Distribution-Based Invariant Deep Networks for Learning Meta-Features

Gwendoline de Bie, Herilalaina Rakotoarison, Gabriel Peyré, Michèle Sebag

Distribution-Based Invariant Deep Networks for Learning Meta-Features

Gwendoline De Bie
TAU - LRI and ENS,
PSL University
debie@dma.ens.fr

Herilalaina Rakotoarison
TAU - LRI, INRIA
heri@lri.fr

Gabriel Peyré
CNRS and ENS,
PSL University
gabriel.peyre@ens.fr

Michele Sebag
CNRS, Paris-Saclay University
sebag@lri.fr

October 20, 2020

Abstract

Recent advances in deep learning from probability distributions successfully achieve classification or regression from distribution samples, thus invariant under permutation of the samples. The first contribution of the paper is to extend these neural architectures to achieve invariance under permutation of the features, too. The proposed architecture, called Dida, inherits the NN properties of universal approximation, and its robustness w.r.t. Lipschitz-bounded transformations of the input distribution is established. The second contribution is to empirically and comparatively demonstrate the merits of the approach on two tasks defined at the dataset level. On both tasks, Dida learns meta-features supporting the characterization of a (labelled) dataset. The first task consists of predicting whether two dataset patches are extracted from the same initial dataset. The second task consists of predicting whether the learning performance achieved by a hyper-parameter configuration under a fixed algorithm (ranging in k-NN, SVM, logistic regression and linear SGD) dominates that of another configuration, for a dataset extracted from the OpenML benchmarking suite. On both tasks, Dida outperforms the state of the art: DSS [23] and Dataset2Vec [16] architectures, as well as the models based on the hand-crafted meta-features of the literature.

1 Introduction

Deep networks architectures, initially devised for structured data such as images [19] and speech [14], have been extended to enforce some invariance or

equivariance properties [35] for more complex data representations. Typically, the network output is required to be invariant with respect to permutations of the input points when dealing with point clouds [29], graphs [13] or probability distributions [7]. The merit of invariant or equivariant neural architectures is twofold. On the one hand, they inherit the universal approximation properties of neural nets [6, 20]. On the other hand, the fact that these architectures comply with the requirements attached to the data representation yields more robust and more general models, through constraining the neural weights and/or reducing their number.

Related works. Invariance or equivariance properties are relevant to a wide range of applications. In the sequence-to-sequence framework, one might want to relax the sequence order [37]. When modelling dynamic cell processes, one might want to follow the cell evolution at a macroscopic level, in terms of distributions as opposed to, a set of individual cell trajectories [12]. In computer vision, one might want to handle a set of pixels, as opposed to a voxellized representation, for the sake of a better scalability in terms of data dimensionality and computational resources [7].

Neural architectures enforcing invariance or equivariance properties have been pioneered by [29, 41] for learning from point clouds subject to permutation invariance or equivariance. These have been extended to permutation equivariance across sets [11]. Characterizations of invariance or equivariance under group actions have been proposed in the finite [10, 4, 31] or infinite case [39, 18].

On the theoretical side, [21, 17] have proposed a general characterization of linear layers enforcing invariance or equivariance properties with respect to the whole permutation group on the feature set. The universal approximation properties of such architectures have been established in the case of sets [41], point clouds [29], equivariant point clouds [34], discrete measures [7], invariant [22] and equivariant [17] graph neural networks. The approach most related to our work is that of [23], handling point clouds and presenting a neural architecture invariant w.r.t. the ordering of points and their features. In this paper, the proposed *distribution-based invariant deep architecture* (Dida) extends [23] as it handles (discrete or continuous) probability distributions instead of point clouds. This enables to leverage the topology of the Wasserstein distance to provide more general approximation results, covering [23] as a special case.

Motivations. A main motivation for Dida is the ability to characterize datasets through *learned meta-features*. Meta-features, aimed to represent a dataset as a vector of characteristcs, have been mentioned in the ML literature for over 40 years, in relation with several key ML challenges: (i) learning a performance model, predicting *a priori* the performance of an algorithm (and the hyper-parameters thereof) on a dataset [32, 38, 15]; (ii) learning a generic model able of quick adaptation to new tasks, e.g. one-shot or few-shot, through the so-called meta-learning approach [9, 40]; (iii) hyper-parameter transfer learning [26], aimed to transfer the performance model learned for a task, to another task.

A large number of meta-features have been manually designed along the years [24], ranging from sufficient statistics to the so-called *landmarks* [28], computing the performance of fast ML algorithms on the considered dataset. Meta-features, expected to describe the joint distribution underlying the dataset, should also be inexpensive to compute. The learning of meta-features has been first tackled by [16] to our best knowledge, defining the DATASET2VEC representation. Specifically, DATASET2VEC is provided two patches of datasets, (two subsets of examples, described by two (different) sets of features), and is trained to predict whether those patches are extracted from the same initial dataset.

Contributions. The proposed DIDA approach extends the state of the art [23, 16] in two ways. Firstly, it is designed to handle discrete or continuous probability distributions, as opposed to point sets (Section 2). As said, this extension enables to leverage the more general topology of the Wasserstein distance as opposed to that of the Haussdorf distance (Section 3). This framework is used to derive theoretical guarantees of stability under bounded distribution transformations, as well as universal approximation results, extending [23] to the continuous setting. Secondly, the empirical validation of the approach on two tasks defined at the dataset level demonstrates the merit of the approach compared to the state of the art [23, 16, 25] (Section 4).

Notations. $[m]$ denotes the set of integers $\{1, \ldots m\}$. Distributions, including discrete distributions (datasets) are noted in bold font. Vectors are noted in italic, with $x[k]$ denoting the k-th coordinate of vector x.

2 Distribution-Based Invariant Networks for Meta-Feature Learning

This section describes the core of the proposed distribution-based invariant neural architectures, specifically the mechanism of mapping a point distribution onto another one subject to sample and feature invariance, referred to as *invariant layer*. For the sake of readability, this section focuses on the case of discrete distributions, referring the reader to Appendix A for the general case of continuous distributions.

2.1 Invariant Functions of Discrete Distributions

Let $\mathbf{z}= \{(x_i, y_i) \in \mathbb{R}^d, i \in [n]\}$ denote a dataset including n labelled samples, with $x_i \in \mathbb{R}^{d_X}$ an instance and $y_i \in \mathbb{R}^{d_Y}$ the associated multi-label. With d_X and d_Y respectively the dimensions of the instance and label spaces, let $d \stackrel{\text{def.}}{=} d_X + d_Y$. By construction, $\mathbf{z}$ is invariant under permutation on the sample ordering; it is viewed as an n-size discrete distribution $\frac{1}{n}\sum_{i=1}^n \delta_{z_i}$ in $\mathbb{R}^d$ with δ_{z_i} the Dirac function at z_i. In the following, $\mathrm{Z}_n(\mathbb{R}^d)$ denotes the space of such n-size point

distributions, with $\mathrm{Z}(\mathbb{R}^d) \stackrel{\text{def.}}{=} \cup_n \mathrm{Z}_n(\mathbb{R}^d)$ the space of distributions of arbitrary size.

Let $G \stackrel{\text{def.}}{=} S_{d_X} \times S_{d_Y}$ denote the group of permutations independently operating on the feature and label spaces. For $\sigma = (\sigma_X, \sigma_Y) \in G$, the image $\sigma(z)$ of a labelled sample is defined as $(\sigma_X(x), \sigma_Y(y))$, with $x = (x[k], k \in [d_X])$ and $\sigma_X(x) \stackrel{\text{def.}}{=} (x[\sigma_X^{-1}(k)], k \in [d_X])$. For simplicity and by abuse of notations, the operator mapping a distribution $\mathbf{z} = (z_i, i \in [n])$ to $\{\sigma(z_i)\} \stackrel{\text{def.}}{=} \sigma_\sharp \mathbf{z}$ is still denoted σ.

Let $\mathrm{Z}(\Omega)$ denote the space of distributions supported on some domain $\Omega \subset \mathbb{R}^d$, with Ω invariant under permutations in G. The goal of the paper is to define and train deep architectures, implementing functions φ on $\mathrm{Z}(\Omega \subset \mathbb{R}^d)$ that are invariant under G, i.e. such that $\forall \sigma \in G, \varphi(\sigma_\sharp \mathbf{z}) = \varphi(\mathbf{z})$[1]. By construction, a multi-label dataset is invariant under permutations of the samples, of the features, and of the multi-labels. Therefore, any meta-feature, that is, a feature describing a multi-label dataset, is required to satisfy the above property.

2.2 Distribution-Based Invariant Layers

The building block of the proposed architecture, the invariant layer meant to satisfy the feature and label invariance requirements, is defined as follows, taking inspiration from [7].

Definition 1. *(Distribution-based invariant layers) Let an interaction functional $\varphi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^r$ be G-invariant:*

$$\forall \sigma \in G, \quad \forall (z_1, z_2) \in \mathbb{R}^d \times \mathbb{R}^d, \quad \varphi(z_1, z_2) = \varphi(\sigma(z_1), \sigma(z_2)).$$

The distribution-based invariant layer f_φ is defined as

$$f_\varphi : \mathbf{z} = (z_i)_{i \in [n]} \in Z(\mathbb{R}^d) \mapsto f_\varphi(\mathbf{z}) \stackrel{\text{def.}}{=} \left[\frac{1}{n} \sum_{j=1}^n \varphi(z_1, z_j), \ldots, \frac{1}{n} \sum_{j=1}^n \varphi(z_n, z_j) \right] \in Z(\mathbb{R}^r). \tag{1}$$

It is easy to see that f_φ is G-invariant. The construction of f_φ is extended to the general case of possibly continuous probability distributions by essentially replacing sums by integrals (Appendix A).

Remark 1. (Varying sample size n). By construction, f_φ is defined on $\mathrm{Z}(\mathbb{R}^d) = \cup_n \mathrm{Z}_n(\mathbb{R}^d)$ (independent of n), such that it supports inputs of arbitrary cardinality n.

Remark 2. (Discussion w.r.t. [23]) The above definition of f_φ is based on the aggregation of pairwise terms $\varphi(z_i, z_j)$. The motivation for using a pairwise φ is twofold. On the one hand, capturing local sample interactions allows to create more expressive architectures, which is important to improve the performance on some complex data sets, as illustrated in the experiments (Section 4). On the

[1] As opposed to *G-equivariant* functions that are characterized by $\forall \sigma \in G, \varphi(\sigma_\sharp \mathbf{z}) = \sigma_\sharp \varphi(\mathbf{z})$

other hand, interaction functionals are crucial to design universal architectures (Appendix C, theorem 2). The proposed theoretical framework relies on the Wasserstein distance (corresponding to the convergence in law of probability distributions), which enables to compare distributions with varying number of points or even with continuous densities. In contrast, [23] do not use interaction functionals, and establish the universality of their DSS architecture for fixed dimension d and number of points n. Moreover, DSS happens to resort to max pooling operators, discontinuous w.r.t. the Wasserstein topology (see Remark 5).

Two particular cases are when φ only depends on its first or second input:

(i) if $\varphi(z, z') = \psi(z')$, then f_φ computes a global "moment" descriptor of the input, as $f_\varphi(\mathbf{z}) = \frac{1}{n}\sum_{j=1}^n \psi(z_j) \in \mathbb{R}^r$.

(ii) if $\varphi(z, z') = \xi(z)$, then f_φ transports the input distribution via ξ, as $f_\varphi(\mathbf{z}) = \{\xi(z_i), i \in [n]\} \in Z(\mathbb{R}^r)$. This operation is referred to as a *push-forward*.

