Does Large Pretrained Dataset Always Help? On the Effect of Dataset Size on Big Transfer Model

Rui Li, Yang Tian, Qibin Chen, Xiangyu Zhu, Yongfei Jia, Hui Zhang, Xue Li, Kai Jiang, Qiang Duan

▶ **To cite this version:**

Rui Li, Yang Tian, Qibin Chen, Xiangyu Zhu, Yongfei Jia, et al.. Does Large Pretrained Dataset Always Help? On the Effect of Dataset Size on Big Transfer Model. 12th International Conference on Intelligent Information Processing (IIP), May 2022, Qingdao, China. pp.136-147, 10.1007/978-3-031-03948-5_12 . hal-04178721

HAL Id: hal-04178721
https://inria.hal.science/hal-04178721v1

Submitted on 8 Aug 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License

This document is the original author manuscript of a paper submitted to an IFIP conference proceedings or other IFIP publication by Springer Nature. As such, there may be some differences in the official published version of the paper. Such differences, if any, are usually due to reformatting during preparation for publication or minor corrections made by the author(s) during final proofreading of the publication manuscript.

Does Large Pretrained Dataset always help? On the Effect of Dataset Size on Big Transfer Model

Xue Li*[1], Kai Jiang*[1], Qiang Duan*[1],
Rui Li[1], Yang Tian[1], Qibin Chen[1], Xiangyu Zhu[1], Yongfei Jia[2], and
Hui Zhang[1] (✉)

[1] Inspur Academy of Science and Technology, Shandong, China
[2] Qilu University of Technology(Shandong Academy of Sciences), Shandong, China
zhanghui@inspur.com
∗ Authors contributed equally to this study.

Abstract. Transfer learning often refers to an approach concerning machine learning in which the programmers relocate an initially developed model as the starting point for a model on a consequent task. It is evident that deep neural networks entail massive datasets to build and train feasible models. To gain massive and qualified datasets, nonetheless, seems expensive and time-consuming. This paper firstly examines the scale of a dataset so as to train an effective model as well as pertinent reactions corresponding to partial changes made to the scale of the dataset. In the practical experiments via training deep neural networks, we simplify hyperparameter tuning via transfer of pre-trained representations, hence promoting sample efficiency. To verify the usefulness of pre-trained models in datasets of different sizes, we have done relevant experiments on two benchmark datasets, cifar10 and cifar100. The results demonstrate that the larger the size of the pre-trained model, the better the fine-tuning effect of the network. With detailed analysis of primary elements contributing to high transfer performance, we aim to utilize pre-trained models with more efficient performance on dataset named as ImageNet-21k to benefit the computer vision research, in contrast to traditional models pre-trained on the smaller dataset, ILSVRC- 2012.

Keywords: EfficientNet · Group Normalization · MixUp · Residual Networks · Transfer Learning · Weight Standardization

1 INTRODUCTION

As the data volume grows rapidly, it is common to build the deep learning model in a fast and convenient way. However, due to the insufficiency of labelled data, it is time-consuming to collect labelled data and build a deep neutral network from the scratch. Thus, it is necessary to make effective use of the model and the data with labels [17]. Given the considerable consumption of compute and time required to develop neural network models pertinent to those problems, it is a popular approach in deep learning to harness pre-trained models as the starting point on computer vision tasks. With a vast and generic dataset established, less

compute and fewer data points will give an edge to initializing subsequent tasks through the imported weights from pre-trained dataset [10].

Employing pre-trained weights and large models is becoming a substantial approach for image classification tasks, which generally considers ImageNet as the training dataset [11]. Scholars like Yosinski et al. simply utilized the ImageNet dataset [19]. Nevertheless, primary differences are manifested between natural image classification and the target tasks in terms of task specifications, features, and sizes, the effects of transfer deserve further study [14]. It can be intriguing to scrutinize whether the traits of transfer learning differ when employing a dataset from a distinctive domain. Our research starts with drawing upon pre-trained model for transfer learning to explore impacts on dataset scale, hence examining the interaction between sizes of architecture and data as well as training hyper-parameters. With such a mechanism, residual neutral networks with 50 layers will be pre-trained on a larger dataset ImageNet-21k including images of 14M and a smaller dataset ILSVRC-2012 containing images of 1.3M.

