Learning More Universal Representations for Transfer-Learning

Youssef Tamaazousti, Hervé Le Borgne, Céline Hudelot, Mohamed-El-Amine Seddik and Mohamed Tamaazousti

Abstract—A representation is supposed universal if it encodes any element of the visual world (e.g., objects, scenes) in any configuration (e.g., scale, context). While not expecting pure universal representations, the goal in the literature is to improve the universality level, starting from a representation with a certain level. To improve that universality level, one can diversify the source-task, but it requires many additive annotated data that is costly in terms of manual work and possible expertise. We formalize such a diversification process then propose two methods to improve the universality of CNN representations that limit the need for additive annotated data. The first relies on human categorization knowledge and the second on re-training using fine-tuning. We propose a new aggregating metric to evaluate the universality in a transfer-learning scheme, that addresses more aspects than previous works. Based on it, we show the interest of our methods on 10 target-problems, relating to classification on a variety of visual domains.

Index Terms—Universal Representations, Universality Evaluation, Transfer-Learning, Visual Recognition.

1 Introduction

Humans are able to recognize scenes and their objects with disconcerting ease in comparison to a machine. According to Atkinson's cognitive study [1], such capabilities result from the development of a powerful internal representation in their infancy, which they re-use later in life to solve multiple problems. Similarly, a neural network learns a representation from data that is then used to solve various tasks and address different domains. However, while human vision is powerful for many different problems, machines are usually able to tackle a particular task only. Some recent works thus seek to "mimic" the ability of human vision, by learning *universal models* that are able to solve every possible task (recognition, detection, segmentation, etc.) [2], [3] or *universal representations* that are able to encode any possible visual element of interest (materials, objects, scenes, etc.) whatever its situation (scale, occlusion, context, etc.) [4], [5], [6]. Since data encoding is at the source of success to solve a task, our interest mainly focus on the representation [7]. However, far from expecting a representation able to encode any information, the actual purpose of our work is to *improve* the universality of a given representation, at a minimal cost in terms of annotated data.

Obviously, one can use more training data to learn a representation with deep neural architectures and thus address more tasks and domains. Hence, [4] proposed to simultaneously learn a wide set of different visual domains, by better considering the additional data, through the use of scaling parameters that learn the statistics of each domain. In the same vein, Subramanian *et al.* [6] considered multiple datasets with different tasks and proposed to find which of the available multi-task learning algorithms and set of tasks allows the best universal level. Such approaches are nevertheless limited by the actual availability of additional learning data, that can be costly to obtain. That is why Conneau

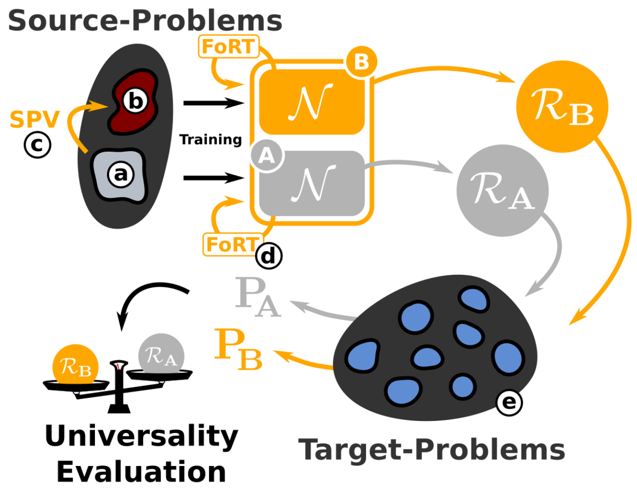


Fig. 1. A universal representation is a set of features learned on a source-problem (SP), that exhibits good performances when it is transferred on many target-problems (TPs). A representation $\mathcal{R}_A$ is obtained by a method Ⓐ that consists of training a network $\mathcal{N}$ on a SP ⓐ. When it is transferred on a large set of TPs ⓔ, it leads to an aggregated performance P_A. We propose to build a more universal representation $\mathcal{R}_B$ by: (i) applying a *source-problem variation* (SPV) ⓒ to transform an initial SP ⓐ into a new one ⓑ (ii) leaning features on (b) with the same method (yellow $\mathcal{N}$) and combining the resulting features with those of Ⓐ (iii) re-training both networks on their initial SP through ⓓ, a method named *Focused ReTraining* (FoRT). From the evaluation on the multiple TPs: $P_B > P_A$, thus the resulting representation $\mathcal{R}_B$ is more universal than $\mathcal{R}_A$.

et al. [5], [8] addressed a more strict problem and proposed to find the best algorithm able to increase the universality from a fixed set of data (tasks and domains). Addressing this question, we argue that training-data can be structured in a way that allows different tasks to learn more or better features with the same learning algorithm.

The overall contribution of this article consists of a method that increases universality from a fixed set of data (Fig. 1). In the vein of MuCale [9], we proposed MuLDiP-Net that automatically varies an initial source-problem (SP) and learn new features

• *Y. Tamaazousti, is at the CSAIL of MIT, USA. E-mail: ytamaaz@mit.edu*
• *H. Le Borgne, M.E.A. Seddik and M. Tamaazousti are at the CEA, LIST, France. E-mail: firstname.lastname@cea.fr*
• *C. Hudelot is at the MICS laboratory of CentraleSupélec (University of Paris-Saclay). E-mail: celine.hudelot@centralesupelec.fr*

Manuscript received September 2, 2018

on it, before combining all of them to form a more universal representation. MulDiP-Net does not need any additional images or annotations. Regarding the variation, we propose to start from a set of categories (*e.g.*, *rottweiler*, *pit-bull*) and group them in more generic ones (*e.g.*, *dog*), according to the upper-levels of a given hierarchy. The proposed formalism is quite general and makes MuCale a particular case of our method, that groups categories according to categorical-levels only [10]. We also introduce *Focused ReTraining* (FoRT), a new method that increases the universality using a principle of retraining based on fine-tuning [11]. By construction, it does not need any additional data, nor supplementary parameters. FoRT can also be used to reduce the dimension of representations. Combined to MulDiP-Net, it thus improves the performances while reducing the network-capacity.

We also address the problem of universality evaluation. Rebuffi *et al.* [12] proposed *Visual Decathlon Challenge* (VDC) to measure universality on multiple domains. They also proposed a metric that coherently aggregates the scores obtained on the multiple domains. However, their evaluation is restricted to a classical end-to-end scheme, that learns a representation and evaluates on the same problem. It thus does not completely match with the cognitive definition of universal representations proposed by Atkinson [1], that implies that the representation of the visual world that is learned by humans in the early years of their development is then re-used later in life to solve many other kinds of problems. Since this definition is closer to the transfer-learning (TL) [13] scheme than the end-to-end learning one, we propose to evaluate universality in a TL scheme. Moreover, we inquire about the desirable criteria to evaluate the universality, and propose new metrics that seem more relevant than existing ones.

The present manuscript differs in many ways from our previous work on this topic [9], including the following new contributions: (1) a new universalizing method that does not require additive data nor parameter (FoRT, Sec. 3); (2) a more general formalism of the MulDiP-Net approach based on the principle of source-problem variation by grouping (Sec. 4), allowing the use of multiple variations and thus making [9] a special case of our approach; (3) the usage of FoRT to reduce the representation dimension of MulDiP-Net (Sec. 4.4), making this later more performing while significantly reducing its network-capacity; (4) the evaluation of universality in a TL scheme and the study of desirable criteria for universality evaluation as well as the proposition of new metrics (Sec. 5); and (5) more extensive and detailed experimental results, including a comparison to the state-of-the-art based on ten available target-problems (Sec. 6), and an ablation study of our approach (Sec. 6.3).

2 Related Work

2.1 Learning Universal Representations

With UberNet, Kokkinos [2] considered a unified architecture that is trained end-to-end to tackle several vision tasks (universal model) and addressed resulting technical challenges. In the same vein, Subramanian *et al.* [6] used multiple datasets with different NLP tasks and proposed to find which of the available multi-task learning algorithms and set of tasks are the best to learn universal representations. Alternatively, Bilen and Vedaldi [4] rather considered that a universal representation should be able to address multiple visual domains. They proposed to learn a compact representation able to perceive a wide set of domains, using scaling parameters to learn the statistics of each. It has been extended in [12], [14] by respectively using sequential and parallel residual adapters. All these works use more data, and it is undoubtedly the most straightforward way towards universality, but it is limited by the availability of more annotated data. It is thus interesting to improve universality from a fixed set of data, which is our goal as well as that of Conneau *et al.* [5], [8]. However, in contrast to their approach that seeks for the best task and algorithm for universality, ours change the data to automatically provide different tasks to learn more universal features.

All the above works consider multiple source-problems (tasks [6], domains [12] or problems with same images but different categories) to learn the universal representation, and use a single network to solve a multi-task, multi-label or recursive [3] learning) problem. In this paper we propose to rely on multiple networks through independent learning, the advantages and drawbacks of these methods being detailed in Sec. 2.2.2.

2.2 Approaches for Universalizing Methods

Many works can be considered as universalizing methods, in the sense that they diversify and increase the amount of feature detectors. We propose a unified view of these approaches, using the notion of Source-Problem Variation (SPV, Sec. 4.1).

