-big datasets (subsets from MNIST or CIFAR10). Let us now discuss the main questions presented in the course of this work in the light of our experimental results.

- *Is the setup able to properly learn such a task in an interpretable way?* Our models were capable of performing the classification tasks, both under fixed features and under the meta-learning paradigm, while profiting from the explainability support that can be provided by our rather simple architecture. Our "data prototypes" that can be used to provide a visual explanation of the model's internal representations, by indicating ambiguous classification regions for example. Some mathematical discussions on how the prototypes induce a partition of the feature space are presented in Appendix A5 and A6. This explanatory power is however limited when using the combined model, since raw features are also considered in prediction. Prioritizing interpretability by using only the direct model yields reduced performances under fixed features, indicating a trade-off between interpretability and classification accuracy in this case.

All together, this is no more than a variant of the notions of prototypes used in the field, but with the objective to both represent the data in an unsupervised way and the supervised information. The name "data prototype" is chosen particularly because they may be considered as virtual samples that summarize what has been learned during the learning phase, acting as a local descriptor in the feature space. Being virtual samples, they could be fed to feature visualization techniques to provide a graphic explanation (here exemplified by taking the closest training samples as a visualization proxy).

Another non negligible aspect is the fact that our proposed architecture, schematized in Fig. 6.1 is "easy to explain" to non-specialists : (i) features are extracted (by the deeper layers), (ii) the data is organized in regions (parameterized by prototypes) and (iii) the classification is performed combining all information. Our will to provide an interpretable and understandable process is not only a wish but corresponds to real interaction with large public targets, including providing didactic resources on these subjects, as reported in Drumond et al (2017a) or available in Kaadoud and Viéville (2016).

- *To which extents does the prototype dimensional reduction impairs the global performances?* Under fixed features, indeed we have a strong dimensionality reduction which impairs the performance of our direct model, which justifies the use of a combination of features and prototype information in our main model. This way it is possible to both benefit from the prototype layer and obtain optimized performances as good as or better than a standard method.

In meta-learning paradigm, we also learn the features by learning the weights of a CNN that feeds our model. In this case, since the features are learned together with our prototype model,

the class representations should become more adapted to a prototype-based discrimination, resulting in higher accuracies than under fixed features even for the direct model. Moreover, the dimensionality reduction was not as strong because we use a 64 dimension embedding and $2N$ prototypes.

Combining these two options allows the user to both obtain state of art performances and interpretable results. With respect to other comparable existing methods, we have obtained similar performances, except when considering meta-learning with a large number of classes. This is explained by the fact that we have not optimized hyper-parameters in this case, which had already been done in the fixed-features paradigm.

We can also hypothesize that our algorithm, since based on a reduced set of prototypes that acts as support-vectors of the classification (See Viéville and Crahay, 2004b, for a development of the link between support-vectors and prototypes), may have good generalization properties, e.g., may limit overfitting which is to be expected in a not-so-big data setup. This is indeed the case for the direct model. For the combined model, as detailed in Appendix A7, we may interpret the softmax layer as piece-wise linear classifier considering $O(L^2)$ prototypes. As a perspective of this work, going in depth in this interpretation may be an interesting issue.

- *Is the hyperparameter adjustment automatic at the end-user level?* The main hyperparameters we introduce are the number of prototypes and a typical regularization strength, this last one being not too critic as long as it is not too strong ($C > 10^2$). Even if the number of prototypes is a non-negligible choice in terms of data representation, with the combined model we eliminate sensitivity to this hyper parameter in terms of performance. However, in terms of interpretability, this leads to choosing a small number of prototypes that might no represent so well classes in the feature space. Conversely, with the direct model, we are led to choose an always large number of prototypes at the risk of overfitting, because at the limit (one prototype by learning sample) the method reduces to a nearest-neighbor classifier, known to be of poor generalization performances. Here, we fix this issue by proposing to choose the number of prototypes in the clustering phase based directly on its unsupervised loss, optimizing it for data representation. We have experimentally verified the feasibility of this method for the different data sets. Comparing to other methods, this seems to be the main new qualitative result to point out.

### 6.5.2 Perspectives and future work

Beyond these outcomes, general or application dependent extensions of the method can be pointed out.

\+ *Consider incremental learning*, where learning can continue when given new samples, by updating the prototypes and the related category prediction. This can be achieved for instance through incremental k-means, which is a standard process (See, e.g., Yadav and Singh, 2015). Applying cross-entropy minimization in an incremental manner is also straightforward,

iteratively adding loss terms for newly arrived samples.

+ *Consider prototype editing*, where (i) redundant prototypes within the same connected component can be deleted, while (ii) learning samples detected as exceptions and generating a classification error can be taken into account via a new prototype. Additionally, and adaptive mechanism for determining the number of prototypes could better explain the data representation, improving interpretability of the feature space, while eliminating one sensitive hyperparameter of the model. In domain-specific applications where classes might be composed of many different subclasses, a sensible number of prototypes could better identify these subclasses and provide means of representing the different subgroups.

+ *Interacting with the deep-learning lower layers*, since the knowledge of the prototypes may help disentangling the lower-layer feature extraction. A class disentanglement is a likely cause behind the improved performances in the meta-learning setting (when compared to using fixed features), since all layers are trained end-to-end. In a transfer learning setting, we could proceed to the fine-tuning of the upper layers of the pre-trained CNN to try to achieve a similar result. Moreover, disentanglement could be enforced implicitly via low dimensional manifold regularization (Zhu et al, 2017b), or using generative latent factor models (Hoogeboom, 2017). Another form of interaction could be the introduction of shortcuts from lower layers to the feature layer in order to increase the prototype components, analogous to other architectures incorporating some form of forward shortcut (Shelhamer et al, 2017; He et al, 2016a; Huang et al, 2017).

+ *Integrate knowledge from unlabeled examples* is also a possibility. These examples can represented in the same feature space as other labeled samples and prototypes, using same embedding CNN to extract this representation. These examples then can be taken into account when computing the prototype coordinates, analogously to what has been demonstrated by Ren et al (2018). Particularly when trying to solve ambiguous regions where the system makes mistakes, a related track is to develop some active learning heuristics and infer which new examples could be of most help if labeled.

## 6.6 Conclusion

We analyzed how considering data prototypes at a different level with respect to the state of the art allows to both reasonably work with small data sets, and with an interpretable view of the obtained results. The proposed architecture is minimal and combine both the performances of a standard method and the capability to observe the feature space representation.

At the methodological level, we also pay attention at exploring in details the performances and limitations of the proposed set-up, in an analogous way as biological experiments are conducted, considering in details all hyperparameters involved in the process. This is somehow different from simply "winning the game" with respect to some deep learning benchmarks.

Chapter 7

# Conclusion

## 7.1 Summary of contributions

### 7.1.1 Understanding object recognition from an interdisciplinary stance

When comparing CNNs and neuroscientific models of vision (reviewed in chapter 2), similarities have been found in multiple interdisciplinary works. On the functional side, early and late layers have been shown to be highly predictive of activity of V1-V2 and V4-IT cortical areas, respectively, considering data from both neurophysiological responses (Cadieu et al, 2014; Yamins and DiCarlo, 2016) and fMRI activity (Eickenberg et al, 2016). On a more "behavioral" side, CNNs have been shown to make similar classification decisions to those made by primates (when trained on sufficiently challenging tasks), having presented confusion matrix patterns similar to both humans and monkeys (Rajalingham et al, 2018).

These similarities are restricted to fast-brain core object recognition, corresponding the first hundred milliseconds of cortical processing. More complex visual processing has been shown to depend on functional principles and architectural paths not modeled in typical CNNs. We have surveyed three main aspects:

- Advanced processing and encoding done in retina and thalamic structures, such as spatio-temporal compressing and computational integration of multiple cortical areas, respectively;
- Interaction between dorsal and ventral streams — thus between recognition and localization. Dorsal and ventral streams share information on motion, color and form, at multiple stages of both hierarchies;
- The role of recurrent processing and feedback connections towards a more refined visual perception under challenging conditions. Local and long-range recurrences — including the previously mentioned connections with thalamic and dorsal areas — are crucial in allowing refinement and disambiguation of object boundaries, interpolating missing information from local spatial recurrences and integrating motion processing cues.

Of these three principles, recurrent processing is the most general one, as it is present at

several levels of the vision hierarchy, which has motivated reviewing how this mechanism has been integrated to CNNs in the recent literature.

Since the first simple and feedforward CNNs, many architectural innovations have been presented over the years, as reviewed in chapters 2 and 4. Both chapters have organized feedforward and feedback recurrent architectures according to their connectivity patterns, resulting in the taxonomy summarized in fig. 7.1, where each family is represented by an archetypal small module of 3-4 layers (in a typical architecture, multiple such modules may be associated to form the full network).

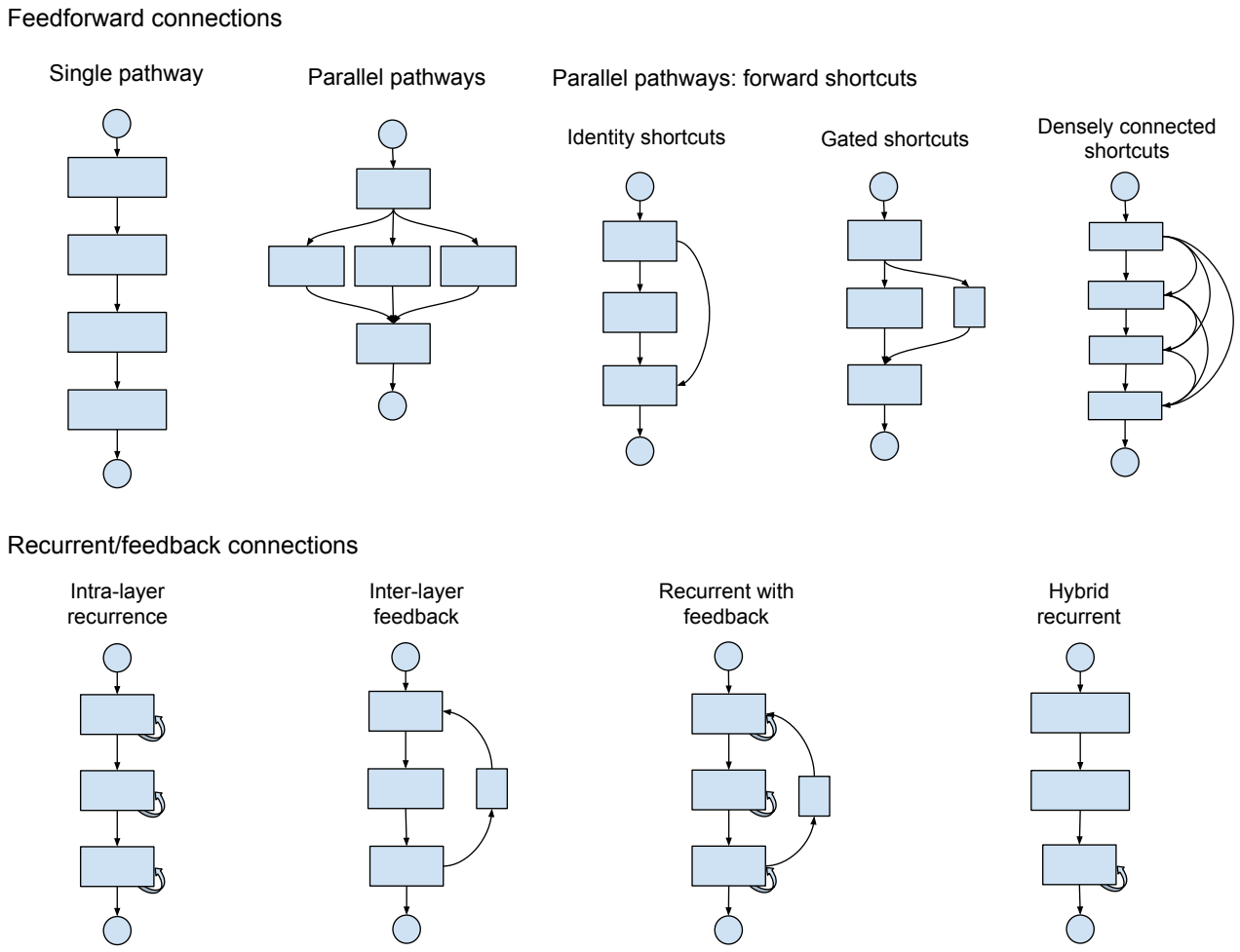



**Figure 7.1** Architectural taxonomy of CNNs: feedforward and feedback. Each family is represented by an archetypal small module of 3-4 layers. Top and bottom circles stand for module's input and output, respectively.

This work has allowed to identify the extent to which bio-inspired principles are taken into consideration for architectures targeted at image classification. Parallels have been made between some architectural innovations — such as multiple feedforward shortcuts and feedback connections — and anatomical/functional mechanisms in the neural visual processing pipeline. Nevertheless, these mechanisms are not well understood regarding the high-level

functionalities they bring to artificial vision on realistic images.

### 7.1.2 Image classification over small datasets with deep models

Having placed our view of the current status of image classification, we have proceeded towards framing the special case of small datasets. As reviewed in chapter 5, among the wide and heterogeneous variety of approaches to the issue, we have identified five general situations possibly falling under the designations "learning on small datasets" or "learning under limited data" (summarized in section 5.2.1 and fig. 5.1). We have proposed this taxonomy after identifying three main levers influencing the level of information scarcity inherent to a dataset which are mainly related to:

- The ratio between the amount of data samples and the data dimensionality;
- The quantity of samples per class and the number of classes;
- The availability of additional data, be it unlabeled samples from the same dataset, data from a similar domain or even cross-domain related data.

The isolation of these three factors has allowed the identification of related tasks, which contribute with methodologies we have taken into account in our review process.

Following this conceptualization step, we have proposed an integrated view of approaches pertinent to solving the multiple facets of the "small data learning" problem. These different small data situations can profit from different strategies, which we have organized in two main axes:

- One data-centered, regrouping methodologies consisting of using additional data in the training process;
- Another model-centered, regrouping methodologies which influence training complexity, either by acting on the architecture or more globally on the training strategy.

This review has allowed us to identify that few works address the issue of small data learning directly. Nevertheless, by identifying its underlying issues, it was possible to rank a considerable amount of possibilities of action, by leveraging related literature in data augmentation, semi-supervision, unsupervised representation learning, transfer learning, metric learning, multi-task learning, meta-learning and curriculum learning. Moreover, boundaries in this framework are not rigid, such that these different aspects in each of this fields may be related to different categories. The presentation of possible solutions is innovative in the sense that it is quite pragmatic and problem oriented: the reader is invited to ask him or herself about his own practical problem, identifying its difficulties and strengths, relating them to the possibilities ranked within the survey.

On the experimental side, we have proposed a novel class-flexible version of Prototypical networks (Snell et al, 2017), an architecture targeted at solving the few-shot learning problem.

In our model the prototypes are no longer fixed-class centroids. Instead, their relationship to each of the classes and their placement in the feature space get both learned from data. This new formulation results in data-representative prototypes, partitioning the feature vector space (as analyzed in section 6.3 and appendix A). Ultimately, data samples will be associated to a feature space partition, each of them associated to class probabilities that can be used to explain the network's predictions, bringing some degree of interpretability to at least the later steps of its decision making process.

This model was analyzed under two small data learning strategies.

- First, a transfer learning paradigm, consisting of reusing a pre-trained CNN as a feature extractor;
- Second, a meta-learning paradigm, similar to the one used in the original prototypical networks.

The first method has an advantage of being faster to train, since features for each data point are fixed, facilitating the study of hyperparameter sensitivity. With this approach, we could demonstrate how the proposed concept of data-prototypes provides a mechanism for explaining network predictions, enhancing interpretability.

The second method involves learning the network end-to-end, thus co-learning feature extraction and the prototype based classification together. Furthermore, with this step we have tackled few-shot learning, one of the many presentation forms of the small data learning problem. This additional experimental track allowed to conclude that adapting the learned representation together is of major importance in achieving higher accuracy, in comparison to the previous fixed-feature method.

## 7.2 Further developments

At the course of this research work, there were many possible paths to be taken, at multiple stages of its development. For each path taken, others inevitably had to be left for future endeavors. Hence some of the possible straightforward extensions and developments following this thesis work are summarized bellow.

- *Study what has been done in other image recognition tasks*: Integrating mechanisms from object detection and segmentation could bring extra information to a classification model trained with limited data, implementing a multi-task learning paradigm. Furthermore, neuroscientific evidence supports the participation of localization information from the dorsal stream to refine recognition in the ventral stream — a successful example of multi-task system. In specific cases, tasks like pose estimation, image inpainting or super-resolution could also be relevant.

- *Study what has been done in video related tasks*: Neuroscience points towards temporal processing being important for disambiguation in segmenting and recognizing an object in a scene, interpolating missing contours and occluded parts. In case short videos can be as easily collected as photographs, integrating motion processing mechanisms and video object segmentation have the potential of helping complicated and ambiguous classification tasks.

- *Study neuro-inspired models not targeted at image classification*: Our visual systems performs different tasks together and simultaneously, with evidence pointing towards many cases of mutual inter-task cooperation. One could expect that while modeling more complex tasks like object tracking could help elucidate fundamental principles underlying the global visual function. One example are predictive learning mechanisms — which involve predicting future stimuli then comparing them to actual perception — that occur all along the visual system. The model proposed by O'Reilly and Rohrlich (2018) for instance relies heavily on this principle, attempting to explain the construction of invariant representations without explicit supervision signals

- *Experiment with recurrent CNN models on small data settings*: Since recurrence can allow to cut down the number of network parameters, it would be interesting to study experimentally if this reduction helps generalization in practice. Before proposing yet another model variant, it would be interesting to test existing propositions reviewed in chapter 4, both in controlled data scenarios — to facilitate the interpretation of results — and in realistic ones — to observe how their practical generalization evolves with decreasing dataset sizes.

- *Experiment with other small data learning methodologies*: Following up on the use of a pre-trained network done in section 6.4.2, an evident next step would be to fine-tune at least its later layers. This could be a less computationally intensive alternative to the end-to-end meta-learning paradigm tested in section 6.4.4. Additionally, unsupervised representation learning methods could be associated to the proposed layer, in a similar pre-training plus fine-tuning paradigm.

- *Experiment with the proposed models on more complex datasets*: Experiments with more complex datasets could maybe indicate whether the prototype class-flexibility could be more useful. A contemporary work, featuring a similar model, with 2 prototypes per class, has shown promising results in a dermatology dataset containing few samples for most of its disease categories (Prabhu et al, 2018). Moreover, it would allow to validate our proposition in more realistic scenarios.

- *Propose a recurrent version of the class-flexible prototype mechanism*: When trained end-to-end, gradients updating networks weights and prototype coordinates are always

computed at every training step. We could for instance have a hybrid arch with recurrent prototype estimation, which would implicitly accelerate prototype refinement in regards to learning on other layers. It would however add a complexity which might not pay off if data is insufficient or of low complexity.

- *Experiment with structuring regularization penalties*: Inducing an underlying structure from the available data samples is one method towards learning a pertinent representation with potential for generalization. It is the case of the LDMNet model (by Zhu et al, 2017b, discussed in section 5.4.1), which encourages a low-dimensional manifold structure onto the last hidden layer, resulting in a lower generalization gap. Analogously, the prototype structure we proposed could be re-formulated as a "soft constraint", with the use of an equivalent regularization penalty (instead of the current "hard" architectural constraint). On one hand, the penalty could involve pushing same-class close-by elements towards the same prototype, but in a multi-modal fashion, similarly to some metric learning losses (e.g. magnet loss Rippel et al, 2016). On the other hand, the loss term could involve enforcing a minimum margin between prototypes to promote sharper boundaries (e.g. Wang et al, 2018b).

## 7.3 Open perspectives

This work has led to the identification of some methodological and technical difficulties surrounding its research topics. Besides more immediate developments, acknowledging these difficulties has motivated broader reflections on open research perspectives, encouraging to rethink the current practices of the field. Hereafter are some comments on the challenges of most relevance to the prospective research ideas proposed in this section:

- *The (excessively) high rate of new publications in DNN literature*: New models are proposed frequently, intending to improve previous results to some extent. However, not that many works try to analyze and bring further understanding onto what has already been created. Most experiments are aimed at reducing error rate a few percent points, often lacking exploratory experiments to understand more about the novel mechanisms proposed (for instance, doing architecture ablation studies, testing robustness to different forms of input perturbation or measuring the generalization gap across reduced dataset sizes).

- *Computational costs*: Powerful GPUs and computing clusters have been playing an important role in the development of deep networks. Despite an increasing access to these computational resources, most actors may never have the capabilities of major tech companies and their leading research groups. On one hand, training multiple variants

of the same model or simply searching for good hyperparameters is necessary. On the other hand, they quickly multiply computational costs associated with any deep learning research or application. Moreover, multiple repetitions would be necessary to potentially obtain statistically meaningful results, increasing computing time even further.

- *Hyperparameter dependency*: There is a strong focus on obtaining practical results (say, high accuracy on benchmark dataset XYZ). Regarding hyperparameters, this focus often implicates on merely finding a good set of values for them, without further reflection on the model's robustness to their variation. Models that are too dependent on them are likely more difficult to be reused in different application domains. Moreover, actors with limited computational budget would be likely inclined towards more robust models, since hyperparameter tuning multiplies computational costs,

- *Statistical soundness of results*: While many search the next-one-percent accuracy improvement, the statistical significance of the obtained results is often unclear. In part due to the computational burden, many works could not repeat experiments over different data splits in order to cross validate their results. Furthermore, the degree to which comparisons of architecturally distinct models are fair or even meaningful is usually vague.

On that account, here we highlight some possibilities of potential research contributions, in relationship with the present work:

- *Take methodological inspiration from neuroscience*: Neuroscience is deeply experimental. To try and understand any given cognitive function, neuroscientists make hypothesis diverse nature, concerning for instance the correlated activity of neuronal populations, the functions they perform locally, the circuits in which the participate and global computations thereby implemented. Experiments are then designed to verify these hypothesis, incrementally ruling out unfitting explanations of the observed brain phenomena. To the end of better specifying functional capabilities of working deep networks, a similar methodological approach could be taken. Exploratory work could look for function specialized neural "regions", "circuits" or "populations". There are already some initiatives encouraging such "reverse engineering" of existing models as a relevant research methodology in the field (e.g. Cammarata et al, 2020). The proximity to mathematical theory may lead the computer science community to dismiss this kind of knowledge as not rigorous enough (in a mathematical sense). Nevertheless, this methodology has allowed great advances in the neuroscientific knowledge, and could bring similar understandings about deep networks. Eventually, experimental insights could inspire novel theoretical explorations (and vice-versa).

