Learning Finer-class Networks for Universal Representations

Julien Girard[1][2]
julien.girard2@cea.fr
Youssef Tamaazousti[1][2][3]
youssef.tamaazousti@cea.fr
Hervé Le Borgne[2]
herve.le-borgne@cea.fr
Céline Hudelot[3]
celine.hudelot@centralesupelec.fr

[1] Both authors contributed equally.
[2] CEA LIST
Vision Laboratory,
Gif-sur-Yvette, France.
[3] CentraleSupélec,
MICS Laboratory,
Châtenay-Malabry, France.

Abstract

Many real-world visual recognition use-cases can not directly benefit from state-of-the-art CNN-based approaches because of the lack of many annotated data. The usual approach to deal with this is to transfer a representation pre-learned on a large annotated source-task onto a target-task of interest. This raises the question of how well the original representation is "universal", that is to say directly adapted to many different target-tasks. To improve such universality, the state-of-the-art consists in training networks on a diversified source problem, that is modified either by adding generic or specific categories to the initial set of categories. In this vein, we proposed a method that exploits finer-classes than the most specific ones existing, for which no annotation is available. We rely on unsupervised learning and a bottom-up split and merge strategy. We show that our method learns more universal representations than state-of-the-art, leading to significantly better results on 10 target-tasks from multiple domains, using several network architectures, either alone or combined with networks learned at a coarser semantic level.

1 Introduction

The state-of-art performances in visual recognition obtained by Convolutional Neural Networks (CNNs) are subject to the availability of a large set of annotated training data to learn the model. Since it is rarely the case for many practical tasks of interest (*target-tasks*), one usually adopts a transfer-learning approach [24, 25, 27] which relies on a CNN pre-trained on a *source task* with sufficient annotated data (often ImageNet [31]) then further truncated to provide the representations of the samples of target-task. Then, even with few annotated data, this last can usually be learned with a linear classifier. Such approaches raise the question of the similarity of the *source-task* on which the representation has been learned and the target-task on which it is used. Although this similarity is not easy to formalize, one has the intuition that the closer the both tasks the better the representation will be adapted to the target-task. This consideration leads to several methods that tend to obtain more *universal* representations [2, 9, 18, 28, 34, 36], that is to say that are more adapted to a large set of diverse target-tasks, in a transfer-learning scenario.

The general idea of these methods is to diversify the classification problem of the source-task in order to obtain more features, able to adequately represent new target-datasets, from more domains, in a larger context. All these approaches vary the problem by creating new categories having an existing label. However, most of them studied the effect of adding categories extracted from ImageNet, either *generic* categories [19, 22, 34, 36] or *specific* ones [1, 2, 28, 45], that are at the bottom of a hierarchy such as ImageNet, except [17, 37, 38] that use web annotations with noisy labels. In general, the usage of specific categories tends to provide better performances than generic ones [2, 28], although combining them can significantly boost the universalizing capacity of the CNN [34, 36]. Yet, even for the most specific categories, it is plausible that it exists a variety of semantics within the class that is not explored (*e.g.*, one could imagine to split the object-class according to the different poses of the object). Clearly, the limiting point is the availability of such finest annotation (*e.g.*, poses, contexts, attributes) for existing specific classes.

In this article, we argue that exploring *finer* classes than the most *specific* existing ones, can significantly increase the diversity of the problem, therefore improve the universality of the representation learned in the internal layers of the CNN. The main difficulty is the lack of annotation below the most specific levels. We propose to rely on unsupervised learning (clustering) to determine these finer categories within each specific category. Our contribution is three-fold. First, we show that the use of finer categories rather than the most specific ones to learn CNNs, improves the universality of the resulting representation, even when the finer classes are determined *randomly* within each specific class. Second, the usage of a K-means based approach leads to slightly better results although the resulting clusters are strongly imbalanced. To fix this, our core contribution splits and merges the specific categories to automatically determine better balanced finer classes, leading to better results. Last, we show that CNNs learned with our approach provide a better complementary to standard CNN representations than those learned on generic categories.

Let note that if the target-task has enough data, the representation can be adapted to the target-task by fine-tuning. This is nevertheless out of the scope of this work, because it is a *complementary* process to the transfer-learning in itself, that will always improve the performances, and especially, because fine-tuning modifies the representations, which leads to a bias that *hides the real ability* of a universalizing method [16]. Hence, in this paper, we are only interested into studying the universality of the representations, independently of many possible refinements of a full adaptation method on each target-task.

Previous works [6, 11, 12, 43] exploited sub-categories in the context of visual recognition. In [11, 12], an object instance affinity graph is computed from intra-class similarities and inter-class ambiguities then the visual subcategories are detected by the graph shift algorithm. The process is nevertheless quite computationally demanding and applied to object detection on small target datasets only. In [6, 43], subcategories that are learned from extrapolated feature maps and fine-tuned on a target-dataset, are used within a CNN to improve region proposal for object detection. To the best of our knowledge, our paper is the first to propose the usage of subcategories determined by unsupervised learning on a source-task, in order to improve universality of representations. More related to universality, [2, 28, 29] added annotated data from more domains as well as domain-specific neurons to an initial set of domain-agnostic ones. Contrary to them, our method only modifies the source-problem at zero cost of annotation. Our work is closer to the approach of [34, 36] that proposes to relabel specific categories into generic ones (that match the upper categorical-levels), to learn an additive CNN with the same architecture. Nevertheless, our approach is interested into the "opposite way", that is, creating finer classes than those at the bottom of a hierarchy