Recent progress in deep learning has been largely attributed to the scale and diversity of data gathered in recent years. Thanks to data augmentation, we are able to considerably multiply the data diversity to enlarge the scale of models for training, acquiring proliferated new data. When it comes to train neural networks of massive size, we usually employ strategies including flipping, padding, and cropping. However, it ought to be noticed that massive deep neural networks tend to reveal unexpected conditions, sometimes negative, in spite of its large size and presumed powerfulness. For instance, they can store and react to some opposite cases. In order to mitigate the negative effects, a brief learning mechanism named MixUp [21] was put forward. This tool regularizes the neural network to sustain reaction of linear correlations amid training cases. It is indicated that MixUp greatly enhances the effectiveness of generalization concerning cutting-edge frameworks of neural network via a series of tests on UCI datasets, CIFAR-10 [21], as well as ImageNet-2012 [2]. In addition to its powerfulness on generalization, deep learning can be more stable and faster by batch normalization [7] which standardizes the inputs to a layer for every sub-batch through re-entering and re-scaling. As a milestone work in deep learning, it is conducive to speeding up the process of training and convergence and preventing the overfitting problem as well. Subsequently, encouraged by the development of batch normalization, layer normalization [1], instance normalization [6], and group normalization [18] emerged one after another, which promotes the development of deep learning in multiple dimensions.

Being an integral part of the computer vision including video surveillance and image retrieval, object detection serves to concisely identify the location and category of a particular object in targeted images or even videos. There are basically two approaches for object detection, that is, one-stage and two-stage (one-stage network is represented by Yolo [12] and two -stage network is rep-

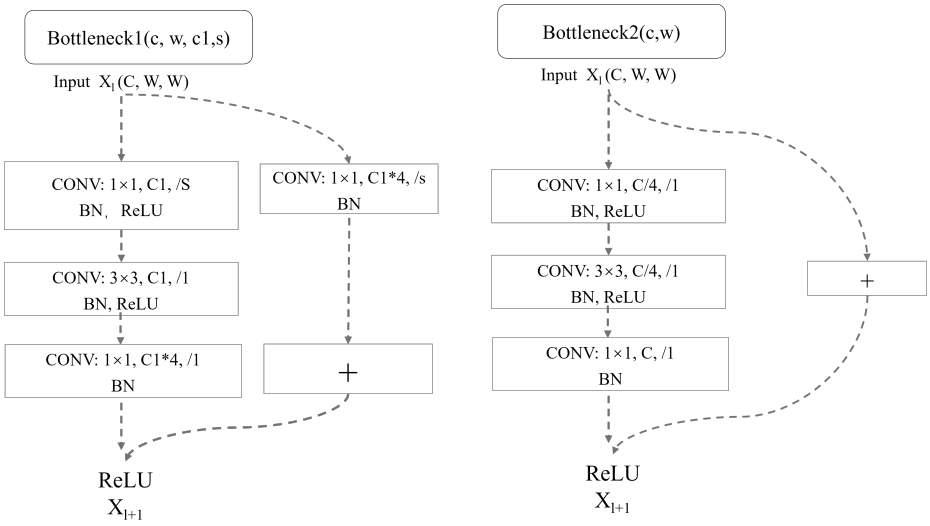

Fig. 1. Bottleneck Architectures

resented by Faster-RCNN [13]). In fact, dramatic progress can be witnessed in object detection algorithm in terms of feature extraction, image expression, classification, and recognition via models of feature extraction and transfer learning capabilities of deep convolutional neural networks. In future studies, we intend to achieve the goals of object detection and semantic segmentation by drawing upon the Big Transfer Model constructed below.

2 APPROACH

The generalizability of features has been examined in several studies, with a success of transfer learning testified. It should be noticeable that fine-tuning serves as a regular scheme for transfer learning. In the wake of re-examining the standards of residual network and fine-tuning the model on assigned objects, our model with larger dataset via diversifying the size of pertinent datasets works better than previous ones.