- *Understand which functionalities different forms of recurrent processing brings to computer vision*: In a similar spirit to Spoerer et al (2017), further studies involving recurrent CNNs could explore how to use recurrence to implement complex visual functions, which are associated with analogous mechanisms present in the brain. Besides testing on controlled data, it is also relevant to verify if the implemented functions translate to realistic data. On another note, it is also important to study how the introduction of recurrence influences the cost function landscape and thus the optimization process, analogously what has been proposed in studies of forward shortcuts and the ResNet architecture (e.g. Li et al, 2017).

- *Experimentally assess the data needs of standard CNN models*: Theoretical efforts in explaining generalization capability of DNNs have indeed advanced, though the quest for tighter generalization bounds continues. Meanwhile, from an experimental perspective, it could be informative to study generalization ability of the main standard models, on decreasing dataset sizes, in order to estimate their sample size-generalization error relation. Methodological aspects to be careful about comparability between different results, regarding hyperparameters choices. One first possibility is to keep hyperparameters from the original works (although some may need adjustments to allow convergence). A larger computational budget could allow a second methodology including hyperparameter tuning for each subset size.

- *Experimentally assess the transferability of standard CNN models*: Many practical applications of deep CNNs to particular application domains can be found if searching for domain specific literature. Fine-tuning pre-trained networks is a practice frequently found, such that a meta-analysis gathering multiple applications could be informative on the transferability of standard architectures. Cross-referencing results could point towards which models have been more suitable for transfer learning, under which circumstances and under which application domains. One could hypothesize that domains with similar recognition challenges would correlate in their preference for a particular model architecture.

- *Develop methodologies to profit from domain specific implicit knowledge*: Leveraging prior knowledge on the application domain can be a valuable method to improve performances on small datasets, as discussed in chapter 5. In many cases however, knowledge employed in an expert's decision are mostly implicit, making it difficult to transform it in explicit architecture adaptations or training methods. Methodologies to model this implicit knowledge computationally (e.g. Kaadoud, 2018) could be useful in identifying principles to be incorporated in CNN models, helping guide learning towards solutions better adapted to the target application.

# Appendix


**A Prototype model derivation and formal interpretation** **151**
A.1 Sample to prototype association . . . 151
A.2 Reminder on softmax function . . . 151
A.3 Deriving the model variational equations . . . 152
A.4 Analysis of the k-means extended metric . . . 152
A.5 Probabilistic interpretation of representing samples by prototypes . . . 153
A.6 Duality between partition, prototypes and metric . . . 154
A.7 Relation between softmax and prototypes . . . 155

**B Methodology for hyperparameter analysis on prototype model with fixed input features** **157**

**C Relevant benchmark datasets in object recognition** **159**
C.1 MNIST . . . 159
C.2 Omniglot . . . 160
C.3 CIFAR 10 and 100 . . . 160
C.4 ImageNet classification 2012 . . . 160

**D Résumé étendu en français** **162**
D.1 Contexte et motivation . . . 162
D.2 Positionnement du travail et plan de la thèse . . . 166
D.3 Résumé des contributions . . . 167

Appendix A

# Prototype model derivation and formal interpretation

## A.1 Sample to prototype association

We approximate the probability $q_j(\mathbf{x})$ for the feature vector $\mathbf{x}$ of a sample, to be associated to the $j$-th prototype writing:

$$\begin{aligned} q_j(\mathbf{x}) &\overset{\text{def}}{=} {}_{\tau}\nu_j\left(-\|\mathbf{x}-\mathbf{p}_j\|^2/2\right) \\ &= {}_{\tau}\nu_j\left(\mathbf{p}_j^T\mathbf{x}-\|\mathbf{p}_j\|^2/2-\|\mathbf{x}\|^2/2\right) && \text{by expansion} \\ &= {}_{\tau}\nu_j\left(\mathbf{p}_j^T\mathbf{x}-\|\mathbf{p}_j\|^2/2\right) && \text{by offset invariance} \end{aligned}$$

using the soft-max function:

$${}_{\tau}\nu_j\left(\mathbf{x}\right) \overset{\text{def}}{=} \frac{\exp(f_j(\mathbf{x})/\tau)}{\sum_{j'}\exp(f_{j'}(\mathbf{x})/\tau)}$$

writing $f_j(\mathbf{x}) \overset{\text{def}}{=} \mathbf{p}_j^T\mathbf{x}-\|\mathbf{p}_j\|^2/2$, thus $\partial_{\mathbf{p}_j} f_j(\mathbf{x})^T = \mathbf{x}-\mathbf{p}_j$. In the core of the paper we omit the temperature $\tau$ without loss of generality since integrated in the $f_j(\cdot)$ parameters. See appendix A.2 for a review of the basic equations used in this paper.

More generally we could consider $f_j(\mathbf{x}) \overset{\text{def}}{=} -\|\mathbf{x}-\mathbf{p}_j\|^d/d$, yielding $\partial_{\mathbf{p}_j} f_j(\mathbf{x}) * T = \|\mathbf{x}-\mathbf{p}_j\|^{d-2}(\mathbf{x}-\mathbf{p}_j)$, in order to deal with more than the $\mathcal{L}^2$ norm (e.g., for sparse estimation).

## A.2 Reminder on softmax function

The softmax function generalizes the logistic function and enjoys the facts that: $0 \leq {}_{\tau}\nu_j(\mathbf{x}) \leq 1$, $\sum_j {}_{\tau}\nu_j(\mathbf{x}) = 1$, ${}_{\tau}\nu_j(\mathbf{x})_{f_j \to f_j+o} = {}_{\tau}\nu_j(\mathbf{x})$, ${}_{\tau/\tau'}\nu_j(\mathbf{x}) = {}_{\tau}\nu_j(\mathbf{x})^{\tau'} \Big/ \sum_{j'}\nu_{j'}(\mathbf{x})^{\tau'}$, in words, yields probability distributions with with offset invariance and power effect of the temperature $\tau$, while:

$${}_{\tau}\nu_j(\mathbf{x}) = \max_j(f_j(\mathbf{x}))_{\tau\to o} = \tfrac{1}{J} + \tfrac{1}{\tau J}\left(f_j(\mathbf{x}) - \tfrac{1}{J}\sum_{j'} f_{j'}(\mathbf{x})\right) + O\left(\tfrac{1}{\tau^2}\right),$$

computes the maximal values at low temperature, while approximating the given affine operator at high temperature. Moreover, for some parameter $\theta_{j'}$ of the $f_{j'}(\mathbf{x})$ we obtain:

$$\tau\,\partial_{\theta_{j'}}{}_{\tau}\nu_{j'}(\mathbf{x}) = {}_{\tau}\nu_j(\mathbf{x})\left(\delta_{j=j'} - {}_{\tau}\nu_{j'}(\mathbf{x})\right)\partial_{\theta_{j'}} f_{j'}(\mathbf{x}),$$

while for some parameter $\theta$ of the $\mathbf{x}$ we obtain:

$$\tau\,\partial_{\theta}{}_{\tau}\nu_j(\mathbf{x}) = {}_{\tau}\nu_j(\mathbf{x})\left(\partial_{\mathbf{x}} f_j(\mathbf{x}) - \sum_{j'}{}_{\tau}\nu_{j'}(\mathbf{x})\,\partial_{\mathbf{x}} f_{j'}(\mathbf{x})\right)\partial_{\theta}\mathbf{x}.$$

Considering an approximate cross-entropy criterion to adjust the values of a standard

soft-max criterion yields the following minimization:

$$\begin{aligned}
\mathcal{C} &= -\int_{\mathcal{X}} p(\mathbf{x})\ \log\left(\nu(\mathbf{x})\right) \\
&\simeq -\tfrac{1}{N}\textstyle\sum_i \log\left(\nu(\mathbf{x}_i)\right) && \text{approximating } p() \text{ on the data samples} \\
&= -\tfrac{1}{N}\textstyle\sum_i \log\left(\nu_{l_i}(\mathbf{x}_i)\right) && \text{since each sample is associated to one class} \\
&= -\tfrac{1}{\tau N}\textstyle\sum_i f_{l_i}(\mathbf{x}_i) + \log\left(\sum_{j'} \exp(f_{j'}(\mathbf{x})/\tau)\right) && \text{by substitution and factorization}
\end{aligned}$$

yielding the gradient, for a parameter $\theta_{j'}$ of the function $f_{j'}(\mathbf{x})$:

$$-\tau\, \partial_{\theta_{j'}} \mathcal{C} = \left(\tfrac{1}{N}\textstyle\sum_i \delta_{j'=l_i} - {}_{\tau}\nu_{j'}(\mathbf{x})\right)\ \partial_{\theta_{j'}} f_{j'}(\mathbf{x}).$$

## A.3 Deriving the model variational equations

Considering an approximate cross-entropy criterion to adjust the parameters given samples $\mathbf{x}_i$ and their label $l_i$ yields the following minimization:

$$\begin{aligned}
\mathcal{C} &= -\int_{\mathcal{X}} p(\mathbf{x})\ \log\left(c(\mathbf{x})\right) \\
&\simeq -\tfrac{1}{N}\textstyle\sum_i \log\left(c_{l_i}(\mathbf{x}_i)\right) \quad \text{approximating } p() \text{ on the data samples}
\end{aligned}$$

yielding:

$$\begin{aligned}
-N\,\tau\,\partial_{\mathbf{w}_l}\mathcal{C}^T &= \textstyle\sum_i \underbrace{\left[\delta_{l=l_i} - c_l(\mathbf{x}_i)\right]}_{\zeta_i}\ \mathbf{q}(\mathbf{x}_i) \\
-N\,\tau\,\partial_{\mathbf{p}_{j'}}\mathcal{C}^T &= \textstyle\sum_i \underbrace{\left[\sum_j \left(w_{l_i j} - \sum_l c_l(\mathbf{x}_i)\, w_{lj}\right)\ q_j(\mathbf{x}_i)\ \left(\delta_{j=j'} - q_{j'}(\mathbf{x}_i)\right)\right]}_{\xi_{ij'}}\ \ (\mathbf{x}_i - \mathbf{p}_{j'})
\end{aligned}$$

so that we can write, for some sufficiently small $\epsilon$:

$$\Delta\mathbf{w}_l = \epsilon \textstyle\sum_i \zeta_i\, \mathbf{q}(\mathbf{x}_i) \text{ and } \mathbf{p}_j = \textstyle\sum_i \xi_{ij}\, \mathbf{x}_i / \sum_i \xi_{ij}$$

at each step, and leading to:

- expectation (estimated the prototypes as weighted mean over the samples),
- minimization (decreasing the criterion by adjusting the soft-max weights) mechanism.

With such an approach there is a weak clustering of the samples with respect to a prototype, e.g., samples with different labels can be related to the same prototype.

## A.4 Analysis of the k-means extended metric

Let us consider the for a given sample $\mathbf{x}_i$ the augmented feature space of dimension $D + J$: $(\mathbf{x}_i, \mathbf{c}_i)$ where $\mathbf{c}_i \in [0, 1]^J$ are the a-priori probabilities for the sample of index $i$ to belong to each category. For a learning sample $(\mathbf{x}_i, l_i)$ of known category $l_i$ we obviously have $c_{ik} = \delta_{l_i=k}$. Such augmented dimensions act as a level-set over the feature space.

Let us consider a standard k-means algorithm on such extended feature space, for some $\beta > 0$:

$$(\mathbf{p}_j, \mathbf{c}_j) = \sum_i \xi_{ij}\,(\mathbf{x}_i, \mathbf{c}_i) / \sum_i \xi_{ij}, \text{ with } j_{\mathbf{x}} = \arg\min \|\mathbf{x}_i - \mathbf{p}_j\|_2 + \beta\,\|\mathbf{c}_i - \mathbf{c}_j\|^2$$

with $\xi_{ij} = \delta_{j=j_{\mathbf{x}}} \in \{0, 1\}$ for a hard k-means algorithm, while more general soft k-means mechanism with $\xi_{ij} \geq 0$ (as in the previous subsection) could be introduced, the prototype being the centroid of the samples belonging to this cluster.

If $\beta = 0$ the augmented dimensions are not taken into account, and we are left with the original k-means mechanism.

If $2\,\beta > M \stackrel{\text{def}}{=} \max_{ii'} |\mathbf{x}_i - \mathbf{x}_{i'}|^2$ then it is easy to verify that two samples with different categories can not be in the same cluster, providing that the number of prototype is not lower than the number of category.

To verify this fact, let us consider two samples $(\mathbf{x}_i, l_i)$ and $(\mathbf{x}_{i'}, l_{i'})$ with $\mathbf{x}_i \neq \mathbf{x}_{i'}$ and $l_i \neq l_{i'}$ so that

$$\|\mathbf{c}_i - \mathbf{c}_{i'}\|^2 = \sum_k (\delta_{l_i=k} - \delta_{l_{i'}=k})^2 = 2$$

since for $k = l_i$ and $k = l_{i'}$ the values in the summation differs by a value of 1. As a consequence, regarding their extended distance

$$\|\mathbf{x}_i - \mathbf{x}_{i'}\|_2 + \beta\,\|\mathbf{c}_i - \mathbf{c}_{i'}\|^2 = \|\mathbf{x}_i - \mathbf{x}_{i'}\|_2 + 2\beta > M$$

it is higher than any sample pair within the same category. Furthermore, if a prototype corresponds to samples of the same category its within-cluster maximal square distance to each sample is lower than $M$, as being the centroid of the samples belonging to this cluster. As consequence, as soon as two prototypes are used in the algorithm, if these two samples are in the same cluster, the within-cluster distance is going to be higher than any solution with clusters only grouping samples of the same category.

Therefore, considering a k-means algorithm with a proper initialization mechanism that minimizes the within-cluster distance, we have a mechanism that weight the importance of taking the a-priori information about category into account.

## A.5 Probabilistic interpretation of representing samples by prototypes

Representing the sample $\mathbf{x}$ by prototypes means approximating the $\mathbf{x}$ distribution by a distribution only function of the prototypes, e.g.:

$$\mathbf{x} \simeq \mathbb{E}_{\hat{q}}(\mathbf{x}) = \sum_j q_j(\mathbf{x})\,\mathbf{p}_j, \text{ with } \hat{q}(\mathbf{x}) \stackrel{\text{def}}{=} \sum_j q_j(\mathbf{x})\,\delta(\mathbf{x} - \mathbf{p}_j)$$

where $\hat{q}(\mathbf{x})$ approximates the true sample probability distribution $p(\mathbf{x})$ as a discrete distribution, given the prototypes.

Another view is to consider a partition $\{\cdots, P_j, \cdots\}$ of the space induced by the prototypes (i.e., with $\mathbf{p}_j \in P_j$, see Appendix C), yielding:

$$q(\mathbf{x}_i) = \sum_j p(\mathbf{x}_i \in P_j)\,p(\mathbf{p}_j|\mathbf{x}_i) = \sum_j \kappa_{ij}\,q_j(\mathbf{x}_i)$$

for some $\kappa_{ij} = p(\mathbf{x}_i \in P_j)$. Here the partition has not to be made explicit, only the $\kappa_{ij}$ have to be estimated.

Very easily, we obtain $\sum_j \kappa_{ij} = 1$ (since $P_j$ forms a partition), while the $\kappa_j \stackrel{\text{def}}{=} \sum_i \kappa_{ij}$ is the

expectation of the number of sample in the partition, i.e. associated to the prototype. If $\kappa_j = 0$ the prototype is inactive, i.e., not associated to any sample. If $\kappa_{ij} \in \{0, 1\}$, i.e, if we know whether $\mathbf{x}_i \in P_j$ or not, i.e., is related to this prototype or not. In such a situation, the constraint $\sum_j \kappa_{ij} = 1$ states that each sample is associated to a unique prototype, while $\kappa_j \overset{\text{def}}{=} \sum_i \kappa_{ij}$ is the number of samples associated to a given prototype. This restrained modeling is not what is proposed in this paper.

Moreover, the fact we consider for $c_l(\mathbf{x})$:

$$p(l|\mathbf{x}) = {}_\tau\nu_l \left(\mathbf{w}_l^T\, p(\mathbf{p}_j \in Q_j|\mathbf{x})\right) \neq \sum_j p(l|\mathbf{p}_j \in Q_j)\, p(\mathbf{p}_j \in Q_j|\mathbf{x})$$

simply means that we do *not* consider that $Q_j$, namely the set of samples associated to the $j$-th prototype, corresponds to a partition of the sample space $\mathcal{X}$, but that the corresponding regions may overlap, while some regions may not correspond to any samples associated to any prototypes.

## A.6 Duality between partition, prototypes and metric

Given a set of prototypes $\mathbf{P} = \{\cdots \mathbf{p}_j \cdots\}$ in a topological space $\mathcal{X}$ we can consider a partition of the space $P = \{\cdots P_j \cdots\}$ around the prototypes, i.e. such that

$$\mathbf{p}_j \in P_j \neq \emptyset, \quad \cup_j P_i = \mathcal{X}, \quad \forall i, j, \mathring{P}_j \cap \mathring{P}_i = \emptyset.$$

The last condition means that we do not require the intersection to be empty but only its interior (i.e., maximal open set). This notion of partition is thus related to a topology. Roughly speaking, this means that we do not take into account what happens at the frontier between two subsets of the partition. For any point, but those at the frontier between two subsets, we can define its partition subset index $j = p(\mathbf{x}) = \{j, \mathbf{x} \in P_j\}$.

Given a set of prototypes, any metric induces such a partition, writing:

$$P_j = \{\mathbf{x} \in \mathcal{X}, d(\mathbf{x}, \mathbf{p}_j) <= min_{j'} d(\mathbf{x}, \mathbf{p}_{j'})\}$$

On the reverse, given a partition and a topology, the ultrametric:

$$d(\mathbf{x}, \mathbf{y}) = \delta_{\mathbf{x} \neq \mathbf{y}} \left(1 + \delta_{p(\mathbf{x}) \neq p(\mathbf{y})}\right) \in \{0, 1, 2\}$$

is a trivial metric compatible with the partition (i.e. fitting in the previous definition).

More interesting is the fact that given any partition with a metric, and choosing a precision $\epsilon$ there exists a countable set of prototypes such that the given partition is a subset of the partition induced by these prototypes, up to the $\epsilon$ precision. If $\mathcal{X}$ is bounded, e.g. compact, the prototype set if finite. Let us consider this case. Any compact set has always a finite cover. As being a metric space, it always has a finite cover by balls of a given radius, say $B(\mathbf{p}_i, \epsilon/2)$. From this construction, we only keep the prototypes corresponding to the center of balls that intersect the initial partition subsets frontiers. It is then easy to prove that this induce, up to $\epsilon$, a partition:

$$\mathbf{p}_j \in P_j \neq \emptyset, \quad \cup_j P_i = \mathcal{X}, \quad \forall i, j, P_j \cap P_i \subset B(\mathbf{p}_i, \epsilon),$$

where $P_j$ is defined from the metric as given above. Such a construction is related to vector

support machines, the given prototypes being somehow the classification support set.

## A.7 Relation between softmax and prototypes

In our model specification we relate a prototype representation to a softmax function writing

$$q_j(\mathbf{x}) = \nu_j\left(-\|\mathbf{x}-\mathbf{p}_j\|^2/2\right).$$

Let us develop here the inverse relation and see to which extent we can link a softmax function to prototypes.

Consider an affine softmax function:

$${}_\tau\nu_l\left(\mathbf{x}\right) \stackrel{\text{def}}{=} \frac{\exp(f_l(\mathbf{x})/\tau)}{\sum_{j'}\exp(f_{j'}(\mathbf{x})/\tau)}$$

with $f_l(\mathbf{x}) \stackrel{\text{def}}{=} \mathbf{w}_l^T\,\mathbf{x} + w_l^0$ and the decision rule

$$l_{\mathbf{x}} = \arg\max_{l'}{}_\tau\nu_{l'}\left(\mathbf{x}\right)$$

This yields a piece-wise linear segmentation of the feature space[1]. Writing $\mathbf{w}_{ll'} = \mathbf{w}_l - \mathbf{w}_{l'}$ we obtain:

$$l_{\mathbf{x}} = \{l, \forall l' \neq l, \mathbf{w}_{ll'}^T\,\mathbf{x} + w_{ll'}^0 \geq 0\}$$

for the $N\,(N-1)/2$ category pairs.

For each inequality we may consider any pair of points $\mathbf{p}_{ll'}, \mathbf{p}_{l'l}$ for which the separation hyperplane defined by $\mathbf{w}_{ll'}^T\,\mathbf{x} + w_{ll'}^0 = 0$ is the median. We can this write[2]:

[1]Since:

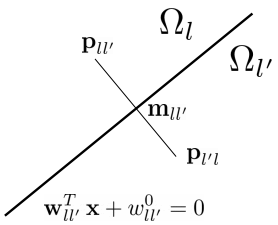
$$\begin{array}{rrcl} & {}_\tau\nu_l\left(\mathbf{x}\right) & \stackrel{>}{<} & {}_\tau\nu_{l'}\left(\mathbf{x}\right) \\ \Leftrightarrow & \frac{\exp(f_l(\mathbf{x})/\tau)}{\sum_{l''}\exp(f_{l''}(\mathbf{x})/\tau)} & \stackrel{>}{<} & \frac{\exp(f_{l'}(\mathbf{x})/\tau)}{\sum_{l''}\exp(f_{l''}(\mathbf{x})/\tau)} \\ \Leftrightarrow & \exp(f_l(\mathbf{x})/\tau) & \stackrel{>}{<} & \exp(f_{l'}(\mathbf{x})/\tau) \\ \Leftrightarrow & f_l(\mathbf{x}) & \stackrel{>}{<} & f_{l'}(\mathbf{x}) \end{array}$$

the denominators being identical and the exponential being a strictly increasing function, we make explicit this piece-wise linear segmentation.