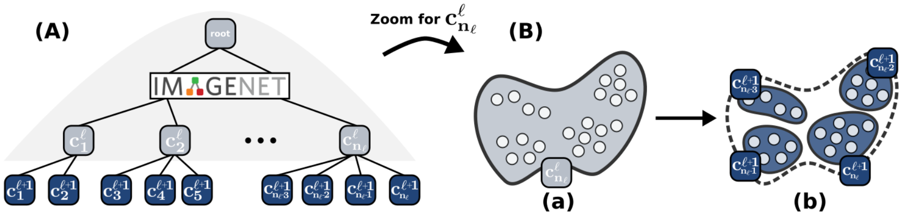

Figure 1: Illustration of our splitting principle that determines a *finer*-level $\ell+1$ containing $n_{\ell+1}$ *finer*-classes (blue nodes in **A**) from the *finest*-level ℓ of a hierarchy (here, ImageNet), containing n_ℓ *specific* categories that the leaf gray nodes in **(A)**. Each finer-class $c_i^{\ell+1}$ ($i \in [n_{\ell+1}]$) is related to a specific-class c_j^ℓ ($j \in [n_\ell]$) through "is-a" relations, since the $c_i^{\ell+1}$ classes are obtained from *each* c_j^ℓ class. In **(B)**, we focus on the particular specific class $c_{n_l}^\ell$ (▨) and its image-representations (○), to determine the finer-classes $\{c_{n_l-3}^{\ell+1}, c_{n_l-2}^{\ell+1}, c_{n_l-1}^{\ell+1}, c_{n_l}^{\ell+1}\}$. In (b), it is splitted into K (here K=4) groups (●) corresponding to the finer-classes $c_j^{\ell+1}$.

(ImageNet), for which no annotation exists and thus their method can not be applied.

We evaluated our proposal on the problem of universality, that is, in a transfer-learning scheme using multiple target-tasks (*i.e.*, ten classification benchmarks from multiple domains, including actions, food, scenes, birds, aircrafts, etc.). In particular, in comparable settings (using ILSVRC as source-task and two architectures, AlexNet [19] and DarkNet [30]), we showed that our method outperforms state-of-the-art ones.

2 Proposed Method

We propose a new universalizing method that consists in training a network on a set of categories that are *finer* than those of the *finest*-level of a hierarchy (*e.g.*, ImageNet hierarchy or any set of categories). In Sec. 2.1, we start by describing its general principle as well as two baselines that splits them either randomly or by clustering their features. With such baseline, the number of finer classes must be a priori fixed, thus we propose a "bottom-up clustering-based merging" approach that determines a better splitting automatically (Sec. 2.2). Furthermore, we propose to combine the features learned on the specific categories and those learned on the finer ones to get an even more universal representation (Sec. 2.3).

2.1 FiNet: Network Trained on Finer-Classes

The leaf nodes of the ImageNet hierarchy represents the *finest* or most *specific* categories that are annotated. More generally, this is the case for the set of categories of any classification dataset. To go towards our goal of automatically obtaining finer categories (without annotations) from the finest ones, a baseline approach consists in using a random partitioning of the specific categories or a simple clustering-based approach of their image-features. The first baseline, randomly assigns every image of a specific category to one of K_i clusters. The second one, first, learns a CNN (noted **SpeNet**) on the specific categories, uses one of its layer as features-extractor for every image, then determines K_i clusters using K-means on these vectors. A more sophisticated way is our final method that is presented in Sec. 2.2. Note that, in all cases, the splitting is performed on specific categories, that already contains quite similar samples/vectors. Once the finer classes obtained, we train another network (denoted **FiNet**) on the *same* images used to train SpeNet, but labeled among the obtained *finer*-classes. The whole set of finer-classes forms the new finest-level of the hierarchy. Our general principle is illustrated in Fig. 1 and presented more formally below.

Let us consider a semantic hierarchy with hyponymy relations, that is to say a set of categories organized according to "is-a" relations (*e.g.*, ImageNet [10] hierarchy). This hierarchy denoted $\mathcal{H}_\ell = (\mathcal{C}_\ell, E)$ is a directed acyclic graph of ℓ levels of nodes, with $\mathcal{C}_\ell$ being all the nodes and E the set of directed edges between the nodes. Each node $c_i^\ell \in \mathcal{C}_\ell$ corresponds to the i-th category at level ℓ in $\mathcal{H}_\ell$ and n_ℓ is the number of categories at level ℓ. A hierarchy-edge $(c_i^\ell, c_j^{\ell+1}) \in E$ indicates that class c_i^ℓ subsumes class $c_j^{\ell+1}$. Let us also consider an initial dataset $\mathcal{D}_N^\ell$ containing a set of N images labeled among the categories at level ℓ and let us denote N_i^ℓ the number of images labeled among the i-th category at level ℓ. Note that, $N = \sum_{i=1}^{n_l} N_i^\ell$. Each image $\mathbf{I}_i^j \in \mathcal{D}_N^\ell$ of the dataset, is associated to a given category c_j^ℓ for $j \in [n_\ell]$[1]. Let us denote $\mathbf{X}_i^{\ell,L} \in \mathbb{R}^{d_\ell}$ the representation of an image $\mathbf{I}_i$ extracted from layer L of the network trained on $\mathcal{D}_N^\ell$ (*i.e.*, SpeNet). Let also $\mathcal{X}_i^\ell = \{\mathbf{X}_j^{\ell,L}\}_{j=1}^{N_i^\ell}$ being the set of features extracted from all the images belonging to the category c_i^ℓ.