2.1 A Parsimonious Model - ResNet 50

Breakthrough regarding image classification can be made due to deep convolutional neural networks. Propelled by an increasing importance of depth, the problem of network degradation arises. To grapple with this tricky problem, He K. et al. [3] proposed a solution in which residual network is established by constructing residual blocks consisting of shortcut connection x and and residual

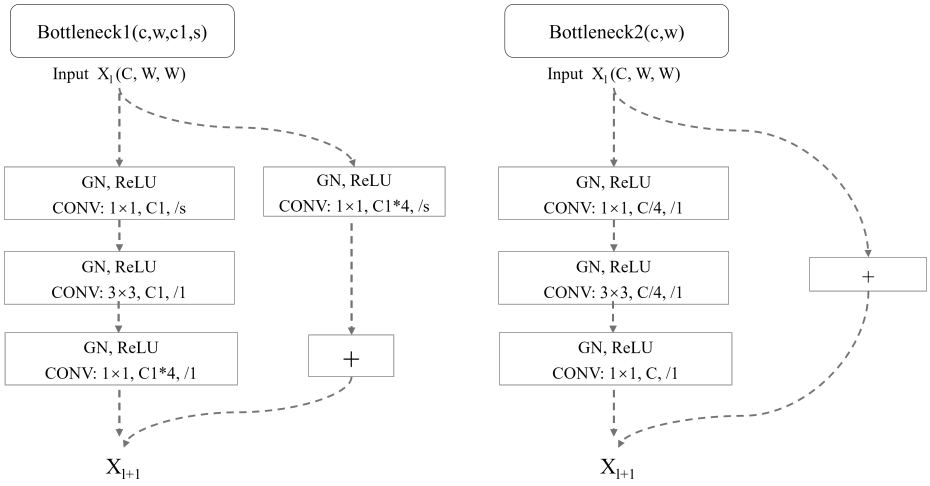

Fig. 2. The Modified Residual Network

function $F(x)$. A residual block can be expressed as: $x_L = x_l + \sum_{i=l}^{L-1} F(x_i, W_i)$. The structure of ResNet is shown in Fig.1 and Fig.3.

After analyzing the propagation formulation of the deep residual module and processing a series of ablation experiments, the modified residual networks [4] put forward, rendering the model training more effective and the ability of generalization stronger. The primary components of ResNet-V2 in our paper are shown in Fig.2.

2.2 The Big Transfer Model

The idea of Big Transfer (BiT) [8] offers a generic method consisting of minimal tactics yet acquiring satisfactory results on more tasks. There are two steps in the construction of BiT: Upstream and Downstream. To be specific, Upstream is the process of pre-training while Downstream refers to the process of fine-tuning. The scale of the Upstream in pre-training model is reflected in the size of each training dataset, instead of the size of the model. According to the size of each dataset, three pre-training models (BiT-S, BiT-M, and BiT-L) corresponding to 5 different architectures (ResNet-50, ResNet-101, ResNet-50x3, ResNet-101x3, and ResNet-152x4) are designed, which matches the datasets of ILSVRC-2012 (1.3m), ImageNet-21k (14m), and JFT (300m) respectively. After training the Upstream network, what we need to do is to transfer to Downstream tasks. Kolesnikov A. et al. [8] use a heuristic method called BiT-HyperRule to select and adjust several important hyperparameters - training schedule length, data resolution as well as whether to use MixUp regularization. Since this method is

