Effective Training of Convolutional Networks using Noisy Web Images

Phong D. Vo, Alexandru Ginsca, Hervé Le Borgne, Adrian Popescu
CEA, LIST, Vision and Content Engineering Laboratory, France
Email : {dinphong.vo,alexandru.ginsca,herve.le-borgne,adrian.popescu}@cea.fr

***Abstract*—Deep convolutional networks have recently shown very interesting performance in a variety of computer vision tasks. Besides network architecture optimization, a key contribution to their success is the availability of training data. Network training is usually done with manually validated data but this approach has a significant cost and poses a scalability problem. Here we introduce an innovative pipeline that combines weakly-supervised image reranking methods and network fine-tuning to effectively train convolutional networks from noisy Web collections. We evaluate the proposed training method versus the conventional supervised training on cross-domain classification tasks. Results show that our method outperforms the conventional method in all of the three datasets. Our findings open opportunities for researchers and practitioners to use convolutional networks with inexpensive training cost.**

I. Introduction

Convolutional networks (convnets) have been a breakthrough in computer vision in the last years. They have achieved superior performances in various problems such as object recognition [1], object detection [2], image segmentation [3], image annotation [4] or image retrieval [5, 6]. Their key success is the capability to learn abstract representations of images from pixel-level, thanks to their deep structure. In a seminal paper [1], a convnet is trained using 1.2 millions images in order to classify 1000 ImageNet synsets. Besides superior accuracy it performs on this challenge, the convnet model in question is highly reusable in other problems and also obtains top performances (see the extensive experiments detailed in [7, 8]). Convnets are easily adapted to new domains; if a target domain has sufficient training data, then a pre-trained convnet could be tuned to improve performance [9].

One important factor that is a key ingredient for the success of convnets is the large volume of training data. Despite the fact that fine-tuning works well on many mid-size datasets, it is not an universal solution for any problems. Furthermore, it is preferable to train a convnet from scratch in order to obtain the best classification accuracy, given a sufficiently large amount of training data [10]. However this solution is expensive and time-consuming.

We consider images on the Web as being the most abundant and comprehensive data source for convnets training. Using Bing and Flickr we can freely download hundreds of thousands images. Downloaded images are pretty noisy; however we can reduce it using image reranking techniques. The simplest one is unsupervised reranking, in which we perform cross-validation [11] over the raw Web images in order to remove noisy images and outliers. Because the selection process of this reranking method is unguided, reranked results may be misled in the case where noisy images are predominant. We therefore investigate two weakly-supervised reranking techniques called Kernel Mean Matching (KMM) [12] and Transductive SVM (TSVM) [13]. For each of these methods to work, few examples are needed to be supplied, so that the algorithm prioritizes the images which are similar to the provided examples towards to the top of the reranked lists. Based on reranking results, several training pipelines are proposed to combine training methodologies (train from scratch or fine-tuning) and data (without reranking, unsupervised or weakly-supervised reranking). We found that merely using Web images for convnet training already gives promising results. If fine-tuning using reranked Web images is applied to a pre-trained convnet, then classification accuracy is significantly improved. For example, a convnet trained with 300K unlabeled Web images gives 53.6% accuracy on ImageNet test data, while finely tuning this convnet with approximately 80K reranked images resulted from running KMM with 10 examples per synset boosts the accuracy up to 60%. On the contrary, in order to obtain 68% accuracy on the same test set, we have to train the same convnet using 100K annotated images. A first interesting finding is that using noisy Web images and 1% of the annotation effort mentioned above already gives us 60% accuracy. Of equal importance is the fact that the convnet trained by our approach provides higher mAP scores when testing on third-party datasets, such as VOC Pascal, MIT Scenes, and Oxford Flowers. The method proposed in this paper opens opportunities for convnets to be trained with easily obtainable data.