In order to construct the K_i finer-categories $\{c_j^{\ell+1}\}_{j=1}^{n^{\ell+1}}$ of *each* category of the previous level c_i^ℓ, we apply a clustering algorithm (*e.g.*, K-means, MeanShift or BUCBAM presented in Sec. 2.2) on $\mathcal{X}_i^\ell$ (that builds K_i centroids) where the feature vector $\mathbf{X}_j^{\ell,L}$ of each image $\mathbf{I}_j$ ($j \in [N_i^\ell]$) is assigned to the nearest centroid (hard-coding [20]), which forms the K_i finer classes. This process is applied for all $i \in [n_\ell]$ which gives the $n_{\ell+1} = K_i \times n_\ell$ finer classes, that forms the nodes of the finer level $\ell+1$. This latter results in a new dataset $\mathcal{D}_N^{\ell+1}$, for which each image $\mathbf{I}_i^j \in \mathcal{D}_N^{\ell+1}$ is associated to a given category $c_j^{\ell+1}$ for $j \in [n_{\ell+1}]$. Note that by construction, every c_i^ℓ subsumes all its finer-categories $\{c_j^{\ell+1}\}_{j=1}^{K_i}$, thus we have $(c_i^\ell, c_j^{\ell+1})_{(i,j)\in[n_\ell]\times[K_i]}$. The whole process results in a new hierarchical level $\ell+1$ that forms the new hierarchy $\mathcal{H}_{\ell+1} = (\mathcal{C}_{\ell+1}, E')$ with $\mathcal{C}_{\ell+1} = \mathcal{C}_\ell \cup \{c_{j,j\in[K_i]}^{\ell+1}\}_{i\in[n_\ell]}$. E' corresponds to the union of E and the edges that connect each category c_i^ℓ to its K_i finer ones $c_j^{\ell+1}$. It is important to point out that, depending on the clustering algorithm, K_i will depend on c_i^ℓ or be the *same* for all categories. This is discussed in the next section.

The new dataset $\mathcal{D}_N^{\ell+1}$ is used to train (softmax cross-entropy loss minimized by SGD) the FiNet network, which has $n_{\ell+1}$ neurons on its last layer. FiNet is then used as features-extractors for the images $\mathbf{I}_i$ of the target-tasks: $\mathbf{X}_i^{\ell+1,L} = \Phi_L^{\ell+1}(\mathbf{I}_i)$.

2.2 Bottom-Up Clustering-Based Merging

We empirically observed (see Fig. 3) that clustering approaches with a fixed K_i for each category (*e.g.*, Kmeans) usually leads to a FiNet that gives better universality results than approaches that adapt K_i to each category (*e.g.*, Affinity-Propagation). Indeed, this latter tends to provide a set of finer classes with many clusters containing few images and a couple of clusters containing a large number of images, leading to an undesirable imbalanced dataset that penalizes the network training. Even if the use of fixed-K_i clustering methods leads to more balanced data, it remains sub-optimal since it sets the *same* amount of clusters for all specific categories ($\forall i \in [n_\ell], K_i = K$), while this may depend on the content of each category. Furthermore, in fixed-K_i clustering methods, the K value is an hyper-parameter that is cross-validated on the target-tasks, which are not accessible during the learning on the

[1] Let $[n]$ denotes $[\![1,n]\!]$, in all the paper.

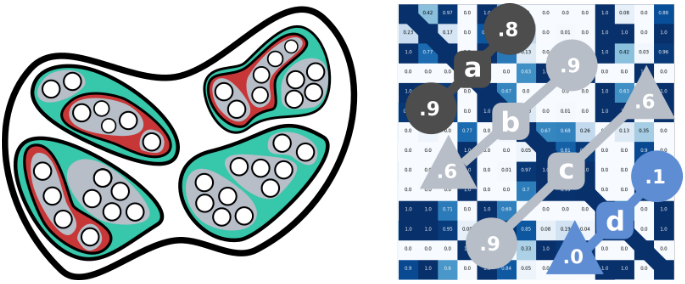

Figure 2: On the left, given image-representations (○) labeled among a specific class (⬭), BUCBAM first performs a clustering with many clusters (⬬), then attaches small clusters to bigger closest ones (⬬) and lastly, merges similar ones (⬬), w.r.t certain strategies. On the right, we describe the latter merging strategies. Given the similarity matrix $\mathbf{M}_j$ (high values: blue, low: white) for one specific class c_j^ℓ, BUCBAM-SS merges *only* clusters that are reciprocally highly similar to each other, respecting constraint (a) with high scoring values in $(\mathbf{M}_j)_{k,l}$ and $(\mathbf{M}_j)_{l,k}$). BUCBAM-AS *also* merges asymmetrically similar clusters (those that respect constraints (a), (b) and (c)). In both cases, dissimilar clusters (d) are let disjoint.

source-task, in the context of universality [36]. Hence, the cross-validation of K should be performed *only* with the *source*-task, which is not trivial (optimal K on the source-task not necessary optimal one for the target-tasks). To overcome the drawbacks of fixed and adapted-K_i clustering methods, we proposed an hybrid one called "Bottom-Up Clustering-BAsed Merging" (**BUCBAM**). It roughly starts with the clusters obtained by a fixed-K_i clustering and *automatically* sets the amount of clusters for *each* specific category by enforcing a more balanced resulting set of finer-classes. Specifically, it consists in three main steps (illustrated on the left of Fig. 2): (i) splitting the specific categories into K clusters (with a large K); (ii) attaching small clusters to the closest bigger ones (to avoid imbalanced data); and (iii) merging the most similar ones, with respect to a proposed similarity criteria.