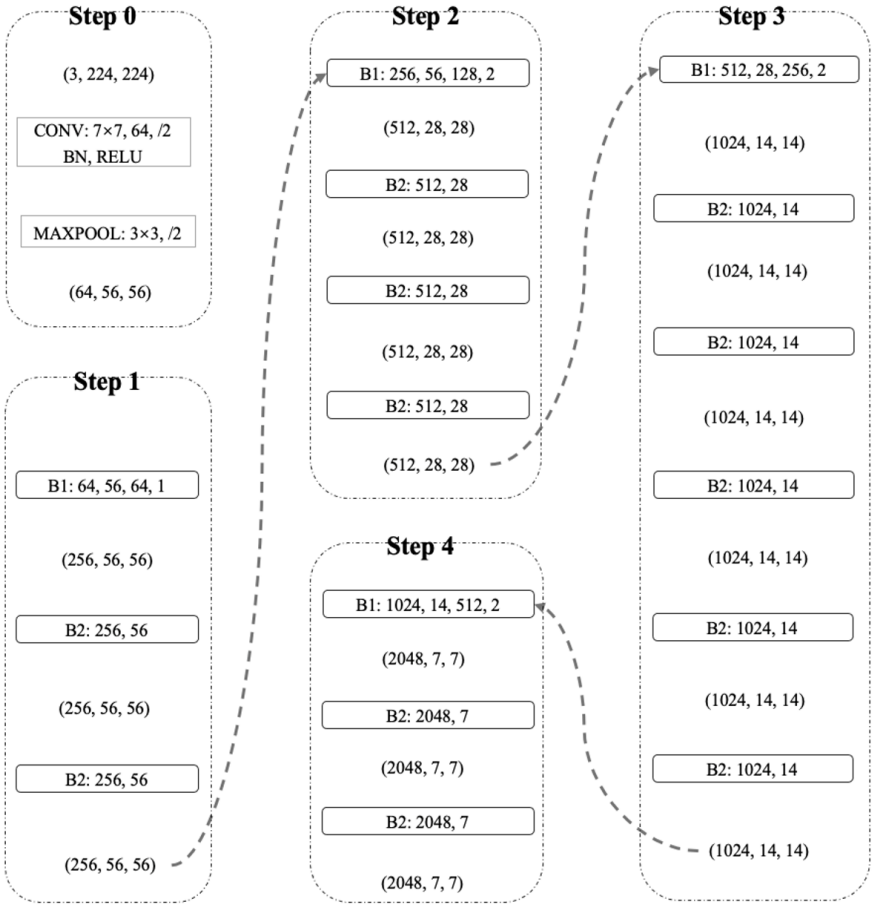

Fig. 3. The ResNet-50 Model

generic, each BiT model only requires one-time pre-training, hence the following tasks of fine-tuning to downstream becoming easier. It should be noticed that BiT in this study wins over the traditional ILSVRC-2012 pre-training via training on the public ImageNet-21k.

2.3 Data Augmentation - MixUp

To multiply the amount of data, we resort to the technique of data augmentation, assuming that $batch_{x1}$ is one of batch sample while $batch_{y1}$ is the label corresponding with it. In this vein, $batch_{x2}$ is the other batch sample while $batch_{y1}$ is the label corresponded with it. Moreover, $\lambda \in [0, 1]$ and the hyperparameter

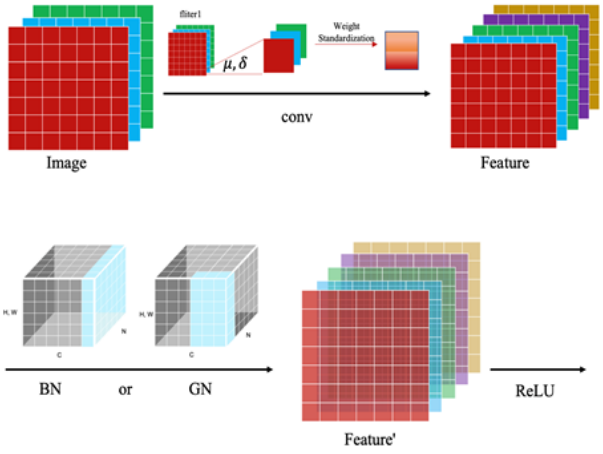

Fig. 4. Weight Standardization: normalize convolution kernels; Normalization Approach: the left part in light blue represents Batch Norm, and the right part in light blue indicates Group Norm.