II. Related Work

The deep network hierarchy of convnets is a stack of learnable layers of convolutions and fixed layers pooling, ReLU (Rectifier Linear Unit), and dropout. The more number of layers, the greater nonlinearity of the convnet; the more feature maps (uniquely determined by the convolutional layers), the higher dimensionality of the learned representation. Convnets are less sensitive against translation variances thanks to convolutional layers. Pooling layers reduce spatial dimensionality and improve feature robustness. ReLU layers resolve the problem of gradient vanishing, which is inherently traditional to neural networks that use sigmoid activation functions. It is important to design networks w.r.t some principles. A typical case-study is the AlexNet presented in [1]. At the lower part of the network, the convolutional, pooling, and ReLU layers are placed interlacingly; such sub-structures are stacked several times to provide higher levels of abstraction for low-level

978-1-4673-6870-4/15/$31.00

(a) Images from Bing (b) Images from Flickr

Fig. 1: Image examples of three synsets (*airframe*, *baker*, *fishnet* in the database *FlickrBing100*). Notice the difference between the two data sources in which Bing images are more document-oriented and Flickr images are more aesthetic and personal.

signals. Placed at the middle are the dropout layers which prevent overfitting. At the top of the network there are three fully-connected layers, taking the role of a softmax classifier. It is not yet understood how convnets learn complex features from raw input data. However, we can get an intuition about those features by visualizing their intermediate layers (for example, see [14]).

A convnet's good generalization property, which means it is highly reusable across domains, has been widely acknowledged. Despite preliminary understandings about convnets, a rigorous explanation on what and how convnets learn complex features remains unanswered. This may slows down research efforts in improving the learning capability of convnets or their capability to learn under weakly supervised settings. In terms of transferring pre-trained convnet models into new domains with mid-size datasets, there have been pioneer works [5, 9, 15] that tune a pretrained convnet into new data domains, for instance by re-training their top layers using data of the target domains. These studies confirm some of the advantages of fine-tuning such as: i) much less data is required for fine-tuning than training from scratch, ii) training time is reduced, iii) a pretrained network can be easily tweaked by modifying its top layers for particular needs. However, fine-tuning is not a long-term solution for much larger-scale problems.

People can create a image collections for new domains. For instance in order to recognize scenes, [10] collect data for 205 scene categories of their Places database. When training from scratch using the Places database, the resulting convnet performs significantly better than the adapted pre-trained models. However creating such a new image collection is daunting and expensive.

An alternative is to use unsupervised learning methods. Autoencoders, deep belief networks, and restricted Boltzmann machines are such examples and they also achieve remarkable successes [16, 17]. Deep unsupervised belief nets can learn the *cat*, *face*, and *human body* concepts from from unlabeled data [16]. However those deep networks require a huge amount of data and computational resources, i.e. 16,000 CPU cores were running in many days. Thus it is impractical to reproduce those results without having sufficient computational resources. In this aspect using supervised learning for convnets are still preferable than the unsupervised methods because the former does not requires much on computational resources.

A recent work [18] proposes to use supervised convnets with surrogated training data. In particular they extract patches from many images and for each of the patches a surrogate class is created and its instances are generated by applying image jittering effects on the patch. By generating up to 8000 surrogate classes, which are equivalent to 8000 original patches, those data are used to train a convnet. High classification accuracy is reported on CIFAR-10 dataset when using feature vectors extracted from the trained convnet to train linear SVM classifiers. This approach however oversimplifies the diversity of image representations that appear in real life scenarios, thus it is unlikely perform well with complex datasets such as ImageNet or Places.

III. FlickrBing100: A Noisy Web Image Database

From our point of view, a plausible way to extend the application range of convnets is to exploit valuable image sources from the Internet. Nowadays, text-based search engines such as Google and Bing or social image sharing platforms such as Flickr and Instagram provide good quality image retrieval services in which retrieved images are quite relevant to their queries.

However, Web images are not ready to be used due to noises. The text part of images, which is used to index those images, may not reflect correctly visual content. As a consequence machine learning methods will be struggling in coping with levels of noise. There are very few studies about training a convnet using noisy data. Recently, [19] proposed to add a denoising layer into a convnet structure in order to correct wrong predictions caused by noisy labeled data. However, [19] only tackles "closed" labels, which means that images may be wrongly labeled by predefined classes of a database. And there are no images coming from classes not mentioned by the training database. This assumption is impractical to the extent that it is unfeasible to use their model to train convnets using Web images which may contain any type of noise.