2.2.1 Learning one Network on a Modified SP

To diversify an initial dataset, a first approach consists of adding new categories and corresponding annotated images. Three kinds of categories are added: specific [4], [13], [15] (*e.g.* rottweiler), generic [16], [17], [18] (*e.g.* dog) and noisy [19], [20]. In some cases, the added categories contain data from multiple domains [4]. These methods slightly increase the capacity of the network by adding parameters specific to each domain. In all cases, these approaches can be quite powerful to increase the universality, but at a high cost since they require many additional data and corresponding annotations. Another limitation lies in the learning methodology. Indeed, the network is often trained jointly on the generic and specific data, resulting into a mix of generic and specific features in the intermediate layers. Since a softmax loss is often used, it considers generic and specific categories as mutually exclusive, that is not the case. At the opposite, we propose to group the categories according to their level and separately learn the features on each to respect the real-world semantics. Alternatives to this proposal are the works of [21], [22] that respectively group hierarchically or by clustering. However (as shown in the experiments), for the sake of universality, it is preferable to group at categorical-levels as we propose.

2.2.2 Learning a Set of Networks on a Set of SPs

The methods of the second approach [23], [24], [25], [26], [27], [28] tackle the problem of undesirably mixed features in joint training. For this, they use an ensemble-model on different source problems and a sequential training procedure. They train a network on an initial problem with either specific or generic categories, then fine-tune it on another problem or on a set of smaller problems. Because of the sequential learning procedure, these approaches are subject to catastrophic forgetting [29], in which features learned on the first task are changed thus lost to be relevant for the following tasks. Hence, they usually need many models (more than 10) to get a significant diversity in the set of features, which is very costly. We propose to answer to this limitation by learning an independent network per problem.

To diversify the source problem, all the mentioned methods [23], [24], [26], [28] re-label specific categories into non-semantic ones, *i.e.*, categories that do not exist in the real world [22]. These non-semantic categories result from a hierarchical clustering based on visual low/mid-level features. These low/mid-level features learned by a network trained on the initial SP, can lead to irrelevant categories when the features fail to capture the dissimilarity between different categories. The non-semantic categories are more general than the specific ones, but do not match to an existing semantics. In a universality perspective, it is better to rely on a grouping process that uses an explicit human categorization expertise in order to reflect semantically-real relations between categories [10], [30].

2.3 Universality Evaluation

Inspired by studies on the visual brain (claim of [1]), authors of [4], [12], [14] evaluated universality of representations as their ability to simultaneously cover a large range of visual domains. Their evaluation consists of learning and testing on the same problem and only the visual domain differs. Moreover, since many problems are considered, they proposed a metric to aggregate the scores in each task, that respects their proposed criterion of significance. While such criterion is undoubtedly useful for universality evaluation, it should be noted that their classical evaluation framework does not completely match with [1]. In contrast to our transfer-learning scheme, their scheme do not consider the terms "re-used later in life" which means that multiple and different problems should be solved by a universal representation. Moreover, in addition to their criterion of significance and the coherent aggregation considered in [6], we proposed a total of six additional criteria that we judged as important for universality evaluation, as well as two metrics that respect almost all the criteria. Nevertheless, while such TL scheme has been already used in the NLP community [5], [6], [8], [31] for universal representations, to the best of our knowledge, we are the first to propose it in the vision community and more importantly, to link it to the claim of [1].

2.4 Cognitive Studies in Computer Vision

To go deeper in the link between computer-vision and cognitive studies, our proposed work is also in line with the use of knowledge on human categorization in computer-vision [32], [33], [34], [35]. Generally, the goal of these works is to output basic-level concepts of images from predicted finer ones. An exception is [35], that is closest to ours since they consider categorical-levels in their representation. Similarly to our work, they consider that even if Humans tend to categorize objects at the subordinate-level, they are still aware of the other categorical-levels [10]. They focus on the low scores of generic classes that is due to their high intra-class variance and fix the balance with the specific classes in a semantic representation [36], [37]. In contrast, we address the problem at the representation level, by modifying the annotated data used to learn the network.

3 Focused ReTraining

We propose a new method that *re-trains* a neural network *on the same problems*, and thus does not need more data [13], [17], [18], nor additive parameters [9], [23], [38], [39]. Our approach relates on the work of [11] who proposes an extensive study of the effect of different *self-training* methods (*i.e.*, re-training a neural network on the same problem it was trained originally). They explored two learning-strategies: (i) frozen-based re-training named Frozen Re-Training (**FrozenRT**) in the following and (ii) fine-tuning based re-training named ReTraining by Fine-Tuning (**RT-FT**). More precisely, both methods re-train an initial network **initNet** with (θ_1^a, θ_2^a) as *trained* weights. Both methods have three stages: (i) the weights of FrozenRT and RT-FT are divided into two sets (θ_1^b and θ_2^b), with θ_1^b containing the weights of the first L layers and θ_2^b, the weights of the last ones (ii) the two sets of weights are initialized differently – in FrozenRT and RT-FT, the first layers are initialized with the weights of the pre-trained initNet and the last layers are initialized randomly –; and finally (iii) the re-training of the weights – FrozenRT retrains only last layers and "freezes" the first ones, while RT-FT retrains them all with the *same* learning-rate. Their extensive study leads to some interesting conclusions that motivate our approach. In particular, they showed that: (i) FrozenRT hurts the performances of initNet because "initNet contained fragile co-adapted features on successive layers", which means that features interact with each other during learning and that co-adaptation can not be relearned by the randomly initialized upper layers alone and (ii) RT-FT slightly increases the performance because "it aims to recover co-adapted features that were trained by the initNet". Simply said, it is important to preserve some knowledge acquired during the learning of the original network, but it is a sub-optimal to *completely* focus the training on the last layers. As a consequence, we propose a new method named **Fo**cused **ReT**raining (FoRT) that can be seen as an hybrid versi noof the two previous ones. As in [11], the re-training principle consists of dividing the weights learned on the initNet into two sets, initializing them differently (first with the pre-trained initNet, others randomly), then jointly minimizing them but with *different* learning-rates for the firs. layets than for thr laste An illustration is given in Fig. 2.

More formally, let us consider a source training database $\mathcal{D}^{\mathcal{S}}$ containing N images x_i with their associated labels y_i. Let us also consider a network $\mathcal{N}$ that was trained on $\mathcal{D}^{\mathcal{S}}$ by minimizing a loss function $\mathcal{L}$ with a certain optimization algorithm. Let us note $\Theta^a{=}(\theta_1^a, \theta_2^a)$ being its set of *trained* weights split into two sets, namely θ_1^a that contains the weights of the first L layers and θ_2^a that contains those of the remaining layers. FoRT re-trains the network $\mathcal{N}$ by minimizing the same loss-function $\mathcal{L}$ as initNet, on the same training database $\mathcal{D}^{\mathcal{S}}$, which is expressed by:

$$\arg\min_{\Theta^b} \mathcal{L}((\Psi^b(\mathbf{x}, \Theta^b = (\theta_1^b, \theta_2^b)); \mathbf{y}), \tag{1}$$

where $\Psi^b(\mathbf{x}, \Theta^b)$ is the predicted probability vector for images $\mathbf{x}$ using learnable weights $\Theta^b{=}(\theta_1^b, \theta_2^b)$. The set of weights θ_1^b is initialized with those of the first layers of initNet ($\theta_1^b{=}\theta_1^a$) and θ_2^b is initialized randomly. With a standard stochastic gradient descent (SGD), the individual weights w_{ij}^b, that form the full sets θ_j^b (with $j \in \{1, 2\}$), are updated through:

$$w_{ij}^b \leftarrow w_{ij}^b - \eta_j \frac{\partial \mathcal{L}}{\partial w_{ij}^b}, \forall w_{ij}^b \in \theta_j^b, \tag{2}$$

where, η_j respectively corresponds to the learning-rate of the first layers (when $j{=}1$) and last ones (when $j{=}2$). More specifically, $\eta_1{=}\alpha{\times}\eta_2$, with $\alpha \in [0, 1]$ is a parameter that can be set by cross-validation. In practice, the SGD can be speed-up with a momentum, the learning rate being smoothed as usual. To summarize, our FoRT method *preserves* some knowledge acquired

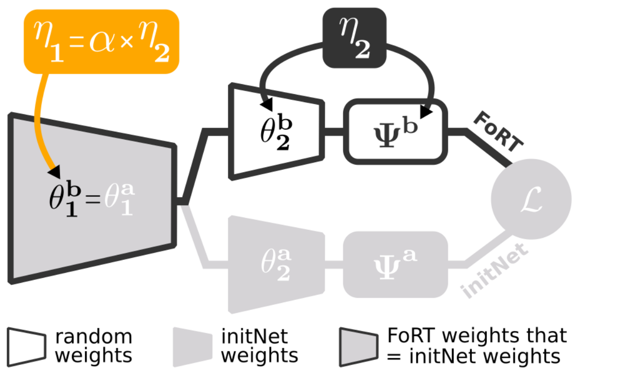


Fig. 2. Focused ReTraining (FoRT). An initial network (gray branch) denoted initNet (first layer's weights denoted θ_1^a, last ones θ_2^a and the classifier Ψ^a) is trained on a source-database, by minimizing the loss-function $\mathcal{L}$. Once initNet trained, another network (black branch) denoted FoRT (first layer's weights denoted θ_1^b, last ones θ_2^b and the classifier Ψ^b) is trained on the *same* source-database and solves the same problem (minimize same loss $\mathcal{L}$). The weights θ_1^b (black block filled in gray) of the first layers of FoRT are initialized with the firsts of initNet (θ_1^a) and *weakly* updated through training. The weights of the last layers θ_2^b (black blocks filled in white) are randomly initialized and *fully* trained.

during the learning of the initNet (by initializing the weights of its first layers, with the firsts of initNet). It also *focuses* the training on the last layers (by completely training the last layers, while allowing, through factor α, some slight change of the first ones).