α determines degree of interpolation. With the relevant variables clarified, the algorithm is presented below,

$$\lambda \sim Beta(\alpha, \alpha), \ for \ \alpha \in (0, \infty)$$

$$mixed_{batch_x} = \lambda \times batch_{x1} + (1 - \lambda) \times batch_{x2}$$
$$mixed_{batch_y} = \lambda \times batch_{y1} + (1 - \lambda) \times batch_{y2}$$

2.4 Group Normalization with Weight Standardization

On the basis of unfixed dimension of batch, batch normalization may vary in effect. For instance, it differs in training and test. Typically, we will move average in the training set to pre-calculate the mean and variance. In the test, these values will not be calculated thereafter, put in use directly. Nevertheless, when the distribution of training and test vary, the pre-calculated parameters on the training set cannot represent the testing data, hence resulting in inconsistency. Through separating the channels into different groups, group normalization computes the variance and mean to standardize the data in every group. The accuracy of computation by group normalization is steady within a considerable batch scales owing to its immunization of batch sizes.

The formulas shown below belong to the conventional normalization operation. Above all, it sets $i = (i_N, i_C, i_H, i_W)$, so x_i refers to a point at a specified position in the feature map.

$$\mu_i = \frac{1}{m} \sum_{k \in S_i} x_k \tag{1}$$

$$\sigma_i = \sqrt{\frac{1}{m}\sum_{k\in S_i}(x_k - \mu_i)^2 + \epsilon} \tag{2}$$

$$\widehat{x_l} = \frac{1}{\sigma_i}(x_i - \mu_i) \tag{3}$$

$$y_i = \gamma\hat{x}_i + \beta_i \tag{4}$$

It can be observed that the distinction between BN and GN lies in the range of S_i in formula 1 and 2. To be specific, S_i in BN is $S_i = \{k \mid k_c = i_c\}$, S_i in GN is $S_i = \{k \mid k_N = i_N, [\frac{k_C}{C/G}] = [\frac{i_C}{C/G}]\}$, in which $k_N = i_N$ makes the calculation on the same graphs' feature map. As G means that C is divided into G groups, G serves as a hyperparameter which is set to 32 by default. Therefore, the idea of GN is to normalize in the same group of the same feature map. Now that the group is only divided in the channel dimension, the operation of normalization is independent of the batch size.

In the three-dimensional cube in Fig. 4, façade C and N represent channel and batch size respectively while the third dimension represents H and W. The dimension size is H * W, which can be elongated into one dimension, so that the whole can be represented by three-dimensional graphics. It can be seen that the calculation of BN is related to the batch size (the light blue area is the part for calculating the mean and variance) while the calculation of GN is not related to the batch size. To avoid the dependence on batch size and realize the mini-batch size, a method called Weight Standardization was proposed directly deal with the weight of convolution kernel. In this sense, there lies pertinent flexibility that the developers can optimize the performance on mini-batch size training with a combination of GN and WS.

3 EXPERIMENTS AND RESULTS

In section 3, we provide an outline of our experiment regarding two upstream models taking backbone as ResNet 50-V2 pre-training on ImageNet21K and ILSVRC-2012. Subsequently, we evaluate their performance on downstream task by referring to CIFAR-10 and CIFAR-100.

3.1 Data Description

The labeled subsets CIFAR-10 and CIFAR-100 are collected by Vinod Nair. et al., extracted from tiny images dataset of 80M [9]. Basically, the CIFAR-10 dataset contains 60000 color images in resolution of 32x32 in 10 classes, with each class including 6000 images. Akin to the CIFAR-10, this dataset possesses 100 classes, each with 600 images (500 training images and 100 testing images