Formally, BUCBAM starts with a *large* amount of K finer-classes per category c_j with $j \in [n_\ell]^\ell$ and K being the *same* for all c_j^ℓ. Let us assume we have $K (\in \mathbb{N}^*)$ finer-categories $\{c_i^{\ell+1}\}_{i \in [K_j]}$ obtained from the category c_j^ℓ of the previous level through a fixed-Ki clustering method. Let denote $\mathcal{X}_i^{\ell+1} = \{\mathbf{X}_i^{\ell,L}\}_{i \in [N_i^{\ell+1}]}$ the whole set of features extracted from the images of a given category $c_i^{\ell+1}$ through the SpeNet Φ_L^ℓ. Note that, $N_i^{\ell+1}$ corresponds to the amount of images in each $c_i^{\ell+1}$[2]. The goal of BUCBAM is to get an amount of clusters K_j depending on the images of *each* category c_j^ℓ. To do so, it first prunes out the *small* clusters (*i.e.*, all the c_i such that $\forall i \in [K_j]$, $\mathrm{Card}(c_i) < S \in \mathbb{N}^*$, with $S \ll N_i$), by re-assigning their samples $\mathbf{I}_{k,k \in [N_i]}$ (that were assigned to c_i) to the category of the closest feature vector $\mathbf{X}^i_{l,l \in [N_i]} = \mathcal{N}(\mathbf{X}^m_{k,k \in [N_m]})$, with $m \neq i$ and $\mathcal{N}(\cdot)$ being a function that provides the closest vector (*e.g.*, k-NN algorithm with Euclidean distance) in the set of features $\{\mathcal{X}_j\}$ belonging to the *other* and *large* clusters (*i.e.*, all c_m with $\mathrm{Card}(c_m) \geqslant S$). Pruning small clusters for all categories c_j^ℓ, results in a set of $K_j^{\mathcal{P}}$ finer-classes $\{c_i\}_{i \in K_j^{\mathcal{P}}}$ per class c_j^ℓ. The last step of BUCBAM is to merge the similar clusters. To do that, a classifier Ψ_i is trained for each cluster c_i – using features of c_i samples as positives and same amount of samples from a *diverse* class c_d as negatives – and evaluated on the images of all other clusters. The diverse category c_d is created by randomly picking elements equiprobably from all the categories

[2] For simplicity, we omit the power indices ℓ, $\ell+1$ and L in the following.

$\{c_i^\ell\}_{i\in[n_\ell]}$. The evaluation of the classifiers provides a similarity matrix $\mathbf{M}_j \in [0,1]^{K_i^{\mathcal{P}} \times K_i^{\mathcal{P}}}$ for each category c_i^ℓ. This last is used to merge similar clusters and let dissimilar ones disjoint. More precisely, a first strategy is to consider clusters c_i and c_m *symmetrically similar* (BUCBAM-SS) if: $\Psi_i(\mathbf{X}^m_{k,k\in[N_m]}) > S_H$ and $\Psi_m(\mathbf{X}^i_{k,k\in[N_i]}) > S_H$, with $m \neq i$ and $S_H \in [0,1]$ a *high* score (close to 1). Another strategy is to consider, clusters *asymmetrically similar* (BUCBAM-AS) if only one constraint is respected and the other is greater than S_M, with $S_M = S_H/2$ a *medium* score. In both cases (that are illustrated in Fig. 2), dissimilar clusters are desirably let disjoint. Merging similar clusters for all classes c_j^ℓ, results in a set of $K_j^{\mathcal{M}}$ finer-classes $\{c_i\}_{i\in K_j^{\mathcal{M}}}$ per category c_j.

2.3 SpeFiNet: Combining Specific and Finer Features

Following the approach of [36] – which roughly consists in training initial features on an initial set of categories, then learning new features on new set of categories and finally combining initial and new features –, we propose to learn the new features with our FiNet (rather than a network trained on *generic* categories [34, 36]) and combine them with the features of the initial SpeNet to get a representation even more universal. This method is denoted **SpeFiNet** in the following. Formally, the final SpeFiNet representation combines specific and finer features and is computed for an image I_i of a target-task as: $\mathbf{X}_i = \mathcal{F}\big(\{\mathcal{Z}(\mathbf{X}_i^\ell), \mathcal{Z}(\mathbf{X}_i^{\ell+1})\}\big)$, where $\mathcal{F}$ is a fusion operator, and $\mathcal{Z}$ is a normalization function. In practice for the normalization and fusion, we respectively choose the L-infinite norm (L-∞) and the concatenation. To the best of our knowledge, we are the first to propose to combine a SpeNet (trained on *specific* categories) and a FiNet (trained on *finer* categories) to get more universal representations.

3 Experimental Results

Universality

Universalizing methods are evaluated in a transfer-learning scheme on multiple target-tasks [4, 9, 36]. More precisely, a source-task is used to train a network that acts as a representation extractor on the data of the target-tasks. Each target-task is trained with a simple predictor on top of the representations extracted from the samples of the target-task. Note that, fine-tuning the representations on the target-tasks could always improve performances but induces a bias avoiding correct evaluation of universality [8, 16, 32, 36]. Hence, following the literature, simple predictors that do not modify the representations learned on the source-task are used. In particular, here for the target-tasks, we used a classification task with datasets from multiple visual domains (presented below) and for the predictor, we used a one-versus-all SVM classifier for each class. Even if [8, 28, 36] initiated a work around universality evaluation, it seems to remain an open problem. Hence here, since we only have benchmarks that are evaluated in terms of accuracy and precision, we evaluate universalizing methods in terms of average of their performances on the multiple benchmarks.