4 Multi Discriminative-Problem Network

The previous method does not add data nor network capacity but always solves the same problem. In this section, we propose another universalizing method that combines different but complementary features learned on different problems. We introduce the principle of source problem variation (SPV) (Sec. 4.1) that defines new source problems (SPs), with a special emphasis on variation with *grouping* (Sec. 4.2). A network is trained on each of these new problems, leading to a set of representations that are combined to obtain a more universal one (Sec. 4.3). It can be combined to the FoRT app orchaof the previous section to reduce the size of the representation (Sec. 4.4). This "Multi Discriminative Problem Network" (MulDiP-Net) approach is illustrated in Fig. 3.

4.1 SPV: Source-Problem Variation

A source problem $\mathcal{D}_k^{\mathcal{S}} = \{(x_i^k, y_i^k)\}_{i=[\![1,N_k]\!]}$, consists of a set of N_k pairs of training image x_i^k and associated label y_i^k. When a CNN is trained on this source problem (SP), it learns features that discriminate specifically between the images of the considered categories $\mathcal{Y}_k = \{c_1^k, \dots, c_{C_k}^k\}$. For example, Zhou *et al.* [15] showed that a CNN trained on scenes learns object detectors while those trained on objects learn object-part detectors. Thus, a change in the source problem diversifies the features that are learned, and is thus likely to be used to improve the universality of .ht leaenedrr presentatione Based on this idea, we propose the principle of *Source Problem Variation* (SPV) that transforms an initial source problem $\mathcal{D}_0^{\mathcal{S}}$ into a new one $\mathcal{D}_{k,k>0}^{\mathcal{S}}$. Obviously, both SPs must differ by at least one element ($\exists (x_i^k, y_i^k)$ s.t. $x_i^k \neq x_j^0$ or $y_i^k \neq y_j^0$) but they must also have some common content ($\exists (x_i^k, y_i^k)$ s.t. $x_i^k = x_j^0$ or $y_i^k = y_j^0$). There are many possible variations, that can act on the images and/or the labels. We particularly consider three types of variations: (i) *adding* images or categories to the initial SP; (ii) *splitting* the initial SP ; (iii) *grouping* categories of the initial SP (Fig. 4).

The SPV principle, although not explicitly described as is, has already been applied in the literature. For instance, variations on the image set while preserving the set of categories was proposed in [40], [41]; *adding* categories was used by [4], [12], [14], [42] (adding specific categories) and [17], [18] (adding generic categories); *splitting* variations on the set of categories was proposed by [13], [23], [24], [27], [28] and finally, variation of the categories by grouping was explored in [21], [22], [43].

4.2 SPV Based on Grouping

Grouping SPV consists of grouping the images of different categories. Contrary to an "adding SVP" it does not need more annotated data. In comparison to the "splitting SPV" it does not decrease the number of images per category thus does not penalize the training of the network. One can consider several principles to group the categories, the most straightforward being randomly or based on clustering. In the considered context where one wants to classify images according to human labelling, we propose to directly benefit from the knowledge on human categorization. Such a knowledge is often represented in the form of hierarchies with hyponymy relations (*i.e.*, a set of categories organized according to "is-a" relations), such as ImageNet [44] or WordNet. We focus on a particular semantic knowledge, namely **categorical-levels** [10], [30], [45] that is a hierarchy of categories reflecting how humans categorize objects. Our approach can nevertheless be applied to *hierarchical*-levels or *clustering*-levels, as explained below.

Such a hierarchy is a directed acyclic graph $\mathcal{H} = (\mathcal{V}, E)$ consisting of a set $\mathcal{V}$ of nodes and directed edges $E \subseteq \mathcal{V} \times \mathcal{V}$. Each node $v \in \mathcal{V}$ is a label and an edge $(v_i, v_j) \in E$ reflects that label v_i subsumes label v_j. We consider the functions $\mathcal{D}_{\mathcal{H}}(v)$ and $\mathcal{A}_{\mathcal{H}}(v)$ that give respectively the sets of descendents and ascendants of a node in $\mathcal{H}$, assuming that $\mathcal{D}_{\mathcal{H}}(v_{leaf}) = v_{leaf}$ and $\mathcal{A}_{\mathcal{H}}(v_{root}) = v_{root}$. Let us consider an initial source-problem $\mathcal{D}_0^{\mathcal{S}}$ with *specific* categories $\mathcal{C}_0^{\mathcal{S}} = \{c_1^0, c_2^0, \dots, c_{C_0}^0\} \subset \mathcal{V}$. We note $\mathcal{B}_L^{cat}$ a set of categories that are at a given categorical level L (*subordinate* level for L=0, *basic* for L=1 and *superordinate* for L=2). Crucially, the categories of $\mathcal{B}_L^{cat}$ do *not* correspond to a given level of the hierarchy $\mathcal{H}$, but it exists a mapping from any of its element $c_i^{cat_L}$ to a node of $\mathcal{H}$. *Grouping* the categories consists in partitioning $\mathcal{C}_0^{\mathcal{S}}$ into G subsets. To do so, we define a partitioning function according to $\mathcal{B}_L^{cat}$ (cardinal G) as:

$$\begin{aligned} P_{cat_L}: \quad & \mathcal{V} && \to 2^{\mathcal{C}_0^{\mathcal{S}}} \\ & c_i^{cat_L} && \mapsto \mathcal{C}_0^{\mathcal{S}} \cap D_{\mathcal{H}}\left(c_i^{cat_L}\right), \end{aligned} \tag{3}$$

Using this function, we obtain G generic categories, with $G \ll C_0$. We define a re-labeling function relative to the set $\mathcal{B}_L^{cat}$ as:

$$\begin{aligned} R_{\mathcal{B}_L^{cat}}: \quad & 2^{\mathcal{C}_0^{\mathcal{S}}} && \to \mathcal{B}_L^{cat} \\ & \mathcal{C}_i && \mapsto \mathcal{B}_L^{cat} \cap \mathcal{A}_{\mathcal{H}}\left(LCA_{\mathcal{H}}\left(\mathcal{C}_i\right)\right), \end{aligned} \tag{4}$$

where $LCA_{\mathcal{H}}(\mathcal{C}_i)$ is the least common ancestor of the categories in the set $\mathcal{C}_i$. To summarize, while Eq. (3) partitions the set of specific categories into "unnamed" generic categories (*i.e.*, do not have association to a humanly understandable word), Eq. (4) re-labels them into existing (and thus "named") categories at the categorical-level L. All the images are re-labeled according to the initial categories they belong to. The grouping SPV ϑ_G we propose is thus the combination of the partitioning and re-labeling functions (Figure 5).

The proposed process can also be applied to re-label into generic categories belonging to $\mathcal{B}_L^h$ at a given hierarchical-levels,

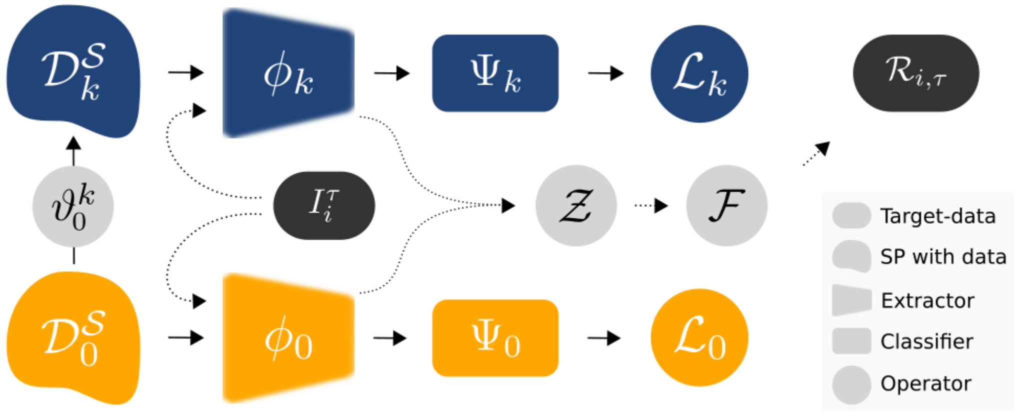


Fig. 3. An *initial* source problem (SP) $\mathcal{D}_0^{\mathcal{S}}$ contains a set of images and their associated labels $(x_i^0, y_i^0)_{i\in[\![1,N]\!]}$. MulDiP-Net has three steps: (i) variation (ϑ) of the initial SP ($\mathcal{D}_0$) into new ones ($\mathcal{D}_k$); (ii) training networks (ϕ and Ψ) on the whole set of SPs ($\{D_0, D_k\}$); and (iii) normalization ($\mathcal{Z}$) followed by combination ($\mathcal{F}$) of the features extracted from each trained network in order to form a more universal representation. Phase (i) is a source problem variation (SPV) (ϑ_0^k) applied on the initial SP ($\mathcal{D}_0^{\mathcal{S}}$), which outputs a new SP ($\mathcal{D}_{k,k>0}^{\mathcal{S}}$). After applying K SPV functions, we get a set of K+1 SPs containing the new SPs and the initial one (only one variation illustrated here, thus K=1). Phase (ii) consists of learning one network for each of the K+1 SPs, resulting in a set of K+1 trained networks: $\{\mathcal{N}_k\}_{k=[\![0,K]\!]}$. Each network $\mathcal{N}_k$ (that is a composition of a features-extractor ϕ_k and a classifier Ψ_k) is trained by minimizing the loss function $\mathcal{L}_k$ computed using the output of the predictor Ψ_k and the ground-truth of the SP $\mathcal{D}_k^{\mathcal{S}}$. Phase (iii) consists of extracting the more universal representations $R_{i,\tau}$ of the images I_i^τ of a target task τ, obtained through an independent normalization ($\mathcal{Z}$) and merging ($\mathcal{F}$) of the features computed by the extractor ϕ_k.