A. Database Acquisition

We want to train convnets using Web images retrieved from Bing and Flickr. We created an image database (*FlickrBing100*) for which 100 synsets are randomly picked from the lexical database Wordnet. Our selection mixes leaf nodes (specific synsets) and inner nodes (general synsets) of the Wordnet hierarchy. A synset may have several synonyms and we use all of them as queries; for each of the query we download the top 1000 images from the returned list. The number of images per synset, thus varies from 1000 to several thousands. Thanks to the Bing API, we can freely download images with a quota of 5000 queries per month

per account. Similarly, the Flickr API allows us to download from 1000 to 5000 images per query without any limitation. Corrupted images in Flickr and Bing collections are removed; duplicating images are detected and removed by comparing SHA-256 signatures. Out of 416K images of the database, Flickr contributes 279K and Bing contributes 137K images.

FlickrBing100 is noisy. For instance, Fig. 1 shows that the Bing images representing the *fishnet* synset contain images not only of fishnet but also images having similar textures such as zebras or somethings called "net" such as woman fishnet socks and logos of Microsoft.NET; Flickr images, on the contrary, are more biased to personal interests and aesthetic quality. Another example is the *airframe* synset; while Bing provides more documentary images about blueprints and diagrams of flight models, Flickr images are more about realistic airplanes and aerial images taken from consumer cameras. This difference is rooted in the ways of how Bing and Flickr index and retrieve images. Bing prioritizes images having rich accompanying text relevant to queries so that it prioritizes images which are picked from books, lectures, articles, etc. On the contrary, Flickr indexes and organizes the images according to users' tags. Flickr images might be very well organized with respect to a concept but they are also highly biased toward individual users.

By adopting images from multiple sources, we have both advantages and challenges. Images retrieved from these sources will cover better intra-variances. However, more data sources induces more noises. Thus it is indispensable to apply image reranking techniques to reduce noise and improve database consistency. Frankly it is so difficultly to completely remove all of noisy images that we are rather interested in seeing whether noise could be helpful to convnet training.

B. Database Evaluation

As *FlickrBing100* is intended as an alternative for convnet training, a baseline evaluation is necessary for our training approach to compare against the typical way of training convnets that uses clean data. The best option for the baseline is to use ImageNet, which is a comprehensive online collection of 14 million labeled images organized based on the Wordnet structure. By drawing 1000 images from each of the 100 synsets of *FlickrBing100*, we construct the baseline training database called *ImNet100*. The baseline test set is also prepared in the same way, with 100 images per synset. This test set is used to evaluate the convnet trained from the baseline training set *ImNet100* and the one trained from *FlickrBing100*. The baseline has the advantage of being trained and tested on the same data domain ImageNet in contrast to the convnets trained by our approach (i.e. trained on Web images and tested on ImageNet images). Nevertheless this coincides to what we are trying to achieve, to train convnets which are good for cross-domain classification.

Since the baseline could easily obtain superior performance, we are more interested in whether a convnet trained on a reranked *FlickrBIng100* close the gap to the baseline. To this end we draw 10 example images from each of the *ImNet100*'s synsets and use those examples as labeled data for the image reranking algorithms. After reranking, *FlickrBing100*, is expected to be cleaner which might facilitate better convnet training. This approach, if it improves the training outcome, will be very easy to reproduce in practice since the cost for annotating examples is negligible comparing to annotating the whole big collection in typical training setting.

Going beyond ImageNet, it is desirable to compare the generalization capability between the convnets trained by *ImNet100* and the ones trained by *FlickrBing100* on third-party datasets which are not biased to any of the two aforementioned training databases. This evaluation will further reveal whether training a convnet using clean images or noisy Web images will result to better generalization.