Datasets

For the source-task, we used ILSVRC [31] and ILSVRC* (half of the former, detailed in [34]). For the target-tasks, we used ten datasets from multiple domains, including general objects (VOC07 [13], NWO [7], CA101 [14], CA256 [15]), scenes (MIT67 [26]), actions (stACT [44]), birds (CUB [39]), plants (FLO [23]), food (FOOD [3]) and airplanes (AIRC [21]). The characteristics of all the datasets are detailed in supplementary material.

Method	VOC07	CA101	CA256	NWO	MIT67	stACT	CUB	FLO	Avg
	mAP	Acc.	Acc.	mAP	Acc.	Acc.	Acc.	Acc.	
SpeNet (REFERENCE)	66.8	71.1	53.2	52.5	36.0	44.3	36.1	50.5	51.3
SPV$_{\mathbf{A}}^{\mathbf{spe}}$ [1]	66.6	74.7	54.7	53.2	37.4	45.1	36.0	51.9	52.4
SPV$_{\mathbf{G}}^{\mathbf{gen}}$ [22, 35]	67.7	73.0	54.3	50.5	37.1	44.9	36.8	50.3	51.8
AMECON [5]	61.1	58.7	40.6	45.8	24.3	32.7	26.1	36.4	44.5
WhatMakes [16]	64.0	69.4	50.1	45.6	33.7	41.9	15.0	42.8	45.3
ISM [42]	62.5	68.8	50.7	28.5	37.9	42.6	34.0	50.0	46.9
GrowingBrain-RWA [41]	69.1	74.8	55.9	50.4	40.0	48.4	38.6	56.1	54.2
FSFT [36]	67.5	73.9	55.0	44.6	40.4	47.1	38.7	56.8	53.0
MuCaLe-Net [34]	69.5	76.0	56.8	**54.7**	41.3	48.5	35.6	54.8	54.6
MulDiP-Net [36]	**69.8**	77.5	58.3	47.9	**43.7**	50.2	37.4	59.7	55.6
FiNet, Random†	66.4	72.4	53.2	51.0	39.7	46.9	35.7	55.9	52.6
FiNet, Cluster†	66.0	73.2	54.6	50.9	40.7	47.2	36.4	55.6	53.1
FiNet, BUCBAM	65.3	75.4	56.0	48.6	41.6	49.4	37.8	59.8	54.2
SpeFiNet, Random†	**69.8**	75.7	57.5	54.6	41.2	50.0	39.8	58.3	55.9
SpeFiNet, Cluster†	68.6	77.9	58.1	53.9	41.3	50.5	40.8	60.1	56.4
SpeFiNet, BUCBAM	69.1	**78.3**	**59.3**	54.0	42.7	**52.0**	**41.8**	**61.7**	**57.4**

Table 1: Comparison of our methods (bottom) to the state-of-the-art (top). All the methods are trained on the data of ILSVRC* with an AlexNet network and compared in terms of average (Avg) performance on the set of eight target-tasks used in [36]. For each benchmark, we highlight the best score in bold and the second is underlined. Methods marked with † are obtained with a parameter cross-validated (on the target-tasks), while our BUCBAM method automatically set this parameter on the source-task. Note that MuCaLe-Net, MulDiP-Net and SpeFiNets use representations which dimension is twice other method's.

Implementation Details

Our method consists in the combination of a SpeNet and FiNet. For both networks, we used two architectures, namely the classical AlexNet [19] and the deeper DarkNet [30]. They are respectively trained on the images of ILSVRC* and ILSVRC. SpeNet is thus respectively trained to recognize $C = 483$ and $C = 1,000$ *specific* categories. In contrast, FiNet is trained to recognize a set of $K_i \times C$ *finer*-classes (K_i depends on the splitting method), for which we used four variants: (i) random splitting with $K_i \in \{2,4,8,16\}$ fixed, denoted **Random-K** (ii) K-means clustering with $K_i \in \{2,4,8,16\}$ fixed, denoted **Cluster-K** (iii) BUCBAM splitting with *asymmetrically similar* clusters merging, denoted **BUCBAM-AS** and (iv) BUCBAM splitting with *symmetrically similar* clusters merging, denoted **BUCBAM-SS**. Note that, the BUCBAM methods leads to a K_i depending on the content of each category. In Sec. 3.2, we provides some statistics of the resulting dataset of each method, including the total amount of finer-classes. In Cluster-K and BUCBAM methods, we extract features from the penultimate layer to represent the samples of each class, which results in features of 4096 dimensions for AlexNet and 1000 for DarkNet. Specific to BUCBAM, the K, S and S_H parameters are respectively set to 32, 15 and 0.8. Indeed, K has to be large, and we found that as long as K is larger than 20 our method provides the same splitting result. $S = 15$ ensures to train a network with at least 15 images per class. We obtained similar results with $S = 50$. The parameters S_H is not critical since similar clusters generally provides very high (close to 1.0) classification scores.