[2]We easily derive:

$$\begin{array}{rrcl} & d(\mathbf{x},\mathbf{p}_{ll'}) & \stackrel{<}{>} & d(\mathbf{x},\mathbf{p}_{l'l}) \\ \Leftrightarrow & d(\mathbf{x},\mathbf{p}_{ll'})^2 & \stackrel{<}{>} & d(\mathbf{x},\mathbf{p}_{l'l})^2 \\ \Leftrightarrow & \|\mathbf{x}\|^2 - 2\mathbf{p}_{ll'}^T\,\mathbf{x} + \|\mathbf{p}_{ll'}\|^2 & \stackrel{<}{>} & \|\mathbf{x}\|^2 - 2\mathbf{p}_{l'l}^T\,\mathbf{x} + \|\mathbf{p}_{l'l}\|^2 \\ \Leftrightarrow & -2(\mathbf{p}_{ll'}-\mathbf{p}_{l'l})^T\,\mathbf{x} + \|\mathbf{p}_{ll'}\|^2 - \|\mathbf{p}_{l'l}\|^2 & \stackrel{<}{>} & 0 \\ \Leftrightarrow & -4\,\lambda\,\mathbf{w}_{ll'}^T\,\mathbf{x} + \|\mathbf{p}_{ll'}\|^2 - \|\mathbf{p}_{l'l}\|^2 & \stackrel{<}{>} & 0 \\ \Leftrightarrow & -4\,\lambda\,\mathbf{w}_{ll'}^T\,\mathbf{x} - 4\,\alpha\,\lambda\|\mathbf{w}_{ll'}\|^2 & \stackrel{<}{>} & 0 \\ \Leftrightarrow & -4\,\lambda\,\left(\mathbf{w}_{ll'}^T\,\mathbf{x} + w_{ll'}^0\right) & \stackrel{<}{>} & 0 \\ \Leftrightarrow & \mathbf{w}_{ll'}^T\,\mathbf{x} + w_{ll'}^0 & \stackrel{>}{<} & 0 \end{array}$$

.




$$\begin{aligned} \mathbf{m}_{ll'} &\stackrel{\text{def}}{=} \mathbf{w}_{ll'}^{\perp} - \alpha\, \mathbf{w}_{ll'} \\ \mathbf{p}_{ll'} &\stackrel{\text{def}}{=} \mathbf{m}_{ll'} + \lambda\, \mathbf{w}_{ll'} \\ \mathbf{p}_{l'l} &\stackrel{\text{def}}{=} \mathbf{m}_{ll'} - \lambda\, \mathbf{w}_{ll'} \\ \alpha &\stackrel{\text{def}}{=} w_{ll'}^0 / \|\mathbf{w}_{ll'}\|^2 \end{aligned}$$

for any $\lambda > 0$ and any vector $\mathbf{w}_{ll'}^{\perp}, \mathbf{w}_{ll'} \perp \mathbf{w}_{ll'}^{\perp} = 0$. These pairs of points $\mathbf{p}_{ll'}, \mathbf{p}_{l'l}$ are thus defined up to $D\,N\,(N-1)/2$ degrees of freedom ($\lambda$ and $\mathbf{w}_{ll'}^{\perp} \in \mathbb{R}^D$ subject to an orthogonality constraint, for each category pair).

Furthermore, from a geometrical point of view, it is coherent to require that $l_{\mathbf{p}_{ll'}} = l$, in words that each prototype belongs to a polytope $\Omega_l$ corresponding to the label $l$, i.e., that $\forall l, l', l'' l \neq l', l \neq l'', \mathbf{w}_{ll''}^T \mathbf{p}_{ll'} + w_{ll''}^0 > 0$. This corresponds to a linear programming problem with $D\,N$ degrees of freedom and $(N\,(N-1)/2)^2$ inequalities, thus with solutions in the general case as soon as $D > N^3/4 + O(N^2)$, i.e., as soon as the number of features is high enough with respect to the number of categories.

Is it possible to reduce the number of prototypes, i.e., that $\mathbf{p}_{ll'} \stackrel{?}{=} \mathbf{p}_{ll''}$ for some of the $N\,(N-1)$ prototypes ? This generates $N$ additional linear constraints for each prototype merge, and the non trivial fact that prototype pairs are to live into the same connected component of a given $\Omega_l$ has to be taken into account. A simple count of the number of degrees of freedom shows that the number of constraints is in the general case twice the number of possible adjustment. Prototype merge is thus possible, but not completely.

The softmax decision rules is thus equivalent to a nearest-neighbor algorithm considering $O(N^2)$ prototypes.

Appendix B

# Methodology for hyperparameter analysis on prototype model with fixed input features

As far as significance and interpretability of the results is concerned a key point is the influence and adjustment of the algorithm hyper-parameters. We have already discussed those for which it was worth studying specifically their influence, but let us now briefly review exhaustively all of them.

Experiments are performed in Python with help of basic scientific libraries, especially the machine learning library scikit-learn (Pedregosa et al, 2011), from which comes implementations for k-means and cross-entropy multinomial (softmax) regression.

Regarding the k-means algorithm and its k-means++ initialization heuristic, we have to consider:

- The number $J$ of clusters, studied in this section.
- The number $R$ of random draws of initial conditions of the expectation-minimization algorithm.
- The maximal number $K$ of iteration of the expectation-minimization algorithm.
- The tolerance $E$ on the criterion variation in order to detect the convergence.
- The extended criterion hyper-parameter $\beta$ introduced in the previous section.
- A choice between two algorithm variants which mainly differ in computational efficiency, the faster one being chosen.

Here given a data set we easily compute the maximal and minimal distances between clusters:

$$M \overset{\text{def}}{=} \max_{ii'} |\mathbf{x}_i - \mathbf{x}_{i'}|^2 \text{ and } m \overset{\text{def}}{=} \min_{ii'} |\mathbf{x}_i - \mathbf{x}_{i'}|^2,$$

allowing us to adjust of fix these hyper-parameters.

We have observed that the number $R$ of random draws is not significant as soon as sufficient (typically $R > 10$). The maximal number of iterations $K$ has not to be bounded because the algorithm always converges. The tolerance $E$ is easy to infer from the data, because as soon as the expectation step of the k-means algorithm does not "redistribute" samples between clusters the algorithm is expected to converge in one step. This occurs as soon as the criterion variation magnitude is lower than the minimal distance between samples, i.e. as soon as $E \ll m$, say 10 times lower.

The number $J$ of clusters is to be adjusted as studied in this section, but we know in which bounds, since we can easily set as minimal number the number of categories (i.e., considering one cluster by category), and set as maximal number the number of learning samples (i.e., falling back to a nearest-neighbor algorithm).

Finally, through $\beta$ is a parameter to be observed and adjusted as studied in this section, we also know in which bounds, since $\beta = 0$ corresponds to the not taking a-priori information into account and $\beta = M$ to hard-wire prototypes on categories, as analyzed in appendix A.4.

Regarding the second step of the method — the cross-entropy minimization — we have to consider similarly:

- The choice of minimization solver.
- The maximal number $K$ of iterations of the minimization algorithm.
- The tolerance $E$ on the criterion variation in order to detect the convergence.
- The choice between $\mathcal{L}^D, D \in \{1, 2\}$ regularization.
- The regularization balance weight $C$.

A softmax function being considered here, we can again calculate the output variation under which the algorithm convergence is negligible. The classification decision does not vary for variations of the output below $c = \min_{jj'} |c_j = c_{j'}|$ as being a comparison between two outputs, while $c_j \in [0, 1]$.

For L2 penalty, L-BFGS solver was chosen for its faster convergence (less iterations). The algorithm always converged allowing us not to consider $K$ as a significant value, at a tolerance $E = 10^{-4}$. Given some restrictions by the library, only one solver option was available for L1 penalties (SAGA solver). Tolerance was relaxed for SAGA solver so as to achieve convergence within the same $K$ iterations ($E = 10^{-1}$). The initial point for both solvers is always taken at zero, thus no hyper-parameter or heuristic is to be considered.

Beyond these parameters, the more significant parameters $C$ and $D$ have been studied in the text.

In addition to these two sets of parameters we have discussed the $\alpha \in [0, 1]$ shortcut gain allowing us to better understand the performances ans limit of our method.

All together, the literature knowledge, simple rule of thumbs on the data values, the concrete understanding of the proposed algorithms and specific numerical studies of more critical parameters allows us to both propose a reproducible piece of experimental results and a method than can be reused without any opaque or application dependent hyper-parameters adjustment, that are not done by the hyper-parameter adjustment layer or the proposed method.

Appendix C

# Relevant benchmark datasets in object recognition

The purpose of this appendix is to provide a brief description on datasets mentioned throughout the thesis. It is meant to be a quick reference for readers, summarizing the most relevant information on each dataset. On each case, the reader is invited to refer to the original dataset companion papers (cited in table C.1 bellow)for more detailed descriptions.

**Table C.1** Information on datasets mentioned throughout this thesis.

| Dataset | Domain | Input space | Number of images (per class) | | | | Number of classes |
|---|---|---|---|---|---|---|---|
| | | | total | training | validation | test | |
| MNIST (Lecun et al, 1998) | Handwritten digits | 28×28 grayscale | 70 000 | 60 000 (6000) | - | 10 000 | 10 |
| CIFAR-10 (Krizhevsky, 2009) | General purpose photographs | 32×32 RGB | 60 000 | 50 000 (5000) | - | 10 000 | 10 |
| CIFAR-100 (Krizhevsky, 2009) | General purpose photographs | 32×32 RGB | 60 000 | 50 000 (500) | - | 10 000 | 100 |
| Oxford Flowers-102 (Nilsback and Zisserman, 2008) | Flower species | RGB, various sizes | 8 189 (40–258 per class) | 1 020 ( 10) | 1 020 ( 10) | 6 149 (20-238, 60 avg) | 102 |
| ImageNet classification (Russakovsky et al, 2015) | General purpose photos from Flicker and alike | RGB, various sizes | +1.4 million | 1 281 167 (732–1300) | 50 000 (50) | 100 000 (100) | 1000 |
| Stanford Cars-196 (Krause et al, 2013) | Car photos categorizes by Make-Model-Year | RGB, various sizes | 16 185 | 8 144 (50% of each class) | - | 8 041 (50% of each class) | 196 |
| CUB-Birds-200 2011 (Wah et al, 2011) | Bird photos categorized by species | RGB, various sizes | 11 788 | 5 994 | - | 5 794 | 200 |

| Dataset | Domain | Input space | Number of images | Number of classes | | | |
|---|---|---|---|---|---|---|---|
| | | | | meta-training | meta-validation | meta-test | total |
| Omniglot (Lake et al, 2011) | Handwritten characters from 50 different alphabets (30 for meta-training, 20 for meta-testing) | 100×100 grayscale | 32 460 (20 per char.) | 1200 characters | - | 423 characters | 1623 characters |
| mini-Imagenet (Ravi and Larochelle, 2017) | Subset of ImageNet 2012 | RGB, 84×84 | (600 samples per class) | 64 | 16 | 20 | 100 |
| mini-Imagenet - closed (Vinyals et al, 2016) | Subset of ImageNet 2012 | RGB, 84×84 | (600 samples per class) | 80 | - | 20 | 100 |

## C.1 MNIST

MNIST is a dataset containing small images of handwritten digits, labeled 0 to 9. Pictures have been basically pre-processed to have digits centered and size-normalized. A large number ($\tilde{6}000$) of straining samples per digit-class is available (see more details in table C.1).

Dataset compilation was part of the work of Lecun et al (1998), which achieved significant success with a backpropagation trained convnet. Nowadays, this dataset is usually serves

as a starting point for tutorials, or a sanity-check debugging dataset to verify minimum function of classification models. It is also often used to demonstrate how different methods learn different embeddings for the same data, and also how different visualization methods differently portray a same high-dimensional embedding space in 2D or 3D.

## C.2 Omniglot

Omniglot dataset was organized by Lake et al (2011). It consists of images of 1622 characters from 50 different alphabets, each drawn by 20 different people (see table C.1 for more details). The small number of samples per class and the large number of classes make it a challenging problem to learn all the classes simultaneously. This dataset is generally used in meta-learning works (few-shot and one-shot learning, see section 5.4.3.1 and section 6.2.2) in which a meta-model is meta-trained on 30 alphabets, then used to learn classifiers over the remaining 20 alphabets.

## C.3 CIFAR 10 and 100

**Images** CIFAR datasets (Krizhevsky, 2009) are composed of tiny 32×32 color images, labeled into 10 or 100 categories. It can be manually downloaded at `https://www.cs.toronto.edu/~kriz/cifar.html`, although most deep learning packages have dedicated functions to download and use within the training code. Original images are issued from the *80 million tiny images* dataset (Torralba et al, 2008)[1].

**Labels** In CIFAR-10, images are labeled into one of 10 categories: airplane, automobile, bird, cat, deer, dog, frog, horse, ship, and truck. For CIFAR-100, images are organized in 20 superclasses, each containing 5 specific classes, resulting in total 100 labels. These finer-grained classes include more animals (like bees, butterflies or kangaroos), small objects (like cups and bowls), tree types (like oak, pine), people, vehicles, among others.

Images having very low resolution, it is expected that some objects are hardly identifiable — even for humans. Therefore close to perfect model performances are not necessarily a goal per se and should be interpreted carefully.

## C.4 ImageNet classification 2012

ImageNet designates a large-scale database, available at `http://image-net.org`, first organized by Princeton University (Deng et al, 2009), later joined by Stanford and others

[1]Tiny Images dataset was later withdrawn due to identification of bias against political minorities (in the work of Prabhu and Birhane, 2020), see `https://groups.csail.mit.edu/vision/TinyImages/` for a retraction note.

(Russakovsky et al, 2015). As of August 2020, the official web page reports 14 197 122 images, related to 21 841 currently indexed concepts (also called "synonym sets" or *synsets*). This database has been the source of datasets feeding the ImageNet large scale visual recognition challenge (ILSVRC), a competition held for 8 years, between 2010 and 2017.

Although the challenge was not restricted to classification tasks — also comprising object localization and detection — it is the classification dataset which got to be commonly referred as *the* "ImageNet" dataset. Its form and content have remained unchanged since the 2012 edition — in which CNNs were top-ranked for the first time — and it is often used as data source for most off-the-shelf pre-trained networks available online.

**Images** Images are photographs having been collected online from Flickr[2] and other search engines. This has resulted in a diverse dataset, both in terms of format and content. Size and aspect ratio vary widely, as well as the the number of objects, clutter, and their pose within the image. Since not all images are free of rights, an individual access demand is required prior to downloading the data. More information is provided in table C.1.

**Labels** Image categories are based upon Wordnet 3.0 — a previously established word hierarchy.Content covers natural scenes and living-beings, as well as man-made objects and settings. The chosen 1000 label set contains both leaf and internal nodes taken from this hierarchy, meaning images can naturally belong to more than one category — through eventual parent-child relations within the concept tree. Additionally, photographs often depict multiple objects within the same scene. Nonetheless, the classification dataset counts with a single ground truth label per image.

### C.4.1 mini-ImageNet

This is a subset devised by Vinyals et al (2016) for meta-learning few-shot experiments. They sampled 100 classes out of the original 1000, collecting 600 examples per class. 80 classes are used for training and 20 for testing. Unfortunately the actual classes and data splits used were not released at the time, leading to the creation of a new subset shortly after: Ravi and Larochelle (2017) have sampled a different set of 100 classes, divided into 64 for meta-training, 16 for meta-validation and 20 for testing (table C.1). Since this second subset has been publicly released[3] it ended up being the most used by follow-up works in few-shot learning.

[2] `https://www.flickr.com/` is an online space for uploading and sharing photographs and graphic content in general.

[3] Available at `https://github.com/twitter/meta-learning-lstm/tree/master/data/miniImagenet`.

Appendix D

# Résumé étendu en français

## D.1 Contexte et motivation

### D.1.1 La revolution du « deep learning »

La reconnaissance d'images a connu une révolution au cours de la dernière décennie, principalement suite aux améliorations de performances obtenues grâce aux réseaux neuronaux profonds. Même si ces modèles existaient depuis la fin des années 80 – début des années 90 (LeCun et al, 1990a), leur application réussie avait été limitée par les exigences en matière de données et de matériel de calcul. Les progrès récents en pouvoir de calcul — avec la possibilité de traiter les opérations à grande matrice en parallèle avec les consommateurs GPUs — et dans la disponibilité des données — avec l'utilisation généralisée des appareils photo numériques et des plateformes de partage de photos en ligne — ont été décisifs pour changer ce scénario. L'amélioration de 12 % en précision dans l'édition 2012 du concours ILSVRC est emblématique de cette révolution, qui a fini par être un jalon d'une utilisation réussie des réseaux convolutifs (CNNs) pour la reconnaissance d'objets à usage général.

Ces modèles d'apprentissage machine ont non seulement atteint une précision sans précédent en matière de reconnaissance d'objets, mais ont également apporté un nouveau paradigme d'extraction et de catégorisation des caractéristiques d'apprentissage en commun, au sein d'un modèle unique. En effet, ce paradigme d'entraînement *de bout en bout*, en faisant correspondre directement les images aux étiquettes classes, est devenu un standard *de facto* dans la plupart des travaux de vision par ordinateur. Il est souvent avancé que cette façon de procéder — en apprenant à partir d'exemples — est plus proche de la façon dont les humains voient et apprennent à reconnaître les objets dans la nature. Cependant, de telles similitudes entre la vision informatique et la vision neurale biologique existent à un niveau plutôt superficiel, alors que nos neurones et nos régions corticales fonctionnent de manière beaucoup plus complexe, mettant en œuvre des fonctions absentes des CNNs typiques.

### D.1.2 Limites de la bio-inspiration

Les réseaux de neurones sont constitués de couches successives de neurones artificiels interconnectés, qui s'inspirent vaguement du fonctionnement des cellules neuronales naturelles. Plus précisément, ils modélisent la propagation de l'information à seuil effectuée par les neurones, lorsqu'ils transmettent sélectivement un potentiel d'action électrique. Un neurone artificiel est une compositions de fonctions mathématiques, qui calcule d'abord une combinaison linéaire de ses entrées, puis fait passer la sortie résultante par une non-linéarité qui imite le mécanisme de seuil biologique. Cette modélisation est très simplifiée et néglige tout aspect temporel pertinent pour le fonctionnement des vrais neurones.

La bio-inspiration est également présente à un niveau plus élevé, dans la façon dont les neurones sont organisés et les modèles qu'ils apprennent. Les réseaux profonds (DNNs) prennent leur prédicat de dénomination de leur architecture archétypique consistant en un empilement de multiples couches de neurones. Chaque couche reçoit les sorties de son homologue précédent, effectuant ensuite une autre étape de traitement de l'information, avant de transmettre le résultat à la couche suivante. Cette structure de traitement incrémentielle, lorsqu'elle est formée sur des ensembles de données d'images naturelles, a été démontré comme fonctionnant de manière hiérarchique, similaire à l'organisation fonctionnelle des différentes régions corticales impliquées dans notre propre vision biologique. Les premières couches ont tendance à apprendre à détecter les caractéristiques des images de bas niveau telles que les parties de bordures dans différentes orientations — dans une analogie fonctionnelle avec notre cortex visuel primaire — tandis que les couches suivantes se spécialisent dans la détection de concepts plus abstraits et, en fin de compte, dans la classe des images — dans un rôle analogue à celui du cortex IT.

Bien que cette similitude fonctionnelle semble être présente, il reste beaucoup de choses inexpliquées dans les CNNs *feedforward* typiques. Les connexions de raccourci entre régions, le traitement local récurrent, les connexions de *feedback* à longue portée, la modulation d'attention — ce sont autant de caractéristiques absentes de ces modèles et réputées pour jouer un rôle important dans le traitement d'une scène visuelle. On sait par exemple que le traitement récurrent permet de reconnaître des objets sous un encombrement visuel, une tâche dans laquelle les CNNs peuvent avoir quelques difficultés. On peut se demander comment l'incorporation de telles fonctionnalités pourrait aider les performances des CNN, à la fois de manière quantitative et qualitative. En outre, on peut se demander laquelle de ces améliorations serait la plus pertinente dans un scénario particulièrement difficile.

Bien qu'une étude détaillée de ce problème ne soit pas l'objectif principal de cette thèse (le lecteur peut se référer à (Medathati et al, 2016) pour un examen détaillé), ce travail est prudent lorsqu'il s'agit de bio-inspiration et de plausibilité des architectures révisées et développées. Sans s'attacher à un niveau particulier de plausibilité biologique, la fonction et la structure biologiques ont servi de référence transversale pour à la fois les développements

expérimentaux et la revue de la littérature, qui sont abordés à plusieurs reprises tout au long de ce travail.

### D.1.3 Le problème des petits corpus de données

Tous ces progrès sont toutefois liés à la disponibilité généralisée de corpus de données à grande échelle, avec des milliers d'échantillons par catégorie d'images, s'élevant à des millions d'images dans l'ensemble des données. Comme les grands réseaux neuronaux ont un nombre énorme de paramètres à apprendre, un grand nombre d'exemples est nécessaire pour que le modèle d'apprendre une représentation pertinente de l'espace de présentation et de généraliser ses prédictions. Sur des ensembles de données plus petits, de tels modèles de grande taille sont susceptibles de sur-ajuster aux données d'entraînement et de ne pas généraliser aux données cibles.

De nombreux domaines, si ce n'est la plupart, sont en fait plus susceptibles de produire de petits corpus de données étiquetés, pour une multitude de raisons. Par exemple, dans le domaine des diagnostics par imagerie médicale, les essais avec chaque patient sont limités, certaines conditions sont rarement rencontrées et l'étiquetage est coûteux car il exige l'avis d'un ou plusieurs experts. Dans l'industrie, le contexte des images prises à partir d'une usine de production, d'une machine particulière ou d'une étape d'un pipeline peut être suffisamment spécifique pour qu'un corpus de données dédié doive être collecté sur place. Ce même corpus de données nécessitera plus tard qu'un technicien expert identifie manuellement toute condition défectueuse ou déviante. Le fait est que dans la plupart des organisations qui ne travaillent pas directement avec la collecte de données, mais simplement désireuse d'automatiser un certain processus interne, la composition d'un corpus de données dédié n'est pas simple, ni en termes de coût ni de logistique. Par conséquent, dans la plupart de ces cas, on peut être confronté à des conditions difficiles en termes de disponibilité des données pour l'application de toute méthode d'apprentissage machine, et l'apprentissage profond en particulier.