Remark 3. (Varying dimensions d_X and d_Y). Both in practice and in theory, it is important that f_φ layers (in particular the first layer of the neural architecture) handle datasets of arbitrary number of features d_X and number of multi-labels d_Y. The proposed approach, used in the experiments (Section 4), is to define φ on the top of a four-dimensional aggregator, as follows. Letting $z = (x, y)$ and $z' = (x', y')$ be two samples in $\mathbb{R}^{d_X} \times \mathbb{R}^{d_Y}$, let u be defined from $\mathbb{R}^4$ onto $\mathbb{R}^t$, consider the sum of $u(x[k], x'[k], y[\ell], y'[\ell])$ for k ranging in $[d_X]$ and ℓ in $[d_Y]$, and apply mapping v from $\mathbb{R}^t$ to $\mathbb{R}^r$ on the sum:

$$\varphi(z, z') = v\left(\sum_{k=1}^{d_X}\sum_{\ell=1}^{d_Y} u(x[k], x'[k], y[\ell], y'[\ell])\right)$$

Remark 4. (Localized computation) In practice, the quadratic complexity of f_φ w.r.t. the number n of samples can be reduced by only computing $\varphi(z_i, z_j)$ for pairs z_i, z_j sufficiently close to each other. Layer f_φ thus extracts and aggregates information related to the neighborhood of the samples.

2.3 Learning Meta-features

The proposed distributional neural architectures defined on point distributions (DIDA) are sought as

$$\mathbf{z} \in \mathrm{Z}(\mathbb{R}^d) \mapsto \mathcal{F}_\zeta(\mathbf{z}) \stackrel{\text{def.}}{=} f_{\varphi_m} \circ f_{\varphi_{m-1}} \circ \ldots \circ f_{\varphi_1}(\mathbf{z}) \in \mathbb{R}^{d_{m+1}} \tag{2}$$

where ζ are the trainable parameters of the architecture (below). Only the case $d_Y = 1$ is considered in the remainder. The k-th layer is built on the top of φ_k, mapping pairs of vectors in $\mathbb{R}^{d_k}$ onto $\mathbb{R}^{d_{k+1}}$, with $d_1 = d$ (the dimension of the input samples). Last layer is built on φ_m, only depending on its second argument;

it maps the distribution in layer $m-1$ onto a vector, whose coordinates are referred to as meta-features.

The G-invariance and dimension-agnosticity of the whole architecture only depend on the first layer f_{φ_1} satisfying these properties. In the first layer, φ_1 is sought as $\varphi_1((x,y),(x',y')) = v(\sum_k u(x[k],x'[k],y,y'))$ (Remark 3), with $u(x[k],x'[k],y,y') = (\rho(A_u \cdot (x[k];x'[k]) + b_u, \mathbb{1}_{y \neq y'})$ in $\mathbb{R}^t \times \{0,1\}$, where ρ is a non-linear activation function, A_u a $(2,t)$ matrix, $(x[k];x'[k])$ the 2-dimensional vector concatenating $x[k]$ and $x'[k]$, and b_u a t-dimensional vector. With $e = \sum_k u(x[k],x'[k],y,y'))$, function v likewise applies a non-linear activation function ρ on an affine transformation of e: $v(e) = \rho(A_v \cdot e + b_v)$, with A_v a (t,r) matrix and b_v a r-dimensional vector.

Note that the subsequent layers need neither be invariant w.r.t. the number of samples, nor handle a varying number of dimensions. Every $\varphi_k, k \geq 2$ is defined as $\varphi_k = \rho(A_k \cdot + b_k)$, with ρ an activation function, A_k a (d_k, d_{k+1}) matrix and b_k a d_{k+1}-dimensional vector. The DIDA neural net thus is parameterized by $\zeta \stackrel{\text{def.}}{=} (A_u, b_u, A_v, b_v, \{A_k, b_k\}_k)$, that is classically learned by stochastic gradient descent from the loss function defined after the task at hand (Section 4).

3 Theoretical Analysis

This section analyzes the properties of invariant-layer based neural architectures, specifically their robustness w.r.t. bounded transformations of the involved distributions, and their approximation abilities w.r.t. the convergence in law, which is the natural topology for distributions. As already said, the discrete distribution case is considered in this section for the sake of readability, referring the reader to Appendix A for the general case of continuous distributions.

3.1 Optimal Transport Comparison of Datasets

Point clouds vs. distributions. Our claim is that datasets should be seen as probability distributions, rather than point clouds. Typically, including many copies of a point in a dataset amounts to increasing its importance, which usually makes a difference in a standard machine learning setting. Accordingly, the topological framework used to define and learn meta-features in the following is that of the convergence in law, with the distance among two datasets being quantified using the Wasserstein distance (below). In contrast, the point clouds setting (see for instance [29]) relies on the Haussdorff distance among sets to theoretically assess the robustness of these architectures. While it is standard for 2D and 3D data involved in graphics and vision domains, it faces some limitations in higher dimensional domains, e.g. due to max-pooling being a non-continuous operator w.r.t. the convergence in law topology.

Wasserstein distance. Referring the reader to [33, 27] for a more comprehensive presentation, the standard 1-Wasserstein distance between two discrete

probability distributions $\mathbf{z}, \mathbf{z}' \in \mathrm{Z}_n(\mathbb{R}^d) \times \mathrm{Z}_m(\mathbb{R}^d)$ is defined as:

$$\mathrm{W}_1(\mathbf{z}, \mathbf{z}') \overset{\text{def.}}{=} \max_{f \in \mathrm{Lip}_1(\mathbb{R}^d)} \frac{1}{n} \sum_{i=1}^{n} f(z_i) - \frac{1}{m} \sum_{j=1}^{m} f(z'_j)$$

with $\mathrm{Lip}_1(\mathbb{R}^d)$ the space of 1-Lipschitz functions $f : \mathbb{R}^d \to \mathbb{R}$. To account for the invariance requirement (making indistinguishable $\mathbf{z} = (z_1, \ldots, z_n)$ and its permuted image $(\sigma(z_1), \ldots, \sigma(z_n)) \overset{\text{def.}}{=} \sigma_\sharp \mathbf{z}$ under $\sigma \in G$), we introduce the G-invariant 1-Wasserstein distance: for $\mathbf{z} \in \mathrm{Z}_n(\mathbb{R}^d), \mathbf{z}' \in \mathrm{Z}_m(\mathbb{R}^d)$:

$$\overline{\mathrm{W}}_1(\mathbf{z}, \mathbf{z}') = \min_{\sigma \in G} \mathrm{W}_1(\sigma_\sharp \mathbf{z}, \mathbf{z}')$$

such that $\overline{\mathrm{W}}_1(\mathbf{z}, \mathbf{z}') = 0$ if and only if $\mathbf{z}$ and $\mathbf{z}'$ belong to the same equivalence class (Appendix A), i.e. are equal in the sense of probability distributions up to sample and feature permutations.

Lipschitz property. In this context, a map f from $\mathrm{Z}(\mathbb{R}^d)$ onto $\mathrm{Z}(\mathbb{R}^r)$ is continuous for the convergence in law (a.k.a. weak convergence on distributions, denoted $\rightharpoonup$) iff for any sequence $\mathbf{z}^{(k)} \rightharpoonup \mathbf{z}$, then $f(\mathbf{z}^{(k)}) \rightharpoonup f(\mathbf{z})$. The Wasserstein distance metrizes the convergence in law, in the sense that $\mathbf{z}^{(k)} \rightharpoonup \mathbf{z}$ is equivalent to $\mathrm{W}_1(\mathbf{z}^{(k)}, \mathbf{z}) \to 0$. Furthermore, map f is said to be C-Lipschitz for the permutation invariant 1-Wasserstein distance iff

$$\forall \mathbf{z}, \mathbf{z}' \in \mathrm{Z}(\mathbb{R}^d), \quad \overline{\mathrm{W}}_1(f(\mathbf{z}), f(\mathbf{z}')) \leqslant C \overline{\mathrm{W}}_1(\mathbf{z}, \mathbf{z}'). \tag{3}$$

The C-Lipschitz property entails the continuity of f w.r.t. its input: if two input distributions are close in the permutation invariant 1-Wasserstein sense, the corresponding outputs are close too.

3.2 Regularity of Distribution-Based Invariant Layers

Assuming the interaction functional to satisfy the Lipschitz property:

$$\forall z \in \mathbb{R}^d, \quad \varphi(z, \cdot) \quad \text{and} \quad \varphi(\cdot, z) \quad \text{are} \quad C_\varphi - \text{Lipschitz}. \tag{4}$$

the robustness of invariant layers with respect to different variations of their input is established (proofs in Appendix B). We first show that invariant layers also satisfy Lipschitz property, ensuring that deep architectures of the form (2) map close inputs onto close outputs.

Proposition 1. *Invariant layer f_φ of type (1) is $(2rC_\varphi)$-Lipschitz in the sense of (3).*

A second result regards the case where two datasets $\mathbf{z}$ and $\mathbf{z}'$ are such that $\mathbf{z}'$ is the image of $\mathbf{z}$ through some diffeomorphism τ ($\mathbf{z} = (z_1, \ldots, z_n)$ and $\mathbf{z}' = \tau_\sharp \mathbf{z} = (\tau(z_1), \ldots, \tau(z_n))$. If τ is close to identity, then the following proposition shows that $f_\varphi(\tau_\sharp \mathbf{z})$ and $f_\varphi(\mathbf{z})$ are close too. More generally, if continuous transformations τ and ξ respectively apply on the input and output space of f_φ, and are close to identity, then $\xi_\sharp f_\varphi(\tau_\sharp \mathbf{z})$ and $f_\varphi(\mathbf{z})$ are also close.

Proposition 2. *Let $\tau : \mathbb{R}^d \to \mathbb{R}^d$ and $\xi : \mathbb{R}^r \to \mathbb{R}^r$ be two Lipschitz maps with respectively Lipschitz constants C_τ and C_ξ. Then,*

$$\forall \mathbf{z} \in Z(\Omega),\ \overline{\mathrm{W}}_1(\xi_\sharp f_\varphi(\tau_\sharp \mathbf{z}), f_\varphi(\mathbf{z})) \leqslant \sup_{x \in f_\varphi(\tau(\Omega))} \|\xi(x) - x\|_2 + 2r\,\mathrm{Lip}(\varphi) \sup_{x \in \Omega} \|\tau(x) - x\|_2$$

$$\forall \mathbf{z}, \mathbf{z}' \in Z(\Omega),\ \textit{if } \tau \textit{ is equivariant, } \overline{\mathrm{W}}_1(\xi_\sharp f_\varphi(\tau_\sharp \mathbf{z}), \xi_\sharp f_\varphi(\tau_\sharp \mathbf{z}')) \leqslant 2r\, C_\varphi\, C_\tau\, C_\xi \overline{\mathrm{W}}_1(\mathbf{z}, \mathbf{z}')$$

3.3 Universality of Invariant Layers

Lastly, the universality of the proposed architecture is established, showing that the composition of an invariant layer (1) and a fully-connected layer is enough to enjoy the universal approximation property, over all functions defined on $Z(\mathbb{R}^d)$ with dimension d less than some D (Remark 3).

Theorem 1. *Let $\mathcal{F} : Z(\Omega) \to \mathbb{R}$ be a G-invariant map on a compact Ω, continuous for the convergence in law. Then $\forall \varepsilon > 0$, there exists two continuous maps ψ, φ such that*

$$\forall \mathbf{z} \in Z(\Omega), \quad |\mathcal{F}(\mathbf{z}) - \psi \circ f_\varphi(\mathbf{z})| < \varepsilon$$

where φ is G-invariant and independent of $\mathcal{F}$.