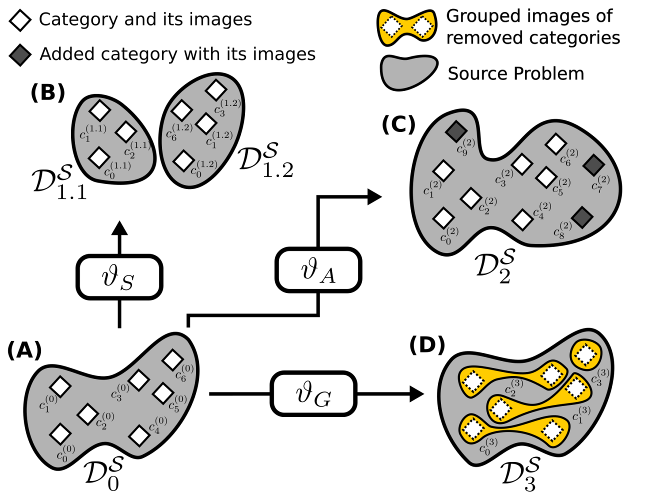

Category and its images
Added category with its images
Grouped images of removed categories
Source Problem
(B)
$\mathcal{D}_{1.1}^{\mathcal{S}}$
$\mathcal{D}_{1.2}^{\mathcal{S}}$
(C)
$\mathcal{D}_2^{\mathcal{S}}$
ϑ_S
ϑ_A
(A)
$\mathcal{D}_0^{\mathcal{S}}$
ϑ_G
(D)
$\mathcal{D}_3^{\mathcal{S}}$

Fig. 4. Illustration of three types of SPV: *Splitting*, *Adding* and *Grouping*. An initial source problem (SP) $\mathcal{D}_0^{\mathcal{S}}$ is illustrated in (A). (B) is the output of the splitting SPV (ϑ_S) which results in two *smaller* sets of training data. In contrast, in (C), an *adding* SPV (ϑ_A) results in a *larger* set of training data (more images and categories). The grouping SPV (ϑ_G) in (D) results in the *same amount* of training-data (*same* images but labeled according *less* categories).

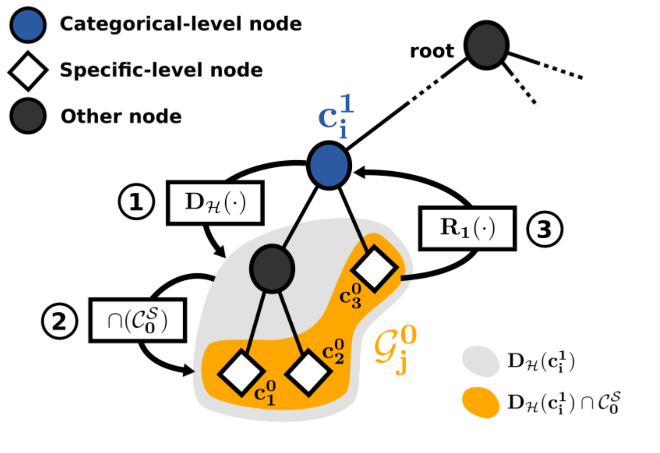

Categorical-level node
Specific-level node
Other node
root
$\mathbf{c_i^1}$
① $\mathbf{D}_{\mathcal{H}}(\cdot)$
② $\cap(\mathcal{C}_0^{\mathcal{S}})$
$\mathbf{R_1}(\cdot)$ ③
$\mathbf{c_3^0}$
$\mathbf{c_2^0}$
$\mathbf{c_1^0}$
$\mathcal{G}_\mathbf{j}^\mathbf{0}$
$\mathbf{D}_{\mathcal{H}}(\mathbf{c_i^1})$
$\mathbf{D}_{\mathcal{H}}(\mathbf{c_i^1}) \cap \mathcal{C}_0^{\mathcal{S}}$

Fig. 5. Illustration of our grouping SPV. Given a set of specific categories (leaf nodes of the hierarchy $\mathcal{H}$ in white diamonds) and a set of generic categories (here c_i^1) at a certain level (here categorical denoted $\mathcal{B}_L^{cat}$), our grouping SPV ϑ_G consists of three steps: computation of all descendants of c_i^1 according to the $\mathcal{H}$ (1); computation of the descendants that belong to the initial set of categories (2), producing a group $\mathcal{G}_j^0$; and relabeling of the categories (as well as their images) of the later group (3) into the categories of $\mathcal{B}_L^{cat}$. The first two points correspond to Eq. (3), while the last one corresponds to Eq. (4).

as well as to a hierarchy obtained through clustering, based on data (see Sec. 6.1).

4.3 MulDiP-Net Training and Representation

Applying the above process on $\mathcal{D}_0^{\mathcal{S}}$, we obtain a set of SPs $\mathcal{D}_\Omega^{\mathcal{S}} = \{\mathcal{D}_0^{\mathcal{S}}, \mathcal{D}_1^{\mathcal{S}}, \cdots, \mathcal{D}_\omega^{\mathcal{S}}\}$. Each SP $\mathcal{D}_k^{\mathcal{S}} = \{(x_i^k, y_i^k)\}_{i\in[\![0,N_k]\!]}$ has N_k images x_i^k labeled among C_k categories c_j^k. MulDiP-Net consists of training one network per SP, with a network architecture $\mathcal{N}_k$[1]. We can consider different architectures for each ω SPs but we limit our study to the special case where they are all the same. Our method does not depend on a particular architecture, thus it can benefit from the advances in this domain. In practice we consider the usual CNN architectures AlexNet [16], VGG [38] and DarkNet [46]. We initialize the weights randomly with a Gaussian distribution, and optimize a softmax loss with stochastic gradient descent (SGD). We have as many cost functions to minimize as the number of SPs in $\mathcal{D}_\Omega^{\mathcal{S}}$ and each cost function $\mathcal{L}_k(\Theta)$ is minimized independently. Since the SPV functions are based on grouping, all the SPs contain the same images as the initial one thus each cost function is minimized on the same set of training images, but with different labels. There is thus no cost in terms of additional data, while the cost to annotate is dramatically reduced since it is reported on each category and not each image. On ILSVRC for instance, it means $1,000$ annotations instead of 1.2 million. At convergence, MulDip-Net is thus $\omega+1$ trained networks $\mathcal{N}_\Omega^* = \{\mathcal{N}_0^*, \ldots, \mathcal{N}_\omega^*\}$.

Let us consider an image I_i^τ of a target task τ. To obtain its representation $\mathcal{R}_{i,\tau}^\Omega$, we extract the features of I_i^τ with the K first layers of each subnetwork, resulting into $\omega+1$ feature vectors

1. In all the documents, $\mathcal{N}$ indicates the architecture of a network, while $\mathcal{N}^*$ indicates a trained network.

$\phi_k^K(I_i^\tau)$. If the K^{th} layer is convolutional, the tensors are flattened or pooled to get some vectors. These vectors are supposed to carry different and complementary pieces of information. However, the features learned on specific data tend to fire with higher values than those learned on generic data. An unnormalized concatenation of these representations would thus result into an unbalanced representation, dominated by the specific representation, that is detrimental to the performances. Each feature vector is thus normalized with an operator $\mathcal{Z}$. All of them are then combined through an operator $\mathcal{F}$. Formally, the MulDiP-Net representation for the image I_i^τ is computed as (Fig. 3):

$$\mathcal{R}_{i,\tau}^{\Omega} = \mathcal{F}_{k \in [\![0,\omega]\!]}\left(\mathcal{Z}(\phi_k^K(I_i^\tau))\right), \tag{5}$$

In practice we normalize with the L_∞ norm then concatenate all the vectors. The independent normalization step is crucial, because it homogenizes the scales of the sub-representations. The same problem of dominating values has been observed in [39], [47], who solved it with a similar normalization process.

4.4 FoRT as a Dimensionality Reduction Method

The more SPs we consider in MulDiP-Net, the more universal representation we get. However, because of its concatenation fusion, the dimension of the fused representation linearly increases with the amount of SPs considered. Also, MulDiP-Net is an ensemble-like method since it contains as many networks as the number of SP considered. As a consequence, MulDiP-Net contains ω+1 *times* more parameters than a standard network. To alleviate this drawback, we proposed to rely on the FoRT method (Sec. 3) to reduce the dimension of the representation. To do so, we re-train each subnetwork of MulDiP-Net on the same initial SP with less neurons on last layers, on which FoRT actually re-trains. Compared to the classical FoRT, the "FoRT as a Dimensionality Reduction method" (**FoRT-DR**) results in a model that has a lower dimension and a limited loss of performance, but that significantly increases the performances compared to the initial network (see Sec. 6.3). Such a behavior is also observed when FoRT-DR is applied on each subnetwork of MulDiP-Net.