IV. Image Reranking

We first define mathematical notations for formulating the problem. Given an arbitrary query, which is a synset name in our context, let us call $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ the set of examples and $\mathcal{U} = \{\mathbf{z}_j\}_{j=1}^n$ the set of retrieved images from a particular source. Here, $\mathbf{x}_i$ and $\mathbf{z}_j$ are the vectorial representation of an labeled and unlabeled images respectively; y_i is the label index of $\mathbf{x}_i$. Furthermore we assume that $m \ll n$. We aim to select a subset $\mathcal{S} \subset \mathcal{U}$ such that elements in $\mathbf{x}_i \in \mathcal{S}$ are relevant, at least to one of the elements in $\mathcal{L}$; the complement $\mathcal{U} \backslash \mathcal{S}$ is considered as being irrelevant to $\mathcal{L}$. In the following, we present three reranking methods used in our experiments. The first method is unsupervised (i.e. dos not require labeled examples), while the rest are weakly-supervised.

A. Cross-Validation (CV)

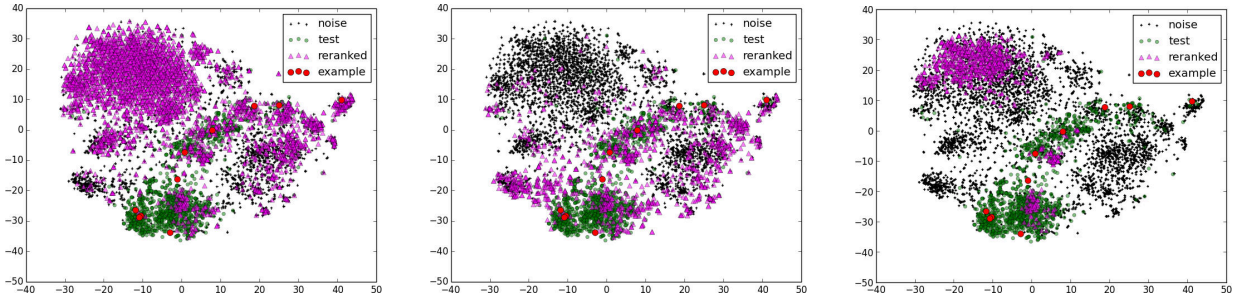
This method first randomly splits $\mathcal{U}$ into K equal disjoint subsets. Then, a binary SVM classifier is trained using $(K-1)$ subsets as positive data, while the remaining subset will be the test set. In order to train classifiers, it is necessary to have negative data in which images are irrelevant to the 100 synsets. We therefore prepare a negative set of approximately 10K images; this set is commonly used between subsets. We apply the cross validation scheme and accumulate prediction scores so that every data point in $\mathcal{U}$ is predicted K times; points with negative scores are considered as noise and rejected. K should not be very large, otherwise the classifier trained from $(K-1)$ positive subsets will classify the whole K-th set as positive. Additionally, setting a low value for the regularization coefficient C allows the SVM hyperplane to seek for low density regions of data, which maximizes its margin.

B. Kernel Mean Matching (KMM)

While cross-validation reranks images using classifiers' scores, KMM re-weights data points $\mathbf{z}_i \in \mathcal{U}$ w.r.t $\mathbf{x}_i \in \mathcal{L}$ for minimizing the mismatch between the empirical means of $\mathcal{L}$ and $\mathcal{U}$. The set $\mathcal{S}$ is constituted by adding $\mathbf{z}_i$ for which the re-weighted score is large. Denoting $\boldsymbol{\alpha} \in \mathbb{R}^n$ as the weight vector, then $\boldsymbol{\alpha}$ is the optimal solution of the following convex quadratic equation:

$$\arg\min_{\boldsymbol{\alpha} \succeq 0} \frac{1}{2} \left\| \frac{1}{m} \sum_{i=1}^{m} \phi(\mathbf{x}_i) - \frac{1}{n} \sum_{j=1}^{n} \alpha_j \phi(\mathbf{z}_j) \right\|_{\mathcal{H}}^2, \qquad (1)$$

where function $\phi(\cdot)$ maps input data $\mathbf{x}_i$'s into a high dimensional Hilbert space $\mathcal{H}$ whose inner product $\langle \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) \rangle = \kappa(\mathbf{x}_i, \mathbf{x}_j)$. Here, $\kappa(\cdot, \cdot)$ is a symmetric positive semidefinite