Proof. The sketch of the proof is as follows (complete proof in Appendix C). Let us define $\varphi = g \circ h$ where: (i) h is the collection of d_X elementary symmetric polynomials in the features and d_Y elementary symmetric polynomials in the labels, which is invariant under G; (ii) a discretization of $h(\Omega)$ on a grid is then considered, achieved thanks to g that aims at collecting integrals over each cell of the discretization; (iii) ψ applies function $\mathcal{F}$ on this discretized measure; this requires h to be bijective, and is achieved by $\tilde{h}$, through a projection on the quotient space S_d/G and a restriction to its image compact Ω'. To sum up, f_φ defined as such computes an expectation which collects integrals over each cell of the grid to approximate measure $h_\sharp \mathbf{z}$ by a discrete counterpart $\widehat{h_\sharp \mathbf{z}}$. Hence ψ applies $\mathcal{F}$ to $\tilde{h}_\sharp^{-1}(\widehat{h_\sharp \mathbf{z}})$. Continuity is obtained as follows: (i) proximity of $h_\sharp \mathbf{z}$ and $\widehat{h_\sharp \mathbf{z}}$ follows from Lemma 1 in [7]) and gets tighter as the grid discretization step tends to 0; (ii) Map $\tilde{h}^{-1}$ is $1/d$-Hölder, after Theorem 1.3.1 from [30]); therefore Lemma 2 entails that $\overline{\mathrm{W}}_1(\mathbf{z}, \tilde{h}_\sharp^{-1}\widehat{h_\sharp \mathbf{z}})$ can be upper-bounded; (iii) since Ω is compact, by Banach-Alaoglu theorem, $\mathrm{Z}(\Omega)$ also is. Since $\mathcal{F}$ is continuous, it is thus uniformly weakly continuous: choosing a discretization step small enough ensures the result. □

Remark 5. (Comparison with [23]) The above proof holds for functionals of arbitrary input sample size n, as well as continuous distributions, generalizing results in [23]. Note that the two types of architectures radically differ (more in Section 4).

Remark 6. (Approximation by an invariant NN) After theorem 1, any invariant continuous function defined on distributions with compact support can be approximated with arbitrary precision by an invariant neural network (Appendix C). The proof involves mainly three steps: (i) an invariant layer f_φ can be

approximated by an invariant network; (ii) the universal approximation theorem [6, 20]; (iii) uniform continuity is used to obtain uniform bounds.

Remark 7. (Extension to different spaces) Theorem 1 also extends to distributions supported on different spaces, via embedding them into a high-dimensional space. Therefore, any invariant function on distributions with compact support in $\mathbb{R}^d$ with $d \leq D$ can be uniformly approximated by an invariant network (Appendix C).

4 Experimental validation

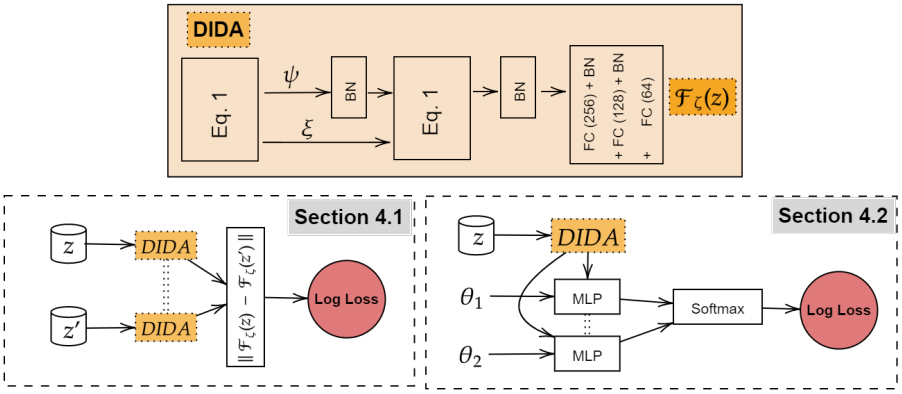

Figure 1: Learning meta-features with DIDA. Top: the DIDA architecture (BN stands for batch norm; FC for fully connected layer). Bottom left: Learning meta-features for patch identification using a Siamese architecture (section 4.1). Bottom right: learning meta-features for performance modelling, specifically to rank two hyper-parameter configurations θ_1 and θ_2 (section 4.2).

The experimental validation presented in this section considers two goals of experiments: (i) assessing the ability of DIDA to learn accurate meta-features; (ii) assessing the merit of the DIDA invariant layer design, building invariant f_φ on the top of an interactional function φ (Eq. 1). As said, this architecture is expected to grasp contrasts among samples, e.g. belonging to different classes; the proposed experimental setting aims to empirically investigate this conjecture. These goals of experiments are tackled by comparing DIDA to three baselines: DSS layers [23]; hand-crafted meta-features (HC) [24] (Table 4 in Appendix D); DATASET2VEC [16]. We implemented DSS (the code being not available) using linear and non-linear invariant layers.[2]. All compared systems are allocated ca the same number of parameters.

[2]The code source of DIDA and (our implementation of) DSS is available in Appendix D.

Experimental setting. Two tasks defined at the dataset level are considered: patch identification (section 4.1) and performance modelling (section 4.2). On both tasks, the same DIDA architecture is considered (Fig 1), involving 2 invariant layers followed by 3 fully connected (FC) layers. Meta-features $\mathcal{F}_\zeta(\mathbf{z})$ consist of the output of the third FC layer, with ζ denoting the trained DIDA parameters. All experiments run on 1 NVIDIA-Tesla-V100-SXM2 GPU with 32GB memory, using Adam optimizer with base learning rate 10^{-3}.

4.1 Task 1: Patch Identification

The patch identification task consists of detecting whether two blocks of data are extracted from the same original dataset [16]. Letting $\mathbf{u}$ denote a n-sample, d-dimensional dataset, a patch $\mathbf{z}$ is constructed from $\mathbf{u}$ by retaining samples with index in $I \subset [n]$ and features with index in $J \subset [d]$. To each pair of patches $\mathbf{z}, \mathbf{z}'$ with same number of instances, is associated a binary meta-label $\ell(\mathbf{z}, \mathbf{z}')$ set to 1 iff $\mathbf{z}$ and $\mathbf{z}$' are extracted from the same initial dataset $\mathbf{u}$. DIDA parameters ζ are trained to minimize the cross-entropy loss of model $\hat{\ell}_\zeta(\mathbf{z}, \mathbf{z}') = \exp(-\|\mathcal{F}_\zeta(\mathbf{z}) - \mathcal{F}_\zeta(\mathbf{z}')\|_2)$, with $\mathcal{F}_\zeta(\mathbf{z})$ and $\mathcal{F}_\zeta(\mathbf{z}')$ the meta-features computed for $\mathbf{z}$ and $\mathbf{z}'$:

$$\text{Minimize} \quad \mathcal{L}(\zeta) = -\sum_{\mathbf{z},\mathbf{z}'} \ell(\mathbf{z}, \mathbf{z}') \log(\hat{\ell}_\zeta(\mathbf{z}, \mathbf{z}')) + (1 - \ell(\mathbf{z}, \mathbf{z}')) \log(1 - \hat{\ell}_\zeta(\mathbf{z}, \mathbf{z}')) \quad (5)$$

The classification results on toy datasets and UCI datasets (Table 1, detailed in Appendix D) show the pertinence of the DIDA meta-features, particularly on the UCI datasets where the number of features widely varies from one dataset to another. The relevance of the interactional invariant layer design is established on this problem as DIDA outperforms both DATASET2VEC and DSS.

Method	TOY	UCI
DATASET2VEC(*)	96.19 %± 0.28	77.58 %± 3.13
DSS layers (Linear aggregation)	89.32 %± 1.85	76.23 %± 1.84
DSS layers (Non-linear aggregation)	96.24 %± 2.04	83.97 %± 2.89
DSS layers (Equivariant+invariant)	96.26 %± 1.40	82.94 %± 3.36
DIDA	**97.2** % ± **0.1**	**89.70** % ± **1.89**

Table 1: Patch identification: performance on 10 runs of DIDA, DSS layers and DATASET2VEC. (*): values reported from [16].

4.2 Task 2: Performance model learning

The performance modelling task aims to assess *a priori* the accuracy of the classifier learned from a given machine learning algorithm with a given configuration θ (vector of hyper-parameters ranging in a hyper-parameter space Θ, Appendix D), on a dataset $\mathbf{z}$ (for brevity, the performance of θ on $\mathbf{z}$) [32].

Method	SGD	SVM	LR	k-NN
Hand-crafted	71.18 ± 0.41	75.39 ± 0.29	86.41 ± 0.419	65.44 ± 0.73
DSS (Linear aggregation)	73.46 ± 1.44	82.91 ± 0.22	87.93 ± 0.58	70.07 ± 2.82
DSS (Equivariant+Invariant)	73.54 ± 0.26	81.29 ± 1.65	87.65 ± 0.03	68.55 ± 2.84
DSS (Non-linear aggregation)	74.13 ± 1.01	83.38 ± 0.37	87.92 ± 0.27	73.07 ± 0.77
DIDA	**78.41 ± 0.41**	**84.14 ± 0.02**	**89.77 ± 0.50**	**81.82 ± 0.91**

Table 2: Pairwise ranking of configurations, for ML algorithms SGD, SVM, LR and k-NN: performance on test set of DIDA, hand-crafted and DSS (average and std deviation on 3 runs).

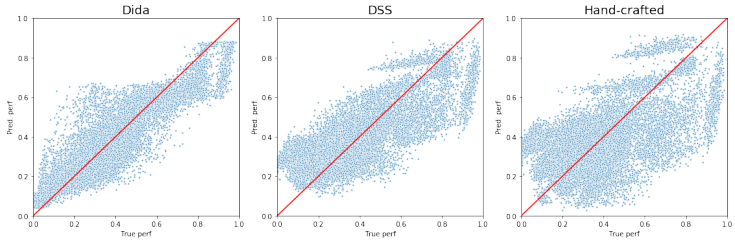

Figure 2: k-NN: True performance vs performance predicted by regression on top of the meta-features (i) learned by DIDA, (ii) DSS or (iii) Hand-crafted statistics.

For each ML algorithm, ranging in Logistic regression (LR), SVM, k-Nearest Neighbours (k-NN), linear classifier learned with stochastic gradient descent (SGD), a set of meta-features is learned to predict whether some configuration θ_1 outperforms some configuration θ_2 on dataset $\mathbf{z}$: to each triplet $(\mathbf{z}, \theta_1, \theta_2)$ is associated a binary value $\ell(\mathbf{z}, \theta_1, \theta_2)$, set to 1 iff θ_2 yields better performance than θ_1 on $\mathbf{z}$. DIDA parameters ζ are trained to build model $\hat{\ell}_\zeta$, minimizing the (weighted version of) cross-entropy loss (5), where $\hat{\ell}_\zeta(\mathbf{z}, \theta_1, \theta_2)$ is a 2-layer FC network with input vector $(\mathcal{F}_\zeta(\mathbf{z}); \theta_1; \theta_2)$, depending on the considered ML algorithm and its configuration space.

In each epoch, a batch made of triplets $(\mathbf{z}, \theta_1, \theta_2)$ is built, with θ_1, θ_2 uniformly drawn in the algorithm configuration space (Table 5) and $\mathbf{z}$ a n-sample d-dimensional patch of a dataset in the OpenML CC-2018 [3] with n uniformly drawn in $[700; 900]$ and d in $[3; 10]$.

The quality of the DIDA meta-features is assessed from the ranking accuracy (Table 2), showing their relevance. The performance gap compared to the baselines is higher for the k-NN modelling task; this is explained as the sought performance model only depends on the local geometry of the examples. Still, good performances are observed over all considered algorithms. A regression setting, where a real-valued $\ell(\mathcal{F}_\zeta(\mathbf{z}), \theta)$ learns the predicted performance, can be successfully considered on top of the learned meta-features $\mathcal{F}_\zeta(\mathbf{z})$ (illustrated on the k-NN algorithm on Figure 2; other results are presented in Appendix D).

5 Conclusion

The theoretical contribution of the paper is the Dida architecture, able to learn from discrete and continuous distributions on $\mathbb{R}^d$, invariant w.r.t. feature ordering, agnostic w.r.t. the size and dimension d of the considered distribution sample (with d less than some upper bound D). This architecture enjoys universal approximation and robustness properties, generalizing former results obtained for point clouds [23]. The merits of Dida are demonstrated on two tasks defined at the dataset level: patch identification and performance model learning, comparatively to the state of the art [23, 16, 25]. The ability to accurately describe a dataset in the landscape defined by ML algorithms opens new perspectives to compare datasets and algorithms, e.g. for domain adaptation [2, 1] and meta-learning [9, 40].