Method	VOC07	CA101	CA256	NWO	MIT67	stACT	CUB	FLO	AIRC	FOOD	Avg.
	mAP	Acc.	Acc.	mAP	Acc.	Acc.	Acc.	Acc.	Acc.	Acc.	
SpeNet (REF.)	82.7	91.0	78.4	70.5	64.8	72.2	59.5	80.0	49.2	47.6	69.6
GenNet [34]	83.2	91.5	78.1	73.2	64.4	72.6	52.5	78.9	48.5	46.2	68.9
MulDiP-Net [36]	**84.1**	**92.7**	**80.1**	**73.9**	66.4	74.5	61.2	82.1	53.5	49.3	71.8
FiNet, Cluster†	82.5	91.8	78.8	70.0	65.8	73.2	60.9	81.9	51.9	47.8	70.5
FiNet, Cluster$(K = 16)$	81.4	91.5	77.4	69.5	64.6	72.2	58.6	81.5	52.3	47.5	69.6
FiNet, BUCBAM	81.3	91.0	77.0	69.7	64.3	72.2	59.1	81.6	52.9	48.9	69.8
SpeFiNet, Cluster†	83.7	92.5	79.8	71.9	**66.7**	**74.8**	**63.6**	83.1	54.5	49.5	**72.0**
SpeFiNet, Cluster$(K = 16)$	83.3	92.2	79.6	71.9	66.6	74.1	62.5	83.0	55.1	49.8	71.8
SpeFiNet, BUCBAM	83.2	92.2	79.6	71.7	66.6	74.5	62.7	**83.4**	**56.1**	**50.0**	**72.0**

Table 2: Comparison of our methods (bottom) to the state-of-the-art (top). All the methods are trained on the data of *full* ILSVRC with a DarkNet network and compared in terms of average performance (Avg) on the set of *ten* target-tasks presented in Sec 3. For each benchmark, we highlight the best score in bold and the second is underlined. Methods marked with † are obtained with a parameter cross-validated on the target-tasks. BUCBAM automatically set this parameter on the source-task.

3.1 Comparison to the State-of-the-Art

We compare the results obtained by our method with those of the literature, in particular, all the methods re-implemented and reported in [36]. For fair comparisons, we followed their training configuration, and trained our method with an AlexNet network and ILSVRC* as source-task. Moreover, instead of using our more diverse set of ten target-datasets, we used the eight ones used in their paper. The results are reported on Table 1. We first observe that our methods are always better than the reference method used in [2, 28, 29, 34, 36], namely SpeNet. In particular our best method (BUCBAM) exhibits a boost of 6 points on average, compared to SpeNet. Let also note that SpeFiNet is always better than FiNet, itself better than SpeNet, regardless the splitting method. More precisely, the BUCBAM splitting method is significantly better than the best Cluster one, without the high cost of cross-validation of the K parameter. Compared to state-of-the-art methods, ours achieves the best performances, that is, almost 2 points of improvement compared to the most competitive MulDiP-Net method [36], while it surpasses all other methods by more than 3 points. A last salient result is the fact that a SpeFiNet (whatever the splitting method) is significantly better than MuCaLe-Net [34] which has been trained on the best generic categories (manually obtained from categorical-levels [33, 34]). This latter clearly demonstrates than combining features trained on specific categories with those trained on *finer* categories is better than combining them with those trained on *generic* categories.

Furthermore, since [36] reported better results with a deeper network (DarkNet) trained on the full ILSVRC, we also implemented our method in the same configuration. Since MulDiP-Net provides the best results of the literature on the problem of universality, we only compare to them for this setting. The results are reported on Table 2. While the improvement is only slightly better, our method still beats the competitive MulDiP-Net. Moreover, a salient observation is that our method tend to be much better than theirs on the fine-grained classification benchmarks, which are more challenging. As in the previous setting, SpeFiNet is better than FiNet which is itself better than SpeNet. We also compared to the GenNet (which is the generic sub-component of MulDiP-Net [36]) and we observe that FiNet-BUCBAM is better by 0.9 points.

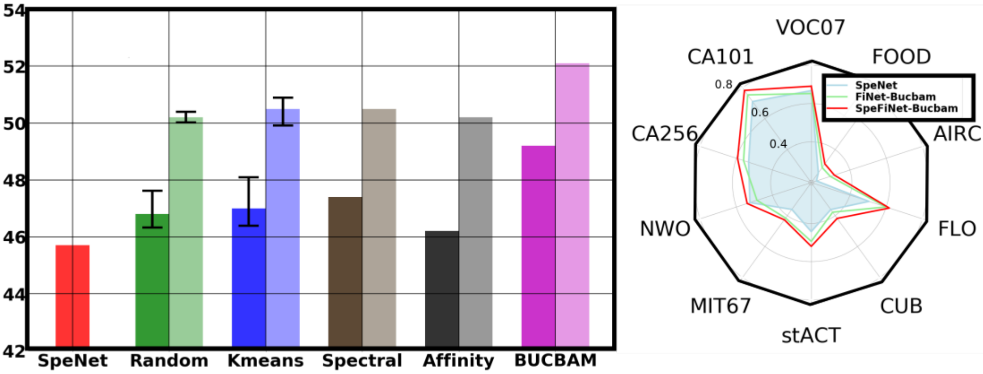

Figure 3: Comparison of our methods to multiple splitting baselines. On the left, we illustrate the average results of each method on the ten target-tasks, were for each splitting method, we illustrate the FiNet in dark color and SpeFiNet in light color. On the right, we plot a diagram for the SpeFiNet-BUCBAM, FiNet-BUCBAM and SpeNet methods, their performances on each target-task. Best view in color

3.2 Analysis and Comparison to Baselines

In this section, we perform an in-depth analysis of our method through an ablation study, a comparison to baselines and visualization of some statistics. In particular, in supplementary we compared our method to multiple clustering baselines, namely Spectral Clustering and Affinity propagation. The former provides a fixed set of clusters per category, while the latter leads to a dynamic set of clusters. A summary of results is presented on the left of Fig. 3, where we plot, a bar for each method, that represents its average performance on the set of 10 benchmarks described in Sec 3. We also tested the Mean-Shift algorithm with many different bandwidth values, but it always led to many clusters with one or two images, and one cluster containing all the remaining images. This setting providing very low results, we did not report them. From these results, we observe that our BUCBAM method is better than all the baselines including other existing algorithms. Rather than the average performances, in the diagram on the right of Fig. 3, we illustrated the *detailed* results (on the ten benchmarks) of the SpeFiNet-BUCBAM, FiNet-BUCBAM and SpeNet methods. We clearly observe that the diagram of our SpeFiNet-BUCBAM overlaps FiNet-BUCBAM, which itself overlaps the reference SpeNet method. In addition, we provide in supplementary material the detailed results of all the methods on all the target-tasks.