Dans de nombreux cas, le premier réflexe d'un ingénieur en apprentissage de machines bien formé peut être de revenir à des modèles moins complexes — comme les arbres de décision par exemple — et en particulier à des versions d'ensemble de ceux-ci — comme les forêts aléatoires ou XGboost. Mais n'oublions pas non plus notre intérêt pour l'exploitation des données sous forme d'images. Les caractéristiques orientées vers les données apprises par les modèles profonds semblent jouer un rôle fondamental dans la les récentes percées en matière de reconnaissance d'images. Les méthodes précédentes de conception de caractéristiques (*feature engineering*) n'ont pas eu un niveau de succès comparable dans la catégorisation des images. En outre, ces caractéristiques spécifiques aux données peuvent être d'autant plus précieuses pour la reconnaissance quand on est dans un domaine très particulier, dans lequel les caractéristiques visuelles générales peuvent ne pas être aussi discriminantes comme le

serait un ensemble de caractéristiques spécialisées liées à une tâche ou à un domaine. Cette volonté conflictuelle de modéliser de petits corpus de données de manière centrée sur les données est la principal motivation derrière le présent travail. Tout au long de ce manuscrit, nous cherchons à comprendre l'apprentissage des caractéristiques hiérarchiques effectué par les réseaux neuronaux profonds — en particulier les architectures convolutives — tout en définissant les circonstances dans lesquelles elles peuvent être exploitées avec succès dans le cadre de la disponibilité limitée des données pour l'entraînement. Une exploration détaillée de ce problème avec une analyse de ses différentes parties constitutives constitue une part importante du travail de cette thèse.

### D.1.4 Litterature sur le « deep learning »

Dans l'ensemble, la recherche dans la communauté de l'apprentissage machine s'est fortement orientée récemment vers le travail en réseau profond. En effet, les conférences spécifiquement liées au sujet (NeurIPS, ICML, IJCNN, entre autres) ont connu une augmentation exponentielle du nombre de soumissions et de la participation globale. En outre, la pré-publication d'articles sur des plateformes en ligne ouvertes, telles que ArXiv, a également donné et accéléré le rythme dans ce domaine, avec des interactions — des citations et des travaux dérivés — qui se passent tous dans un rythme beaucoup plus rapide qu'un cycle de publication typique, avec examen par les pairs. Il s'agit d'une question difficile et longue en soi de naviguer à travers ce corpus littéraire toujours croissant, en essayant de discerner les tendances prometteuses des idées éphémères.

En essayant de se concentrer sur le problème spécifique de l'apprentissage avec petits corpus de données — *small data learning*, on finira par se rendre compte qu'elle entraîne quelques difficultés plus particulières. Premièrement, il ne s'agit pas d'un seul mais de plusieurs problèmes. On peut parler de plusieurs contextes pratiques différents comme étant une forme d'apprentissage *small data*. Deuxièmement, peu d'ouvrages ont tenté de mesurer et d'aborder directement la difficulté de l'entraînement CNNs sur de petits corpus de données, en abordant plutôt les difficultés similaires rencontrées dans d'autres sous-domaines (par exemple, catégorisation fine, apprentissage métrique, apprentissage par transfert, méta-apprentissage, pour n'en citer que quelques-uns). La combinaison de ces deux aspects conduit à une troisième observation générale : l'état actuel des connaissances dans ce domaine n'est pas facilement disponible à toute personne ayant un intérêt naissant pour ce problème. Au lieu de cela, il faut lire des papiers de plusieurs domaines connexes afin d'établir les défis et les leviers d'action face à toute forme particulière de rareté des données. Il est donc primordial de revisiter ces différents domaines, en structurant et en reliant des études apparemment déconnectées qui couvrent ces multiples aspects du sujet. Cet effort d'organisation des connaissances est l'un des aspects majeurs du travail de cette thèse.

## D.2 Positionnement du travail et plan de la thèse

Dans la première section de cette introduction, nous avons établi la difficulté et l'intérêt général d'appliquer l'apprentissage approfondi à de petits ensembles de données. Même en limitant le champ d'application à la reconnaissance d'objets, et en particulier à l'image il existe encore une grande diversité d'approches dans ce domaine. Compte tenu de l'évolution rapide de la recherche sur l'apprentissage profond et du vaste corpus de littérature, il est indispensable de faire le point sur l'état de l'art afin de placer toute proposition nouvelle. En outre, comme les scénarios de données de petite taille sont multiples, il devient nécessaire de clarifier et de distinguer les différentes versions de ces problèmes avant d'aborder un cas particulier. Nous avons donc abordé cette question à travers deux axes méthodologiques principaux (voir figure 1.1) :

- Un travail d'analyse documentaire approfondi visant à présenter une vision claire et concise du domaine et le problème à traiter (cf. chapitres **??**) ;
- Propositions expérimentales explorant de nouveaux mécanismes dans le cadre révisé (cf. chapitre 6).

**Plan** Il ressort également de notre discussion que le problème de la reconnaissance d'objets avec des CNN sur de petits ensembles de données est en fait un méta-problème englobant de multiples sous-problèmes. En tant que tel, il est nécessaire (i) d'identifier les parties qui le composent et (ii) de décider lesquelles et comment traiter chacune d'entre elles. Nous avons donc décidé d'organiser ce travail en deux parties, en abordant d'abord la reconnaissance des objets (part I), suivi d'un traitement spécifique du petit cas de données (part II).

Dans la part I, nous avons abordé la reconnaissance d'objets d'un point de vue interdisciplinaire, en discutant des CNN artificielles ainsi que de leurs aspects d'inspiration biologique :

- Tout d'abord, nous présentons le fonctionnement de base de la reconnaissance d'objets, à la fois dans les CNN et dans le cerveau (cf. chapitre 2). En particulier, nous passons en revue les travaux récents comparant les CNN et les représentations corticales, et résumer les caractéristiques fonctionnelles et architecturales non modélisées par les CNN typiques.
- Comme les capacités modernes de reconnaissance d'objets ont été réalisées avec les CNNs feedforward, cette catégorie de modèles est examinée et organisée dans le chapitre 3. Alors que d'autres examens sur le sujet ont été publiés au cours de ce travail de thèse, ils n'ont pas nécessairement formulé le sujet d'une manière pertinente pour le travail développé ici. Il était donc en tout cas pertinent de réviser l'histoire du progrès des principes et des innovations architecturales, en mettant en évidence les tendances pertinentes pour le présent travail.

- Complétant le travail de révision précédent, le chapitre 4 traite des modèles CNN récurrents et de rétroaction pour la catégorisation des images. Le chapitre vise à organiser les modèles examinés en fonction de leur utilisation de la récurrence, d'un point de vue fonctionnel et architectural. En plus de proposer une étude de la littérature avec des perspectives et des points de vue innovants, l'étude des traitements récurrents dans les CNN pour la catégorisation des images illustre également le potentiel d'échange de connaissances

Après avoir posé notre paradigme sur la reconnaissance d'objets, nous abordons les conditions correspondant à l'apprentissage sur *small data* dans II en deux étapes :

- Une description plus détaillée du problème est présentée dans le chapitre 5, y compris un examen des principales stratégies présentes dans la littérature sur l'apprentissage approfondi. Ce chapitre est un effort d'organisation des connaissances existantes dans le domaine, tout en les reliant à des tâches connexes présentant des difficultés similaires.
- Finalement, le travail est complété par des propositions expérimentales dans le cadre des *small data*, y compris la proposition de nouveaux mécanismes et perspectives, en s'appuyant sur les modèles profonds existants.

En fin de compte, un travail de thèse typique a tendance à ouvrir beaucoup plus de questions de recherche qu'il n'en ferme. Après un travail d'examen approfondi, il est naturel d'identifier les espaces de contribution et d'interaction possibles entre les domaines qui pourraient contribuer à faire progresser les connaissances. Comme ceux-ci pourraient faire l'objet de nombreuses autres thèses, ces perspectives ouvertes seront exposées et discutées dans la conclusion de la thèse.

## D.3 Résumé des contributions

### D.3.1 Étude et compréhension de la reconnaissance d'objets d'un point de vue interdisciplinaire

En comparant les CNN et les modèles neuroscientifiques de la vision (passés en revue au chapitre 2), Des similitudes ont été trouvées dans de multiples travaux interdisciplinaires. Sur le plan fonctionnel, il a été démontré que les couches précoces et tardives sont hautement prédictives de l'activité des zones corticales V1-V2 et V4-IT, respectivement, en considérant les données des deux réponses neurophysiologiques (**?**Yamins and DiCarlo, 2016) et l'activité IRMf (Eickenberg et al, 2016). D'un point de vue plus « comportemental », Il a été démontré que les CNN prennent des décisions de catégorisation similaires à celles prises par les primates

(lorsqu'ils sont entraînés à des tâches suffisamment difficiles), ayant présenté des modèles de matrice de confusion similaires à ceux des humains et des singes (Rajalingham et al, 2018).

Ces similitudes se limitent à la reconnaissance rapide des objets du noyau cérébral, correspondant aux cent premières millisecondes de traitement cortical. Il a été démontré que des traitements visuels plus complexes dépendent de principes fonctionnels et de chemins architecturaux non modélisés dans les CNN typiques. Nous avons étudié trois aspects principaux :

- Traitement et codage avancés effectués respectivement dans la rétine et les structures thalamiques, tels que la compression spatio-temporelle et l'intégration computationnelle de plusieurs zones corticales ;
- Interaction entre les flux dorsal et ventral — donc entre la reconnaissance et la localisation. Les flux dorsaux et ventaux partagent des informations sur le mouvement, la couleur et la forme, à plusieurs niveaux des deux hiérarchies ;
- Le rôle des connexions récurrentes de traitement et de rétroaction vers une perception visuelle plus raffinée dans des conditions difficiles. Récurrences locales et à long terme — y compris les connexions susmentionnées avec les zones thalamiques et dorsales — sont cruciales pour permettre d'affiner et de désambiguïser les frontières entre les objets, l'interpolation des informations manquantes à partir des récurrences spatiales locales et l'intégration des signaux de traitement des mouvements.

De ces trois principes, le traitement récurrent est le plus général, car il est présent à plusieurs niveaux de la hiérarchie de la vision, ce qui a motivé l'examen de la manière dont ce mécanisme a été intégré aux CNN dans la littérature récente.

Depuis les premiers CNN simples et à flux continu, de nombreuses innovations architecturales ont été présentées au fil des ans, comme le montrent les chapitres **??**. Les deux chapitres ont organisé les architectures récurrentes de feedforward et de feedback en fonction de leurs modèles de connectivité, ce qui donne la taxonomie résumée dans la figure 7.1, où chaque famille est représentée par un petit module archétypal de 3-4 couches (dans une architecture typique, plusieurs de ces modules peuvent être associés pour former le réseau complet).

Ce travail a permis de déterminer dans quelle mesure les principes bio-inspirés sont pris en considération pour les architectures ciblées sur la catégorisation des images. Des parallèles ont été établis entre certaines innovations architecturales — tels que les multiples raccourcis en avant (*feedforward*) et les connexions de retour d'information (*feedback*) — et des mécanismes anatomiques/fonctionnels dans le pipeline de traitement visuel neural. Néanmoins, ces mécanismes ne sont pas bien compris en ce qui concerne les fonctionnalités de haut niveau qu'ils apportent à la vision artificielle sur des images réalistes.

### D.3.2 Classification d'images sur des corpus restreints avec des réseaux profonds

Après avoir placé notre vue sur l'état actuel de la classification des images, nous avons procédé à l'encadrement du cas particulier des petits ensembles de données. Comme nous l'avons vu dans le chapitre 5, parmi la grande variété et l'hétérogénéité des approches de la question, nous avons identifié cinq situations générales susceptibles de relever des dénominations "Apprendre sur de petits ensembles de données ou "apprentissage sous données limitées" (résumé dans la section 5.2.1 et la figure 5.1). Nous avons proposé cette taxonomie après avoir identifié trois principaux leviers influençant le niveau de rareté de l'information inhérent à un corpus de données qui sont principalement liés à

- Le rapport entre la quantité d'échantillons et la dimensionnalité des données ;
- La quantité d'échantillons par classe et le nombre de classes ;
- La disponibilité de données supplémentaires, qu'il s'agisse d'échantillons non-étiquetés provenant du même corpus de données, des données d'un domaine similaire ou même des données liées à plusieurs domaines.

L'isolement de ces trois facteurs a permis d'identifier des tâches connexes, qui contribuent aux méthodologies que nous avons prises en compte dans notre processus de révision.

Suite à cette étape de conceptualisation, nous avons proposé une vision intégrée des approches pertinent pour résoudre les multiples facettes du problème de l'apprentissage sur *small data*. Ces différentes situations de petites données peuvent bénéficier de différentes stratégies, que nous avons organisées en deux axes principaux :

- Le premier centré sur les données, regroupant les méthodologies consistant à utiliser des données supplémentaires dans le processus d'entraînement ;
- Le deuxième centré sur le modèle, regroupant les méthodologies qui influencent la complexité de l'entraînement, soit en agissant sur l'architecture, soit plus globalement sur la stratégie d'entraînement.

Cet examen nous a permis de constater que peu d'ouvrages abordent directement la question de l'apprentissage sur *small data*. Néanmoins, en identifiant ses problèmes subjacents, il a été possible de classer un nombre considérable de possibilités d'action, en tirant parti de la littérature connexe dans augmentation des données, semi-supervision, l'apprentissage de la représentation non supervisée, l'apprentissage par transfert, l'apprentissage de métrique, l'apprentissage multitâche, le méta-apprentissage et l'apprentissage via curriculum. En outre, les frontières dans ce cadre ne sont pas rigides, de sorte que ces différents aspects dans chacun de ces domaines peuvent être liés à différentes catégories. Cette présentation des solutions possibles est innovante dans le sens qu'elle est assez pragmatique et axée sur les problèmes : le lecteur est invité à s'interroger sur son propre problème pratique, en identifiant ses difficultés et ses points forts, en les mettant en relation avec les possibilités discutés dans la revue.

Sur le plan expérimental, nous avons proposé une nouvelle version des réseaux prototypiques (Snell et al, 2017) (une architecture visant à résoudre le problème de l'apprentissage en quelques coups), mais plus souple au niveau des classes. Dans notre modèle, les prototypes ne sont plus des centroïdes de classe fixe. Au lieu de cela, leur relation avec chacune des classes et leur placement dans l'espace de caractéristiques sont tous les deux appris des données. Cette nouvelle formulation permet d'obtenir des prototypes représentatifs des données, en partitionnant l'espace vectoriel des caractéristiques (comme analysé dans la section 6.3 et l'annexe A). En fin de compte, les échantillons de données seront associés à une partition de l'espace des caractéristiques, chacun d'entre eux étant associé à des probabilités de classe, qui peuvent être utilisées pour expliquer les prédictions du réseau, apportant un certain degré de l'interprétabilité, au moins aux étapes ultérieures de son processus décisionnel.

Ce modèle a été analysé dans le cadre de deux stratégies d'apprentissage *small data* :

- Premièrement, un paradigme d'apprentissage par transfert, consistant à réutiliser une CNN préformée comme extracteur de caractéristiques ;
- Second, un paradigme de méta-apprentissage, similaire à celui utilisé dans les réseaux prototypiques originaux.

La première méthode a l'avantage d'être plus rapide à mettre en œuvre, puisque les caractéristiques de chaque point de données sont fixes, ce qui facilite l'étude de la sensibilité des hyperparamètres. Avec cette approche, nous avons pu démontrer comment le concept proposé de *data-prototypes* fournit un mécanisme permettant d'expliquer les prévisions du réseau, améliorer l'interprétabilité.

La deuxième méthode consiste à apprendre le réseau de bout en bout, Ainsi, l'extraction des caractéristiques de co-apprentissage et la catégorisation basée sur les prototypes sont combinées. De plus, avec cette étape, nous avons abordé l'apprentissage en quelques clics, l'une des nombreuses formes de présentation du petit problème d'apprentissage des données. Cette piste expérimentale supplémentaire a permis de conclure que l'adaptation commune de la représentation apprise est d'une importance majeure en obtenant une précision plus élevée, par rapport à la méthode précédente des caractéristiques fixes.

# Index


active learning, 97
    acquisition function, 97
    querry batch selection, 97

bio-inspiration, 28
    limits, 17, 42

convolutional neural networks, 29
    convolutional layers, 29
    pooling, 30
Convolutional recurrent networks
    Conv. GRU, 71
    Conv. LSTM, 71
cortical hierarchy, 32

data augmentation, 91

extra-cortical processing
    retina, 43
    thalamus, 43

ImageNet, 28
Inception, 53

neural networks
    activations, 31
    autoencoders, 22
    GAN, 22
    generative models, 22
    GRU, 22
    LSTM, 22
    recurrent networks, 22
    ReLU, 31

object recognition
    object detection, 20
    object localization, 20
    semantic segmentation, 21

recurrent processing in vision, 44
    attentional modulation, 44
    perceptual grouping, 44
    retina, 43
    spatial refinement, 44
    time-scale, 44
retina, 37
    retinal processing, 43

transfer learning, 89
two-stream model of vision, 33
    dorsal pathway, 33
    interactions, 45
    ventral pathway, 33, 34

# Bibliography


Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y. and Zheng, X. (2016). "TensorFlow: A system for large-scale machine learning". In *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, pp. 265–283. Cited on page 54.

Agarwal, S., Terrail, J. O. D. and Jurie, F. (2018). "Recent Advances in Object Detection in the Age of Deep Convolutional Neural Networks". ARXIV: 1809.03193. Cited on page 21.

Alom, M. Z., Hasan, M., Yakopcic, C. and Taha, T. M. (2017). "Inception Recurrent Convolutional Neural Network for Object Recognition". ARXIV: 1704.07709. Cited on page 64.

Andreopoulos, A. and Tsotsos, J. K. (2013). "50 Years of object recognition: Directions forward". *Computer Vision and Image Understanding*, 117, no. 8, 827–891. ISSN 10773142. DOI: 10.1016/j.cviu.2013.04.005. Cited on page 27.

Antoniou, A., Storkey, A. and Edwards, H. (2018). "Data Augmentation Generative Adversarial Networks". *arXiv:1711.04340* [*cs, stat*]. ARXIV: 1711.04340. Cited on page 93.

Ash, J. T., Zhang, C., Krishnamurthy, A., Langford, J. and Agarwal, A. (2019). "Deep Batch Active Learning by Diverse, Uncertain Gradient Lower Bounds". *arXiv:1906.03671* [*cs, stat*]. ARXIV: 1906.03671. Cited on page 98.

Ba, J., Mnih, V. and Kavukcuoglu, K. (2015). "Multiple Object Recognition with Visual Attention". In *International Conference on Learning Representations (ICRL)*, pp. 1–10. ISBN 978-1-4244-6516-3. ARXIV: 1412.7755v2. Cited on pages 67, 70, 78, and 108.

Bach, S., Binder, A., Montavon, G., Klauschen, F., Müller, K.-R. and Samek, W. (2015). "On Pixel-Wise Explanations for Non-Linear Classifier Decisions by Layer-Wise Relevance Propagation". *PLOS ONE*, 10, no. 7, e0130140. ISSN 1932-6203. DOI: 10.1371/journal.pone.0130140. Cited on pages 120 and 121.

Badrinarayanan, V., Kendall, A. and Cipolla, R. (2017). "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, no. 12, 2481–2495. ISSN 01628828. DOI: 10.1109/tpami.2016.2644615. ARXIV: 1511.00561. Cited on pages 51, 52, and 100.

Ballas, N., Yao, L., Pal, C. and Courville, A. (2015). "Delving Deeper into Convolutional Networks for Learning Video Representations". ARXIV: 1511.06432. Cited on pages 63, 71, and 77.

Banerjee, A., Merugu, S., Dhillon, I. S. and Ghosh, J. (2005). "Clustering with Bregman Divergences". *Journal of Machine Learning Research*, 6, 1705–1749. ISSN 08856125. DOI: 10.1007/s10994-005-5825-6. Cited on pages 118 and 124.

Bao, J., Chen, D., Wen, F., Li, H. and Hua, G. (2017). "CVAE-GAN: Fine-Grained Image Generation through Asymmetric Training". *arXiv:1703.10155* [*cs*]. ARXIV: 1703.10155. Cited on page 93.

Baur, C., Albarqouni, S. and Navab, N. (2018). "Generating Highly Realistic Images of Skin Lesions with GANs". *arXiv:1809.01410* [*cs, eess*]. ARXIV: 1809.01410. Cited on page 93.

Belharbi, S., Chatelain, C., Hérault, R. and Adam, S. (2017). "Neural Networks Regularization Through Class-wise Invariant Representation Learning". *arXiv:1709.01867* [*cs, stat*]. ARXIV: 1709.01867 . Cited on page 105.

Belkin, M. and Niyogi, P. (2004). "Semi-supervised learning on riemannian manifolds". *Machine Learning*, 56, no. 1-3, 209–239. ISSN 08856125. DOI: 10.1023/b:mach.0000033120.25363.1e . Cited on page 95.

Bell, S., Zitnick, C. L., Bala, K. and Girshick, R. (2016). "Inside-Outside Net: Detecting Objects in Context with Skip Pooling and Recurrent Neural Networks". In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 2016-Decem, pp. 2874–2883. IEEE. ISBN 978-1-4673-8851-1. DOI: 10.1109/cvpr.2016.314 . ARXIV: 1512.04143 . Cited on pages 65, 73, and 74.

Bellet, A., Habrard, A. and Sebban, M. (2013). "A Survey on Metric Learning for Feature Vectors and Structured Data". Tech. Rep., Laboratoire Hubert Curien UMR 5516. ARXIV: 1306.6709 . Cited on page 119.

Beluch, W. H., Genewein, T., Nurnberger, A. and Kohler, J. M. (2018). "The Power of Ensembles for Active Learning in Image Classification". In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9368–9377. DOI: 10.1109/CVPR.2018.00976 . Cited on page 97.

Bengio, Y. and LeCun, Y. (2007). "Scaling Learning Algorithms towards {AI}". *Large Scale Kernel Machines*, , no. 1, 321–360. ISSN 00099104. ARXIV: 1011.1669v3 . Cited on page 115.

Bengio, Y., Lee, D.-H., Bornschein, J. and Lin, Z. (2016). "Towards Biologically Plausible Deep Learning". Tech. Rep.. ARXIV: 1502.04156 . Cited on page 116.

Bengio, Y., Louradour, J., Collobert, R. and Weston, J. (2009). "Curriculum learning". In *Proceedings of the 26th Annual International Conference on Machine Learning - ICML '09*, pp. 41–48. ACM Press, Montreal, Quebec, Canada. ISBN 978-1-60558-516-1. DOI: 10.1145/1553374.1553380 . ARXIV: 1011.1669v3 . Cited on page 110.

Bengio, Y., Simard, P. and Frasconi, P. (1994). "Learning long-term dependencies with gradient descent is difficult". *IEEE Transactions on Neural Networks*, 5, no. 2, 157–166. ISSN 1941-0093. DOI: 10.1109/72.279181 . Cited on page 22.

Bergstra, J. and Bengio, Y. (2012). "Random Search for Hyper-Parameter Optimization". *Journal of Machine Learning Research*, 13, 281–305. ISSN 1532-4435. Cited on page 75.