(a) Cross-Validation (b) Kernel Mean Matching (c) Transductive SVM

Fig. 2: This visualization highlights the differences between the three methods (a), (b), and (c) in selecting relevant images from the image synset *salmon*. Method (a) is unsupervised so that red circles (examples) have no effect in selection. Method (b) selects (triangle pink) points in order to close the mean gap between the reranked data and the examples. Method (c) selects points based on density and the fixed ratio r. Red and green points come from *ImNet100* while black and pink points come from *FlickrBing100*. Note that both (a) and (c) use additional negative images, which are not visualized here.

kernel. For increased efficiency, we use linear kernel which means $\mathcal{H} \equiv \mathbb{R}^n$ and $\phi(\mathbf{x}) = \mathbf{x}$. Thus Eq. (1) could be re-written as the following quadratic form:

$$\begin{array}{ll} \arg\min\limits_{\boldsymbol{\alpha}} & \frac{1}{2}\boldsymbol{\alpha}'\left(\mathbf{Z}'\mathbf{Z}\right)\boldsymbol{\alpha} - \frac{n}{m}(\mathbf{Z}'\mathbf{X}\mathbf{1})'\boldsymbol{\alpha} \\ \text{s.t} & 0 \leq \boldsymbol{\alpha} \leq B \\ & m(1-\epsilon) \leq \boldsymbol{\alpha}'\mathbf{1} \leq m(1+\epsilon) \end{array}, \quad (2)$$

where the first constraint defines a scope bounding discrepancy between the distributions $P_{\mathcal{L}}$ and $P_{\mathcal{U}}$. The bigger B is, the lesser points $\mathbf{z}_i$ are weighted with high values. As $B \to 0$, an unweighted solution is obtained, which is unwanted. The second constraint ensures the measurement $\boldsymbol{\alpha}(x)P(x)$ is close to a probability distribution (for further details see [20]).

C. Transductive Support Vector Machine (TSVM)

This algorithm is the application of SVM into a transductive setting, where both m training samples and n test samples (their labels are unknown) are simultaneously used to learn a classifier and predict labels for the test data. The assumption $m \ll n$ of TSVM is very appropriate for our reranking context. The goal therefore is to minimize the following problem:

$$\arg\min_{\mathbf{w},\{t_j\}} \frac{1}{2}\|\mathbf{w}\|_2^2 + \frac{\alpha}{m}\sum_{i=1}^{m}\ell(y_i\mathbf{w}'\mathbf{x}_i) + \frac{\beta}{n}\sum_{j=1}^{n}\ell(t_j\mathbf{w}'\mathbf{z}_j), \quad (3)$$

where the hinge loss $\ell(x) = \max[0, 1-x]$ is used by default; non-negative coefficients α and β control the influence of positive examples $\{\mathbf{x}_i\}$ and unlabeled images $\{\mathbf{z}_i\}$ on the classifier $\mathbf{w}$. Labels $\{t_j\}$ of $\{\mathbf{z}_j\}$ are valued at running time. In order to make sure that not all of the unlabeled data belongs to either the positive or negative class, a ratio $r = \frac{1}{n}\sum_{j=1}^{n}\max[0, \text{sign}(\mathbf{w}'\mathbf{z}_j)]$ is kept fixed during optimizing (3). This ratio reflects the belief of how many clean images are found in $\mathcal{U}$. The idea of TSVM is to separate clean images from noise by finding a (sub)optimal partition $\{t_j\}$ conditioned on the labeled set $\mathcal{L}$ and unlabeled set $\mathcal{U}$.

D. Experiment Settings

In order to rerank images, our first step is to extract the descriptors. We use the pretrained convnet AlexNet as the extractor. In particular, each image is forwarded from the input data layer so that its corresponding output at `fc7` layer is flattened into a feature vector of size 4096. Every feature vector is then L_2-norm normalized. Other parameters are specifically set as follows.

CV: We SVM with linear kernel, $C = 1$, and $K = 5$. The same negative training examples set of size 10K is used for each of the synset.