More formally, a feature-extractor ϕ_k trained on a certain SP $\mathcal{D}_k^{\mathcal{S}}$ with a network architecture $\mathcal{N}_k$ has a penultimate layer of dimension n. FoRT-DR re-trains the last layers of ϕ_k according to Eq. 2 on the same source-problem $\mathcal{D}_k^{\mathcal{S}}$ but with a different network-architecture $\mathcal{N}_k'$ that has a penultimate layer of dimension $n' < n$, resulting into a new features-extractor ϕ_k'. In the classical formulation of FoRT n'=n. By construction FoRT-DR reduces the number of parameters of the final network $\mathcal{N}_k'$. This is particularly the case when FoRT-DR is applied to fully-connected layers, that usually represent the majority of the parameters of the CNN. For instance, AlexNet has 62.35 millions parameters with 3.75 millions in the convolutional layers and 58.6 millions in the fully-connected ones (about 94% of the total). To apply FoRT-DR to MulDiP-Net, we consider a set $\mathcal{N}_\Omega^* = \{\mathcal{N}_0^*, \ldots, \mathcal{N}_\omega^*\}$ of ω+1 trained networks. FoRT-DR is applied to each subnetwork $\mathcal{N}_{k \in [\![0,\omega]\!]}^*$ by setting the last layer of ϕ_k' to be of size $n' = \lceil n/(\omega+1) \rceil$, with n the dimension of the original representation ϕ_k. The rest of the pipeline is similar to MulDiP-Net.

5 Evaluation of Universalizing Methods

Inspired by Atkinson [1], several authors evaluated the universality of representations by their ability to simultaneously cover a large range of domains, like objects, faces, animals, etc. [4]. However, their evaluation scheme does not completely match with [1] since learning and testing are conducted on the same problem, while in [1], the terms "re-used later in life" mean that universality implies to work well on many but different problems than those used to learn the representation. We thus propose to consider a transfer-learning (TL) scheme that naturally fits with it. Indeed, in TL, the source-task is used to learn the representation (the universal one developed in the early years) and the target-tasks are used to evaluate it (the problems solved later in life). To better match the claim of [1] (re-use "as-is"), we propose to *not* modify the representation (no fine-tuning) for each target-task, but rather learn a simple task-predictor on top of it. Indeed, when humans develop their representation, they do not have access to future problems, thus the target tasks should not be available during the learning of the representation. In that sense, our evaluation framework based on TL is closer to [1] than that of [4] which lies in a classical end-to-end learning scheme for all target tasks.

Criterion	Avg	RG [6]	VDC [12]	BC	mNRG
Coherent aggr.		✓	✓	✓	✓
Significance			✓		
Merit bonus			✓		✓
Penalty malus					✓
Penalty for damage	✓	✓		✓	✓
Indep. to outliers				✓	✓
Indep. to reference	✓			✓	
Time consistency	✓	✓	✓		✓

TABLE 1
Comparison of universality evaluation-metrics. The criteria are detailed in Sec. 5, highlighted in bold.

Evaluating universality requires to aggregate the scores of a method on multiple target-problems. This step is not straightforward, since metrics of each problem could be different (*e.g.*, error-rate, mAP, F1-score) or even be increasing (*e.g.*, recall) or decreasing (*e.g.*, median rank). Thus, naively averaging the scores would result into incoherences. To get **coherent-aggregation**, Subramanian *et al.* [6] proposed to average the gain over the scores of a reference universal representation across the set of multiple problems: relative gain (**RG**) computed as: $U_i^{RG} = \frac{1}{P}\sum_{j=1}^{P} s_j^{ref} - s_j^i$, with s_j^{ref} and s_j^i being respectively the performance of a reference method M_{ref} and the method to evaluate M_i on the problem P_j. Such aggregation is coherent, but **depends on a reference method** that needs to be arbitrarily and manually chosen. Rebuffi *et al.* [12] went further by identifying one additional criterion for universality: the metric should better rewards significant gain compared to the reference scores (called **significance** criterion). To do so, they proposed the Visual Decathlon Challenge (**VDC**), which is computed as: $U_i^{VDC} = \sum_{j=1}^{P} \alpha_j \max\{0, E_j^{max} - E_j\}^{\gamma_j}$, with E_j^{max} and E_j respectively being the error-rate of the actual method M_i and the best reachable one on the problem P_j, α_j is just a normalizing factor, and $\gamma_j \geqslant 1$ models the significance.

In addition to the properties already put on evidence, we introduce four more criteria. The **merit-bonus** indicates that the metric should reward proportionally with the score of the reference method. For instance an improvement of 1% is more rewarding if it starts from a reference at 90% than one at 10%, because it is easier to improve a baseline when it has low performances.

Moreover, improving a high reference consists of addressing the hardest cases. The **penalty for damage** reflects the fact that a method could improve the performance on some tasks but decrease it on others. In such a case, the metric should penalize them. The **penalty malus** lies with the fact that the aggregated improvement (over the reference) of some methods could be lower than the aggregated decrease. For example, with VDC negative values are not allowed (because of the max), thus it does not respect the penalty for damage nor the penalty malus. In addition, the significance criterion is modeled through a power γ, which not only prevents the merit-bonus, but even exhibits the inverse behavior. Indeed, VDC will reward more a method that gains 3% starting from a reference at 10%, than one that gains 2.5% starting from 90%. It thus favours large relative gain, while we consider it is better to reward method that address hardest cases. Last, the metric should be **independent to outlier methods**, in the sense that it should detect methods that are well suited to a given benchmark (gain a lot of points) that could compensate (when aggregated) a insignificant gain on the other benchmarks.

To respect these constraints, we propose the median normalized relative gain (**mNRG**): $U_i^{mNRG} = \underset{j \in [\![1,P]\!]}{\text{median}}(s_j - s_j^{ref})/(s_j^{max} - s_j^{ref})$. The numerator is the simple RG metric, thus it respects the same criteria (coherent aggregation and penalty for damage because negative scores are allowed). The denominator acts as a normalization and naturally handles the merit-bonus and penalty malus criteria. Finally, instead of an average to aggregate, we used the median to make our metric independent to outliers.

Finally, none of the above metrics is independent to a reference method, thus we propose an alternative metric based on Borda Count (**BC**), which is a voting method that uses an ordinal scale, with each benchmark considered as an independent voter. Each voter can use its own measure to estimate the performance of the methods, which provides coherent aggregation. Second, it becomes insensitive to some outliers such as the exceptional fit of a method to a particular benchmark. Moreover, we consider that reporting the number of times a method M_1 is better than a method M_2 can be a more reliable information than the average difference of scores (as long as these score are actually comparable). Borda Count lacks the **consistency with time** since its score differs with the chosen comparison methods. Formally, let us consider M methods to rank, relying on the information provided by P problems. Each method is then ranked according to each problem, resulting into a rank r_i^j with $M \in [\![1,i]\!]$ and $P \in [\![1,j]\!]$. It is converted into a score $M - r_i^j$ that is itself averaged to give the final score of the method: $S_i = \sum_{j=1}^{P} M - r_i^j$.

All the metrics are compared in Table 1 according to all the mentioned criteria. None of them address all the criteria but the proposed mNRG is the most complete of them.

6 Experimental Results

6.1 Experimental Settings

The evaluation of all the methods is conducted according to a transfer-learning scheme on 10 target-problems, that are described in detail in the supplementary material and summarised in Table 2. The methods are trained with the architectures AlexNet [16] (default), VGG [38] and DarkNet [46]), using SGD with a momentum of 0.9, a weight decay of 10^{-4} and minibatches of size 256. By default, we train on ILSVRC* because it is smaller than ILSVRC, making the process faster. We usually not use fine-tuning, this last being studied in Sec. 6.5. Once the representations extracted, we learn each class with a *one-vs-all* linear SVM classifier. The soft margin is optimized on each dataset through cross-validation with the usual train/val splits. The performances of each target-problem are evaluated with standard splittings and metrics, namely the mean Average Precision (mAP) for multi-label datasets and Accuracy (Acc.) for mono-label ones. To evaluate the universality, we use the proposed **mNRG** evaluation-metric (Sec. 5).

Datasets	(1)	(2)	(3)	(4)	(5)	(6)
ILSVRC*	objects	483	✗	569,000	48,299	Acc.
ILSVRC	objects	1K	✗	1.2M	50,000	Acc.
VOC07	objects	20	✓	5,011	4,952	mAP
VOC12	objects	20	✓	11,540	10,991	mAP
NWO	objects	31	✓	21,709	14,546	mAP
CA101	objects	102	✗	3,060	3,022	Acc.
CA256	objects	257	✗	15,420	15,187	Acc.
MIT67	scenes	67	✗	5,360	1,340	Acc.
stACT	actions	40	✗	4,000	5,532	Acc.
CUB	birds	200	✗	5,994	5,794	Acc.
stCA	cars	196	✗	8,144	8,041	Acc.
FLO	plants	102	✗	1,020	6,149	Acc.

TABLE 2
Source-datasets (top) and target datasets (bottom) used in this paper. The columns report: (1) images-domain; (2) number of categories; (3): multiple categories per image (✓) or not (✗); (4) number of training samples; (5): number of test samples; and (6) evaluation-metric (**Acc.** or **mAP**). See the supplementary material for further details.