In Figure 4, we visualize some of the clusters obtained by each splitting method (random, clustering and BUCBAM). To do so, we highlight three clusters for two specific categories (two blocks of three rows of five images). On the left, the clusters are determined from a random distribution within the full specific category, leading to clusters that contain its full diversity. On the contrary, with the K-means clustering (middle), the clusters exhibits a more coherent aspect. For example, for the *goldfish* category, the c_3^1 cluster report close-up views of fish that are rather seen on their profile. We have a similar behaviour for the *banjo* category with cluster c_1^3 and c_2^3. With the proposed BUCBAM method (right), the clusters are even more specific than in the K-means case. For instance, for the *goldfish* category, we clearly identify a cluster that represents “many golfishes” (c_1^1), “on goldfish in a close-up view” (c_2^1) and some images on which the fish tank is visible (c_3^1). Also for the *banjo* class,

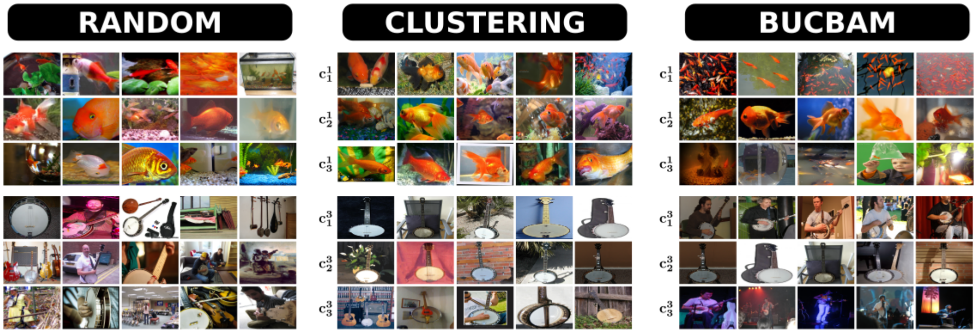

Figure 4: Illustration of some finer categories obtained from two specific categories (two row blocks) with the different methods: random split (left), Kmeans clustering with K=16 (middle) and our BUCBAM proposal (right). In every block, a line (from the three) shows the five most representative images of a cluster at the new finest-level. Best view in PDF.

we also clearly observe that our method identified a cluster that represents "person playing banjo" c_1^3 and even "person playing banjo in a concert" c_3^3. Importantly, while the clustering method tend to results in duplicate clusters (*e.g.*, c_2^1 with c_3^1; c_1^3 with c_2^3; etc.), ours tend to provide only dissimilar results, thank to our merging process.

In supplementary material, we also provided some statistics of our method and baselines (*i.e.*, histograms of average amount of clusters per category, histograms of intra-class variance of the clusters) and more visualizations of the obtained clusters, through the visualization of some images in some clusters and the features of each image of the clusters in a 2D dimensional space, after performing a PCA on their full features. This highlights the clear interest of our method, in terms of cluster relevancy and the balance of resulting data, compared to the random and clustering baselines.

4 Conclusions

In this paper, we tackled the problem of universality of representations with a new method relying on categories that are finer than the most specific ones of the ImageNet hierarchy. These last being the finest that are annotated, we proposed a method that automatically add a hierarchical-level to the ImageNet hierarchy. A network trained on the categories of such finer-level provides a more universal representation than with the upper levels. In practice, it leads to significantly better results in a transfer-learning scheme, on 10 publicly available datasets from diverse domains.

We also showed that a K-means and, surprisingly, a random partitioning of the leaf nodes of ImageNet already gives interesting results, although below than the proposed approach. It nevertheless suggests that the general principle highlighted in this article could be fruitful to design new CNN-based representations that are more universal in a transfer-learning context. Furthermore, it should be noted that our principle is neither limited to the ImageNet hierarchy nor to the classification task. Indeed, it could be applied to any hierarchy or dataset and on other tasks, such as detection, segmentation or keypoint estimation, as considered in [40].

References

[1] Hossein Azizpour, Ali Razavian, Josephine Sullivan, Atsuto Maki, and Stefan Carlsson. Factors of transferability for a generic convnet representation. *PAMI*, 2015.

[2] H. Bilen and A. Vedaldi. Universal representations: The missing link between faces, text, planktons, and cat breeds. *arXiv:1701.07275*, 2017.

[3] Lukas Bossard, Matthieu Guillaumin, and Luc Van Gool. Food-101 – mining discriminative components with random forests. In *ECCV*, 2014.

[4] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

[5] Ines Chami, Youssef Tamaazousti, and Hervé Le Borgne. Amecon: Abstract meta concept features for text-illustration. In *ICMR*, 2017.

[6] Tao Chen, Shijian Lu, and Jiayuan Fan. S-cnn: Subcategory-aware convolutional networks for object detection. *PAMI*, 2017.

[7] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yan-Tao. Zheng. Nus-wide: A real-world web image database from national university of singapore. In *ACM Conference on Image and Video Retrieval*, CIVR, 2009.