Berry, M. J., Brivanlou, I. H., Jordan, T. A. and Meister, M. (1999). "Anticipation of moving stimuli by the retina". *Nature*, 398, no. 6725, 334–338. ISSN 00280836. DOI: 10.1038/18678 . Cited on page 43.

Berthelot, D., Carlini, N., Goodfellow, I., Papernot, N., Oliver, A. and Raffel, C. A. (2019). "MixMatch: A Holistic Approach to Semi-Supervised Learning". In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 5050–5060. Curran Associates, Inc. URL: http://papers.nips.cc/paper/8749-mixmatch-a-holistic-approach-to-semi-supervised-learning.pdf. Cited on pages 92 and 106.

Boney, R. and Ilin, A. (2017). "Semi-Supervised Few-Shot Learning with Prototypical Networks". NIPS Workshops, pp. 1–5. ARXIV: 1711.10856 . Cited on page 95.

Boyd, S., Parikh, N., Chu, E., Peleato, B. and Eckstein, J. (2011). "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers". 3, no. 1, 1–122. DOI: 10.1561/2200000016 . Cited on page 105.

Bullier, J. (2001). "Integrated model of visual processing". *Brain research reviews*, 36, no. 2-3, 96–107. ISSN 01650173. DOI: 10.1016/s0165-0173(01)00085-6 . Cited on page 115.

Cadieu, C. F., Hong, H., Yamins, D. L. K., Pinto, N., Ardila, D., Solomon, E. A., Majaj, N. J. and DiCarlo, J. J. (2014). "Deep Neural Networks Rival the Representation of Primate IT Cortex for Core Visual Object Recognition". *PLoS Computational Biology*, 10, no. 12, e1003963. ISSN 15537358. DOI: 10.1371/journal.pcbi.1003963 .

ARXIV: 1406.3284 . Cited on pages 41 and 142.

Cammarata, N., Carter, S., Goh, G., Olah, C., Petrov, M. and Schubert, L. (2020). "Thread: Circuits". *Distill*, 5, no. 3, e24. ISSN 2476-0757. DOI: 10.23915/distill.00024 . Cited on page 148.

Cao, C., Liu, X., Yang, Y., Yu, Y., Wang, J. and Wang, Z. (2015). "Look and Think Twice : Capturing Top-Down Visual Attention with Feedback". *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2956–2964. URL: http://openaccess.thecvf.com/content_iccv_2015/html/Cao_Look_and_Think_ICCV_2015_paper.html. Cited on pages 66, 68, 69, 70, 75, and 78.

Cardon, D., Cointet, J.-P. and Mazières, A. (2018). *La Revanche Des Neurones*, vol. 211. ISBN 978-2-348-04068-9. DOI: 10.3917/res.211.0173 . Cited on page 28.

Caruana, R. (1997). "Multitask Learning". *Machine Learning*, 28, no. 1, 41–75. ISSN 1573-0565. DOI: 10.1023/A:1007379606734 . Cited on page 112.

Carvajal, C. (2014). "Dynamic interplay between standard and non-standard retinal pathways in the early thalamocortical visual system : A modeling study". These de doctorat, Université de Lorraine. URL: https://www.theses.fr/2014LORR0209. Cited on page 34.

Carvajal, C., Viéville, T. and Alexandre, F. (2013). "Impact of the Konio pathway in the thalamocortical visual system: A modeling study". *BMC Neuroscience*, 14, no. S1, P6. ISSN 1471-2202. DOI: 10.1186/1471-2202-14-s1-p6 . Cited on page 44.

Casagrande, V. (1994). "A third parallel visual pathway to primate area V1". *Trends in Neurosciences*, 17, no. 7, 305–310. ISSN 0166-2236. DOI: 10.1016/0166-2236(94)90065-5 . Cited on pages 34 and 45.

Caswell, I., Shen, C. and Wang, L. (2016). "Loopy Neural Nets : Imitating Feedback Loops in the Human Brain". Tech. Rep.. Cited on pages 66, 69, 70, 72, 78, and 80.

Chapelle, O., Schölkopf, B. and Zien, A. (2006a). "Introduction to Semi-Supervised Learning". In *Semi-Supervised Learning*, Adaptive Computation and Machine Learning, pp. 1–12. MIT Press, Cambridge, Mass. ISBN 978-0-262-03358-9. Cited on page 94.

Chapelle, O., Schölkopf, B. and Zien, A. (eds.) (2006b). *Semi-Supervised Learning*. Adaptive Computation and Machine Learning. MIT Press, Cambridge, Mass. ISBN 978-0-262-03358-9. Cited on page 94.

Chatfield, K., Simonyan, K., Vedaldi, A. and Zisserman, A. (2014). "Return of the Devil in the Details: Delving Deep into Convolutional Nets". ISSN 1-901725-52-9. DOI: 10.5244/c.28.6. ARXIV: 1405.3531 . Cited on page 91.

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K. and Yuille, A. L. (2016). "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". ARXIV: 1606.00915 . Cited on pages 30 and 100.

Chen, X. and Gupta, A. (2015). "Webly Supervised Learning of Convolutional Networks". In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1431–1439. URL: https://openaccess.thecvf.com/content_iccv_2015/html/Chen_Webly_Supervised_Learning_ICCV_2015_paper.html. Cited on page 96.

Cheng, G., Zhou, P. and Han, J. (2018). "Duplex Metric Learning for Image Set Classification". *IEEE Transactions on Image Processing*, 27, no. 1, 281–292. ISSN 1057-7149. DOI: 10.1109/tip.2017.2760512 . Cited on page 105.

Cho, K., van Merrienboer, B., Gülçehre, Ç., Bougares, F., Schwenk, H. and Bengio, Y. (2014). "Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation". *arXiv:1406.1078 [cs, stat]*, abs/1406.1. ARXIV: 1406.1078 . Cited on page 22.

Chollet, F. (2016). "Xception: Deep Learning with Depthwise Separable Convolutions". ARXIV: 1610.02357 .

Cited on pages 53, 54, and 107.

Ciresan, D., Meier, U. and Schmidhuber, J. (2012). "Multi-column deep neural networks for image classification". In *IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642–3649. IEEE. ISBN 978-1-4673-1228-8. DOI: 10.1109/cvpr.2012.6248110 . Cited on page 52.

Clouâtre, L. and Demers, M. (2019). "FIGR: Few-shot Image Generation with Reptile". *arXiv:1901.02199* [*cs, stat*]. ARXIV: 1901.02199 . Cited on page 94.

Cooper, L. N., Intrator, N., Blais, B. S. and Shouval, H. Z. (2004). *Theory of Cortical Plasticity*. World Scientific Publishing. Cited on page 116.

Cortes, N. (2012). "The role of the Pulvinar in the transmission of information in the visual hierarchy". Ph.D. thesis, UMPC. Cited on pages 11, 33, 36, and 115.

Creswell, A., White, T., Dumoulin, V., Arulkumaran, K., Sengupta, B. and Bharath, A. A. (2017). "Generative Adversarial Networks: An Overview". Tech. Rep.. ARXIV: 1710.07035 . Cited on page 23.

Dalal, N. and Triggs, B. (2005). "Histograms of Oriented Gradients for Human Detection". In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893. IEEE. ISBN 0-7695-2372-2. DOI: 10.1109/cvpr.2005.177 . Cited on page 27.

Demšar, J. (2006). "Statistical Comparisons of Classifiers over Multiple Data Sets". *Journal of Machine Learning Research*, 7, no. Jan, 1–30. ISSN ISSN 1533-7928. URL: http://www.jmlr.org/papers/v7/demsar06a.html. Cited on page 132.

Deng, J., Dong, W., Socher, R., Li, L.-J., Kai Li, Li Fei-Fei, Wei Dong, Jia Deng, Kai Li, Socher, R. and Li-Jia Li (2009). "ImageNet: A large-scale hierarchical image database". In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 248–255. IEEE. ISBN 978-1-4244-3992-8. DOI: 10.1109/cvpr.2009.5206848 . Cited on page 160.

DeVries, T. and Taylor, G. W. (2017). "Dataset Augmentation in Feature Space". ARXIV: 1702.05538 . Cited on page 92.

DiCarlo, J. J. and Cox, D. D. (2007). "Untangling invariant object recognition". *Trends in Cognitive Sciences*, 11, no. 8, 333–341. ISSN 13646613. DOI: 10.1016/j.tics.2007.06.010 . Cited on page 35.

DiCarlo, J. J., Zoccolan, D. and Rust, N. C. (2012). "How Does the Brain Solve Visual Object Recognition?" *Neuron*, 73, no. 3, 415–434. ISSN 0896-6273. DOI: 10.1016/j.neuron.2012.01.010 . Cited on pages 34, 35, 37, and 40.

Doersch, C., Gupta, A. and Efros, A. A. (2015). "Unsupervised Visual Representation Learning by Context Prediction". In *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, ICCV '15, pp. 1422–1430. IEEE Computer Society, Santiago, Chile. ISBN 978-1-4673-8391-2. DOI: 10.1109/ICCV.2015.167 . Cited on page 99.

Donahue, C., McAuley, J. and Puckette, M. (2019). "Adversarial Audio Synthesis". *arXiv:1802.04208* [*cs*]. ARXIV: 1802.04208 . Cited on page 23.

Donahue, J., Hendricks, L. A., Rohrbach, M., Venugopalan, S., Guadarrama, S., Saenko, K. and Darrell, T. (2017). "Long-Term Recurrent Convolutional Networks for Visual Recognition and Description". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, no. 4, 677–691. ISSN 01628828. DOI: 10.1109/tpami.2016.2599174 . Cited on pages 63 and 77.

Dong, X., Zheng, L., Ma, F., Yang, Y. and Meng, D. (2017). "Few-shot Object Detection". pp. 1–11. ARXIV: 1706.08249 . Cited on page 114.

Doshi-Velez, F. and Kim, B. (2017). "Towards A Rigorous Science of Interpretable Machine Learning". Tech. Rep.. ARXIV: 1702.08608 . Cited on page 120.

Drumond, T., Viennot, L., Viéville, T. and François, V. (2017a). "Jouez avec les neurones de la machine". *Blog Binaire LeMonde.fr*, pp. 1–3. URL: https://hal.inria.fr/hal-01620451. Cited on page 139.

Drumond, T. F., Viéville, T. and Alexandre, F. (2017b). "Using prototypes to improve convolutional networks interpretability". In *Annual Conference on Neural Information Processing Systems: Transparent and Interpretable Machine Learning in Safety Critical Environments Workshop*. Long Beach, United States. URL: https://hal.inria.fr/hal-01651964. Cited on pages 114 and 117.

Drumond, T. F., Viéville, T. and Alexandre, F. (2019). "Bio-inspired Analysis of Deep Learning on Not-So-Big Data Using Data-Prototypes". *Frontiers in Computational Neuroscience*, 12, 100. ISSN 1662-5188. DOI: 10.3389/fncom.2018.00100 . Cited on page 113.

Dube, P., Bhattacharjee, B., Petit-Bois, E. and Hill, M. (2018). "Improving Transferability of Deep Neural Networks". *arXiv:1807.11459* [*cs, stat*]. ARXIV: 1807.11459 . Cited on page 100.

Eickenberg, M., Gramfort, A., Varoquaux, G. and Thirion, B. (2016). "Seeing it all: Convolutional network layers map the function of the human visual system". *NeuroImage*. ISSN 10538119. DOI: 10.1016/j.neuroimage.2016.10.001 . Cited on pages 41, 142, and 167.

Eigen, D., Rolfe, J., Fergus, R. and LeCun, Y. (2013). "Understanding Deep Architectures using a Recursive Convolutional Network". ARXIV: 1312.1847 . Cited on page 63.

Elman, J. L. (1990). "Finding Structure in Time". *Cognitive Science*, 14, no. 2, 179–211. ISSN 1551-6709. DOI: 10.1207/s15516709cog1402_1 . Cited on page 22.

Esteva, A., Kuprel, B., Novoa, R. A., Ko, J., Swetter, S. M., Blau, H. M. and Thrun, S. (2017). "Dermatologist-level classification of skin cancer with deep neural networks". *Nature*, 542, no. 7639, 115–118. ISSN 0028-0836. DOI: 10.1038/nature21056 . Cited on pages 99 and 126.

Fabre-Thorpe, M., Richard, G. and Thorpe, S. J. (1998). "Rapid categorization of natural images by rhesus monkeys." *Neuroreport*, 9, no. 2, 303–308. ISSN 0959-4965. DOI: 10.1097/00001756-199801260-00023 . Cited on pages 34, 35, and 115.

Fan, J., Ma, C. and Zhong, Y. (2019). "A Selective Overview of Deep Learning". ARXIV: 1904.05526 . Cited on page 103.

Fayek, H. M., Cavedon, L. and Wu, H. R. (2018). "On the Transferability of Representations in Neural Networks Between Datasets and Tasks". *arXiv:1811.12273* [*cs, stat*]. ARXIV: 1811.12273 . Cited on page 100.

Fedus, W., Goodfellow, I. and Dai, A. M. (2018). "MaskGAN: Better Text Generation via Filling in the_____". *arXiv:1801.07736* [*cs, stat*]. ARXIV: 1801.07736 . Cited on page 23.

Felleman, D. J. and Van Essen, D. C. (1991). "Distributed hierarchical processing in the primate cerebral cortex". *Cerebral Cortex*. ISSN 14602199. DOI: 10.1093/cercor/1.1.1 . Cited on pages 11, 32, and 36.

Fergus, R., Fei-Fei, L., Perona, P. and Zisserman, A. (2010). "Learning Object Categories From Internet Image Searches". *Proceedings of the IEEE*, 98, no. 8, 1453–1466. ISSN 1558-2256. DOI: 10.1109/JPROC.2010.2048990 . Cited on page 96.

Finn, C., Abbeel, P. and Levine, S. (2017). "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks". In *International Conference on Machine Learning*, pp. 1126–1135. Sydney. ISBN 978-1-5108-5514-4. ARXIV: 1703.03400 . Cited on pages 108, 109, and 138.

Follmann, P. and Bottger, T. (2018). "A Rotationally-Invariant Convolution Module by Feature Map Back-Rotation". In *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pp. 784–792. DOI: 10.1109/WACV.2018.00091 . Cited on page 107.

Folmsbee, J., Liu, X., Brandwein-Weber, M. and Doyle, S. (2018). "Active deep learning: Improved training efficiency of convolutional neural networks for tissue classification in oral cavity cancer". In *2018 IEEE 15th*

*International Symposium on Biomedical Imaging (ISBI 2018)*, pp. 770–773. DOI: 10.1109/ISBI.2018.8363686 . Cited on page 97.

Fort, S. (2017). "Gaussian Prototypical Networks for Few-Shot Learning on Omniglot". *arXiv preprint*. ARXIV: 1708.02735 . Cited on pages 117 and 119.

Fournier, J., Monier, C., Pananceau, M. and Frégnac, Y. (2011). "Adaptation of the simple or complex nature of V1 receptive fields to visual statistics". *Nature Neuroscience*, 14, no. 8, 1053–1060. ISSN 10976256. DOI: 10.1038/nn.2861 . Cited on page 38.

Freiwald, W. A., Tsao, D. Y. and Livingstone, M. S. (2009). "A face feature space in the macaque temporal lobe". *Nature Neuroscience*, 12, no. 9, 1187–1196. ISSN 1097-6256. DOI: 10.1038/nn.2363 . Cited on page 39.

Frosst, N. and Hinton, G. (2017). "Distilling a Neural Network Into a Soft Decision Tree". *NIPS*. ARXIV: 1711.09784 . Cited on page 121.

Fu, J., Zheng, H. and Mei, T. (2017). "Look closer to see better: Recurrent attention convolutional neural network for fine-grained image recognition". *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017-Janua, 4476–4484. DOI: 10.1109/cvpr.2017.476 . Cited on pages 67, 70, 77, 78, and 107.

Fukushima, K. (1980). "Neocognitron: A self organizing neural network model for a mechanism of pattern recognition unaffected by shift in position." *Biological cybernetics*, 36, no. 4, 193–202. ISSN 0340-1200. DOI: 10.1007/bf00344251 . ARXIV: 1011.1669v3 . Cited on page 28.

Fukushima, K. (1988). "Neocognitron: A Hierarchical Neural Network Capable of Visual Pattern Recognition". *Neural Networks*, 1, no. 2, 119–130. ISSN 08936080. DOI: 10.1016/0893-6080(88)90014-7 . Cited on page 38.

Fukushima, K. (2005). "Restoring partly occluded patterns: A neural network model". *Neural Networks*, 18, no. 1, 33–43. ISSN 08936080. DOI: 10.1016/j.neunet.2004.05.001 . Cited on page 68.

Gal, Y., Islam, R. and Ghahramani, Z. (2017). "Deep Bayesian active learning with image data". In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, pp. 1183–1192. JMLR.org, Sydney, NSW, Australia. Cited on page 97.

Garcia, V. and Estrach, J. B. (2018). "Few-Shot Learning with Graph Neural Networks". In *International Conference on Learning Representations*, pp. 1–13. URL: https://openreview.net/forum?id=BJj6qGbRW. Cited on page 138.

Garcia-Garcia, A., Orts-Escolano, S., Oprea, S., Villena-Martinez, V. and Garcia-Rodriguez, J. (2017). "A Review on Deep Learning Techniques Applied to Semantic Segmentation". ARXIV: 1704.06857 . Cited on pages 21 and 65.

Giese, M. A. and Poggio, T. (2003). "Neural mechanisms for the recognition of biological movements". *Nature Reviews Neuroscience*, 4, no. 3, 179–192. ISSN 1471-003X. DOI: 10.1038/nrn1057 . Cited on page 45.

Girshick, R., Donahue, J., Darrell, T. and Malik, J. (2014). "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 580–587. ISSN 10636919. DOI: 10.1109/cvpr.2014.81 . ARXIV: 1311.2524 . Cited on page 51.

Gissin, D. and Shalev-Shwartz, S. (2019). "Discriminative Active Learning". *arXiv:1907.06347 [cs, stat]*. ARXIV: 1907.06347 . Cited on page 98.

Glorot, X. and Bengio, Y. (2010). "Understanding the difficulty of training deep feedforward neural networks". *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, 9, 249–256. ISSN 15324435. Cited on page 31.

Gollisch, T. and Meister, M. (2010). "Eye Smarter than Scientists Believed: Neural Computations in Circuits of

the Retina". *Neuron*, 65, no. 2, 150–164. ISSN 08966273. DOI: 10.1016/j.neuron.2009.12.009 . Cited on pages 34, 43, and 75.

Goodale, M. A. and Milner, D. A. (1992). "Separate visual pathways for perception and action". *Trends in Neurosciences*, 15, no. 1, 20–25. ISSN 0166-2236. DOI: 10.1016/0166-2236(92)90344-8 . Cited on page 34.

Goodfellow, I. (2016). "NIPS 2016 Tutorial: Generative Adversarial Networks". Tech. Rep.. ARXIV: 1701.00160 . Cited on page 23.

Goodfellow, I., Bengio, Y. and Courville, A. (2016a). "Autoencoder". In *Deep Learning*. MIT Press. URL: http://www.deeplearningbook.org. Cited on pages 23 and 124.

Goodfellow, I., Bengio, Y. and Courville, A. (2016b). "Convolutional Networks". In *Deep Learning*, pp. 330–372. MIT Press. URL: http://www.deeplearningbook.org. Cited on pages 30 and 115.

Goodfellow, I., Bengio, Y. and Courville, A. (2016c). "Deep Feedforward Networks". In *Deep Learning*, pp. 169–229. MIT Press. URL: http://www.deeplearningbook.org. Cited on page 113.

Goodfellow, I., Bengio, Y. and Courville, A. (2016d). "Deep Generative Models". In *Deep Learning*, pp. 658–729. MIT Press. ARXIV: 1504.06787v3 . Cited on page 22.

Goodfellow, I., Bengio, Y. and Courville, A. (2016e). "Regularization for Deep Learning". In *Deep Learning*, pp. 216–261. MIT Press. URL: http://www.deeplearningbook.org. Cited on page 103.

Goodfellow, I., Bengio, Y. and Courville, A. (2016f). "Representation Learning". In *Deep Learning*, pp. 528–559. MIT Press. URL: http://www.deeplearningbook.org. Cited on pages 99 and 118.

Goodfellow, I., Bengio, Y. and Courville, A. (2016g). "Sequence Modeling : Recurrent and Recursive Nets". In *Deep Learning*, pp. 373–422. MIT Press. URL: http://www.deeplearningbook.org. Cited on page 22.

Goodfellow, I., Pouget-Abadie, J. and Mirza, M. (2014). "Generative Adversarial Networks". *arXiv preprint arXiv: ...*, pp. 1–9. ISSN 10495258. ARXIV: 1406.2661v1 . Cited on page 23.

Graves, A. and Schmidhuber, J. (2012). "Offline Handwriting Recognition with Multidimensional Recurrent Neural Networks". In *NIPS*, pp. 1–8. DOI: 10.1007/978-1-4471-4072-6_12 . Cited on page 73.

Graves, A., Wayne, G., Danihelka, I. and Deepmind, G. (2014). "Neural Turing Machines". ARXIV: 1410.5401 . Cited on page 120.

Greff, K., Srivastava, R. K. and Schmidhuber, J. (2016). "Highway and Residual Networks learn Unrolled Iterative Estimation". ARXIV: 1612.07771 . Cited on pages 57 and 59.

Guidotti, R., Monreale, A., Ruggieri, S., Turini, F., Pedreschi, D. and Giannotti, F. (2018). "A Survey Of Methods For Explaining Black Box Models". ISSN 03600300. DOI: 10.1145/3236009 . ARXIV: 1802.01933 . Cited on page 69.

Gulcehre, C., Moczulski, M., Visin, F. and Bengio, Y. (2016). "Mollifying Networks". *arXiv:1608.04980* [*cs*]. ARXIV: 1608.04980 . Cited on page 110.

Hacohen, G. and Weinshall, D. (2019). "On The Power of Curriculum Learning in Training Deep Networks". *arXiv:1904.03626* [*cs, stat*]. ARXIV: 1904.03626 . Cited on pages 110 and 111.

Han, K., Wen, H., Zhang, Y., Fu, D., Culurciello, E. and Liu, Z. (2018). "Deep Predictive Coding Network with Local Recurrent Processing for Object Recognition". ARXIV: 1805.07526 . Cited on pages 67 and 70.

Han, S., Pool, J., Tran, J. and Dally, W. J. (2015). "Learning both Weights and Connections for Efficient Neural Networks". pp. 1–9. ISSN 01406736. DOI: 10.1016/s0140-6736(95)92525-2 . ARXIV: 1506.02626 . Cited on page 107.