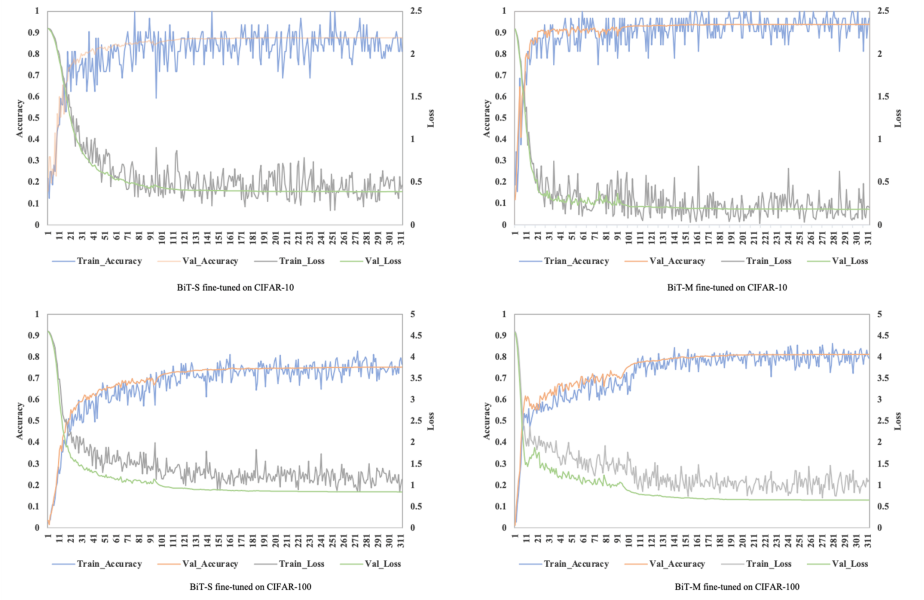

Fig. 5. Results with Bit-S and Bit-M

per class). With 100 classes in the CIFAR-100 integrated into 20 superclasses, each of its image is labeled "fine" (certain class to which it should be categorized) and "coarse" (certain superclass to which it should be categorized).

3.2 Details about Hyperparameters in Upstream and Downstream

Resembling separate units of nervous system, upstream part in deep learning is engaged in pre-training while downstream is used for fine-tuning to a specific task. With regard to the upstream section where this study draws upon the warm-up method to render the learning rate climb to $0.03 \times \frac{batchsize}{256}$ and the weight decay of the optimizer be 0.0001, we seek reference from procedures taken by Kolesnikov, A.et al. [8] who drew upon unified hyperparameters: SGD with momentum of 0.9, initial learning rate of 0.03, batch size of 4096, and the size of input data being 224×224. With 90 epochs trained by BiT-S and BiT-M, the learning rate was divided by 10 in the 30th, 60th, and 80th generations.

After training the upstream part, we need to fine-tune it into the downstream task. To perform pertinent functions Kolesnikov, A.et al. proposed a heuristic approach named BiT-Hyperrule. The SGD with momentum was set to 0.9, learning rate to 0.003, and batch size to 512. It entails selecting and adjusting