[8] Alexis Conneau and Douwe Kiela. Senteval: An evaluation toolkit for universal sentence representations. *arXiv preprint arXiv:1803.05449*, 2018.

[9] Alexis Conneau, Douwe Kiela, Holger Schwenk, Loic Barrault, and Antoine Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

[10] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR*, 2009.

[11] Jian Dong, Wei Xia, Qiang Chen, Jianshi Feng, Zhongyang Huang, and Shuicheng Yan. Subcategory-aware object classification. In *CVPR*, 2013.

[12] Jian Dong, Qiang Chen, Jiashi Feng, Kui Jia, Zhongyang Huang, and Shuicheng Yan. Looking inside category: subcategory-aware object recognition. *Transactions on Circuits and Systems for Video Technology*, 2015.

[13] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge. *IJCV*, 2010.

[14] Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *PAMI*, 2006.

[15] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007.

[16] Minyoung Huh, Pulkit Agrawal, and Alexei A Efros. What makes imagenet good for transfer learning? *arXiv:1608.08614*, 2016.

[17] Armand Joulin, Laurens van der Maaten, Allan Jabri, and Nicolas Vasilache. Learning visual features from large weakly supervised data. In *ECCV*, 2016.

[18] Iasonas Kokkinos. Ubernet: Training a universal convolutional neural network for low-, mid-, and high-level vision using diverse datasets and limited memory. In *CVPR*, 2017.

[19] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[20] Lingqiao Liu, Lei Wang, and Xinwang Liu. In defense of soft-assignment coding. In *ICCV*, 2011.

[21] S. Maji, J. Kannala, E. Rahtu, M. Blaschko, and A. Vedaldi. Fine-grained visual classification of aircraft. Technical report, 2013.

[22] Pascal Mettes, Dennis Koelma, and Cees G. M. Snoek. The imagenet shuffle: Reorganized pre-training for video event detection. In *ICMR*, 2016.

[23] Maria-Elena Nilsback and Andrew Zisserman. Automated flower classification over a large number of classes. In *IEEE Computer Vision, Graphics & Image Processing*, 2008.

[24] Maxime Oquab, Leon Bottou, Ivan Laptev, and Josef Sivic. Learning and transferring mid-level image representations using convolutional neural networks. In *CVPR*, 2014.

[25] Adrian Popescu, Gadeski Etienne, and Hervé Le Borgne. Scalable domain adaptation of convolutional neural networks. preprint arXiv:1512.02013, 2015.

[26] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, 2009.

[27] Ali Sharif Razavian, Hossein Azizpour, Josephine Sullivan, and Stefan Carlsson. Cnn features off-the-shelf: an astounding baseline for recognition. *CoRR*, 2014.

[28] S-A. Rebuffi, H. Bilen, and A. Vedaldi. Learning multiple visual domains with residual adapters. In *NIPS*, 2017.

[29] Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. Efficient parametrization of multi-domain deep neural networks. In *CVPR*, 2018.

[30] Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *CVPR*, 2017.

[31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. Imagenet large scale visual recognition challenge. *IJCV*, 2015.

[32] Sandeep Subramanian, Adam Trischler, Yoshua Bengio, and Christopher J Pal. Learning general purpose distributed sentence representations via large scale multi-task learning. In *ICLR*, 2018.

[33] Youssef Tamaazousti, Hervé Le Borgne, and Céline Hudelot. Diverse concept-level features for multi-object classification. In *ICMR*, 2016.

[34] Youssef Tamaazousti, Hervé Le Borgne, and Céline Hudelot. Mucale-net: Multi categorical-level networks to generate more discriminating features. In *CVPR*, 2017.

[35] Youssef Tamaazousti, Hervé Le Borgne, Adrian Popescu, Etienne Gadeski, Alexandru Ginsca, and Céline Hudelot. Vision-language integration using constrained local semantic features. *CVIU*, 2017.

[36] Youssef Tamaazousti, Hervé Le Borgne, Céline Hudelot, Mohamed El Amine Seddik, and Mohamed Tamaazousti. Learning more universal representations for transfer-learning. *arXiv:1712.09708*, 2018.

[37] Phong Vo, Alexandru Lucian Ginsca, Hervé Le Borgne, and Adrian Popescu. Effective training of convolutional networks using noisy web images. In *proc. 13th International Workshop on Content-Based Multimedia Indexing (CBMI 2015)*, 2015.

[38] Phong D Vo, Alexandru Ginsca, Hervé Le Borgne, and Adrian Popescu. Harnessing noisy web images for deep representation. *CVIU*, 2017.

[39] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The caltech-ucsd birds-200-2011 dataset, 2011.

[40] Jingyan Wang, Olga Russakovsky, and Deva Ramanan. The more you look, the more you see: towards general object understanding through recursive refinement. In *WACV*, 2018.

[41] Yu-Xiong Wang, Deva Ramanan, and Martial Hebert. Growing a brain: Fine-tuning by increasing model capacity. In *CVPR*, 2017.

[42] Yue Wu, Jun Li, Yu Kong, and Yun Fu. Deep convolutional neural network with independent softmax for large scale face recognition. In *ACM*, 2016.

[43] Yu Xiang, Wongun Choi, Yuanqing Lin, and Silvio Savarese. Subcategory-aware convolutional neural networks for object proposals and detection. In *WACV*, 2017.

[44] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas Guibas, and Li Fei-Fei. Human action recognition by learning bases of action attributes and parts. In *ICCV*, 2011.

[45] Bolei Zhou, Agata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *NIPS*, 2014.