KMM: Setting B high ($B > 10$) will increase visual consistency of $\mathcal{S}$ w.r.t $\mathcal{L}$ which leads to fewer images being chosen ($|\mathcal{S}|$ is small). Since a convnet prefers abundant training data, we set $B = 5$. Notice that we do not set $B = 1$ because KMM will give equal weights to all of the data. An image will be added to $\mathcal{S}$ if its re-weighted score is larger than 0.01.

TSVM: We set $r = 1000/n$ simply because we expect there are 1000 relevant images out of n. The rest of the coefficients are set as follows: $\alpha = 1$ and $\beta = 0.0001$, meaning Eq.(3) must give priority to correctly classified examples.

E. Qualitative Results

Since we have no groundtruth for Web images, the effectiveness of reranking methods can be indirectly measured based on the convnet's performance. However, we can still get some intuition on the reranking results by visualizing the data using t-SNE [21]. Fig.2 shows the visualization of the embedding obtained by reducing the dimensionality of *salmon* data. In Fig. 2a, we can observe that the method CV tends to reject clusters which lie remotely from bigger clusters. It also rejects points at low density areas.

In Fig. 2b, KMM does exactly what it is designed for. It selects the (magenta) points such that they form a "Gaussian shaped" cloud covering (red) examples and therefore the two empirical means are better matched. Notice how KMM reranks images regardless of the nature of data clustering.

In Fig. 2c, TSVM reranks images quite different from other methods. On one hand, it selects a portion of images from each of cluster containing the examples, which is quite similar to KMM. On the other hand, it also selects data points from high density clusters, although there is no example coming from those, which is similar to the behavior of CV. Since TSVM learns classifiers to rerank images, images near the examples (positive training data) will be highly ranked.

Furthermore, TSVM must obey the fixed ratio $r = 1000/n$ (see Section IV-C) so that approximately 1000 images must be classified as positive; this explains why data points of large clusters are included in the reranking result.

Visualizations allow us to derive some hypotheses about reranking algorithms: i) both CV and TSVM are able to maintain the diversity of reranked data, however CV is less noise resilient than TSVM; ii) KMM is better than TSVM in reranking images w.r.t examples. However, using KMM leads to reduced visual diversity in the reranked list; iii) the fixed ratio r used in TSVM is quite restrictive because it may prevent selecting more relevant images as well as rejecting noisy images. Notice that all of the three methods may obtain better reranking results if non-linear kernels are used. However, we will not run into suboptimality since a slightly better reranking may be negligible to convnet training.

V. Training and Evaluation

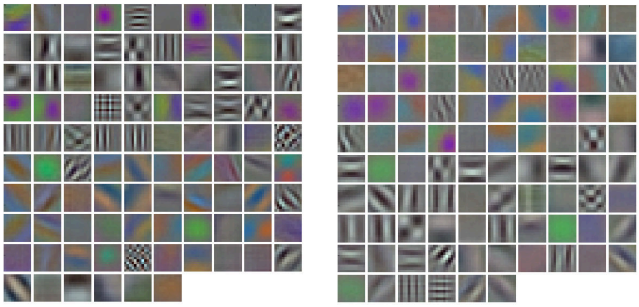
To evaluate our approach, we train four convnets for an image classification task using different training databases: FB (the original $FlickrBing100$ without reranking), FB_{CV} (reranked FB by CV), FB_{KMM} (reranked FB by KMM), and FB_{TSVM} (reranked FB by TSVM). Results of these convnets are compared to each other and against the $baseline$ convnet (trained from $ImNet100$), using the testset extracted from the ImageNet collection. The number of training images is listed in the first row of Table I.

Normally, the convnets trained from Web images could not compete against the $baseline$. Therefore we seek for a more efficient training pipeline. The best performing convnet(s) trained from Web images is used as a pre-trained model to do fine-tuning using reranked subsets. Going beyond ImageNet evaluation, we further evaluate the generalization capability of those convnets on three third-party datasets: VOC Pascal 2007, Oxford 102Flowers, and MIT 67Scenes. Results on these datasets will reveal which training approaches are better for cross-domain problems.