Acknowledgements

The work of G. De Bie is supported by the Région Ile-de-France. H. Rakotoarison acknowledges funding from the ADEME #1782C0034 project NEXT. The work of G. Peyré was supported by the European Research Council (ERC project NORIA) and by the French government under management of Agence Nationale de la Recherche as part of the "Investissements d'avenir" program, reference ANR19-P3IA-0001 (PRAIRIE 3IA Institute).

References

[1] Ben-David, S., Blitzer, J., Crammer, K., Kulesza, A., Pereira, F., and Vaughan, J. W. A theory of learning from different domains. *Machine Learning 79*, 1 (2010), 151–175.

[2] Ben-David, S., Blitzer, J., Crammer, K., and Pereira, F. Analysis of representations for domain adaptation. *Advances in Neural Information Processing Systems 19* (2007), 137–144.

[3] Bischl, B., Casalicchio, G., Feurer, M., Hutter, F., Lang, M., Mantovani, R. G., van Rijn, J. N., and Vanschoren, J. Openml benchmarking suites. *arXiv preprint arXiv:1708.03731* (2019).

[4] Cohen, T., and Welling, M. Group equivariant convolutional networks. *Proceedings of The 33rd International Conference on Machine Learning 48* (20–22 Jun 2016), 2990–2999.

[5] Cox, D. A., Little, J., and O'Shea, D. *Ideals, Varieties, and Algorithms: An Introduction to Computational Algebraic Geometry and Commutative Algebra, 3/e (Undergraduate Texts in Mathematics)*. Springer-Verlag, Berlin, Heidelberg, 2007.

[6] Cybenko, G. Approximation by superpositions of a sigmoidal function. *Mathematics of control, signals and systems 2*, 4 (1989), 303–314.

[7] De Bie, G., Peyré, G., and Cuturi, M. Stochastic deep networks. *Proceedings of the 36th International Conference on Machine Learning* (2019), 1556–1565.

[8] Dua, D., and Graff, C. UCI machine learning repository.

[9] Finn, C., Xu, K., and Levine, S. Probabilistic model-agnostic meta-learning. *Advances in Neural Information Processing Systems 31* (2018), 9516–9527.

[10] Gens, R., and Domingos, P. M. Deep symmetry networks. *Advances in Neural Information Processing Systems 27* (2014), 2537–2545.

[11] Hartford, J., Graham, D. R., Leyton-Brown, K., and Ravanbakhsh, S. Deep models of interactions across sets. *Proceedings of the 35th International Conference on Machine Learning* (2018).

[12] Hashimoto, T., Gifford, D., and Jaakkola, T. Learning population-level diffusions with generative rnns. *Proceedings of The 33rd International Conference on Machine Learning 48* (20–22 Jun 2016), 2417–2426.

[13] Henaff, M., Bruna, J., and LeCun, Y. Deep convolutional networks on graph-structured data. *ArXiv abs/1506.05163* (2015).

[14] Hinton, G., Deng, L., Yu, D., Dahl, G. E., Mohamed, A.-r., Jaitly, N., Senior, A., Vanhoucke, V., Nguyen, P., Sainath, T. N., et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine 29*, 6 (2012), 82–97.

[15] Hutter, F., Kotthoff, L., and Vanschoren, J., Eds. *Automated Machine Learning: Methods, Systems, Challenges*. Springer, 2018. In press, available at http://automl.org/book.

[16] Jomaa, H. S., Grabocka, J., and Schmidt-Thieme, L. Dataset2vec: Learning dataset meta-features. *arXiv abs/1905.11063* (2019).

[17] Keriven, N., and Peyré, G. Universal invariant and equivariant graph neural networks. *Advances in Neural Information Processing Systems 32* (2019), 7090–7099.

[18] Kondor, R., and Trivedi, S. On the generalization of equivariance and convolution in neural networks to the action of compact groups. *Proceedings of The 35th International Conference on Machine Learning* (2018).

[19] Krizhevsky, A., Sutskever, I., and Hinton, G. E. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems* (2012), 1097–1105.

[20] Leshno, M., Lin, V. Y., Pinkus, A., and Schocken, S. Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural networks 6*, 6 (1993), 861–867.

[21] Maron, H., Ben-Hamu, H., Shamir, N., and Lipman, Y. Invariant and equivariant graph networks. *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019* (2019).

[22] Maron, H., Fetaya, E., Segol, N., and Lipman, Y. On the universality of invariant networks. *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA* (2019), 4363–4371.

[23] Maron, H., Litany, O., Chechik, G., and Fetaya, E. On learning sets of symmetric elements. *International Conference on Machine Learning (ICML)* (2020).

[24] Muñoz, M. A., Villanova, L., Baatar, D., and Smith-Miles, K. Instance spaces for machine learning classification. *Machine Learning 107*, 1 (Jan. 2018), 109–147.

[25] Muñoz, M. A., Villanova, L., Baatar, D., and Smith-Miles, K. Instance spaces for machine learning classification. *Machine Learning 107*, 1 (2018), 109–147.

[26] Perrone, V., Jenatton, R., Seeger, M. W., and Archambeau, C. Scalable hyperparameter transfer learning. *Advances in Neural Information Processing Systems 31* (2018), 6845–6855.

[27] Peyré, G., and Cuturi, M. Computational optimal transport. *Foundations and Trends® in Machine Learning 11*, 5-6 (2019), 355–607.

[28] Pfahringer, B., Bensusan, H., and Giraud-Carrier, C. G. Meta-learning by landmarking various learning algorithms. *Proceedings of the Seventeenth International Conference on Machine Learning* (2000), 743–750.

[29] Qi, C. R., Su, H., Mo, K., and Guibas, L. J. Pointnet: Deep learning on point sets for 3d classification and segmentation. *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE* (2017).

[30] Rahman, Q. I., and Schmeisser, G. Analytic theory of polynomials. *Oxford University Press* (2002).

[31] Ravanbakhsh, S., Schneider, J., and Póczos, B. Equivariance through parameter-sharing. *Proceedings of the 34th International Conference on Machine Learning 70* (2017), 2892–2901.

[32] Rice, J. R. The algorithm selection problem. *Advances in Computers 15* (1976), 65–118.

[33] Santambrogio, F. Optimal transport for applied mathematicians. *Birkäuser, NY* (2015).

[34] Segol, N., and Lipman, Y. On universal equivariant set networks. *8th International Conference on Learning Representations, ICLR 2020* (2019).

[35] Shawe-Taylor, J. Symmetries and discriminability in feedforward network architectures. *IEEE Transactions on Neural Networks 4*, 5 (Sep. 1993), 816–826.

[36] Springenberg, J. T., Klein, A., Falkner, S., and Hutter, F. Bayesian optimization with robust bayesian neural networks. *Advances in Neural Information Processing Systems 29* (2016), 4134–4142.

[37] Vinyals, O., Bengio, S., and Kudlur, M. Order matters: Sequence to sequence for sets. *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings* (2016).

[38] Wolpert, D. H. The lack of A priori distinctions between learning algorithms. *Neural Computation 8*, 7 (1996), 1341–1390.

[39] Wood, J., and Shawe-Taylor, J. Representation theory and invariant neural networks. *Discrete applied mathematics 69*, 1-2 (1996), 33–60.

[40] Yoon, J., Kim, T., Dia, O., Kim, S., Bengio, Y., and Ahn, S. Bayesian model-agnostic meta-learning. *Advances in Neural Information Processing Systems 31* (2018), 7332–7342.

[41] Zaheer, M., Kottur, S., Ravanbakhsh, S., Poczos, B., Salakhutdinov, R. R., and Smola, A. J. Deep sets. *Advances in Neural Information Processing Systems 30* (2017), 3391–3401.

A Extension to arbitrary distributions

Overall notations. Let $X \in \mathcal{R}(\mathbb{R}^d)$ denote a random vector on $\mathbb{R}^d$ with $\alpha_X \in \mathcal{P}(\mathbb{R}^d)$ its law (a positive Radon measure with unit mass). By definition, its expectation denoted $\mathbb{E}(X)$ reads $\mathbb{E}(X) = \int_{\mathbb{R}^d} x \mathrm{d}\alpha_X(x) \in \mathbb{R}^d$, and for any continuous function $f : \mathbb{R}^d \to \mathbb{R}^r$, $\mathbb{E}(f(X)) = \int_{\mathbb{R}^d} f(x)\mathrm{d}\alpha_X(x)$. In the following, two random vectors X and X' with same law α_X are considered indistinguishable, noted $X' \sim X$. Letting $f : \mathbb{R}^d \mapsto \mathbb{R}^r$ denote a function on $\mathbb{R}^d$, the push-forward operator by f, noted $f_\sharp : \mathcal{P}(\mathbb{R}^d) \mapsto \mathcal{P}(\mathbb{R}^r)$ is defined as follows, for any g continuous function from $\mathbb{R}^d$ to $\mathbb{R}^r$ (g in $\mathcal{C}(\mathbb{R}^d; \mathbb{R}^r)$):

$$\forall g \in \mathcal{C}(\mathbb{R}^d; \mathbb{R}^r) \quad \int_{\mathbb{R}^r} g \mathrm{d}(f_\sharp \alpha) \stackrel{\text{def.}}{=} \int_{\mathbb{R}^d} g(f(x))\mathrm{d}\alpha(x)$$

Letting $\{x_i\}$ be a set of points in $\mathbb{R}^d$ with $w_i \geqslant 0$ such that $\sum_i w_i = 1$, the discrete measure $\alpha_X = \sum_i w_i \delta_{x_i}$ is the sum of the Dirac measures δ_{x_i} weighted by w_i.

Invariances. In this paper, we consider functions on probability measures that are *invariant with respect to permutations of coordinates*. Therefore, denoting S_d the d-sized permutation group, we consider measures over a symmetrized compact $\Omega \subset \mathbb{R}^d$ equipped with the following equivalence relation: for α, $\beta \in \mathcal{P}(\Omega), \alpha \sim \beta \iff \exists \sigma \in S_d, \beta = \sigma_\sharp \alpha$, such that a measure and its permuted counterpart are indistinguishable in the corresponding quotient space, denoted alternatively $\mathcal{P}(\Omega)_{/\sim}$ or $\mathcal{R}(\Omega)_{/\sim}$. A function $\varphi : \Omega^n \to \mathbb{R}$ is said to be invariant (by permutations of coordinates) iff $\forall \sigma \in S_d, \varphi(x_1, \ldots, x_n) = \varphi(\sigma(x_1), \ldots, \sigma(x_n))$ (Definition 1).

Tensorization. Letting X and Y respectively denote two random vectors on $\mathcal{R}(\mathbb{R}^d)$ and $\mathcal{R}(\mathbb{R}^p)$, the tensor product vector $X \otimes Y$ is defined as: $X \otimes Y \stackrel{\text{def.}}{=} (X', Y') \in \mathcal{R}(\mathbb{R}^d \times \mathbb{R}^p)$, where X' and Y' are independent and have the same law as X and Y, i.e. $\mathrm{d}(\alpha_{X\otimes Y})(x, y) = \mathrm{d}\alpha_X(x)\mathrm{d}\alpha_Y(y)$. In the finite case, for $\alpha_X = \frac{1}{n}\sum_i \delta_{x_i}$ and $\alpha_Y = \frac{1}{m}\sum_j \delta_{y_j}$, then $\alpha_{X\otimes Y} = \frac{1}{nm}\sum_{i,j} \delta_{x_i, y_j}$, weighted sum of Dirac measures on all pairs (x_i, y_j). The $k-$fold tensorization of a random vector $X \sim \alpha_X$, with law $\alpha_X^{\otimes k}$, generalizes the above construction to the case of k independent random variables with law α_X. Tensorization will be used to define the law of datasets, and design universal architectures (Appendix C).