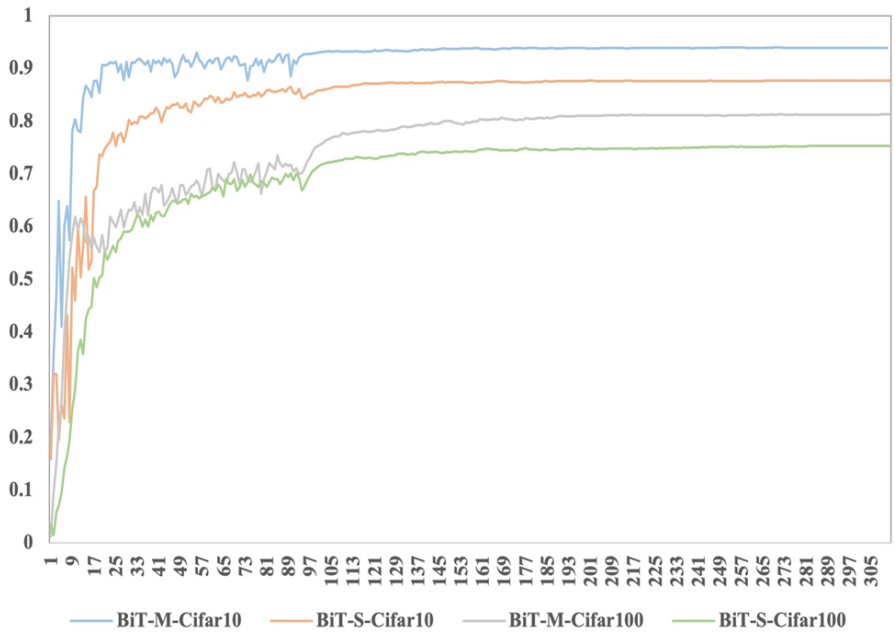

Fig. 6. Improvement in Test Accuracy

several critical hyperparameters containing schedule length in which the small tasks (less than 20k) were trained with 500 steps while the medium tasks (ranging from 20K to 500K) with 10K steps, images' resolution as well as necessity of using MixUp data augmentation in which the α is set to 0.1. In particular, the procedures of images' resolution are akin to the standard process of handling data, that is, resizing, cropping, and flipping. To be specific, the target image can be rectified into a square, randomly cropped out of a smaller one, and finally flipped in horizontal at random during the training period, which differs from the fixed size reshaped during the testing period.

3.3 Results

BiT-M represents model pre-trained on ImageNet-21k and BiT-S means model pre-trained on ILSVRC-2012. The dataset pretrained on BiT-M is 10 times bigger than BiT-S. In the fine-tune task, we obtain the accuracy of 93.9% and 87.7% on CIFAR-10 and the accuracy of 81.3% and 75.3% on CIFAR-100 after running fine-tuning of the model BiT-M and BiT-S. The details are presented in Fig. 6. As seen from this figure, the performance of the model is logarithmically refined along with the rise in the number of pre-training data.

It can be seen from Fig. 5 that the visual representations of BiT-M have been

significantly refined with a training on ImageNet-21k, satisfactory accuracy can be obtained with large-scale pre-training models even when the sample data per category is small. However, in our practice, we show that the accuracy is related to the class of the dataset, and when the class is very different from the dataset of the pretrained model and the samples are small, the training with the pre-trained model is not only low in accuracy but also time-consuming.

3.4 Compared with the EfficientNet Model

To validate the learning ability in the Big Transfer Model, we compare those experimental results mentioned above with the EfficientNet [15] model in sub-section 4. Typically, if the model is designed to be too wide [20], deep [3], or with high resolution [5], it will soon be saturated, and the efficiency will be deteriorated despite being useful at the beginning. To address such an issue, we employ the EfficientNet where these features are scaled in a more organized way. The NAS (neural network search) is used to find a better backbone, that is, EfficientNet-b0. By scaling the width, depth, and resolution of the B0 model at the same time, they gain the EfficientNet b1-b7.

As we can see from Fig.7, the test accuracy is about 95% on CIFAR-10, which is very similar to the BiT-M fine-tuned on CIFAR-10, but it converges faster in 50 epochs. Thus, it testifies that the BiT model has better transfer learning ability and the EfficientNet model has stronger convergence ability.

4 DISCUSSIONS AND FUTURE WORK

The smaller datasets of computer vision like ILSVRC-2012 which contains images of 1.28M tend to produce proper outcome when following traditional training steps. Due to the advancement in computing techs, such standard procedures enable datasets of larger size such as ImageNet-21k which includes images of 14.2M to be processed. Nonetheless, it is deficient of existing mechanisms for training datasets of larger scale. To address this issue, on the one hand, we scrutinized Big Transfer model with architecture ResNet-50 V2 on various datasets, and on the other hand, we propose a few fundamental principles for handling larger datasets so as to increase the image's classification accuracy. The results acquired in this research verify the weight of dataset scale concerning visual representations. In this sense, it will provide a deep insight for those who have access to either large or small computational resources.

As a novel discipline, there are still quite an amount of work in both theoretical and practical aspects of deep learning [22]. With the application of deep neural networks to computer vision, it is likely to produce results that rival against or even surpass performance of current state-of-the-art methods [16]. Of course,