We adopt the AlexNet network configuration for all of the convnets and use the Caffe toolbox [22] to train them on a single GTX Titan Black graphics card. The starting learning rate for all of the training from-scratch is fixed to $\eta = 0.01$ and reduced by a magnitude of $1/\gamma = 10$ every $50K$ iterations. The training stops at 400K iterations. When performing fine-tuning, those parameters are $\eta = 0.001$, $1/\gamma = 10$, $30K$ iterations, and $120K$ iterations, respectively. The validations sets for training or fine-tuning are extracted from 10% of the corresponding training databases.

A. Evaluation on ImageNet data

The second row of Table I reports the classification accuracy of the baseline convnet and that trained with our data. Obviously the baseline performs best among them, at 67.8%. The FB_{CV} comes next with 55% and FB is in the 3^{rd} position, with 53.6%. This means that the cross-validation method is useful for eliminating around $100K$ spurious images. However, the two reranking methods KMM and TSVM are not helpful when doing the training from-scratch. By removing more than 75% of the FB in order to keep FB_{KMM} and FB_{TSVM} well aligned w.r.t examples, their classification accuracy drop to 48% and 49.6% respectively. Therefore it is not that cleaner (and smaller) data was the most important contribution accounting for network's performance but bigger data. The FB_{KMM} is somehow more consistent w.r.t synset examples than FB (as illustrated in Fig. 2). However, FB_{KMM} is four times smaller than FB so that FB_{KMM} could not help improving the corresponding convnet model. The $baseline$ database is approximately the same size as FB_{KMM} but its images are manually annotated so that it does not present interest for our approach. The take-home message for Web images-based training is *"quantity comes first."*



(a) Our convnet (b) $baseline$

Fig. 3: Visualizations of `conv1` layers. Low-level features learned from web images are more comprehensive than from ImageNet. Many Gabor-like filters just present in (a). Similarly, some high-frequency filters present in (a) but not in (b).

We then wonder whether training convnets using Web images could be further improved. Recall that the major advantage of Web images is the abundance of images which provides diversified representations for visual synsets. This characteristic promotes primitive feature learning of convnets at low layers (see Fig. 3). However, the lack of labeled data makes convnets difficult to learn more synset-oriented features thus reducing discrimination power at the softmax output layer. In order to make our convnets perform better on ImageNet data, we have to fine-tune their top layers using reranked data. Two training pipelines are dedicated for this purpose, in which the model to be tuned was trained from FB_{CV}. FB_{KMM} and FB_{TSVM} are the two fine-tuning databases. Particularly,

$$\begin{array}{ll} \frac{|FB_{KMM}|\cap|FB_{CV}|}{|FB_{KMM}|} = 0.79, & \frac{|FB_{TSVM}|\cap|FB_{CV}|}{|FB_{TSVM}|} = 0.98 \\ \frac{|FB_{KMM}|\cap|FB_{TSVM}|}{|FB_{TSVM}|} = 0.41, & \frac{|FB_{KMM}|\cap|FB_{TSVM}|}{|FB_{KMM}|} = 0.5 \end{array} \quad (4)$$

While these two databases are approximately the same size and share 50% duplicate images, FB_{TSVM} has more images in common with FB_{CV} (0.98) than FB_{KMM} (0.79). Based on the observations for the two reranking methods KMM and TSVM made in Section IV-E, we expect that FB_{KMM} tunes the model such that it classifies better ImageNet images. Compared to FB_{KMM}, FB_{TSVM} brings little new data into the fine-tuning process, however, FB_{TSVM} fine-tunes the model to be better at classifying predominant visual objects that appear in the FB_{CV} database.