Invariant layers. In the general case, a G-invariant layer f_φ with invariant map $\varphi : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}^r$ such that φ satisfies

$$\forall (x_1, x_2) \in (\mathbb{R}^d)^2, \forall \sigma \in G, \varphi(\sigma(x_1), \sigma(x_2)) = \varphi(x_1, x_2)$$

is defined as

$$f_\varphi : X \in \mathcal{R}(\mathbb{R}^d)_{/\sim} \mapsto \mathbb{E}_{X' \sim X}[\varphi(X, X')] \in \mathcal{R}(\mathbb{R}^r)_{/\sim}$$

where the expectation is taken over $X' \sim X$. Note that considering the couple (X, X') of independent random vectors $X' \sim X$ amounts to consider the tensorized law $\alpha_X \otimes \alpha_X$.

Remark 8. Taking as input a discrete distribution $\alpha_X = \sum_{i=1}^n w_i \delta_{x_i}$, the invariant layer outputs another discrete distribution $\alpha_Y = \sum_{i=1}^n w_i \delta_{y_i}$ with $y_i = \sum_{j=1}^n w_j \varphi(x_i, x_j)$; each input point x_i is mapped onto y_i summarizing the pairwise interactions with x_i after φ.

Remark 9. (Generalization to arbitrary invariance groups) The definition of invariant φ can be generalized to arbitrary invariance groups operating on $\mathbb{R}^d$, in particular sub-groups of the permutation group S_d. After [23] (Thm 5), a simple and only way to design an invariant linear function is to consider $\varphi(z, z') = \psi(z + z')$ with ψ being G-invariant. How to design invariant functions in the general non-linear case is left for further work.

Remark 10. Invariant layers can also be generalized to handle higher order interactions functionals, namely $f_\varphi(X) \stackrel{\text{def.}}{=} \mathbb{E}_{X_2,\ldots,X_N \sim X}[\varphi(X, X_2, \ldots, X_N)]$, which amounts to consider, in the discrete case, N-uple of inputs points $(x_{j_1}, \ldots, x_{j_N})$.

B Proofs on Regularity

Wasserstein distance. The regularity of the involved functionals is measured w.r.t. the 1-Wasserstein distance between two probability distributions $(\alpha, \beta) \in \mathcal{P}(\mathbb{R}^d)$

$$\mathrm{W}_1(\alpha, \beta) \stackrel{\text{def.}}{=} \min_{\pi_1=\alpha, \pi_2=\beta} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\| \mathrm{d}\pi(x, y) \stackrel{\text{def.}}{=} \min_{X \sim \alpha, Y \sim \beta} \mathbb{E}(\|X - Y\|)$$

where the minimum is taken over measures on $\mathbb{R}^d \times \mathbb{R}^d$ with marginals $\alpha, \beta \in \mathcal{P}(\mathbb{R}^d)$. W_1 is known to be a norm [33], that can be conveniently computed using

$$\mathrm{W}_1(\alpha, \beta) = \mathrm{W}_1(\alpha - \beta) = \max_{\mathrm{Lip}(g) \leqslant 1} \int_{\mathbb{R}^d} g \mathrm{d}(\alpha - \beta),$$

where $\mathrm{Lip}(g)$ is the Lipschitz constant of $g : \mathbb{R}^d \to \mathbb{R}$ with respect to the Euclidean norm (unless otherwise stated). For simplicity and by abuse of notations, $\mathrm{W}_1(X, Y)$ is used instead of $\mathrm{W}_1(\alpha, \beta)$ when $X \sim \alpha$ and $Y \sim \beta$. The convergence in law denoted $\rightharpoonup$ is equivalent to the convergence in Wasserstein distance in the sense that $X_k \rightharpoonup X$ is equivalent to $\mathrm{W}_1(X_k, X) \to 0$.

Permutation-invariant Wasserstein distance. The Wasserstein distance is quotiented according to the permutation-invariance equivalence classes: for $\alpha, \beta \in \mathcal{P}(\mathbb{R}^d)$

$$\overline{\mathrm{W}}_1(\alpha, \beta) \stackrel{\text{def.}}{=} \min_{\sigma \in S_d} \mathrm{W}_1(\sigma_\sharp \alpha, \beta) = \min_{\sigma \in S_d} \max_{\mathrm{Lip}(g) \leqslant 1} \int_{\mathbb{R}^d} g \circ \sigma \mathrm{d}\alpha - \int_{\mathbb{R}^d} g \mathrm{d}\beta$$

such that $\overline{\mathrm{W}}_1(\alpha, \beta) = 0 \iff \alpha \sim \beta$. $\overline{\mathrm{W}}_1$ defines a norm on $\mathcal{P}(\mathbb{R}^d)_{/\sim}$.

Lipschitz property. A map $f : \mathcal{R}(\mathbb{R}^d) \to \mathcal{R}(\mathbb{R}^r)$ is continuous for the convergence in law (aka the weak* of measures) if for any sequence $X_k \rightharpoonup X$, then $f(X_k) \rightharpoonup f(X)$. Such a map is furthermore said to be C-Lipschitz for the permutation invariant 1-Wasserstein distance if

$$\forall\, (X, Y) \in (\mathcal{R}(\mathbb{R}^d)_{/\sim})^2,\ \overline{\mathrm{W}}_1(f(X), f(Y)) \leqslant C\overline{\mathrm{W}}_1(X, Y). \tag{6}$$

Lipschitz properties enable us to analyze robustness to input perturbations, since it ensures that if the input distributions of random vectors are close in the permutation invariant Wasserstein sense, the corresponding output laws are close, too.

Proofs of section 3.2.

Proof. (Proposition 1). For $\alpha, \beta \in \mathcal{P}(\mathbb{R}^d)$, Proposition 1 from [7] yields $\mathrm{W}_1(f_\varphi(\alpha), f_\varphi(\beta)) \leqslant 2r\,\mathrm{Lip}(\varphi)\,\mathrm{W}_1(\alpha, \beta)$, hence, for $\sigma \in G$,

$$\begin{aligned}\mathrm{W}_1(\sigma_\sharp f_\varphi(\alpha), f_\varphi(\beta)) &\leqslant \mathrm{W}_1(\sigma_\sharp f_\varphi(\alpha), f_\varphi(\alpha)) + \mathrm{W}_1(f_\varphi(\alpha), f_\varphi(\beta)) \\ &\leqslant \mathrm{W}_1(\sigma_\sharp f_\varphi(\alpha), f_\varphi(\alpha)) + 2r\,\mathrm{Lip}(\varphi)\,\mathrm{W}_1(\alpha, \beta)\end{aligned}$$

hence, taking the infimum over σ yields

$$\begin{aligned}\overline{\mathrm{W}}_1(f_\varphi(\alpha), f_\varphi(\beta)) &\leqslant \overline{\mathrm{W}}_1(f_\varphi(\alpha), f_\varphi(\alpha)) + 2r\,\mathrm{Lip}(\varphi)\,\mathrm{W}_1(\alpha, \beta) \\ &\leqslant 2r\,\mathrm{Lip}(\varphi)\,\mathrm{W}_1(\alpha, \beta)\end{aligned}$$

Since f_φ is invariant, for $\sigma \in G$, $f_\varphi(\mathbf{z}) = f_\varphi(\sigma_\sharp \mathbf{z})$,

$$\overline{\mathrm{W}}_1(f_\varphi(\alpha), f_\varphi(\beta)) \leqslant 2r\,\mathrm{Lip}(\varphi)\,\mathrm{W}_1(\sigma_\sharp\alpha, \beta)$$

Taking the infimum over σ yields the result. □

Proof. (Proposition 2). To upper bound $\overline{\mathrm{W}}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), f_\varphi(\alpha))$ for $\alpha \in \mathcal{P}(\mathbb{R}^d)$, we proceed as follows, using proposition 3 from [7] and proposition 1:

$$\begin{aligned}\mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha_\varphi(\alpha)), f_\varphi(\alpha)) &\leqslant \mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), f_\varphi(\tau_\sharp\alpha)) + \mathrm{W}_1(f_\varphi(\tau_\sharp\alpha), f_\varphi(\alpha)) \\ &\leqslant \|\xi - id\|_{L^1(f_\varphi(\tau_\sharp\alpha))} + \mathrm{Lip}(f_\varphi)\,\mathrm{W}_1(\tau_\sharp\alpha, \alpha) \\ &\leqslant \sup_{y \in f_\varphi(\tau(\Omega))} \|\xi(y) - y\|_2 + 2r\,\mathrm{Lip}(\varphi) \sup_{x\in\Omega} \|\tau(x) - x\|_2\end{aligned}$$

For $\sigma \in G$, we get

$$\mathrm{W}_1(\sigma_\sharp\xi_\sharp f_\varphi(\tau_\sharp\alpha), f_\varphi(\alpha)) \leqslant \mathrm{W}_1(\sigma_\sharp\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\alpha)) + \mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), f_\varphi(\alpha))$$

Taking the infimum over σ yields

$$\begin{aligned}\overline{\mathrm{W}}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), f_\varphi(\alpha)) &\leqslant \mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), f_\varphi(\alpha)) \\ &\leqslant \sup_{y \in f_\varphi(\tau(\Omega))} \|\xi(y) - y\|_2 + 2rC(\varphi) \sup_{x\in\Omega} \|\tau(x) - x\|_2\end{aligned}$$

Similarly, for $\alpha, \beta \in (\mathcal{P}(\mathbb{R}^d))^2$,

$$\begin{aligned}\mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\beta)) &\leqslant \mathrm{Lip}(\xi)\,\mathrm{W}_1(f_\varphi(\tau_\sharp\alpha), f_\varphi(\tau_\sharp\beta)) \\ &\leqslant \mathrm{Lip}(\xi)\,\mathrm{Lip}(f_\varphi)\,\mathrm{W}_1(\tau_\sharp\alpha, \tau_\sharp\beta) \\ &\leqslant 2r\,\mathrm{Lip}(\varphi)\,\mathrm{Lip}(\xi)\,\mathrm{Lip}(\tau)\,\mathrm{W}_1(\alpha,\beta)\end{aligned}$$

hence, for $\sigma \in G$,

$$\begin{aligned}\mathrm{W}_1(\sigma_\sharp\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\beta)) \leqslant\ & \mathrm{W}_1(\sigma_\sharp\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\alpha)) \\ &+ \mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\beta))\end{aligned}$$

and taking the infimum over σ yields

$$\begin{aligned}\overline{\mathrm{W}}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\beta)) &\leqslant \mathrm{W}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\beta)) \\ &\leqslant 2r\,\mathrm{Lip}(\varphi)\,\mathrm{Lip}(\xi)\,\mathrm{Lip}(\tau)\,\mathrm{W}_1(\alpha,\beta)\end{aligned}$$

Since τ is equivariant: namely, for $\alpha \in \mathcal{P}(\mathbb{R}^d)$, $\sigma \in G, \tau_\sharp(\sigma_\sharp\alpha) = \sigma_\sharp(\tau_\sharp\alpha)$, hence, since f_φ is invariant, $f_\varphi(\tau_\sharp(\sigma_\sharp\alpha)) = f_\varphi(\sigma_\sharp(\tau_\sharp\alpha)) = f_\varphi(\tau_\sharp\alpha)$, hence for $\sigma \in G$,

$$\overline{\mathrm{W}}_1(\xi_\sharp f_\varphi(\tau_\sharp\alpha), \xi_\sharp f_\varphi(\tau_\sharp\beta)) \leqslant 2r\,\mathrm{Lip}(\varphi)\,\mathrm{Lip}(\xi)\,\mathrm{Lip}(\tau)\,\mathrm{W}_1(\sigma_\sharp\alpha,\beta)$$

Taking the infimum over σ yields the result. □

C Proofs on Universality

Detailed proof of Theorem 1. This paragraph details the result in the case of S_d-invariance, while the next one focuses on invariances w.r.t. products of permutations. Before providing a proof of Theorem 1 we first state two useful lemmas. Lemma 1 is mentioned for completeness, referring the reader to [7], Lemma 1 for a proof.