For the SPV, using ILSVRC or ILSVRC* as initial source-problem, we consider three generic grouping methods: categorical, hierarchical and clustering. For the two first methods, we rely on the ImageNet hierarchy for the functions of partitioning (Eq. (3)) and re-labeling (Eq. (4)). A part of the categorical-level categories of ILSVRC is obtained from the list released in [44] and the other part is re-labeled by ourselves as depicted in [9]. It results into 480 generic categories for the $1,000$ specific ones of ILSVRC (200 generic for the 483 specific of ILVRC*). For the hierarchical-method, we follow the bottom-up approach of [21] and re-labeled the categories to higher levels of the ImageNet hierarchy. For the clustering method, we follow [22] and clustered the data of the categories with a Kmeans, setting K from 50 to 300 with steps of 50. Regarding the extraction and combination of the features (Eq. 5), we used one label-set per grouping SPV (*i.e.*, basic-level, 7^{th} hierarchical and the clustering with K=100). The later choice results in a set of four SPs (including the initial one). For the extraction of the representations of images of target-tasks, we always use the penultimate layer of each subnetwork. Regarding the normalization step before combining the representations, we used the L_∞ norm. In the FoRT methods, the settings are: L=6, meaning that we focus the retraining on the two last layers; η_2=10^{-2} as the learning rate used to train the original network; and α=0.1, meaning that we train the last layers 10 times faster than the first ones. For FoRT we also use the penultimate layer as a representation of the target images.

6.2 Comparison to the State-Of-The-Art

We compare the proposed FoRT and MulDiP+FoRT to state-of-the-art methods that could be used as universalizing methods:

- **REFERENCE**: CNN trained on the initial source-problem, that contains 483 specific categories. Since it is a classical

Method	VOC07	VOC12	CA101	CA256	NWO	MIT67	stACT	CUB	stCA	FLO	mNRG
	mAP	mAP	Acc.	Acc.	mAP	Acc.	Acc.	Acc.	Acc.	Acc.	
REFERENCE	66.8	67.3	71.1	53.2	52.5	36.0	44.3	36.1	14.4	50.5	0.0
SPV$_{\mathbf{A}}^{\mathbf{spe}}$ [4], [13], [15]	66.6	67.5	74.7	54.7	53.2	37.4	45.1	36.0	13.7	51.9	+1.5
SPV$_{\mathbf{G}}^{\mathbf{gen}}$ [16], [17], [18]	67.7	68.1	73.0	54.3	50.5	37.1	44.9	36.8	14.6	50.3	+1.4
AMECON [22]	61.1	62.1	58.7	40.6	45.8	24.3	32.7	26.1	13.1	36.4	-17.7
WhatMakes [21]	64.0	62.7	69.4	50.1	45.6	33.7	41.9	15.0	12.5	42.8	-7.5
ISM [27]	62.5	65.4	68.8	50.7	28.5	37.9	42.6	34.0	13.3	50.0	-4.3
GrowBrain-WA [39]	68.4	68.3	73.1	54.7	49.3	38.4	46.5	37.5	14.7	54.8	+3.5
GrowBrain-RWA [39]	69.1	69.0	74.8	55.9	50.4	40.0	48.4	38.6	14.8	56.1	+6.0
MuCaLe-Net [9]	69.5	69.8	76.0	56.8	**54.7**	41.3	48.5	35.6	15.7	54.8	+7.7
FoRT (Ours)	67.5	67.4	73.9	55.0	44.6	40.4	47.1	**38.7**	15.8	56.8	+4.0
MulDiP+FoRT (Ours)	**69.8**	**70.0**	**77.5**	**58.3**	47.9	**43.7**	**50.2**	37.4	**16.1**	**59.7**	**+9.8**

TABLE 3
Comparison of our methods (bottom) to **state-of-the-art universalizing-methods** (top). The comparison is carried in a transfer-learning scheme on ten target-datasets. Methods are compared in terms of their individual scores on each benchmark (with standard metrics) and especially their aggregated scores, with our nMRG (blue scores in last column). The universalizing methods are compared to a reference one for which we colored its scores in red. In each column, we highlight the highest score in bold, and the second one is underlined. All the methods have been learned with the same AlexNet architecture on the same initial SP (ILSVRC*).

method that works quite well on many problems, we use it as reference to evaluate universality of other methods.

- **SPV$_{\mathbf{A}}^{\mathbf{spe}}$** [4], [13], [15]: a network trained on 583 specific categories resulting from an *adding* SPV of 100 specific categories randomly obtained from the leaf nodes of ImageNet (100K images each).
- **SPV$_{\mathbf{A}}^{\mathbf{gen}}$** [16], [17], [18]: Same as the previous method but with a generic adding SPV that adds 100 generic categories (with their 100K images), obtained from random internal-nodes of the ImageNet hierarchy. It results in training a network on 583 specific and generic classes.
- **WhatMakes** [21]: A *grouping* SPV followed by the training of a network on the obtained SP. Specifically, the grouping SPV corresponds to a relabeling of specific categories into internal hierarchical-levels of the ImageNet hierarchy. We performed it for all the levels and report the results for the best one (6^{th} level starting from the leaf-node's one).
- **AMECON** [22]: Similar to the previous method but it differs by its kind of grouping SPV. Indeed, the grouping is performed by clustering. Specifically, all the images of each specific categories are used to compute the mean features (obtained through the $fc7$ layer of the pre-trained reference network) for the categories. Then, a Kmeans algorithm is used to cluster this set of category mean features. We applied this method with different amount of clusters (K from 50 to 300 with a step of 50) and report the best results (K=100).
- **ISM** [27]: This method trains an ensemble-model with N networks, one for each SP obtained from a splitting SPV. We split the initial SP in two balanced subsets. Once the networks are trained, we normalize and concatenate the features extracted from each of them.
- **GrowBrain-WA** [39]: This method performs fine-tuning of a pre-trained network on the source-problem it was trained originally, after growing the network capacity (wider or deeper). The best setting is the width augmented (WA) growing that add $2,048$ neurons to the $fc7$ layer. We implemented the proposed normalization and scaling step for the new and old layers, because they are crucial to make this method performing. The used representation is the $6,192$-dimensional $fc7$ layer. **GrowBrain-RWA** is an extended version that performs a recursive growing of the network capacity. The best setting reported is to add $1,024$ neurons on the $fc6$ layer and $2,048$ on the $fc7$ one.
- **MuCaLe-Net** [9]: it normalizes then concatenates the features extracted from two CNNs, one trained on data labeled according specific categories and one according categorical-levels. It results in a $8,192$-dimensional representation.

As depicted in Sec. 5, the methods are evaluated in terms of the proposed mNRG score, in a transfer-learning scheme on a set of ten target-datasets from different domains. The results of the comparison are presented in Table 3. As expected, the methods that add annotated images (SPV$_A^{spe}$ and SPV$_A^{gen}$) as well as those that increase the capacity of the network (GrowBrain) learn more universal representations than the reference method (positive mNRG score). Surprisingly, the ISM method is not very good. This might be due to the need for very large source-problems, and the half-million images used here are not sufficient. The low performances of AMECON may result from the specific categories we used (leaf node of ImageNet), that are not as specific as those they use (captions). Regarding WhatMakes, the original study showed that a network trained on data labeled among generic categories is almost as performing as one trained on specific categories. It was positively evaluated on target-tasks from three domains (general objects, actions and scenes), but our experiments tend to show it is no longer the case when one considers a larger scope, in particular fine-grained objects. Finally, we observe that our MulDiP+FoRT method significantly performs better than all other methods (highest mNRG score). It also significantly outperforms MuCaLe-Net (by 2 points of mNRG), showing the interest of combining MulDiP-Net with the proposed FoRT method.

We also remark that FoRT increases universality witout increasing the network capacity nor needing any additive annotation. It nevertheless outperforms the methods that add data and their annotations (SPV$_A^{spe}$ and SPV$_A^{gen}$) as well as one that increases the network-capacity (GrowBrain-WA).

6.3 Comparison to Baseline Methods

Here, we take further experiments to analyse the performances achieved by the proposed methods (MulDiP-Net and FoRT),

Method	VOC07 mAP	CA101 Acc.	CA256 Acc.	NWO mAP	MIT67 Acc.	stACT Acc.	CUB Acc.	FLO Acc.	mNRG
REFERENCE	66.8	71.1	53.2	52.5	36.0	44.3	36.1	50.5	0.0
Ensemble	67.8	72.2	54.5	52.0	37.2	45.0	34.7	51.8	+2.3
Multi-Task	61.5	61.8	45.4	49.4	30.7	36.4	25.6	38.7	-16.2
Multi-Label	44.7	46.8	26.4	25.1	27.2	28.0	15.2	38.1	-45.0
Recursive	65.3	68.6	50.8	52.4	33.4	50.8	29.4	45.5	-4.8
MulDiP-Net*	69.5	76.0	56.8	54.7	41.3	48.5	35.6	54.8	+7.9
FrST	62.3	64.3	47.3	50.0	30.9	38.4	29.3	41.1	-11.6
SFT	67.0	71.5	52.5	49.8	36.2	44.0	36.4	50.1	-0.1
FoRT$_{\text{DR}}$*	67.6	73.3	54.8	47.2	37.9	46.9	38.2	56.3	+3.4
FoRT*	67.5	73.9	55.0	44.6	40.4	47.1	**38.7**	56.8	+4.5
MulDiP+FoRT$_{\text{DR}}$*	**69.8**	76.6	57.3	**49.9**	41.6	48.7	38.0	59.2	+8.8
MulDiP+FoRT*	**69.8**	**77.5**	**58.3**	47.9	**43.7**	50.2	37.4	**59.7**	**+10.7**

TABLE 4
Comparison of our universalizing-methods to several **baselines**, in terms of mNRG (last column in blue). All the methods have been learned with the same AlexNet architecture on the same initial SP (ILSVRC*). Methods marked with * are ours.