Hardt, M. and Ma, T. (2016). "Identity Matters in Deep Learning". pp. 1–20. ARXIV: 1611.04231 . Cited on page 60.

Hartline, H. K. (1940). "The Receptive Fields of Optic Nerve Fibers". *American Journal of Physiology-Legacy Content*, 130, no. 4, 690–699. ISSN 0002-9513. DOI: 10.1152/ajplegacy.1940.130.4.690 . Cited on page 38.

Hassibi, B., Stork, D. G. and Wolff, G. J. (1993). "Optimal brain surgeon and general network pruning". In *IEEE International Conference on Neural Networks - Conference Proceedings*, vol. 1993-Janua, pp. 293–299. ISBN 0-7803-0999-5. DOI: 10.1109/icnn.1993.298572 . Cited on page 107.

Håstad, J. and Goldmann, M. (1991). "On the power of small-depth threshold circuits". *Computational Complexity*, 1, no. 2, 113–129. ISSN 1016-3328, 1420-8954. DOI: 10.1007/bf01272517 . Cited on page 113.

He, K., Zhang, X., Ren, S. and Sun, J. (2015). "Spatial Pyramid Pooling in Deep Convolutional Networks for Visual Recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, no. 9, 1904–1916. ISSN 01628828. DOI: 10.1109/tpami.2015.2389824 . ARXIV: 1406.4729 . Cited on page 67.

He, K., Zhang, X., Ren, S. and Sun, J. (2016a). "Deep Residual Learning for Image Recognition". In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. DOI: 10.1109/cvpr.2016.90 . ARXIV: 1512.03385 . Cited on pages 29, 31, 41, 42, 53, 55, 56, 60, 75, 104, 113, and 141.

He, K., Zhang, X., Ren, S. and Sun, J. (2016b). "Identity Mappings in Deep Residual Networks". In *ECCV 2016: Computer Vision – ECCV 2016*, 1, pp. 630–645. ISBN 978-3-319-46493-0. DOI: 10.1007/978-3-319-46493-0_38 . ARXIV: 1603.05027 . Cited on pages 55 and 56.

Hecht, T. and Gepperth, A. (2016). "Computational advantages of deep prototype-based learning". *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 9887 LNCS, 121–127. ISSN 16113349. DOI: 10.1007/978-3-319-44781-0_15 . Cited on page 118.

Hernández-García, A. and König, P. (2019). "Data augmentation instead of explicit regularization". *arXiv:1806.03852 [cs]*. ARXIV: 1806.03852 . Cited on page 103.

Hershey, S., Chaudhuri, S., Ellis, D. P. W., Gemmeke, J. F., Jansen, A., Moore, R. C., Plakal, M., Platt, D., Saurous, R. A., Seybold, B., Slaney, M., Weiss, R. J. and Wilson, K. (2017). "CNN architectures for large-scale audio classification". In *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 131–135. DOI: 10.1109/ICASSP.2017.7952132 . Cited on page 29.

Herzog, M. H. and Clarke, A. M. (2014). "Why vision is not both hierarchical and feedforward". *Frontiers in Computational Neuroscience*, 8, 135. ISSN 1662-5188. DOI: 10.3389/fncom.2014.00135 . Cited on pages 44 and 45.

Hochreiter, S. and Schmidhuber, J. (1997). "Long Short-Term Memory". *Neural Comput.*, 9, no. 8, 1735–1780. ISSN 0899-7667. DOI: 10.1162/neco.1997.9.8.1735 . Cited on page 22.

Hong, H., Yamins, D. L. K., Majaj, N. J. and DiCarlo, J. J. (2016). "Explicit information for category-orthogonal object properties increases along the ventral stream". *Nature Neuroscience*, 19, no. 4, 613–622. ISSN 1097-6256. DOI: 10.1038/nn.4247 . Cited on pages 42 and 81.

Hong, Y., Niu, L., Zhang, J. and Zhang, L. (2020a). "MatchingGAN: Matching-based Few-shot Image Generation". *arXiv:2003.03497 [cs, eess]*. ARXIV: 2003.03497 . Cited on page 94.

Hong, Y., Niu, L., Zhang, J., Zhao, W., Fu, C. and Zhang, L. (2020b). "F2GAN: Fusing-and-Filling GAN for Few-shot Image Generation". *arXiv:2008.01999 [cs]*. ARXIV: 2008.01999 . Cited on page 94.

Hoogeboom, E. (2017). "Few-shot Classification by Learning Disentangled Representations". MSc, University of Amsterdam. Cited on page 141.

Howard, A. G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M. and Adam, H.

(2017). "MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications". ARXIV: 1704.04861 . Cited on pages 52 and 107.

Huang, G., Liu, Z., v. d. Maaten, L. and Weinberger, K. Q. (2017). "Densely Connected Convolutional Networks". In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2261–2269. DOI: 10.1109/cvpr.2017.243 . ARXIV: 1608.06993 . Cited on pages 42, 58, and 141.

Hubel, D. H. and Wiesel, T. N. (1959). "Receptive fields of single neurones in the cat's striate cortex". *The Journal of Physiology*, 148, no. 3, 574–591. ISSN 00223751. DOI: 10.1113/jphysiol.1959.sp006308 . Cited on pages 38 and 45.

Hubel, D. H. and Wiesel, T. N. (1962). "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". *The Journal of Physiology*. ISSN 14697793. DOI: 10.1113/jphysiol.1962.sp006837 . ARXIV: 1011.1669v3 . Cited on pages 28, 37, and 38.

Hyvärinen, A., Hurri, J. and Hoyer, P. O. (2009). *Natural Image Statistics*. ISBN 978-1-84882-490-4. Cited on page 27.

Iandola, F. N., Han, S., Moskewicz, M. W., Ashraf, K., Dally, W. J. and Keutzer, K. (2016). "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and". ARXIV: 1602.07360 . Cited on pages 52 and 56.

Iscen, A., Tolias, G., Avrithis, Y. and Chum, O. (2019). "Label Propagation for Deep Semi-Supervised Learning". In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 5070–5079. URL: http://openaccess.thecvf.com/content_CVPR_2019/papers/Iscen_Label_Propagation_for_Deep_Semi-Supervised_Learning_CVPR_2019_paper. Cited on page 98.

Issa, E. B., Cadieu, C. F. and Dicarlo, J. J. (2018). "Neural dynamics at successive stages of the ventral visual stream are consistent with hierarchical error signals". *eLife*, 7, no. 1, 1–24. ISSN 2050084X. DOI: 10.7554/elife.42870 . Cited on page 63.

Jaderberg, M., Simonyan, K., Zisserman, A. and Kavukcuoglu, K. (2015). "Spatial Transformer Networks". In *Nips*, pp. 1–14. DOI: 10.1038/nbt.3343 . ARXIV: 1506.02025 . Cited on page 67.

Jakubovitz, D., Giryes, R. and Rodrigues, M. R. D. (2018). "Generalization Error in Deep Learning". ISSN 0167-577X. DOI: 10.1016/j.matlet.2011.09.042 . ARXIV: 1808.01174 . Cited on pages 85, 86, and 103.

Jarrett, K., Kavukcuoglu, K., Ranzato, M. and LeCun, Y. (2009). "What is the best multi-stage architecture for object recognition?" *Proceedings of the IEEE International Conference on Computer Vision*, pp. 2146–2153. ISSN 1550-5499. DOI: 10.1109/iccv.2009.5459469 . Cited on page 31.

Jetley, S., Romera-Paredes, B., Jayasumana, S. and Torr, P. (2015). "Prototypical Priors: From Improving Classification to Zero-Shot Learning". In Xianghua Xie Mark W. Jones and G. K. L. Tam (eds.), *Proceedings of the British Machine Vision Conference (BMVC)*, pp. 120.1–120.12. BMVA Press. ISBN 1-901725-53-7. DOI: 10.5244/c.29.120 . ARXIV: 1512.01192 . Cited on page 117.

Kaadoud, I. C. (2018). "Apprentissage de séquences et extraction de règles de réseaux récurrents : application au traçage de schémas techniques." Ph.D. thesis, Université de Bordeaux. URL: https://tel.archives-ouvertes.fr/tel-01771685. Cited on page 149.

Kaadoud, I. C. and Viéville, T. (2016). "L'apprentissage profond : une idée à creuser ?" *Interstices*. URL: https://interstices.info/lapprentissage-profond-une-idee-a-creuser/. Cited on page 139.

Kar, K., Kubilius, J., Schmidt, K., Issa, E. B. and DiCarlo, J. J. (2019). "Evidence that recurrent circuits are critical to the ventral stream's execution of core object recognition behavior". *Nature Neuroscience*, 22, no. 6, 974–983. ISSN 1097-6256. DOI: 10.1038/s41593-019-0392-5 . Cited on page 82.

Kastner, D. B. and Baccus, S. A. (2013). "Insights from the retina into the diverse and general computations of

adaptation, detection, and prediction". *Current Opinion in Neurobiology*. ISSN 09594388. DOI: 10.1016/j.conb.2013.11.012. Cited on page 43.

Kaur, P., Sikka, K. and Divakaran, A. (2017). "Combining Weakly and Webly Supervised Learning for Classifying Food Images". *arXiv:1712.08730* [*cs*]. ARXIV: 1712.08730. Cited on page 96.

Kaya, M. and Bilge, H. Ş. (2019). "Deep Metric Learning: A Survey". *Symmetry*, 11, no. 9, 1066. DOI: 10.3390/sym11091066. Cited on page 99.

Khaligh-Razavi, S.-M. and Kriegeskorte, N. (2014). "Deep Supervised, but Not Unsupervised, Models May Explain IT Cortical Representation". *PLoS Computational Biology*, 10, no. 11, e1003915. ISSN 1553-7358. DOI: 10.1371/journal.pcbi.1003915. Cited on pages 41, 42, and 81.

Kim, B., Khanna, R. and Koyejo, O. O. (2016). "Examples are not Enough, Learn to Criticize! Criticism for Interpretability". *Advances in Neural Information Processing Systems 29*, pp. 2280–2288. Cited on page 114.

Kim, Y. (2014). "Convolutional Neural Networks for Sentence Classification". *arXiv:1408.5882* [*cs*]. ARXIV: 1408.5882. Cited on page 29.

Kingma, D. P. and Welling, M. (2014). "Auto-Encoding Variational Bayes". *arXiv:1312.6114* [*cs, stat*]. ARXIV: 1312.6114. Cited on page 23.

Kirsch, A., van Amersfoort, J. and Gal, Y. (2019). "BatchBALD: Efficient and Diverse Batch Acquisition for Deep Bayesian Active Learning". In H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox and R. Garnett (eds.), *Advances in Neural Information Processing Systems 32*, pp. 7024–7035. Curran Associates, Inc. URL: http://papers.nips.cc/paper/8925-batchbald-efficient-and-diverse-batch-acquisition-for-deep-bayesian-active-learning.pdf. Cited on page 98.

Koch, G., Zemel, R. and Salakhutdinov, R. (2015). "Siamese Neural Networks for One-Shot Image Recognition". In *ICML - Deep Learning Workshop*, vol. 37. Cited on pages 100, 105, and 119.

Koga, T., Nonaka, N., Sakuma, J. and Seita, J. (2018). "General-to-Detailed GAN for Infrequent Class Medical Images". *arXiv:1812.01690* [*cs, stat*]. ARXIV: 1812.01690. Cited on page 94.

Krause, J., Sapp, B., Howard, A., Zhou, H., Toshev, A., Duerig, T., Philbin, J. and Fei-Fei, L. (2016). "The Unreasonable Effectiveness of Noisy Data for Fine-Grained Recognition". In *Lecture Notes in Computer Science* (*Including Subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics*), vol. 9907 LNCS, pp. 301–320. ISBN 978-3-319-46486-2. DOI: 10.1007/978-3-319-46487-9_19. ARXIV: 1511.06789. Cited on page 96.

Krause, J., Stark, M., Deng, J. and Fei-Fei, L. (2013). "3D object representations for fine-grained categorization". In *4th International IEEE Workshop on 3D Representation and Recognition* (*3dRR-13*). Sydney, Australia. DOI: 10.1109/ICCVW.2013.77. Cited on pages 88 and 159.

Krizhevsky, A. (2009). "Learning Multiple Layers of Features from Tiny Images". Tech. Rep.. Cited on pages 159 and 160.

Krizhevsky, A., Sutskever, I. and Hinton, G. E. (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In F. P. a. C. J. C. B. a. L. B. a. K. Q. Weinberger (ed.), *Advances in Neural Information Processing Systems 25*, pp. 1097–1105. Curran Associates, Inc. Cited on pages 28, 41, 49, and 113.

Kubilius, J., Schrimpf, M., Nayebi, A., Bear, D., Yamins, D. L. K. and DiCarlo, J. J. (2018). "CORnet: Modeling the Neural Mechanisms of Core Object Recognition". DOI: 10.1101/408385. BIORXIV: 408385. Cited on page 42.

Kubilius, J., Wagemans, J. and Op de Beeck, H. P. (2015). "A conceptual framework of computations in mid-level vision". *Frontiers in computational neuroscience*, 8, no. December, 158. ISSN 1662-5188. DOI: 10.3389/fncom.2014.00158. Cited on page 39.

Kuen, J., Kong, X., Wang, G. and Tan, Y.-P. (2016). "DelugeNets: Deep Networks with Efficient and Flexible Cross-layer Information Inflows". ARXIV: 1611.05552 . Cited on pages 58 and 61.

Kuffler, S. W. (1953). "Discharge Patterns And Functional Organization of Mammalian Retina". *Journal of Neurophysiology*, 16, no. 1, 37–68. ISSN 0022-3077. DOI: 10.1152/jn.1953.16.1.37 . Cited on page 38.

Kukačka, J., Golkov, V. and Cremers, D. (2017). "Regularization for Deep Learning: A Taxonomy". *arXiv:1710.10686* [*cs, stat*]. ARXIV: 1710.10686 . Cited on pages 103 and 104.

Lake, B. M., Salakhutdinov, R. R., Gross, J. and Tenenbaum, J. B. (2011). "One shot learning of simple visual concepts". In L. Carlson, C. Hoelscher and T. F. Shipley (eds.), *Proceedings of the 33rd Annual Conference of the Cognitive Science Society* (*CogSci 2011*), vol. 1, pp. 2568–2573. Curran Associates, Inc., Boston, Massachusetts, USA. Cited on pages 125, 159, and 160.

Lamme, V. A. F., Supèr, H. and Spekreijse, H. (1998). "Feedforward, horizontal, and feedback processing in the visual cortex". *Current Opinion in Neurobiology*, 8, no. 4, 529–535. ISSN 09594388. DOI: 10.1016/s0959-4388(98)80042-1 . Cited on page 44.

Land, M. F. (1999). "Motion and vision: Why animals move their eyes". *Journal of Comparative Physiology A: Sensory, Neural, and Behavioral Physiology*, 185, no. 4, 341–352. ISSN 0340-7594, 1432-1351. DOI: 10.1007/s003590050393 . Cited on page 37.

Laptev, D., Savinov, N., Buhmann, J. M. and Pollefeys, M. (2016). "TI-POOLING: Transformation-Invariant Pooling for Feature Learning in Convolutional Neural Networks". In *2016 IEEE Conference on Computer Vision and Pattern Recognition* (*CVPR*), pp. 289–297. DOI: 10.1109/CVPR.2016.38 . Cited on page 107.

Larochelle, H. and Hinton, G. E. (2010). "Learning to combine foveal glimpses with a third-order Boltzmann machine". URL: https://papers.nips.cc/paper/4089-learning-to-combine-foveal-glimpses-with-a-third-order-boltzmann-machine. Cited on page 67.

Larsson, G., Maire, M. and Shakhnarovich, G. (2016). "FractalNet: Ultra-Deep Neural Networks without Residuals". ARXIV: 1605.07648 . Cited on page 57.

Le, Q. V., Ngiam, J., Chen, Z., Chia, D., Koh, P. W. and Ng, A. Y. (2010). "Tiled convolutional neural networks". In *Advances in Neural Information Processing Systems 23* (*NIPS 2010*). Cited on page 30.

LeCun, Y., Bengio, Y. and Hinton, G. (2015). "Deep learning". *Nature*, 521, no. 7553, 436–444. ISSN 0028-0836. DOI: 10.1038/nature14539 . ARXIV: 1312.6184v5 . Cited on page 113.

LeCun, Y., Boser, B., Denker, J. S., Henderson, D., Howard, R. E., Hubbard, W. and Jackel, L. D. (1990a). "Handwritten Digit Recognition with a Back-Propagation Network". *ADVANCES IN NEURAL INFORMATION PROCESSING SYSTEMS*, 2, 396–404. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.32.5076. Cited on pages 17, 28, and 162.

Lecun, Y., Bottou, L., Bengio, Y. and Haffner, P. (1998). "Gradient-based learning applied to document recognition". *Proceedings of the IEEE*, 86, no. 11, 2278–2324. ISSN 00189219. DOI: 10.1109/5.726791 . Cited on pages 91, 113, and 159.

LeCun, Y., Denker, J. S. and Solla, S. A. (1990b). "Optimal brain damage". In D. S. Touretzky (ed.), *Advances in Neural Information Processing Systems*, vol. 2, pp. 598–605. Morgan-Kaufmann. URL: http://papers.nips.cc/paper/250-optimal-brain-damage.pdf. Cited on page 107.

Lehky, S. R., Kiani, R., Esteky, H. and Tanaka, K. (2014). "Dimensionality of Object Representations in Monkey Inferotemporal Cortex". *Neural Computation*, 26, no. 10, 2135–2162. ISSN 0899-7667. DOI: 10.1162/neco_a_00648 . Cited on page 40.

Leroy, A. (1967). "L'apprentissage de la lecture chez les jeunes enfants : Acquisition des lettres de l'alphabet et maturité mentale." *Enfance; psychologie, pedagogie, neuropsychiatrie, sociologie*, 20, no. 1, 27–55. ISSN 0013-7545. DOI: 10.3406/enfan.1967.2408 . Cited on page 114.

Li, S., Jiao, J., Han, Y. and Weissman, T. (2017). "Demystifying ResNet". pp. 1–13. ARXIV: 1611.01186v2. Cited on pages 60, 73, and 149.

Li, X., Jie, Z., Feng, J., Liu, C. and Yan, S. (2018). "Learning with rethinking: Recurrently improving convolutional neural networks through feedback". *Pattern Recognition*, 79, 183–194. ISSN 00313203. DOI: 10.1016/j.patcog.2018.01.015. ARXIV: 1708.04483. Cited on pages 69, 70, 75, and 78.

Liang, M. and Hu, X. (2015). "Recurrent convolutional neural network for object recognition". In *Computer Vision and Pattern Recognition (CVPR), 2015 IEEE Conference On*, vol. 07-12-June, pp. 3367–3375. ISBN 978-1-4673-6964-0. DOI: 10.1109/cvpr.2015.7298958. ARXIV: 1306.2795v1. Cited on pages 59, 64, 69, 70, 73, and 78.

Liao, Q. and Poggio, T. (2016). "Bridging the Gaps Between Residual Learning, Recurrent Neural Networks and Visual Cortex". Tech. Rep. 047, CBMM. ARXIV: 1604.03640. Cited on pages 59, 64, 69, 72, 75, 76, 80, and 82.

Lipton, Z. C. (2016). "The Mythos of Model Interpretability". In *ICML Workshop on Human Interpretability in Machine Learning*. ARXIV: 1606.03490v3. Cited on page 120.

Liu, N. and Han, J. (2018). "A Deep Spatial Contextual Long-Term Recurrent Convolutional Network for Saliency Detection". *IEEE Transactions on Image Processing*, 27, no. 7, 3264–3274. ISSN 10577149. DOI: 10.1109/tip.2018.2817047. Cited on page 66.

Liu, Y., Jain, A., Eng, C., Way, D. H., Lee, K., Bui, P., Kanada, K., Marinho, G. d. O., Gallegos, J., Gabriele, S., Gupta, V., Singh, N., Natarajan, V., Hofmann-Wellenhof, R., Corrado, G. S., Peng, L. H., Webster, D. R., Ai, D., Huang, S., Liu, Y., Dunn, R. C. and Coz, D. (2019). "A deep learning system for differential diagnosis of skin diseases". *arXiv:1909.05382 [cs, eess]*. ARXIV: 1909.05382. Cited on page 92.

Lowe, D. G. (2004). "Distinctive image features from scale-invariant keypoints". *International Journal of Computer Vision*. ISSN 09205691. DOI: 10.1023/b:visi.0000029664.99615.94. ARXIV: cs/0112017. Cited on page 27.

Luan, S., Chen, C., Zhang, B., Han, J. and Liu, J. (2018). "Gabor Convolutional Networks". *IEEE Transactions on Image Processing*, 27, no. 9, 4357–4366. ISSN 1941-0042. DOI: 10.1109/TIP.2018.2835143. ARXIV: 1705.01450v3. Cited on page 107.

Lundberg, S. M. and Lee, S.-I. (2017). "A Unified Approach to Interpreting Model Predictions". In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4765–4774. Curran Associates, Inc. ARXIV: 1705.07874. Cited on page 121.

Lundervold, A. S. and Lundervold, A. (2019). "An overview of deep learning in medical imaging focusing on MRI". *Zeitschrift für Medizinische Physik*, 29, no. 2, 102–127. ISSN 0939-3889. DOI: 10.1016/j.zemedi.2018.11.002. Cited on page 88.

Lyu, S. and Simoncelli, E. P. (2009). "Nonlinear Extraction of Independent Components of Natural Images Using Radial Gaussianization". *Neural Computation*, 21, no. 6, 1485–1519. ISSN 0899-7667, 1530-888X. DOI: 10.1162/neco.2009.04-08-773. Cited on page 122.

Maaten, L. V. D. and Hinton, G. (2008). "Visualizing Data using t-SNE". *Journal of Machine Learning Research 1*, 620, no. 1, 267–84. ISSN 1940-6029. DOI: 10.1007/s10479-011-0841-3. ARXIV: 1307.1662. Cited on page 127.

Majaj, N. J., Hong, H., Solomon, E. A. and DiCarlo, J. J. (2015). "Simple Learned Weighted Sums of Inferior Temporal Neuronal Firing Rates Accurately Predict Human Core Object Recognition Performance". *Journal of Neuroscience*, 35, no. 39, 13402–13418. ISSN 0270-6474, 1529-2401. DOI: 10.1523/JNEUROSCI.5181-14.2015. Cited on page 39.