Lemma 1. *Let $(S_j)_{j=1}^N$ be a partition of a domain including Ω ($S_j \subset \mathbb{R}^d$) and let $x_j \in S_j$. Let $(\varphi_j)_{j=1}^N$ a set of bounded functions $\varphi_j : \Omega \to \mathbb{R}$ supported on S_j, such that $\sum_j \varphi_j = 1$ on Ω. For $\alpha \in \mathcal{P}(\Omega)$, we denote $\hat{\alpha}_N \overset{\text{def.}}{=} \sum_{j=1}^N \alpha_j \delta_{x_j}$ with $\alpha_j \overset{\text{def.}}{=} \int_{S_j} \varphi_j \mathrm{d}\alpha$. One has, denoting $\Delta_j \overset{\text{def.}}{=} \max_{x\in S_j} \|x_j - x\|$,*

$$\mathrm{W}_1(\hat{\alpha}_N, \alpha) \leqslant \max_{1\leqslant j\leqslant N} \Delta_j.$$

Lemma 2. *Let $f : \mathbb{R}^d \to \mathbb{R}^q$ a $1/p$-Hölder continuous function ($p \geqslant 1$), then there exists a constant $C > 0$ such that for all $\alpha, \beta \in \mathcal{P}(\mathbb{R}^d)$, $\mathrm{W}_1(f_\sharp\alpha, f_\sharp\beta) \leqslant C\,\mathrm{W}_1(\alpha,\beta)^{1/p}$.*

Proof. For any transport map π with marginals α and β, $1/p$-Hölderness of f with constant C yields $\int ||f(x) - f(y)||_2 \mathrm{d}\pi(x,y) \leqslant C \int ||x-y||_2^{1/p} \mathrm{d}\pi(x,y) \leqslant C\left(\int ||x-y||_2 \mathrm{d}\pi(x,y)\right)^{1/p}$ using Jensen's inequality ($p \leqslant 1$). Taking the infimum over π yields $\mathrm{W}_1(f_\sharp\alpha, f_\sharp\beta) \leqslant C\,\mathrm{W}_1(\alpha,\beta)^{1/p}$. □

Now we are ready to dive into the proof. Let $\alpha \in \mathcal{P}(\mathbb{R}^d)$. We consider:

- $h : x = (x_1, \dots, x_d) \in \mathbb{R}^d \mapsto \left(\sum_{1 \leqslant j_1 < \dots < j_i \leqslant d} x_{j_1} \cdot \dots \cdot x_{j_i} \right)_{i=1\dots d} \in \mathbb{R}^d$ the collection of d elementary symmetric polynomials; h does not lead to a loss in information, in the sense that it generates the ring of S_d-invariant polynomials (see for instance [5], chapter 7, theorem 3) while preserving the classes (see the proof of Lemma 2, appendix D from [23]);
- h is obviously not injective, so we consider $\pi : \mathbb{R}^d \to \mathbb{R}^d/S_d$ the projection onto $\mathbb{R}^d/S_d$: $h = \tilde{h} \circ \pi$ such that $\tilde{h}$ is bijective from $\pi(\Omega)$ to its image Ω', compact of $\mathbb{R}^d$; $\tilde{h}$ and $\tilde{h}^{-1}$ are continuous;
- Let $(\varphi_i)_{i=1\dots N}$ the piecewise affine P1 finite element basis, which are hat functions on a discretization $(S_i)_{i=1\dots N}$ of $\Omega' \subset \mathbb{R}^d$, with centers of cells $(y_i)_{i=1\dots N}$. We then define $g : x \in \mathbb{R}^d \mapsto (\varphi_1(x), \dots, \varphi_N(x)) \in \mathbb{R}^N$;
- $f : (\alpha_1, \dots, \alpha_N) \in \mathbb{R}^N \mapsto \mathcal{F}\left(\sum_{i=1}^N \alpha_i \delta_{\tilde{h}^{-1}(y_i)} \right) \in \mathbb{R}$.

We approximate $\mathcal{F}$ using the following steps:

- Lemma 1 (see Lemma 1 from [7]) yields that $h_\sharp \alpha$ and $\widehat{h_\sharp \alpha} = \sum_{i=1}^N \alpha_i \delta_{y_i}$ are close: $\mathrm{W}_1(h_\sharp \alpha, \widehat{h_\sharp \alpha}) \leqslant \sqrt{d}/N^{1/d}$;
- The map $\tilde{h}^{-1}$ is regular enough ($1/d$-Hölder) such that according to Lemma 2, there exists a constant $C > 0$ such that

  $$\mathrm{W}_1(\tilde{h}_\sharp^{-1}(h_\sharp \alpha), \tilde{h}_\sharp^{-1}\widehat{h_\sharp \alpha}) \leqslant C\, \mathrm{W}_1(h_\sharp \alpha, \widehat{h_\sharp \alpha})^{1/d} \leqslant C d^{1/2d}/N^{1/d^2}$$

  Hence $\overline{\mathrm{W}}_1(\alpha, \tilde{h}_\sharp^{-1}\widehat{h_\sharp \alpha}) := \inf_{\sigma \in S_d} \mathrm{W}_1(\sigma_\sharp \alpha, \tilde{h}_\sharp^{-1}\widehat{h_\sharp \alpha}) \leqslant C d^{1/2d}/N^{1/d^2}$.

  Note that h maps the roots of polynomial $\prod_{i=1}^d (X - x^{(i)})$ to its coefficients (up to signs). Theorem 1.3.1 from [30] yields continuity and $1/d$-Hölderness of the reverse map. Hence $\tilde{h}^{-1}$ is $1/d$-Hölder.
- Since Ω is compact, by Banach-Alaoglu theorem, we obtain that $\mathcal{P}(\Omega)$ is weakly-* compact, hence $\mathcal{P}(\Omega)_{/\sim}$ also is. Since $\mathcal{F}$ is continuous, it is thus uniformly weak-* continuous: for any $\varepsilon > 0$, there exists $\delta > 0$ such that $\overline{\mathrm{W}}_1(\alpha, \tilde{h}_\sharp^{-1}\widehat{h_\sharp \alpha}) \leqslant \delta$ implies $|\mathcal{F}(\alpha) - \mathcal{F}(\tilde{h}_\sharp^{-1}\widehat{h_\sharp \alpha})| < \varepsilon$. Choosing N large enough such that $C d^{1/2d}/N^{1/d^2} \leqslant \delta$ therefore ensures that $|\mathcal{F}(\alpha) - \mathcal{F}(\tilde{h}_\sharp^{-1}\widehat{h_\sharp \alpha})| < \varepsilon$.

Extension of Theorem 1 to products of permutation groups.

Corollary 1. *Let $\mathcal{F} : \mathcal{P}(\Omega)_{/\sim} \to \mathbb{R}$ a continuous $S_{d_1} \times \dots \times S_{d_n}$-invariant map ($\sum_i d_i = d$), where Ω is a symmetrized compact over $\mathbb{R}^d$. Then $\forall \varepsilon > 0$, there exists three continuous maps f, g, h such that*

$$\forall \alpha \in \mathcal{M}_+^1(\Omega)_{/\sim}, |\mathcal{F}(\alpha) - f \circ \mathbb{E} \circ g(h_\sharp \alpha)| < \varepsilon$$

where h is invariant; g, h are independent of $\mathcal{F}$.

Proof. We provide a proof in the case $G = S_d \times S_p$, which naturally extends to any product group $G = S_{d_1} \times \ldots \times S_{d_n}$. We trade h for the collection of elementary symmetric polynomials in the first d variables; and in the last p variables: $h : (x_1, \ldots, x_d, y_1, \ldots, y_p) \in \mathbb{R}^{d+p} \mapsto ([\sum_{1 \leqslant j_1 < \ldots < j_i \leqslant d} x_{j_1} \ldots x_{j_i}]_{i=1}^{d}; [\sum_{1 \leqslant j_1 < \ldots < j_i \leqslant p} y_{j_1} \ldots y_{j_i}]_{i=1}^{p}) \in \mathbb{R}^{d+p}$ up to normalizing constants (see Lemma 4). Step 1 (in Lemma 3) consists in showing that h does not lead to a loss of information, in the sense that it generates the ring of $S_d \times S_p-$invariant polynomials. In step 2 (in Lemma 4), we show that $\tilde{h}^{-1}$ is $1/\max(d,p)-$Hölder. Combined with the proof of Theorem 1, this amounts to showing that the concatenation of Hölder functions (up to normalizing constants) is Hölder. With these ingredients, the sketch of the previous proof yields the result. □

Lemma 3. *Let the collection of symmetric invariant polynomials* $[P_i(X_1, \ldots, X_d)]_{i=1}^{d} \overset{\text{def.}}{=} [\sum_{1 \leqslant j_1 < \ldots < j_i \leqslant d} X_{j_1} \ldots X_{j_i}]_{i=1}^{d}$ *and* $[Q_i(Y_1, \ldots, Y_p)]_{i=1}^{p} = [\sum_{1 \leqslant j_1 < \ldots < j_i \leqslant p} Y_{j_1} \ldots Y_{j_i}]_{i=1}^{p}$. *The* $d+p-$*sized family* $(P_1, \ldots, P_d, Q_1, \ldots, Q_p)$ *generates the ring of* $S_d \times S_p-$*invariant polynomials.*

Proof. The result comes from the fact the fundamental theorem of symmetric polynomials (see [5] chapter 7, theorem 3) does not depend on the base field. Every $S_d \times S_p-$invariant polynomial $P(X_1, \ldots, X_d, Y_1, \ldots, Y_p)$ is also $S_d \times I_p-$invariant with coefficients in $\mathbb{R}[Y_1, \ldots, Y_p]$, hence it can be written $P = R(Y_1, \ldots, Y_p)(P_1, \ldots, P_d)$. It is then also S_p-invariant with coefficients in $\mathbb{R}[P_1, \ldots, P_d]$, hence it can be written $P = S(Q_1, \ldots, Q_p)(P_1, \ldots, P_d) \in \mathbb{R}[P_1, \ldots, P_d, Q_1, \ldots, Q_p]$. □

Lemma 4. *Let* $h : (x, y) \in \Omega \subset \mathbb{R}^{d+p} \mapsto (f(x)/C_1, g(y)/C_2) \in \mathbb{R}^{d+p}$ *where* Ω *is compact,* $f : \mathbb{R}^d \to \mathbb{R}^d$ *is* $1/d-$*Hölder with constant* C_1 *and* $g : \mathbb{R}^p \to \mathbb{R}^p$ *is* $1/p-$*Hölder with constant* C_2. *Then* h *is* $1/\max(d,p)-$*Hölder.*

Proof. Without loss of generality, we consider $d > p$ so that $\max(d,p) = d$, and f, g normalized (f.i. $\forall x, x_0 \in (\mathbb{R}^d)^2, \|f(x) - f(x_0)\|_1 \leqslant \|x - x_0\|_1^{1/d}$). For $(x,y), (x_0,y_0) \in \Omega^2$, $\|h(x,y) - h(x_0,y_0)\|_1 \leqslant \|f(x) - f(x_0)\|_1 + \|g(y) - g(y_0)\|_1 \leqslant \|x - x_0\|_1^{1/d} + \|y - y_0\|_1^{1/p}$ since both f, g are Hölder. We denote D the diameter of Ω, such that both $\|x - x_0\|_1 / D \leqslant 1$ and $\|y - y_0\|_1 / D \leqslant 1$ hold. Therefore $\|h(x,y) - h(x_0,y_0)\|_1 \leqslant D^{1/d} \left(\frac{\|x-x_0\|_1}{D}\right)^{1/d} + D^{1/p} \left(\frac{\|y-y_0\|_1}{D}\right)^{1/p} \leqslant 2^{1-1/d} D^{1/p-1/d} \|(x,y) - (x_0,y_0)\|_1^{1/d}$ using Jensen's inequality, hence the result. □

In the next two paragraphs, we focus the case of S_d-invariant functions for the sake of clarity, without loss of generality. Indeed, the same technique applies to $G-$invariant functions as h in that case has the same structure: its first d_X components are $S_{d_X}-$invariant functions of the first d_X variables and its last d_Y components are $S_{d_Y}-$invariant functions of the last variables.