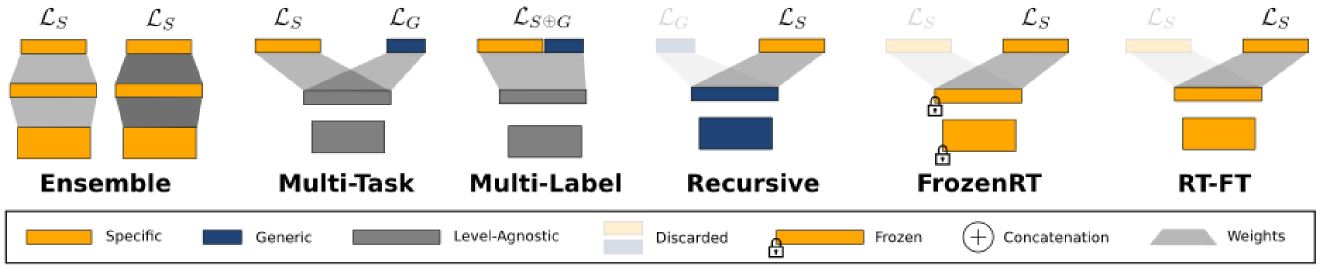


Fig. 6. Illustration of the different baseline methods.

through their comparison to several baseline-methods. All the baselines are described below, illustrated in Fig. 6 and the results are reported in Table 4. In this section (and supplementary), the comparisons are conducted on eight of the ten target-datasets presented in Sec. 6.1 (VOC12 and stCAR removed because evaluation-servers limited to 1 run per day needed).

We first assess whether the gain of universality obtained by our MulDiP-Net method is caused by the ensemble-model component. To do so, we compare it to a baseline that consists of an ensemble-model with two networks trained on the same specific SP but with different random initialization of the weights (**Ensemble**). While it significantly increases the universality compared to the reference, our MulDiP-Net gives much better results. This later means that the performances of our method does not come from the ensemble-model aspect only, but also by the combination of features learned with data labeled among generic and specific categories. We also compare our method to three variants of joint training on two SPs (generic and specific): (i) a **Multi-Task** approach that minimizes a sum of softmax losses, one for each SP; (ii) a **Multi-Label** approach that considers each image annotated both with the generic and specific label; and (iii) a **Recursive** approach that trains the network on the generic SP then continues the training on the specific SP. As reported in table 4, the first two baselines perform poorly. The Recursive method has similar performances to the reference one, probably because the network trained on the second specific problem "forgets" the features learned on the former generic one, what is known as catastrophic forgetting [29]. Globally, the comparison to these baselines emphasizes the advantage of learning networks independently rather than jointly, and the importance of growing the capacity of the ensemble model to benefit from multiple SPs.

We compare FoRT to the methods proposed by Yosinski *et al.* [11], Frozen ReTraining (**FrozenRT**) and ReTraining by Fine-Tuning (**RT-FT**), that also re-train a network on the same problem. Similarly to the original study, we observe a performance drop of FrozenRT compared to the reference, explained by the fragile co-adaptation neurons learned in the original network (Sec. 3). A slight drop of performance is also observed for RT-FT, meaning that fine-tuning does not always recover all the co-adapted neurons. In contrast, FoRT increases the performance, even when we reduce its representation to 2,048 (**FoRT$_{\text{DR}}$**), highlighting its capacity to adequately balance between recovering co-adapted neurons of the original network and training the new ones.

Finally, we notice that when FoRT is combined to MulDiP, the performances are significantly boosted, even when the capacity and the representation dimension are reduced with FoRT$_{\text{DR}}$.

6.4 Deeper Networks and More Data

Our approach is independent from the network architecture used and can benefit from progresses in this domain. Using wider or larger networks often lead to better performances, although it usually requires more data to train. We thus apply our approach to three different networks, namely AlexNet VGG-16 and DarkNet-20 , and learn them on the full ILSVRC. The results of these experiments are presented in Table 5. With twice more data than in Table 4, MulDiP-Net still significantly increases universality compared to the reference, showing that it benefits to additive data as previous works that increase domains [4] or tasks [6]. We also observe that the deeper architecture do not always learn more universal representation since Net-S with VGG-16 is better than Net-S with DarkNet-20. It shows that increasing the capacity is not in itself enough to improve the representation universality. Finally, we observe that Net-G is always below Net-S, except for DarkNet. This is surprising since one could imagine that training

Method	**Network**	**VOC07** mAP	**CA101** Acc.	**CA256** Acc.	**NWO** mAP	**MIT67** Acc.	**stACT** Acc.	**CUB** Acc.	**FLOW** Acc.	**mNRG**
Net-S (Ref.)	AlexNet	71.7	79.7	62.4	58.3	46.9	51.2	**36.3**	58.4	0.0
Net-G	AlexNet	71.5	77.4	60.4	57.8	42.8	49.3	19.5	52.4	-7.7
MulDiP-Net	AlexNet	**74.4**	**82.5**	**65.2**	**60.8**	**47.4**	**54.2**	36.1	**62.5**	+7.4
Net-S	VGG-16	86.1	88.8	78.0	71.8	66.7	73.5	69.8	78.9	+44.8
Net-G	VGG-16	85.7	87.6	76.9	70.3	65.8	72.2	67.0	75.0	+38.9
MulDiP-Net	VGG-16	**87.5**	**92.0**	**80.9**	**72.6**	**68.9**	**75.0**	**71.5**	**81.9**	**+55.3**
Net-S	DarkNet-20	82.7	91.0	78.4	70.5	64.8	72.2	59.5	80.0	+38.9
Net-G	DarkNet-20	83.2	91.5	78.1	73.2	64.4	72.6	52.5	78.9	+40.6
MulDiP-Net	DarkNet-20	**84.1**	**92.7**	**80.1**	**73.9**	**66.4**	**74.5**	**61.2**	**82.1**	+47.1

TABLE 5
MulDiP-Net performances with **different network architectures and more training data**. To compute the mNRG scores (last column in blue), we used the Net-S of AlexNet as reference. All the methods have been learned on the same initial SP (whole ILSVRC).

Approach (Representation)	**VOC07** mAP	**CA101** Acc.	**CA256** Acc.	**NWO** mAP	**MIT67** Acc.	**stACT** Acc.	**CUB** Acc.	**FLO** Acc.	**mNRG**
Standard-TL (Net-S)	75.4	82.8	66.0	68.1	45.4	56.8	40.8	65.0	0.0
Standard-TL (MulDip-Net)	77.7	85.8	70.2	70.8	50.8	60.6	43.9	70.9	+9.6
FineTuning (Net-S)	78.9	83.5	62.4	75.3	60.9	60.1	64.7	76.3	+18.4
FineTuning (MulDip-Net)	79.8	84.8	65.2	77.2	63.1	65.2	65.0	77.6	+24.0
ResAdapters (Net-S) [14]	80.2	89.5	71.9	75.0	63.7	67.9	66.0	77.4	+29.6
ResAdapters (MulDip-Net)	**81.2**	**90.2**	**73.8**	**77.5**	**66.4**	**70.8**	**67.7**	**80.8**	**+35.4**

TABLE 6
Performances of MulDip compared to Net-S in three cases: **standard transfer learning** (top), with an additive **fine-tuning** (middle) or **residual adapters** (bottom). Following [14], all the methods are based on the same Resnet-26 architecture, trained on ILSVRC with 72x72px images.

with finer categories leads to better results, while it seems that it also depends on the actual network used. Their combination through MulDiP-Net remains nevertheless beneficial in any case.

6.5 Fine Tuning and Residual Adapters

In the previous sections, sthe pre-trained representations are used "as is" on the target datasets, meaning that only the classifiers are learned on top of them. However, it is well known that fine-tuning the whole network on the target dataset improves the performances. More interestingly, adding some residual parameters [12], [14], [42] to a fixed backbone, then learning them jointly with the classifiers has been showed to be even better. In any case, such approaches are complimentary to MulDip-Net, that provides a more universal representation (*i.e.*, better backbone network).

We thus studied the effect of fine tuning and residual adapters on our approach. Following [14], we used a ResNet-26 backbone, that is trained on ILSVRC with images of size 72x72. MulDip-Net was compared to a Net-S for: (i) a standard transfer learning: only the classifiers are learned on top of the representation, (ii) fine-tuning: the representation is fine tuned on the target dataset and finally (iii) residual adapters: some parameters are added to the fixed representation to adapt the new parameters to each target dataset independently. Following [14], residual adapters were added in parallel to each 3×3 convolution. For MulDip-Net, we minimized the sum of a loss specific to each subnetwork and that of a joint loss during training. During the test phase, only the joined output is used. As reported in Table 6, MulDip-Net is still beneficial, whatever the adaptation approach used. Our independent implementation also confirms the interest of residual adapters in comparison to standard fine tuning.