Makhzani, A. and Frey, B. (2014). "K-Sparse Autoencoders". *arXiv:1312.5663* [*cs*]. ARXIV: 1312.5663. Cited on page 23.

Marcus, G. (2018). "Deep Learning: A Critical Appraisal". pp. 1–27. ARXIV: 1801.00631. Cited on page 116.

Marr, D. (1982). *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information*. Henry Holt and Co., Inc., New York, NY, USA. ISBN 0-7167-1567-8. Cited on page 32.

Masland, R. H. (2001). "Neuronal diversity in the retina". *Current Opinion in Neurobiology*, 11, no. 4, 431–436. ISSN 09594388. DOI: 10.1016/s0959-4388(00)00230-0. Cited on page 43.

Matusov, E. (2001). "Vygotskij's Theory of Human Development and New Approaches to Education". In N. J. Smelser and P. B. Baltes (eds.), *International Encyclopedia of the Social and Behavioral Sciences*, pp. 16339–16343. Pergamon, Oxford. ISBN 978-0-08-043076-8. DOI: 10.1016/B0-08-043076-7/02407-4. Cited on page 110.

McHaffie, J. G., Stanford, T. R., Stein, B. E., Coizet, V. and Redgrave, P. (2005). "Subcortical loops through the basal ganglia". *Trends in Neurosciences*, 28, no. 8, 401–407. ISSN 01662236. DOI: 10.1016/j.tins.2005.06.006. Cited on page 116.

Medathati, N. V. K., Neumann, H., Masson, G. S. and Kornprobst, P. (2016). "Bio-inspired computer vision: Towards a synergistic approach of artificial and biological vision". *Computer Vision and Image Understanding*, 150, 1–30. ISSN 10773142. DOI: 10.1016/j.cviu.2016.04.009. Cited on pages 18, 44, 45, 115, 116, and 163.

Mehrotra, A. and Dukkipati, A. (2017). "Generative Adversarial Residual Pairwise Networks for One Shot Learning". Tech. Rep.. ARXIV: 1703.08033. Cited on page 56.

Mensch, A., Mairal, J., Bzdok, D., Thirion, B. and Varoquaux, G. (2017). "Learning Neural Representations of Human Cognition across Many fMRI Studies". In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 5883–5893. Curran Associates, Inc. ARXIV: 1710.11438. Cited on page 116.

Mensink, T., Verbeek, J., Perronnin, F. and Csurka, G. (2013). "Distance-Based Image Classification: Generalizing to New Classes at Near-Zero Cost". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35, no. 11, 2624–2637. ISSN 0162-8828. DOI: 10.1109/tpami.2013.83. Cited on page 119.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S. and Dean, J. (2013). "Distributed Representations of Words and Phrases and their Compositionality". In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 26*, pp. 3111–3119. Curran Associates, Inc. URL: http://papers.nips.cc/paper/5021-distributed-representations-of-words-and-phrases-and-their-compositionality.pdf. Cited on pages 22 and 101.

Milner, D. and Goodale, M. (1995). *The Visual Brain in Action*. Oxford University Press, Great Clarendon Street, Oxford OX2 6DP, second edn.. ISBN 0-19-852472-2. URL: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0198524722. Cited on page 115.

Mishkin, M., Ungerleider, L. G. and Macko, K. A. (1983). "Object vision and spatial vision: Two cortical pathways". *Trends in Neurosciences*. ISSN 01662236. DOI: 10.1016/0166-2236(83)90190-x. Cited on page 33.

Mnih, V., Heess, N., Graves, A. and Kavukcuoglu, K. (2014). "Recurrent Models of Visual Attention". ARXIV: 1406.6247. Cited on page 67.

Morerio, P., Cavazza, J., Volpi, R., Vidal, R. and Murino, V. (2017). "Curriculum Dropout". *arXiv:1703.06229* [*cs, stat*]. ARXIV: 1703.06229. Cited on page 110.

Mouret, J.-B. (2016). "Micro-Data Learning: The Other End of the Spectrum". ARXIV: 1610.00946. Cited on page 114.

Movshovitz-Attias, Y., Toshev, A., Leung, T. K., Ioffe, S. and Singh, S. (2017). "No Fuss Distance Metric Learning using Proxies". *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 360–368. ISSN 15505499. DOI: 10.1109/iccv.2017.47. ARXIV: 1703.07464. Cited on page 105.

Nayebi, A., Bear, D., Kubilius, J., Kar, K., Ganguli, S., Sussillo, D., DiCarlo, J. J. and Yamins, D. L. K. (2018). "Task-Driven Convolutional Recurrent Models of the Visual System". pp. 5290–5301. ARXIV: 1807.00053. Cited on pages 42, 63, 64, 72, 75, 81, and 82.

Neyshabur, B., Li, Z., Bhojanapalli, S., LeCun, Y. and Srebro, N. (2019). "The role of over-parametrization in generalization of neural networks". In *International Conference on Learning Representations*. URL: https://openreview.net/forum?id=BygfghAcYX. Cited on page 103.

Ng, J. Y.-H., Hausknecht, M., Vijayanarasimhan, S., Vinyals, O., Monga, R. and Toderici, G. (2015). "Beyond Short Snippets: Deep Networks for Video Classification". ARXIV: 1503.08909. Cited on page 77.

Nilsback, M.-E. and Zisserman, A. (2008). "Automated flower classification over a large number of classes". In *Indian Conference on Computer Vision, Graphics and Image Processing*. Cited on page 159.

Ning, G., Zhang, Z., Huang, C., Ren, X., Wang, H., Cai, C. and He, Z. (2017). "Spatially supervised recurrent convolutional neural networks for visual object tracking". In *2017 IEEE International Symposium on Circuits and Systems (ISCAS)*, pp. 1–4. IEEE. ISBN 9781467368520. DOI: 10.1109/ISCAS.2017.8050867. Cited on page 63.

Noh, H., Hong, S. and Han, B. (2016). "Learning deconvolution network for semantic segmentation". *Proceedings of the IEEE International Conference on Computer Vision*, 11-18-Dece, 1520–1528. ISSN 15505499. DOI: 10.1109/iccv.2015.178. ARXIV: 1505.04366. Cited on page 100.

Olah, C., Mordvintsev, A. and Schubert, L. (2017). "Feature Visualization". *Distill*, 2, no. 11, e7. ISSN 2476-0757. DOI: 10.23915/distill.00007. Cited on pages 38 and 120.

van den Oord, A., Kalchbrenner, N. and Kavukcuoglu, K. (2016). "Pixel Recurrent Neural Networks". *International Conference on Machine Learning (ICML)*. ARXIV: 1601.06759. Cited on page 66.

O'Reilly, R. and Rohrlich, J. (2018). "Deep Predictive Learning in Vision". In *2018 Conference on Cognitive Computational Neuroscience*. Cognitive Computational Neuroscience, Philadelphia, Pennsylvania, USA. DOI: 10.32470/CCN.2018.1242-0. Cited on page 146.

Orhan, A. E. and Pitkow, X. (2017). "Skip Connections Eliminate Singularities". pp. 1–16. ARXIV: 1701.09175. Cited on page 60.

Pan, S. J. and Yang, Q. (2010). "A Survey on Transfer Learning". *IEEE Transactions on Knowledge and Data Engineering*, 22, no. 10, 1345–1359. ISSN 1041-4347. DOI: 10.1109/tkde.2009.191. Cited on page 89.

Papernot, N. and McDaniel, P. (2018). "Deep k-Nearest Neighbors: Towards Confident, Interpretable and Robust Deep Learning". ARXIV: 1803.04765. Cited on page 97.

Pavel, M. S., Schulz, H. and Behnke, S. (2015). "Recurrent convolutional neural networks for object-class segmentation of RGB-D video". *Proceedings of the International Joint Conference on Neural Networks*, 2015-Septe. ISSN 08936080. DOI: 10.1109/ijcnn.2015.7280820. Cited on page 22.

Pedamonti, D. (2018). "Comparison of non-linear activation functions for deep neural networks on MNIST classification task". , no. 3. ARXIV: 1804.02763. Cited on page 31.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. and Duchesnay, E. (2011). "Scikit-learn: Machine Learning in Python". *Journal of Machine Learning Research*, 12, 2825–2830. Cited on page 157.

Perry, C. J. and Fallah, M. (2014). "Feature integration and object representations along the dorsal stream visual

hierarchy". *Frontiers in Computational Neuroscience*, 8, 84. ISSN 1662-5188. DOI: 10.3389/fncom.2014.00084 . Cited on pages 39 and 45.

Pesteie, M., Abolmaesumi, P. and Rohling, R. N. (2019). "Adaptive Augmentation of Medical Data Using Independently Conditional Variational Auto-Encoders". *IEEE Transactions on Medical Imaging*, 38, no. 12, 2807–2820. ISSN 1558-254X. DOI: 10.1109/TMI.2019.2914656 . Cited on page 93.

Pezeshki, M., Fan, L., Brakel, P., Courville, A. and Bengio, Y. (2015). "Deconstructing the Ladder Network Architecture". In *Proceedings of Machine Learning Research*, vol. 48. ARXIV: 1511.06430 . Cited on page 76.

Philipp, G., Carbonell, J. G. and Song, D. (2018). "Gradients explode - deep networks are shallow - resnet explained". , no. 2017. Cited on page 60.

Pinheiro, P. and Collobert, R. (2014). "Recurrent convolutional neural networks for scene labeling". *Proceedings of The 31st International Conference ...*, 32, no. June, 82–90. ARXIV: 1306.2795 . Cited on page 63.

Pinheiro, P. O. and Collobert, R. (2015). "From image-level to pixel-level labeling with Convolutional Networks". In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1713–1721. IEEE. ISBN 978-1-4673-6964-0. DOI: 10.1109/cvpr.2015.7298780 . Cited on page 51.

Poggio, T. A. and Vetter, T. (1992). "Recognition and structure from one 2D model view: Observations on prototypes, object classes and symmetries". Tech. Rep., MIT. Cited on page 91.

Prabhu, V., Kannan, A., Ravuri, M., Chablani, M., Sontag, D. and Amatriain, X. (2018). "Prototypical Clustering Networks for Dermatological Disease Diagnosis". ARXIV: 1811.03066 . Cited on pages 109 and 146.

Prabhu, V. U. and Birhane, A. (2020). "Large image datasets: A pyrrhic win for computer vision?" *arXiv:2006.16923 [cs, stat]*. ARXIV: 2006.16923 . Cited on page 160.

Prémont-Schwarz, I., Ilin, A., Hao, T. H., Rasmus, A., Boney, R. and Valpola, H. (2017). "Recurrent Ladder Networks". In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 6009–6019. Curran Associates, Inc. ARXIV: 1707.09219 . Cited on page 76.

Purves, D., Monson, B. B., Sundararajan, J. and Wojtach, W. T. (2014). "How biological vision succeeds in the physical world." *Proceedings of the National Academy of Sciences of the United States of America*, 111, no. 13, 4750–5. ISSN 1091-6490. DOI: 10.1073/pnas.1311309111 . Cited on page 39.

Rahman, S., Khan, S. and Porikli, F. (2018). "A Unified Approach for Conventional Zero-Shot, Generalized Zero-Shot, and Few-Shot Learning". *IEEE Transactions on Image Processing*, 27, no. 11, 5652–5667. ISSN 10577149. DOI: 10.1109/tip.2018.2861573 . ARXIV: 1706.08653 . Cited on pages 101 and 114.

Rajalingham, R., Issa, E. B., Bashivan, P., Kar, K., Schmidt, K. and DiCarlo, J. J. (2018). "Large-Scale, High-Resolution Comparison of the Core Visual Object Recognition Behavior of Humans, Monkeys, and State-of-the-Art Deep Artificial Neural Networks". *The Journal of Neuroscience*, 38, no. 33, 7255–7269. ISSN 0270-6474. DOI: 10.1523/jneurosci.0388-18.2018 . Cited on pages 41, 142, and 168.

Rasmus, A., Valpola, H., Honkala, M., Berglund, M. and Raiko, T. (2015). "Semi-Supervised Learning with Ladder Networks". In *NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems*, vol. 2, pp. 3546–3554. MIT Press Cambridge, MA, USA, Montreal, Canada. ARXIV: 1507.02672 . Cited on page 76.

Ravi, S. and Larochelle, H. (2017). "Optimization as a model for few-shot learning". In *International Conference on Learning Representations (ICRL)*, pp. 1–11. URL: https://openreview.net/forum?id=rJY0-Kcll&noteId=rJY0-Kcll. Cited on pages 108, 109, 119, 159, and 161.

Razavian, A. S., Azizpour, H., Sullivan, J., Carlsson, S., Sharif, A., Hossein, R., Josephine, A., Stefan, S. and Royal, K. T. H. (2014). "CNN Features off-the-shelf : An Astounding Baseline for Recognition". *CVPRW '14 Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 512–519. ISSN

21607516. DOI: 10.1109/cvprw.2014.131. ARXIV: 1403.6382. Cited on pages 51, 98, and 99.

Ren, M., Triantafillou, E., Ravi, S., Snell, J., Swersky, K., Tenenbaum, J. B., Larochelle, H. and Zemel, R. S. (2018). "Meta-Learning for Semi-Supervised Few-Shot Classification". In *ICLR*, pp. 1–15. ARXIV: 1803.00676. Cited on pages 119 and 141.

Ren, S., He, K., Girshick, R. and Sun, J. (2015). "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". *Nips*, pp. 1–10. ISSN 01689002. DOI: 10.1016/j.nima.2015.05.028. ARXIV: 1506.01497v1. Cited on page 65.

Ribeiro, M. T., Singh, S. and Guestrin, C. (2016). "Why Should {I} Trust You?: Explaining the Predictions of Any Classifier". In *22nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining*. ARXIV: 1602.04938. Cited on page 121.

Riesenhuber, M. and Poggio, T. (1999). "Hierarchical models of object recognition in cortex." *Nature neuroscience*, 2, no. 11, 1019–25. ISSN 1097-6256. DOI: 10.1038/14819. Cited on pages 38 and 41.

Rippel, O., Paluri, M., Dollar, P. and Bourdev, L. (2016). "Metric Learning with Adaptive Density Discrimination". In *International Conference on Learning Representations (ICRL)*. ARXIV: 1511.05939. Cited on pages 125 and 147.

Roelfsema, P. R. (2006). "Cortical Algorithms for Perceptual Grouping". *Annual Review of Neuroscience*, 29, no. 1, 203–227. ISSN 0147-006X. DOI: 10.1146/annurev.neuro.29.051605.112939. Cited on pages 34 and 35.

Ronneberger, O., Fischer, P. and Brox, T. (2015). "U-Net: Convolutional Networks for Biomedical Image Segmentation". In N. Navab, J. Hornegger, W. M. Wells and A. F. Frangi (eds.), *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, vol. LNCS 9351, pp. 234–241. Springer International Publishing, Cham. ISBN 978-3-319-24574-4. DOI: 10.1007/978-3-642-40763-5. Cited on page 51.

Rubinstein, R., Bruckstein, A. M. and Elad, M. (2010). "Dictionaries for Sparse Representation Modeling". *Proceedings of the IEEE*, 98, no. 6, 1045–1057. ISSN 0018-9219. DOI: 10.1109/jproc.2010.2040551. Cited on pages 118 and 123.

Ruder, S. (2017a). "An Overview of Multi-Task Learning in Deep Neural Networks". *arXiv:1706.05098* [*cs, stat*]. ARXIV: 1706.05098. Cited on pages 111 and 112.

Ruder, S. (2017b). "Transfer Learning - Machine Learning' s Next Frontier". URL: http://ruder.io/transfer-learning/. Cited on page 89.

Rumelhart, D. E. and McClelland, J. L. (1987). "Learning Internal Representations by Error Propagation". In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*, pp. 318–362. MITP. ISBN 978-0-262-29140-8. URL: http://ieeexplore.ieee.org/document/6302929. Cited on page 22.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., Berg, A. C. and Fei-Fei, L. (2015). "ImageNet large scale visual recognition challenge". *International Journal of Computer Vision (IJCV)*, 115, no. 3, 211–252. DOI: 10.1007/s11263-015-0816-y. ARXIV: 1409.0575. Cited on pages 159 and 161.

Saalmann, Y. B., Pinsk, M. A., Wang, L., Li, X. and Kastner, S. (2012). "The Pulvinar Regulates Information Transmission Between Cortical Areas Based on Attention Demands". *Science*, 337, no. 6095, 753–756. ISSN 0036-8075. DOI: 10.1126/science.1223082. Cited on page 115.

Sajjadi, M., Javanmardi, M. and Tasdizen, T. (2016). "Regularization With Stochastic Transformations and Perturbations for Deep Semi-Supervised Learning". In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon and R. Garnett (eds.), *Advances in Neural Information Processing Systems 29*, pp. 1163–1171. Curran Associates, Inc. URL: http://papers.nips.cc/paper/6333-regularization-with-stochastic-transformations-and-perturbations-for-deep-semi-supervised-learning.pdf. Cited on

page 106.

Santoro, A., Bartunov, S., Botvinick, M., Wierstra, D. and Lillicrap, T. (2016). "One-shot Learning with Memory-Augmented Neural Networks". *arXiv preprint*. ISSN 19449224. DOI: 10.1002/2014gb005021. ARXIV: 1605.06065. Cited on pages 119 and 120.

Savarese, P. H. P. and Maire, M. (2019). "Learning Implicitly Recurrent CNNs Through Parameter Sharing". ARXIV: 1902.09701. Cited on page 63.

Schrimpf, M., Kubilius, J., Hong, H., Majaj, N. J., Rajalingham, R., Issa, E. B., Kar, K., Bashivan, P., Prescott-Roy, J., Schmidt, K., Yamins, D. L. K. and DiCarlo, J. J. (2018). "Brain-Score: Which Artificial Neural Network for Object Recognition is most Brain-Like?" DOI: 10.1101/407007. BIORXIV: 407007. Cited on page 42.

Schroff, F., Kalenichenko, D. and Philbin, J. (2015). "FaceNet: A unified embedding for face recognition and clustering". In *Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 07-12-June, pp. 815–823. IEEE. DOI: 10.1109/cvpr.2015.7298682. ARXIV: 1503.03832. Cited on pages 105 and 106.

Schwarz, M., Milan, A., Lenz, C., Mu, A., Periyasamy, A. S., Schreiber, M., Sch, S. and Behnke, S. (2017). "NimbRo Picking : Versatile Part Handling for Warehouse Automation". *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3032–3039. DOI: 10.1109/icra.2017.7989348. Cited on page 51.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R. and LeCun, Y. (2013). "OverFeat: Integrated Recognition, Localization and Detection using Convolutional Networks". ARXIV: 1312.6229. Cited on pages 41 and 51.

Serre, T., Wolf, L., Bileschi, S., Riesenhuber, M. and Poggio, T. (2007). "Robust object recognition with cortex-like mechanisms". *IEEE transactions on pattern analysis and machine intelligence*, 29, no. 3, 411–426. DOI: 10.1109/TPAMI.2007.56. Cited on page 118.

Serre, T., Wolf, L. and Poggio, T. (2005). "Object recognition with features inspired by visual cortex". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2, 994–1000. ISSN 10636919. DOI: 10.1109/cvpr.2005.254. ARXIV: cs/0112017. Cited on page 38.

Settles, B. (2010). "Active Learning Literature Survey". Tech. Rep.. Cited on page 97.

Shalev-Shwartz, S. and Ben-David, S. (2014). *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press. ISBN 1-107-05713-2. DOI: 10.1017/CBO9781107298019. Cited on pages 85 and 123.

Shelhamer, E., Long, J. and Darrell, T. (2017). "Fully Convolutional Networks for Semantic Segmentation". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 39, no. 4, 640–651. ISSN 0162-8828. DOI: 10.1109/tpami.2016.2572683. ARXIV: 1605.06211. Cited on pages 52, 55, 100, and 141.

Shen, F., Gan, R. and Zeng, G. (2016). "Weighted residuals for very deep networks". In *International Conference on Systems and Informatics, ICSAI*, pp. 936–941. ISBN 978-1-5090-5521-0. DOI: 10.1109/icsai.2016.7811085. ARXIV: 1605.08831. Cited on page 56.

Shetty, R. and Laaksonen, J. (2015). "Video captioning with recurrent networks based on frame- and video-level features and visual content classification". *arXiv:1512.02949 [cs]*. ARXIV: 1512.02949. Cited on page 22.

Shin, H.-C., Roth, H. R., Gao, M., Lu, L., Xu, Z., Nogues, I., Yao, J., Mollura, D. and Summers, R. M. (2016). "Deep Convolutional Neural Networks for Computer-Aided Detection: CNN Architectures, Dataset Characteristics and Transfer Learning". *IEEE Transactions on Medical Imaging*, 35, no. 5, 1285–1298. DOI: 10.1109/tmi.2016.2528162. Cited on pages 99 and 126.

Shrikumar, A., Greenside, P. and Kundaje, A. (2017). "Learning Important Features Through Propagating Activation Differences". ARXIV: 1704.02685. Cited on page 121.

Shrivastava, A. and Gupta, A. (2016). "Contextual Priming and Feedback for Faster R-CNN". In B. Leibe,

J. Matas, N. Sebe and M. Welling (eds.), *Computer Vision – ECCV 2016*, vol. 9905, pp. 330–348. Springer International Publishing, Cham. ISBN 978-3-319-46447-3. DOI: 10.1007/978-3-319-46448-0_20 . Cited on page 65.

Shu, J., Xu, Z. and Meng, D. (2018). "Small Sample Learning in Big Data Era". ISSN 1872826X. DOI: arXiv:1808.04572v1 . ARXIV: 1808.04572 . Cited on page 108.

Shui, C., Zhou, F., Gagné, C. and Wang, B. (2019). "Deep Active Learning: Unified and Principled Method for Query and Training". *arXiv:1911.09162* [*cs, stat*]. ARXIV: 1911.09162 . Cited on page 98.

Siam, M., Valipour, S., Jagersand, M. and Ray, N. (2017). "Convolutional gated recurrent networks for video segmentation". In *2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3090–3094. DOI: 10.1109/ICIP.2017.8296851 . Cited on page 22.

Simard, P. Y., Steinkraus, D. and Platt, J. (2003). "Best Practices for Convolutional Neural Networks Applied to Visual Document Analysis". URL: https://www.microsoft.com/en-us/research/publication/best-practices-for-convolutional-neural-networks-applied-to-visual-document-analysis/. Cited on page 91.