Extension of Theorem 1 to distributions on spaces of varying dimension.

Corollary 2. *Let $I = [0;1]$ and, for $k \in [1; d_m], \mathcal{F}_k : \mathcal{P}(I^k) \to \mathbb{R}$ continuous and S_k-invariant. Suppose $(\mathcal{F}_k)_{k=1\dots d_m-1}$ are restrictions of $\mathcal{F}_{d_m}$, namely, $\forall \alpha_k \in \mathcal{P}(I^k), \mathcal{F}_k(\alpha_k) = \mathcal{F}_{d_m}(\alpha_k \otimes \delta_0^{\otimes d_m - k})$. Then functions f and g from Theorem 1 are uniform: there exists f, g continuous, $h_1, \dots, h_{d_m}$ continuous invariant such that*

$$\forall k = 1 \dots d_m, \forall \alpha_k \in \mathcal{P}(I^k), |\mathcal{F}_k(\alpha_k) - f \circ \mathbb{E} \circ g(h_{k\sharp}\alpha_k)| < \varepsilon.$$

Proof. Theorem 1 yields continuous f, g and a continuous invariant h_{d_m} such that $\forall \alpha \in \mathcal{P}(I^{d_m}), |\mathcal{F}_{d_m} - f \circ \mathbb{E} \circ g(h_{d_m\sharp}\alpha)| < \varepsilon$. For $k = 1 \dots d_m - 1$, we denote $h_k : (x_1, \dots, x_k) \in \mathbb{R}^k \mapsto ((\sum_{1 \leqslant j_1 < \dots < j_i \leqslant k} x^{(j_1)} \cdot \dots \cdot x^{(j_i)})_{i=1\dots k}, 0 \dots, 0) \in \mathbb{R}^{d_m}$. With the hypothesis, for $k = 1 \dots d_m - 1$, $\alpha_k \in \mathcal{P}(I^k)$, the fact that $h_{k\sharp}(\alpha_k) = h_{d_m\sharp}(\alpha_k \otimes \delta_0^{\otimes d_m - k})$ yields the result. □

Approximation by invariant neural networks. Based on theorem 1, $\mathcal{F}$ is uniformly close to $f \circ \mathbb{E} \circ g \circ h$:

- We approximate f by a neural network $f_\theta : x \in \mathbb{R}^N \mapsto C_1\lambda(A_1 x + b_1) \in \mathbb{R}$, where p_1 is an integer, $A_1 \in \mathbb{R}^{p_1 \times N}, C_1 \in \mathbb{R}^{1 \times p_1}$ are weights, $b_1 \in \mathbb{R}^{p_1}$ is a bias and λ is a non-linearity.
- Since each component φ_j of $\varphi = g \circ h$ is permutation-invariant, it has the representation $\varphi_j : x = (x_1, \dots, x_d) \in \mathbb{R}^d \mapsto \rho_j\left(\sum_{i=1}^d u(x_i)\right)$ [41] (which is a special case of our layers with a base function only depending on its first argument, see section 2.2), $\rho_j : \mathbb{R}^{d+1} \to \mathbb{R}$, and $u : \mathbb{R} \to \mathbb{R}^{d+1}$ independent of j (see [41], theorem 7).
- We can approximate ρ_j and u by neural networks $\rho_{j,\theta} : x \in \mathbb{R}^{d+1} \mapsto C_{2,j}\lambda(A_{2,j}x + b_{2,j}) \in \mathbb{R}$ and $u_\theta : x \in \mathbb{R}^d \mapsto C_3\lambda(A_3 x + b_3) \in \mathbb{R}^{d+1}$, where $p_{2,j}, p_3$ are integers, $A_{2,j} \in \mathbb{R}^{p_{2,j} \times (d+1)}, C_{2,j} \in \mathbb{R}^{1 \times p_{2,j}}, A_3 \in \mathbb{R}^{p_3 \times 1}, C_3 \in \mathbb{R}^{(d+1) \times p_3}$ are weights and $b_{2,j} \in \mathbb{R}^{p_{2,j}}, b_3 \in \mathbb{R}^{p_3}$ are biases, and denote $\varphi_\theta(x) = (\varphi_{j,\theta}(x))_j \overset{\text{def.}}{=} (\rho_{j,\theta}(\sum_{i=1}^d u_\theta(x_i)))_j$.

Indeed, we upper-bound the difference of interest $|\mathcal{F}(\alpha) - f_\theta\left(\mathbb{E}_{X\sim\alpha}\left(\varphi_\theta(X)\right)\right)|$ by triangular inequality by the sum of three terms:

- $|\mathcal{F}(\alpha) - f\left(\mathbb{E}_{X\sim\alpha}\left(\varphi(X)\right)\right)|$
- $|f\left(\mathbb{E}_{X\sim\alpha}\left(\varphi(X)\right)\right) - f_\theta\left(\mathbb{E}_{X\sim\alpha}\left(\varphi(X)\right)\right)|$
- $|f_\theta\left(\mathbb{E}_{X\sim\alpha}\left(\varphi(X)\right)\right) - f_\theta\left(\mathbb{E}_{X\sim\alpha}\left(\varphi_\theta(X)\right)\right)|$

and bound each term by $\frac{\varepsilon}{3}$, which yields the result. The bound on the first term directly comes from theorem 1 and yields a constant N which depends on ε. The bound on the second term is a direct application of the universal approximation theorem (UAT) [6, 20]. Indeed, since α is a probability measure, input values of f lie in a compact subset of $\mathbb{R}^N$: $\| \int_\Omega g \circ h(x) \mathrm{d}\alpha \|_\infty \leqslant \max_{x\in\Omega} \max_i |g_i \circ h(x)|$, hence the theorem is applicable as long as λ is a nonconstant, bounded and continuous

activation function. Let us focus on the third term. Uniform continuity of f_θ yields the existence of $\delta > 0$ s.t. $||u - v||_1 < \delta$ implies $|f_\theta(u) - f_\theta(v)| < \frac{\varepsilon}{3}$. Let us apply the UAT: each component φ_j of h can be approximated by a neural network $\varphi_{j,\theta}$. Therefore:

$$\begin{aligned}\|\mathbb{E}_{X\sim\alpha}\left(\varphi(X) - \varphi_\theta(X)\right)\|_1 &\leqslant \mathbb{E}_{X\sim\alpha}\|\varphi(X) - \varphi_\theta(X)\|_1 \leqslant \sum_{j=1}^{N}\int_\Omega |\varphi_j(x) - \varphi_{j,\theta}(x)|\mathrm{d}\alpha(x)\\ &\leqslant \sum_{j=1}^{N}\int_\Omega |\varphi_j(x) - \rho_{j,\theta}(\sum_{i=1}^{d} u(x_i))|\mathrm{d}\alpha(x)\\ &+ \sum_{j=1}^{N}\int_\Omega |\rho_{j,\theta}(\sum_{i=1}^{d} u(x_i)) - \rho_{j,\theta}(\sum_{i=1}^{d} u_\theta(x_i))|\mathrm{d}\alpha(x)\\ &\leqslant N\frac{\delta}{2N} + N\frac{\delta}{2N} = \delta\end{aligned}$$

using the triangular inequality and the fact that α is a probability measure. The first term is small by UAT on ρ_j while the second also is, by UAT on u and uniform continuity of $\rho_{j,\theta}$. Therefore, by uniform continuity of f_θ, we can conclude.

Universality of tensorization. This complementary theorem provides insight into the benefits of tensorization for approximating invariant regression functionals, as long as the test function is invariant.

Theorem 2. *The algebra*

$$\mathcal{A}_\Omega \overset{\text{def.}}{=} \left\{\mathcal{F} : \mathcal{P}(\Omega)_{/\sim} \to \mathbb{R}, \exists n \in \mathbb{N}, \exists \varphi : \Omega^n \to \mathbb{R} \text{ invariant}, \forall \alpha, \mathcal{F}(\alpha) = \int_{\Omega^n} \varphi \mathrm{d}\alpha^{\otimes n}\right\}$$

where $\otimes n$ denotes the n-fold tensor product, is dense in $\mathcal{C}(\mathcal{M}_+^1(\Omega)_{/\sim})$.

Proof. This result follows from the Stone-Weierstrass theorem. Since Ω is compact, by Banach-Alaoglu theorem, we obtain that $\mathcal{P}(\Omega)$ is weakly-* compact, hence $\mathcal{P}(\Omega)_{/\sim}$ also is. In order to apply Stone-Weierstrass, we show that $\mathcal{A}_\Omega$ contains a non-zero constant function and is an algebra that separates points. A (non-zero, constant) 1-valued function is obtained with $n = 1$ and $\varphi = 1$. Stability by scalar is straightforward. For stability by sum: given $(\mathcal{F}_1, \mathcal{F}_2) \in \mathcal{A}_\Omega^2$ (with associated functions (φ_1, φ_2) of tensorization degrees (n_2, n_2)), we denote $n \overset{\text{def.}}{=} \max(n_1, n_2)$ and $\varphi(x_1, \ldots, x_n) \overset{\text{def.}}{=} \varphi_1(x_1, \ldots, x_{n_1}) + \varphi_2(x_1, \ldots, x_{n_2})$ which is indeed invariant, hence $\mathcal{F}_1 + \mathcal{F}_2 = \int_{\Omega^n} \varphi \mathrm{d}\alpha^{\otimes n} \in \mathcal{A}_\Omega$. Similarly, for stability by product: denoting this time $n = n_1 + n_2$, we introduce the invariant $\varphi(x_1, \ldots, x_n) = \varphi_1(x_1, \ldots, x_{n_1}) \times \varphi_2(x_{n_1+1}, \ldots, x_n)$, which shows that $\mathcal{F} = \mathcal{F}_1 \times \mathcal{F}_2 \in \mathcal{A}_\Omega$ using Fubini's theorem. Finally, $\mathcal{A}_\Omega$ separates points: if $\alpha \neq \nu$, then there exists a symmetrized domain S such that $\alpha(S) \neq \nu(S)$: indeed, if for all symmetrized domains S, $\alpha(S) = \nu(S)$, then $\alpha(\Omega) = \nu(\Omega)$ which is absurd. Taking $n = 1$ and $\varphi = 1_S$ (invariant since S is symmetrized) yields an $\mathcal{F}$ such that $\mathcal{F}(\alpha) \neq \mathcal{F}(\nu)$. □

D Experimental validation, supplementary material

DIDA and DSS source code are provided in the last file of the supplementary material.

D.1 Benchmark Details

Three benchmarks are used (Table 3): TOY and UCI, taken from [16], and OpenML CC-18. TOY includes 10,000 datasets, where instances are distributed along mixtures of Gaussian, intertwinning moons and rings in $\mathbb{R}^2$, with 2 to 7 classes. UCI includes 121 datasets from the UCI Irvine repository [8]. Each benchmark is divided into 70%-30% training-test sets.

	# datasets	# samples	# features	# labels	test ratio
Toy Dataset	10000	[2048, 8192]	2	[2, 7]	0.3
UCI	121	[10, 130064]	[3, 262]	[2, 100]	0.3
OpenML CC-18	71	[500, 100000]	[5, 3073]	[2, 46]	0.5

Table 3: Benchmarks characteristics

D.2 Baseline Details

Dataset2Vec details. We used the available implementation of DATASET2VEC[3]. In the experiments, DATASET2VEC hyperparameters are set to their default values except size and number of patches, set to same values as in DIDA.