7 Conclusion

In this paper, we proposed four contributions: (i) a new challenge of learning more universal representations from a fixed set of data (domains and tasks); (ii) the evaluation of universality in a more suitable scheme (transfer-learning), as well as a new metric respecting most of the highlighted desirable criteria; (iii) a new method based on the re-training of networks, by focusing the training on some parameters; and finally (iv) a new method based on a general formalism of source problem variation and training of multiple networks. We demonstrated the effectiveness of our universalizing methods, in a transfer-learning scheme, through our evaluation-metric, on ten target-datasets from different domains. An in-depth analysis has also been conducted to highlight some important insights of our methods. We hope that our contributions will further support the creation of other methods to get more universal representations and open doors for many less explored aspects of transfer-learning such as learning the source-problem, using Human knowledge or even through more realistic multi-modal and dynamic environments such as HOME [48].

References

[1] J. Atkinson, *The developing visual brain*. Oxford University Press UK, 2002.
[2] I. Kokkinos, "Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory," in *CVPR*, 2017.
[3] J. Wang, O. Russakovsky, and D. Ramanan, "The more you look, the more you see: towards general object understanding through recursive refinement," in *WACV*, 2018.
[4] H. Bilen and A. Vedaldi, "Universal representations: The missing link between faces, text, planktons, and cat breeds," *arXiv:1701.07275*, 2017.
[5] A. Conneau, H. Schwenk, L. Barrault, and Y. Lecun, "Very deep convolutional networks for text classification," in *ACL*, 2017.
[6] S. Subramanian, A. Trischler, Y. Bengio, and C. J. Pal, "Learning general purpose distributed sentence representations via large scale multi-task learning," in *ICLR*, ser. ICLR, 2018.
[7] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *TPAMI*, 2013.
[8] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *EMNLP*, 2017.

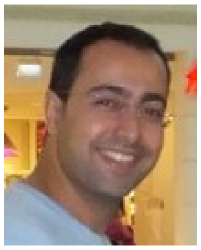
[9] Y. Tamaazousti, H. Le Borgne, and C. Hudelot, "Mucale-net: Multi categorical-level networks to generate more discriminating features," in *CVPR*, 2017.
[10] E. Rosch, "Principles of categorization," *Cognition and Categorization*, 1978.
[11] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *NIPS*, 2014.
[12] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Learning multiple visual domains with residual adapters," in *NIPS*, 2017.
[13] H. Azizpour, A. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "Factors of transferability for a generic convnet representation," *TPAMI*, 2015.
[14] S.-A. Rebuffi, H. Bilen, and A. Vedaldi, "Efficient parametrization of multi-domain deep neural networks," in *CVPR*, ser. CVPR, 2018.
[15] B. Zhou, A. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *NIPS*, 2014.
[16] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.
[17] P. Mettes, D. Koelma, and C. G. M. Snoek, "The imagenet shuffle: Reorganized pre-training for video event detection," in *ICMR*, 2016.
[18] Y. Tamaazousti, H. Le Borgne, A. Popescu, E. Gadeski, A. Ginsca, and C. Hudelot, "Vision-language integration using constrained local semantic features," *CVIU*, 2017.
[19] P. D. Vo, A. Ginsca, H. Le Borgne, and A. Popescu, "Harnessing noisy web images for deep representation," *CVIU*, 2017.
[20] A. Joulin, L. van der Maaten, A. Jabri, and N. Vasilache, "Learning visual features from large weakly supervised data," in *ECCV*, 2016.
[21] M. Huh, P. Agrawal, and A. A. Efros, "What makes imagenet good for transfer learning?" *arXiv:1608.08614*, 2016.
[22] I. Chami, Y. Tamaazousti, and H. Le Borgne, "Amecon: Abstract meta concept features for text-illustration," in *International Conference on Multimedia Retrieval*, ser. ICMR, 2017.
[23] K. Ahmed, M. H. Baig, and L. Torresani, "Network of experts for large-scale image categorization," in *ECCV*, 2016.
[24] G. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *arXiv:1503.02531*, 2015.
[25] V. N. Murthy, V. Singh, T. Chen, R. Manmatha, and D. Comaniciu, "Deep decision network for multi-class image classification," in *CVPR*, 2016.
[26] W. Ouyang, X. Wang, C. Zhang, and X. Yang, "Factors in finetuning deep model for object detection with long-tail distribution," in *CVPR*, 2016.
[27] Y. Wu, J. Li, Y. Kong, and Y. Fu, "Deep convolutional neural network with independent softmax for large scale face recognition," in *ACM*, 2016.
[28] Z. Yan, H. Zhang, R. Piramuthu, V. Jagadeesh, D. DeCoste, W. Di, and Y. Yu, "Hd-cnn: hierarchical deep convolutional neural networks for large scale visual recognition," in *ICCV*, 2015.
[29] R. M. French, "Catastrophic forgetting in connectionist networks," *Trends in cognitive sciences*, vol. 3, no. 4, pp. 128–135, 1999.
[30] P. Jolicoeur, M. A. Gluck, and S. M. Kosslyn, "Pictures and names: Making the connection," *Cognitive Psychology*, 1984.
[31] A. Conneau and D. Kiela, "Senteval: An evaluation toolkit for universal sentence representations," in *LREC*, 2018.
[32] J. Deng, J. Krause, A. C. Berg, and L. Fei-Fei, "Hedging your bets: Optimizing accuracy-specificity trade-offs in large scale visual recognition," in *CVPR*, 2012.
[33] A. Mathews, L. Xie, and X. He, "Choosing basic-level concept names using visual and language context," in *WACV*, 2015.
[34] V. Ordonez, W. Liu, J. Deng, Y. Choi, A. C. Berg, and T. L. Berg, "Predicting entry-level categories," *IJCV*, 2015.
[35] Y. Tamaazousti, H. Le Borgne, and C. Hudelot, "Diverse concept-level features for multi-object classification," in *ICMR*, 2016.
[36] A. L. Ginsca, A. Popescu, H. Le Borgne, N. Ballas, P. Vo, and I. Kanellos, "Large-scale image mining with flickr groups," in *MM*, 2015.
[37] Y. Tamaazousti, H. Le Borgne, and A. Popescu, "Constrained local enhancement of semantic features by content-based sparsity," in *ICMR*, 2016.
[38] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *ICLR*, 2015.
[39] Y.-X. Wang, D. Ramanan, and M. Hebert, "Growing a brain: Fine-tuning by increasing model capacity," in *CVPR*, 2017.
[40] L. Herranz, S. Jiang, and X. Li, "Scene recognition with cnns: objects, scales and dataset bias," in *CVPR*, 2016.
[41] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, "Overfeat: Integrated recognition, localization and detection using convolutional networks," in *ICLR*, 2014.
[42] A. Rosenfeld and J. K. Tsotsos, "Incremental learning through deep adaptation," *TPAMI*, 2018.
[43] A. Salvador, N. Hynes, Y. Aytar, J. Marin, F. Ofli, I. Weber, and A. Torralba, "Learning cross-modal embeddings for cooking recipes and food images," in *CVPR*, ser. CVPR, 2017.
[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *IJCV*, 2015.
[45] J. W. Tanaka and M. Taylor, "Object categories and expertise: Is the basic level in the eye of the beholder?" *Cognitive Psychology*, 1991.
[46] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *CVPR*, 2017.
[47] W. Liu, A. Rabinovich, and A. C. Berg, "Parsenet: Looking wider to see better," in *ICLR workshop*, 2016.
[48] S. Brodeur, E. Perez, A. Anand, F. Golemo, L. Celotti, F. Strub, J. Rouat, H. Larochelle, and A. Courville, "Home: A household multimodal environment," *arXiv preprint arXiv:1711.11017*, 2017.

Youssef Tamaazousti is a Postdoctoral Research Associate at the Computer Science and Artificial Intelligence Lab (CSAIL) of Massachusetts Institute of Technology (MIT). Previously, he received his PhD from CentraleSupelec (University of Paris-Saclay) and CEA LIST, in 2018. His research interests span the areas of visual recognition, neural networks and representations-learning.

Hervé Le Borgne is a researcher at the CEA LIST since 2006, carrying out research on computer vision and multimedia mining. Previously, he received his PhD from the INP Grenoble in 2004 and worked as a post-doc at Dublin City university until 2006. He published more than 40 articles in international conferences and journals and is co-inventor of seven patents. His research interest deals with information extraction from visual and textual documents, and relating this information to the human user needs.

Céline Hudelot is a Full Professor at the Mathematics and Interaction with Computer Science Laboratory of CentraleSupelec (University of Paris-Saclay), in charge of the research axis on formal methods for semantic multimedia understanding. She obtained her Ph.D from INRIA and the University of Nice Sophia Antipolis in 2005 and her Habilitation from Université Paris-Sud in 2014. Her research interests include knowledge and ontological engineering for semantic image analysis, 2D and 3D image processing, information fusion, formal logics, graph-based representation and reasoning, spatial reasoning and machine learning.

Mohamed-El-Amine Seddik received a Master of Engineering in Data Science from Institut Mines-Telecom de Lille (with the final year completed at Telecom ParisTech) and a Master Degree in Vision and Machine Learning from ENS Cachan in 2017. He is currently a PhD student in the computer vision laboratory of CEA LIST, interested in random matrix theory for machine learning and scene understanding.



Mohamed Tamaazousti received his Master's Degree in applied mathematical from the University of Orléans in 2009 and the Ph.D. degree in computer vision from the University Blaise Pascal in 2013. He is currently a permanent researcher at CEA LIST. His main research interests include structure from motion for rigid scenes, real time vision-based localization and reconstruction (SLAM) for autonomous system. He is also interested in augmented and diminished reality applications.