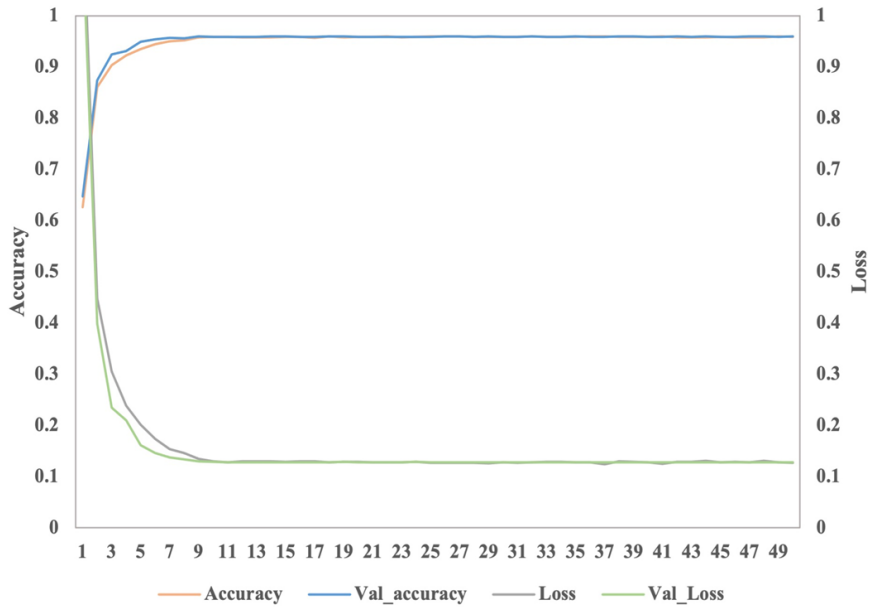

Fig. 7. EfficientNet on CIFAR-100

there still lie some limitations concerning this technology. For instance, problematic behaviors can occur in deep learning architectures , including sorting blurring images into a designated category of typical images and misclassifying correct images out of minor changes, which can be ascribed to the limits of internal representations and misreading of image semantics.

Thus, instead of seeking for all-encompassing solutions in the future work, we ought to conduct a thorough study of the reproducibility of cutting-edge outcomes and of correlations between the scale of training dataset as well as the influence of diverse neural models in terms of generalizability.

References

1. J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.
2. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009.
3. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.
4. K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks. In *European conference on computer vision*, pages 630–645. Springer, 2016.

5. Y. He, J. Lin, Z. Liu, H. Wang, L.-J. Li, and S. Han. Amc: Automl for model compression and acceleration on mobile devices. In *Proceedings of the European conference on computer vision (ECCV)*, pages 784–800, 2018.
6. X. Huang and S. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1501–1510, 2017.
7. S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International conference on machine learning*, pages 448–456. PMLR, 2015.
8. A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, and N. Houlsby. Big transfer (bit): General visual representation learning. In *Computer Vision–ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part V 16*, pages 491–507. Springer, 2020.
9. A. Krizhevsky, G. Hinton, et al. Learning multiple layers of features from tiny images. 2009.
10. S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. doi: 10.1109/TKDE.2009.191.
11. M. Raghu, C. Zhang, J. Kleinberg, and S. Bengio. Transfusion: Understanding transfer learning for medical imaging. *arXiv preprint arXiv:1902.07208*, 2019.
12. J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.
13. S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28:91–99, 2015.
14. D. Soekhoe, P. Van Der Putten, and A. Plaat. On the impact of data set size in transfer learning using deep neural networks. In *International symposium on intelligent data analysis*, pages 50–60. Springer, 2016.
15. M. Tan and Q. Le. Efficientnet: Rethinking model scaling for convolutional neural networks. In *International Conference on Machine Learning*, pages 6105–6114. PMLR, 2019.
16. A. Voulodimos, N. Doulamis, A. Doulamis, and E. Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.
17. K. Weiss, T. M. Khoshgoftaar, and D. Wang. A survey of transfer learning. *Journal of Big data*, 3(1):1–40, 2016.
18. Y. Wu and K. He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018.
19. J. Yosinski, J. Clune, Y. Bengio, and H. Lipson. How transferable are features in deep neural networks? *arXiv preprint arXiv:1411.1792*, 2014.
20. S. Zagoruyko and N. Komodakis. Wide residual networks. *arXiv preprint arXiv:1605.07146*, 2016.
21. H. Zhang, M. Cisse, Y. N. Dauphin, and D. Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.
22. Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu. Object detection with deep learning: A review. *IEEE transactions on neural networks and learning systems*, 30(11):3212–3232, 2019.