DSS layer details. We built our own implementation of invariant DSS layers, as follows. Linear invariant DSS layers (see [23], Theorem 5, 3.) are of the form

$$L_{inv} : X \in \mathbb{R}^{n \times d} \mapsto L^H(\sum_{j=1}^{n} x_j) \in \mathbb{R}^K \tag{7}$$

where $L^H : \mathbb{R}^d \to \mathbb{R}^K$ is a linear H-invariant function. Our applicative setting requires that the implementation accommodates to varying input dimensions d as well as permutation invariance, hence we consider the Deep Sets representation (see [41], Theorem 7)

$$L^H : x = (x_1, \ldots, x_d) \in \mathbb{R}^d \mapsto \rho\left(\sum_{i=1}^{d} \varphi(x_i)\right) \in \mathbb{R}^K \tag{8}$$

where $\varphi : \mathbb{R} \to \mathbb{R}^{d+1}$ and $\rho : \mathbb{R}^{d+1} \to \mathbb{R}^K$ are modelled as (i) purely linear functions; (ii) FC networks, which extends the initial linear setting (7). In

[3] DATASET2VEC code is available at `https://github.com/hadijomaa/dataset2vec`

our case, $H = S_{d_X} \times S_{d_Y}$, hence, two invariant layers of the form (7-8) are combined to suit both feature- and label-invariance requirements. Both outputs are concatenated and followed by an FC network to form the DSS meta-features. The last experiments use DSS equivariant layers (see [23], Theorem 1), which take the form

$$L_{eq} : X \in \mathbb{R}^{n \times d} \mapsto \left(L^1_{eq}(x_i) + L^2_{eq}(\sum_{j \neq i} x_j) \right)_{i \in [n]} \in \mathbb{R}^{n \times d} \tag{9}$$

where L^1_{eq} and L^2_{eq} are linear H-equivariant layers. Similarly, both feature- and label-equivariance requirements are handled via the Deep Sets representation of equivariant functions (see [41], Lemma 3) and concatenated to be followed by an invariant layer, forming the DSS meta-features. All methods are allocated the same number of parameters to ensure fair comparison. We provide our implementation of the DSS layers in the supplementary material.

Hand-crafted meta-features. For the sake of reproducibility, the list of meta-features used in section 4 is given in Table 4. Note that meta-features related to missing values and categorical features are omitted, as being irrelevant for the considered benchmarks. Hand-crafted meta-features are extracted using the BYU `metalearn` library[4].

D.3 Performance Prediction

Experimental setting. Table 5 details all hyper-parameter configurations Θ considered in Section 4.2. As said, the learnt meta-features $\mathcal{F}_\zeta(\mathbf{z})$ can be used in a regression setting, predicting the performance of various ML algorithms on a dataset $\mathbf{z}$. Several performance models have been considered on top of the meta-features learnt in Section 4.2, for instance (i) a BOHAMIANN network [36]; (ii) Random Forest models, trained under a Mean Squared Error loss between predicted and true performances.

Results. Table 6 reports the Mean Squared Error on the test set with performance model BOHAMIANN [36], comparatively to DSS and hand-crafted ones. Replacing the surrogate model with Random Forest concludes to the same ranking as in Table 6. Figure 3 complements Table 6 in assessing the learnt DIDA meta-features for performance model learning. It shows DIDA's ability to capture more expressive meta-features than both DSS and hand-crafted ones, for all ML algorithms considered.

[4] See `https://github.com/byu-dml/metalearn`

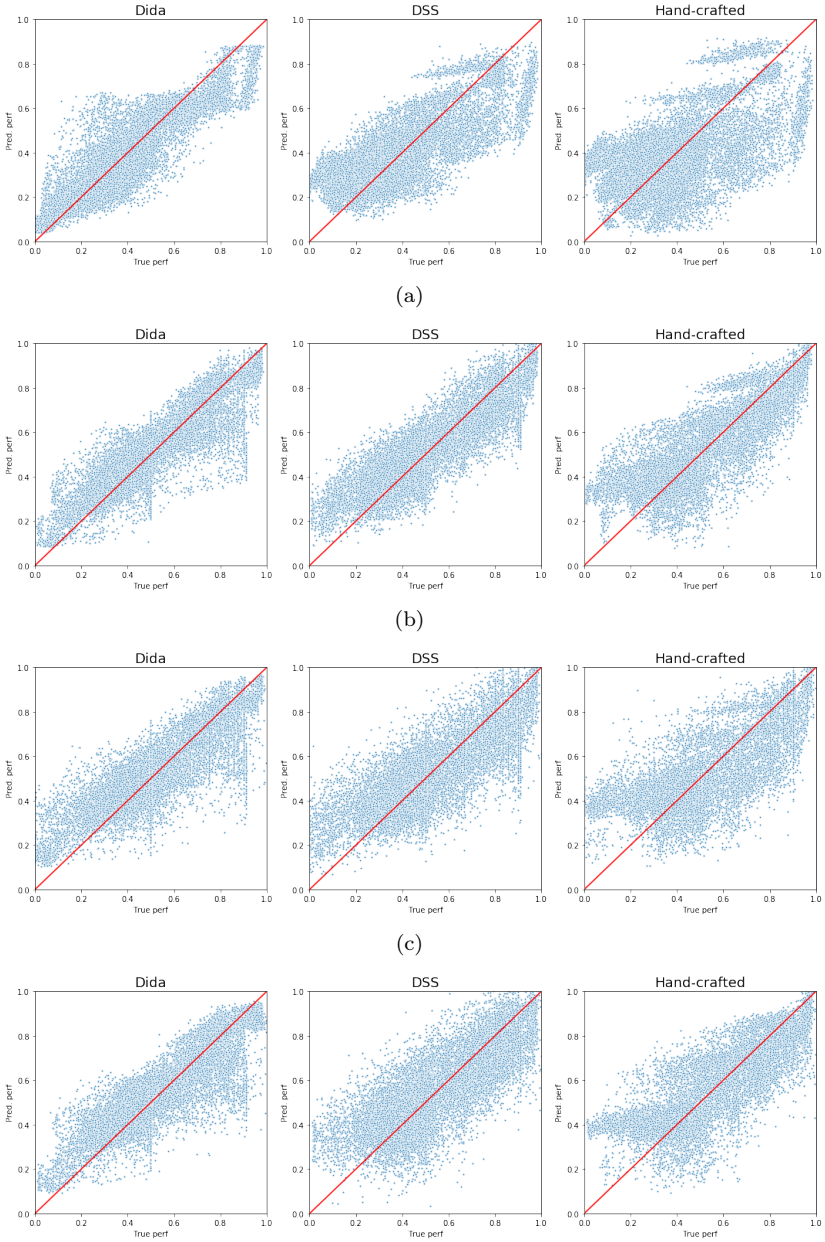

Figure 3: Comparison between the true performance and the performance predicted by the trained surrogate model on DIDA, DSS or Hand-crafted meta-features, for various ML algorithms: (a) k-NN; (b) Logistic Regression; (c) SVM; (d) Linear classifier learnt with stochastic gradient descent.

Meta-features	**Mean**	**Min**	**Max**
Quartile2ClassProbability	0.500	0.75	0.25
MinorityClassSize	487.423	426.000	500.000
Quartile3CardinalityOfNumericFeatures	224.354	0.000	976.000
RatioOfCategoricalFeatures	0.347	0.000	1.000
MeanCardinalityOfCategoricalFeatures	0.907	0.000	2.000
SkewCardinalityOfNumericFeatures	0.148	-2.475	3.684
RatioOfMissingValues	0.001	0.000	0.250
MaxCardinalityOfNumericFeatures	282.461	0.000	977.000
Quartile2CardinalityOfNumericFeatures	185.555	0.000	976.000
KurtosisClassProbability	-2.025	-3.000	-2.000
NumberOfNumericFeatures	3.330	0.000	30.000
NumberOfInstancesWithMissingValues	2.800	0.000	1000.000
MaxCardinalityOfCategoricalFeatures	0.917	0.000	2.000
Quartile1CardinalityOfCategoricalFeatures	0.907	0.000	2.000
MajorityClassSize	512.577	500.000	574.000
MinCardinalityOfCategoricalFeatures	0.879	0.000	2.000
Quartile2CardinalityOfCategoricalFeatures	0.915	0.000	2.000
NumberOfCategoricalFeatures	1.854	0.000	27.000
NumberOfFeatures	5.184	4.000	30.000
Dimensionality	0.005	0.004	0.030
SkewCardinalityOfCategoricalFeatures	-0.050	-4.800	0.707
KurtosisCardinalityOfCategoricalFeatures	-1.244	-3.000	21.040
StdevCardinalityOfNumericFeatures	68.127	0.000	678.823
StdevClassProbability	0.018	0.000	0.105
KurtosisCardinalityOfNumericFeatures	-1.060	-3.000	12.988
NumberOfInstances	1000.000	1000.000	1000.000
Quartile3CardinalityOfCategoricalFeatures	0.916	0.000	2.000
NumberOfMissingValues	2.800	0.000	1000.000
Quartile1ClassProbability	0.494	0.463	0.500
StdevCardinalityOfCategoricalFeatures	0.018	0.000	0.707
MeanClassProbability	0.500	0.500	0.500
NumberOfFeaturesWithMissingValues	0.003	0.000	1.000
MaxClassProbability	0.513	0.500	0.574
NumberOfClasses	2.000	2.000	2.000
MeanCardinalityOfNumericFeatures	197.845	0.000	976.000
SkewClassProbability	0.000	-0.000	0.000
Quartile3ClassProbability	0.506	0.500	0.537
MinCardinalityOfNumericFeatures	138.520	0.000	976.000
MinClassProbability	0.487	0.426	0.500
RatioOfInstancesWithMissingValues	0.003	0.000	1.000
Quartile1CardinalityOfNumericFeatures	160.748	0.000	976.000
RatioOfNumericFeatures	0.653	0.000	1.000
RatioOfFeaturesWithMissingValues	0.001	0.000	0.250

Table 4: Hand-crafted meta-features

	Parameter	Parameter values	Scale
LR	warm start	True, Fase	
	fit intercept	True, Fase	
	tol	[0.00001, 0.0001]	
	C	[1e-4, 1e4]	log
	solver	newton-cg, lbfgs, liblinear, sag, saga	
	max_iter	[5, 1000]	
SVM	kernel	linear, rbf, poly, sigmoid	
	C	[0.0001, 10000]	log
	shrinking	True, False	
	degree	[1, 5]	
	coef0	[0, 10]	
	gamma	[0.0001, 8]	
	max_iter	[5, 1000]	
KNN	n_neighbors	[1, 100]	log
	p	[1, 2]	
	weights	uniform, distance	
SGD	alpha	[0.1, 0.0001]	log
	average	True, False	
	fit_intercept	True, False	
	learning rate	optimal, invscaling, constant	
	loss	hinge, log, modified_huber, squared_hinge, perceptron	
	penalty	l1, l2, elasticnet	
	tol	[1e-05, 0.1]	log
	eta0	[1e-7, 0.1]	log
	power_t	[1e-05, 0.1]	log
	epsilon	[1e-05, 0.1]	log
	l1_ratio	[1e-05, 0.1]	log

Table 5: Hyper-parameter configurations considered in Section 4.2.

Method	SGD	SVM	LR	KNN
Hand-crafted	0.016 ± 0.001	0.021 ± 0.001	0.018 ± 0.002	0.034 ± 0.001
DSS (Linear aggregation)	0.015 ± 0.007	0.020 ± 0.002	0.019 ± 0.001	0.025 ± 0.010
DSS (Equivariant+Invariant)	0.014 ± 0.002	0.017 ± 0.003	0.015 ± 0.003	0.028 ± 0.003
DSS (Non-linear aggregation)	0.015 ± 0.009	0.016 ± 0.003	0.014 ± 0.001	0.020 ± 0.005
DIDA	**0.012 ± 0.001**	**0.015 ± 0.001**	**0.010 ± 0.001**	**0.009 ± 0.000**

Table 6: Performance modelling, comparative results of DIDA, DSS and Hand-crafted (HC) meta-features: Mean Squared Error (average over 5 runs) on test set, between the true performance and the performance predicted by the trained BOHAMIANN surrogate model, for ML algorithms SVM, LR, kNN, SGD (see text).