Siméoni, O., Budnik, M., Avrithis, Y. and Gravier, G. (2019). "Rethinking deep active learning: Using unlabeled data at model training". *arXiv:1911.08177* [*cs*]. ARXIV: 1911.08177 . Cited on pages 97 and 98.

Simonyan, K. and Zisserman, A. (2015). "Very Deep Convolutional Networks for Large-Scale Image Recognition". In *International Conference on Learning Representations (ICRL)*, pp. 1–14. ARXIV: 1409.1556 . Cited on pages 41, 51, and 104.

Sironi, A., Tekin, B., Rigamonti, R., Lepetit, V. and Fua, P. (2015). "Learning Separable Filters". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37, no. 1, 94–106. ISSN 0162-8828, 2160-9292. DOI: 10.1109/TPAMI.2014.2343229 . Cited on page 107.

Sivic, J. and Zisserman, A. (2003). "Video Google: A text retrieval approach to object matching in videos". In *Proceedings Ninth IEEE International Conference on Computer Vision*, pp. 1470–1477 vol.2. IEEE, Nice, France. ISBN 978-0-7695-1950-0. DOI: 10.1109/ICCV.2003.1238663 . Cited on page 27.

Snell, J., Swersky, K. and Zemel, R. (2017). "Prototypical Networks for Few-shot Learning". In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan and R. Garnett (eds.), *Advances in Neural Information Processing Systems 30*, pp. 4077–4087. Curran Associates, Inc. ARXIV: 1703.05175 . Cited on pages 14, 95, 109, 117, 119, 121, 132, 137, 138, 144, and 170.

Soekhoe, D., van der Putten, P. and Plaat, A. (2016). "On the Impact of Data Set Size in Transfer Learning Using Deep Neural Networks". In H. Boström, A. Knobbe, C. Soares and P. Papapetrou (eds.), *Advances in Intelligent Data Analysis XV: 15th International Symposium, IDA 2016, Stockholm, Sweden, October 13-15, 2016, Proceedings*, pp. 50–60. Springer International Publishing, Cham. ISBN 978-3-319-46349-0. DOI: 10.1007/978-3-319-46349-0_5 . Cited on page 100.

Sohn, K. (2016). "Improved Deep Metric Learning with Multi-class N-pair Loss Objective". *30th Conference on Neural Information Processing Systems*, , no. Nips, 1857–1865. ISSN 10495258. DOI: 10.1080/02678373.2017.1304463 . Cited on page 105.

Sønderby, S. K., Sønderby, C. K., Maaløe, L. and Winther, O. (2015). "Recurrent Spatial Transformer Networks". pp. 1–9. ARXIV: 1509.05329 . Cited on page 67.

Song, H. O., Jegelka, S., Rathod, V. and Murphy, K. (2017). "Deep Metric Learning via Facility Location". In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5382–5390. ARXIV: 1612.01213 . Cited on pages 117 and 119.

Soufi, N. and Valdenegro-Toro, M. (2019). "Data augmentation with Symbolic-to-Real Image Translation GANs for Traffic Sign Recognition". *arXiv:1907.12902* [*cs, stat*]. ARXIV: 1907.12902 . Cited on page 94.

Spoerer, C. J., McClure, P. and Kriegeskorte, N. (2017). "Recurrent Convolutional Neural Networks: A Better Model Of Biological Object Recognition Under Occlusion". *Frontiers in Psychology*, 8, 1551. ISSN 1664-1078. DOI: 10.3389/fpsyg.2017.01551 . Cited on pages 63, 64, 68, 69, 83, and 149.

Springenberg, J. T., Dosovitskiy, A., Brox, T. and Riedmiller, M. (2014). "Striving for Simplicity: The All Convolutional Net". ARXIV: 1412.6806 . Cited on pages 30, 31, and 38.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. and Salakhutdinov, R. (2014). "Dropout: Prevent NN from overfitting". *Journal of Machine Learning Research*, 15, 1929–1958. ISSN 15337928. DOI: 10.1214/12-aos1000 . ARXIV: 1102.4807 . Cited on page 104.

Srivastava, R. K., Greff, K. and Schmidhuber, J. (2015). "Highway Networks". ARXIV: 1505.00387 . Cited on pages 56 and 57.

Stollenga, M. F., Masci, J., Gomez, F. and Schmidhuber, J. J. (2014). "Deep Networks with Internal Selective Attention through Feedback Connections". pp. 3545–3553. ARXIV: 1407.3068 . Cited on pages 69, 70, 75, and 78.

Sundararajan, M., Taly, A. and Yan, Q. (2017). "Axiomatic Attribution for Deep Networks". In *International Conference on Machine Learning*. ARXIV: 1703.01365 . Cited on page 120.

Sundermeyer, M., Ney, H. and Schlüter, R. (2015). "From Feedforward to Recurrent LSTM Neural Networks for Language Modeling". *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 23, no. 3, 517–529. ISSN 2329-9304. DOI: 10.1109/TASLP.2015.2400218 . Cited on page 22.

Szegedy, C., Ioffe, S., Vanhoucke, V. and Alemi, A. (2016a). "Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning". *Arxiv*, p. 12. ARXIV: 1602.07261 . Cited on pages 41, 54, and 56.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V. and Rabinovich, A. (2015). "Going deeper with convolutions". In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9. ISBN 978-1-4673-6964-0. DOI: 10.1109/cvpr.2015.7298594 . ARXIV: 1409.4842 . Cited on pages 29, 31, 41, 53, and 126.

Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. (2016b). "Rethinking the Inception Architecture for Computer Vision". *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826. ARXIV: 1512.00567 . Cited on pages 53 and 54.

Targ, S., Almeida, D. and Lyman, K. (2016). "Resnet in Resnet: Generalizing Residual Architectures". ARXIV: 1603.08029 . Cited on page 57.

Teftef, E., Carvajal, C., Viéville, T. and Alexandre, F. (2013). "When early vision in the retina attempts to take decisions about visual motion events : The role of konio cells". URL: https://hal.inria.fr/hal-00826099. Cited on pages 34 and 43.

Theis, L., van den Oord, A. and Bethge, M. (2016). "A note on the evaluation of generative models". *arXiv:1511.01844 [cs, stat]*. ARXIV: 1511.01844 . Cited on page 93.

Torralba, A., Fergus, R. and Freeman, W. (2008). "80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30, no. 11, 1958–1970. ISSN 0162-8828. DOI: 10.1109/TPAMI.2008.128 . Cited on page 160.

Torralba, A. and Oliva, A. (2003). "Statistics of natural image categories". *Network (Bristol, England)*, 14, no. 3, 391–412. ISSN 0954-898X. DOI: 10.1088/0954-898x/14/3/302 . Cited on page 27.

Triantafillou, E., Zemel, R. and Urtasun, R. (2017). "Few-Shot Learning Through an Information Retrieval Lens". Tech. Rep.. ARXIV: 1707.02610 . Cited on pages 88, 109, and 118.

Tsai, Y.-H., Hamsici, O. C. and Yang, M.-H. (2015). "Adaptive region pooling for object detection". In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 07-12-June, pp. 731–739. IEEE. ISBN

978-1-4673-6964-0. DOI: 10.1109/cvpr.2015.7298673. Cited on page 31.

Ungerleider, L. G. and Haxby, J. V. (1994). ‘"What" and "where" in the human brain”. *Current Opinion in Neurobiology*, 4, no. 2, 157–165. ISSN 0959-4388. DOI: 10.1016/0959-4388(94)90066-3. Cited on page 34.

Vallet, A. and Sakamoto, H. (2016). “Convolutional Recurrent Neural Networks for Better Image Understanding”. In *2016 International Conference on Digital Image Computing: Techniques and Applications (DICTA)*, pp. 1–7. IEEE. ISBN 978-1-5090-2896-2. DOI: 10.1109/dicta.2016.7797026. Cited on pages 64, 67, 70, 71, 72, 74, 78, and 81.

Valpola, H. (2014). “From neural PCA to deep unsupervised learning”. pp. 1–23. ARXIV: 1411.7783. Cited on page 76.

Vanschoren, J. (2018). “Meta-Learning: A Survey”. *arXiv:1810.03548 [cs, stat]*. ARXIV: 1810.03548. Cited on page 108.

Veit, A., Wilber, M. and Belongie, S. (2016). “Residual Networks Behave Like Ensembles of Relatively Shallow Networks”. ISSN 10495258. ARXIV: 1605.06431. Cited on page 59.

Verma, V., Lamb, A., Kannala, J., Bengio, Y. and Lopez-Paz, D. (2019). “Interpolation Consistency Training for Semi-Supervised Learning”. *arXiv:1903.03825 [cs, stat]*. ARXIV: 1903.03825. Cited on page 92.

Viéville, T. and Crahay, S. (2004a). “A deterministic biologically plausible classifier”. *Neurocomputing*, 58-60, 923–928. ISSN 09252312. DOI: 10.1016/j.neucom.2004.01.147. Cited on page 35.

Viéville, T. and Crahay, S. (2004b). “Using an Hebbian Learning Rule for Multi-Class SVM Classifiers”. *Journal of Computational Neuroscience*, 17, no. 3, 271–287. ISSN 0929-5313. DOI: 10.1023/b:jcns.0000044873.20850.9c. Cited on pages 115, 118, and 140.

Viéville, T., Hinaut, X., Drumond, T. F. and Alexandre, F. (2017). “Recurrent neural network weight estimation through backward tuning”. Research Report RR-9100, INRIA. URL: https://hal.inria.fr/hal-01610735. Cited on page 115.

Vincent, P., Larochelle, H., Bengio, Y. and Manzagol, P.-A. (2008). “Extracting and composing robust features with denoising autoencoders”. In *Proceedings of the 25th International Conference on Machine Learning - ICML ’08*, pp. 1096–1103. ACM Press, Helsinki, Finland. ISBN 978-1-60558-205-4. DOI: 10.1145/1390156.1390294. Cited on page 23.

Vinyals, O., Blundell, C., Lillicrap, T., Kavukcuoglu, K. and Wierstra, D. (2016). “Matching Networks for One Shot Learning”. *NIPS*. ARXIV: 1606.04080. Cited on pages 14, 109, 119, 138, 159, and 161.

Visin, F., Kastner, K., Cho, K., Matteucci, M., Courville, A. and Bengio, Y. (2015). “ReNet: A Recurrent Neural Network Based Alternative to Convolutional Networks”. ARXIV: 1505.00393. Cited on pages 64, 65, 70, 73, and 78.

Visin, F., Romero, A., Cho, K., Matteucci, M., Ciccone, M., Kastner, K., Bengio, Y. and Courville, A. (2016). “ReSeg: A Recurrent Neural Network-Based Model for Semantic Segmentation”. *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 426–433. ISSN 21607516. DOI: 10.1109/cvprw.2016.60. ARXIV: 1511.07053v1. Cited on pages 70 and 78.

Wah, C., Branson, S., Welinder, P., Perona, P. and Belongie, S. (2011). “The Caltech-UCSD Birds-200-2011 Dataset”. Tech. Rep. CNS-TR-2011-001, California Institute of Technology. URL: http://www.vision.caltech.edu/visipedia/papers/CUB_200_2011.pdf. Cited on pages 88 and 159.

Walker, R. A. (2010). “Sociocultural Issues in Motivation”. In P. Peterson, E. Baker and B. McGaw (eds.), *International Encyclopedia of Education (Third Edition)*, pp. 712–717. Elsevier, Oxford. ISBN 978-0-08-044894-7. DOI: 10.1016/B978-0-08-044894-7.00629-1. Cited on page 110.

Wan, L., Zeiler, M., Zhang, S., LeCun, Y. and Fergus, R. (2013). "Regularization of neural networks using dropconnect". In *International Conference on Machine Learning*, 1, pp. 109–111. ARXIV: 1509.08985 . Cited on page 104.

Wang, B., Luo, X., Li, Z., Zhu, W., Shi, Z. and Osher, S. J. (2018a). "Deep Neural Nets with Interpolating Function as Output Activation". ARXIV: 1802.00168 . Cited on page 31.

Wang, K., Zhang, D., Li, Y., Zhang, R. and Lin, L. (2017a). "Cost-Effective Active Learning for Deep Image Classification". *IEEE Transactions on Circuits and Systems for Video Technology*, 27, no. 12, 2591–2600. ISSN 1051-8215. DOI: 10.1109/TCSVT.2016.2589879 . Cited on page 97.

Wang, P., Chen, P., Yuan, Y., Liu, D., Huang, Z., Hou, X. and Cottrell, G. (2017b). "Understanding Convolution for Semantic Segmentation". ARXIV: 1702.08502 . Cited on page 30.

Wang, Q., Zhang, J., Song, S. and Zhang, Z. (2014). "Attentional Neural Network: Feature Selection Using Cognitive Feedback". In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence and K. Q. Weinberger (eds.), *Advances in Neural Information Processing Systems 27*, pp. 2033–2041. Curran Associates, Inc. ARXIV: 1411.5140 . Cited on pages 70 and 78.

Wang, Y., Wu, X.-M., Li, Q., Gu, J., Xiang, W., Zhang, L. and Li, V. O. K. (2018b). "Large Margin Few-Shot Learning". ARXIV: 1807.02872 . Cited on page 147.

Weinshall, D., Cohen, G. and Amir, D. (2018). "Curriculum Learning by Transfer Learning: Theory and Experiments with Deep Networks". *arXiv:1802.03796* [*cs*]. ARXIV: 1802.03796 . Cited on pages 110 and 111.

Weiss, K., Khoshgoftaar, T. M. and Wang, D. (2016). "A survey of transfer learning". *Journal of Big Data*, 3, no. 1, 9. ISSN 2196-1115. DOI: 10.1186/s40537-016-0043-6 . Cited on pages 89 and 114.

Wen, H., Han, K., Shi, J., Zhang, Y., Culurciello, E. and Liu, Z. (2018). "Deep Predictive Coding Network for Object Recognition". ARXIV: 1802.04762 . Cited on pages 64, 66, 70, 74, 78, 81, and 83.

Wu, Y., Schuster, M., Chen, Z., Le, Q. V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, Ł., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M. and Dean, J. (2016). "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation". *arXiv:1609.08144* [*cs*]. ARXIV: 1609.08144 . Cited on page 22.

Wyatte, D., Jilk, D. J. and O'Reilly, R. C. (2014). "Early recurrent feedback facilitates visual object recognition under challenging conditions". *Frontiers in Psychology*, 5, no. JUL, 674. ISSN 16641078. DOI: 10.3389/fpsyg.2014.00674 . Cited on page 63.

Xian, Y., Lampert, C. H., Schiele, B. and Akata, Z. (2018). "Zero-Shot Learning - A Comprehensive Evaluation of the Good, the Bad and the Ugly". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, , no. c. ISSN 0162-8828. DOI: 10.1109/tpami.2018.2857768 . Cited on pages 101 and 102.

Xiao, D., Huang, Y., Qin, C., Liu, Z., Li, Y. and Liu, C. (2019). "Transfer learning with convolutional neural networks for small sample size problem in machinery fault diagnosis". *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science*, 233, no. 14, 5131–5143. ISSN 0954-4062. DOI: 10.1177/0954406219840381 . Cited on page 99.

Xie, Q., Dai, Z., Hovy, E., Luong, M.-T. and Le, Q. V. (2019). "Unsupervised Data Augmentation for Consistency Training". *arXiv:1904.12848* [*cs, stat*]. ARXIV: 1904.12848 . Cited on pages 91, 92, and 95.

Xie, S., Girshick, R., Dollár, P., Tu, Z. and He, K. (2017a). "Aggregated residual transformations for deep neural networks". In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, pp. 5987–5995. ISBN 978-1-5386-0457-1. DOI: 10.1109/cvpr.2017.634 . ARXIV: 1611.05431 . Cited on page 56.

Xie, W., Noble, A. and Zisserman, A. (2017b). "Layer Recurrent Neural Networks". pp. 1–17. URL: https://ayearofai.com/rohan-lenny-3-recurrent-neural-networks-10300100899b. Cited on pages 65, 70, 73, 77, and 78.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A., Salakhutdinov, R., Zemel, R. and Bengio, Y. (2015). "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention". ARXIV: 1502.03044. Cited on page 67.

Xu, M., Gao, M., Chen, Y.-T., Davis, L. S. and Crandall, D. J. (2019). "Temporal Recurrent Networks for Online Action Detection". In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 5532–5541. URL: https://openaccess.thecvf.com/content_ICCV_2019/html/Xu_Temporal_Recurrent_Networks_for_Online_Action_Detection_ICCV_2019_paper.html. Cited on page 22.

Yadav, A. and Singh, G. (2015). "Incremental k-means clustering algorithms: A review". *International Journal of Latest Trends in Engineering and Technology*, 5, no. 4. URL: http://portal.acm.org/citation.cfm?id=1596883. Cited on page 140.

Yamins, D. L. and DiCarlo, J. J. (2016). "Eight open questions in the computational modeling of higher sensory cortex". *Current Opinion in Neurobiology*, 37, 114–120. ISSN 0959-4388. DOI: 10.1016/j.conb.2016.02.001. Cited on pages 41, 142, and 167.

Yamins, D. L. K., Hong, H., Cadieu, C. F., Solomon, E. A., Seibert, D. and DiCarlo, J. J. (2014). "Performance-optimized hierarchical models predict neural responses in higher visual cortex". *Proceedings of the National Academy of Sciences*, 111, no. 23, 8619–8624. ISSN 0027-8424. DOI: 10.1073/pnas.1403112111. ARXIV: 0706.1062v1. Cited on page 41.

Yang, J., Jiang, Y.-G., Hauptmann, A. G. and Ngo, C.-W. (2007). "Evaluating bag-of-visual-words representations in scene classification". DOI: 10.1145/1290082.1290111. Cited on page 28.

Yin, C., Qian, B., Cao, S., Li, X., Wei, J., Zheng, Q. and Davidson, I. (2017). "Deep Similarity-Based Batch Mode Active Learning with Exploration-Exploitation". In *2017 IEEE International Conference on Data Mining (ICDM)*, pp. 575–584. IEEE, New Orleans, LA. ISBN 978-1-5386-3835-4. DOI: 10.1109/ICDM.2017.67. Cited on page 98.

Yoshida, Y. and Miyato, T. (2017). "Spectral Norm Regularization for Improving the Generalizability of Deep Learning". *arXiv:1705.10941 [cs, stat]*. ARXIV: 1705.10941. Cited on page 104.

Yu, F. and Koltun, V. (2016). "Multi-Scale Context Aggregation by Dilated Convolutions". In *International Conference on Learning Representations (ICRL)*. ARXIV: 1511.07122. Cited on page 100.

Yu, Y., Si, X., Hu, C. and Zhang, J. (2019). "A Review of Recurrent Neural Networks: LSTM Cells and Network Architectures". *Neural Computation*, 31, no. 7, 1235–1270. ISSN 0899-7667. DOI: 10.1162/neco_a_01199. Cited on page 22.

Zagoruyko, S. and Komodakis, N. (2016). "Wide Residual Networks". *Arxiv*. ARXIV: 1605.07146. Cited on page 62.

Zamir, A. R., Wu, T. L., Sun, L., Shen, W. B., Shi, B. E., Malik, J. and Savarese, S. (2017). "Feedback networks". In *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-Janua, pp. 1808–1817. ISBN 978-1-5386-0457-1. DOI: 10.1109/cvpr.2017.196. ARXIV: 1612.09508. Cited on pages 11, 66, 70, 71, 72, 74, 78, 79, 80, 108, and 111.

Zeiler, M. (2014). "Hierarchical Convolutional Deep Learning in Computer Vision". Ph.D. thesis, New York University. Cited on page 62.

Zeiler, M. D. and Fergus, R. (2014). "Visualizing and Understanding Convolutional Networks". *Lecture Notes in Computer Science*, 8689, 818–833. ISSN 978-3-319-10589-5. DOI: 10.1007/978-3-319-10590-1_53. ARXIV: 1311.2901. Cited on pages 30, 38, 41, 50, 100, and 120.

Zhang, C., Bengio, S., Hardt, M., Recht, B. and Vinyals, O. (2016a). "Understanding deep learning requires

rethinking generalization". ARXIV: 1611.03530. Cited on page 103.

Zhang, H., Cisse, M., Dauphin, Y. N. and Lopez-Paz, D. (2018). "Mixup: Beyond Empirical Risk Minimization". *arXiv:1710.09412* [*cs, stat*]. ARXIV: 1710.09412. Cited on page 92.

Zhang, K., Sun, M., Han, T. X., Yuan, X., Guo, L. and Liu, T. (2016b). "Residual Networks of Residual Networks: Multilevel Residual Networks". 14, no. 8, 1–12. ISSN 1051-8215. DOI: 10.1109/tcsvt.2017.2654543. ARXIV: 1608.02908. Cited on page 57.

Zhao, Y. and Park, I. M. (2016). "Interpretable Nonlinear Dynamic Modeling of Neural Trajectories". In D. D. Lee, M. Sugiyama, U. V. Luxburg, I. Guyon and R. Garnett (eds.), *Advances in Neural Information Processing Systems 29*, pp. 3333–3341. Curran Associates, Inc. ARXIV: 1608.06546. Cited on page 114.

Zheng, L., Yang, Y. and Tian, Q. (2018). "SIFT Meets CNN: A Decade Survey of Instance Retrieval". *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40, no. 5, 1224–1244. ISSN 0162-8828. DOI: 10.1109/tpami.2017.2709749. ARXIV: 1608.01807. Cited on pages 27 and 28.

Zhu, J.-Y., Park, T., Isola, P. and Efros, A. A. (2017a). "Unpaired Image-to-Image Translation Using Cycle-Consistent Adversarial Networks". In *2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2242–2251. DOI: 10.1109/ICCV.2017.244. Cited on page 23.

Zhu, W., Qiu, Q., Huang, J., Calderbank, R., Sapiro, G. and Daubechies, I. (2017b). "LDMNet: Low Dimensional Manifold Regularized Neural Networks". ARXIV: 1711.06246. Cited on pages 95, 105, 141, and 147.

Zou, Z., Shi, Z., Guo, Y. and Ye, J. (2019). "Object Detection in 20 Years: A Survey". ARXIV: 1905.05055. Cited on page 21.

Zuo, Z., Shuai, B., Wang, G., Liu, X., Wang, X., Wang, B. and Chen, Y. (2016). "Learning Contextual Dependence With Convolutional Hierarchical Recurrent Neural Networks". *IEEE Transactions on Image Processing*, 25, no. 7, 2983–2996. ISSN 10577149. DOI: 10.1109/tip.2016.2548241. ARXIV: 1509.03877. Cited on pages 65, 70, 73, 74, and 78.

This bibliography contains 352 references.