The network structure of fine-tuning pipelines is unchanged with the exception of the last fully connected layer, `fc8`, which will be learned from scratch (random weight initialization) but with the learning rate $\eta_{fc8} = 0.01$ – 10 times larger

	Training					Fine-tuning	
						FB_{CV}	
	baseline	FB	FB_{CV}	FB_{KMM}	FB_{TSVM}	↙	↘
#images (x1K)	(112.7)	(406.5)	(291)	(81.3)	(98.9)	FB_{KMM}	FB_{TSVM}
baseline-test	**67.8**	53.6	55.0	48.0	49.6	**60.0**	58.9
VocPascal07	54.8	58.9	**58.3**	47.3	52.4	58.8	**60.4**
102Flowers	52.6	53.9	**55.7**	43.6	51.2	57.8	**58.8**
MITScene67	32.6	32.2	**34.3**	25.5	30.3	34.3	**36.7**

TABLE I: Experiment results of training and fine-tuning convnets using noisy Web images.

than the global learning rate used in fine-tuning. Our solution improves the accuracy on $baseline-test$ from 53.6% up to 58.9% for the pipeline $FB_{CV} \rightarrow FB_{TSVM}$ and up to 60% for the pipeline $FB_{CV} \rightarrow FB_{KMM}$. The annotation cost spent on reranking methods is negligible compared to that of $ImNet100$, i.e. 1K versus 100K annotated images. According to our best knowledge, our approach is the first efort to train convnets using noisy Web images.

B. Convnets and Cross-domain Evaluation

If convnets are preferred in practice, it is due to their cross-domain capability. In this experiment, we use three third-party datasets in order to evaluate trained models presented in Section V-A. Images in each dataset are forwarded into the trained networks to extract `fc7` features. These normalized features are then used to train one-vs-rest linear SVM classifiers. mAP scores are computed on the test sets of each dataset and reported in the last rows of Table I. The result is surprising: our approach outperforms the baseline for all of the three datasets. Moreover, fine-tuning using FB_{TSVM} gives higher mAP than using FB_{KMM} and this somewhat contradicts the experiment in Section V-A. Our explanation lies in (4), where up to 98% images of FB_{TSV} belong to FB_{CV}; this percentage is just 79% in the case of FB_{KMM}. Clearly fine-tuning on a cleaner subset of the data used in gross training may increase the expressiveness of convnet's high-level features. In overall our conclusion is *"image reranking and network fine-tuning are good companions."*

VI. Conclusion

We introduced a new approach that effectively trains convolutional networks using noisy Web images, which are cheap to obtain and easily available. The paper conveys both the uses of unsupervised and weakly-supervised image reranking methods in order to improve classification results on three public datasets. Extensive experiments have shown that our training approach leads to very promising results. It opens opportunities for convnets to be widely used with affordable training cost. We plan to scale up Web image collections in order to verify our hypothesis and elaborate more on effective training methods with limited supervision. [1]

References

[1] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *NIPS*, 2012.

[2] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, "Scalable object detection using deep neural networks," in *CVPR*, 2014.

[3] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *CoRR*, 2014.

[4] Y. Gong, Y. Jia, T. Leung, A. Toshev, and S. Ioffe, "Deep convolutional ranking for multilabel image annotation," *CoRR*, 2013.

[5] A. Babenko, A. Slesarev, A. Chigorin, and V. S. Lempitsky, "Neural codes for image retrieval," in *ECCV*, 2014.

[6] J. Wang, Y. Song, T. Leung, C. Rosenberg, J. Wang, J. Philbin, B. Chen, and Y. Wu, "Learning fine-grained image similarity with deep ranking," in *CVPR*, 2014.

[7] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: an astounding baseline for recognition," *CoRR*.

[8] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," in *BMVC*, 2014.

[9] J. Hoffman, E. Tzeng, J. Donahue, Y. Jia, K. Saenko, and T. Darrell, "One-shot adaptation of supervised deep convolutional models," *CoRR*, 2013.

[10] B. Zhou, À. Lapedriza, J. Xiao, A. Torralba, and A. Oliva, "Learning deep features for scene recognition using places database," in *NIPS*, 2014.

[11] F. Schroff, A. Criminisi, and A. Zisserman, "Harvesting image databases from the web," *PAMI*, 2011.

[12] J. Huang, A. J. Smola, A. Gretton, K. M. Borgwardt, and B. Schölkopf, "Correcting sample selection bias by unlabeled data," in *NIPS*, 2006.

[13] V. Sindhwani and S. S. Keerthi, "Large scale semi-supervised linear svms," in *SIGIR*, 2006.

[14] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *ECCV*, 2014.

[15] E. Rodner, J. Hoffman, J. Donahue, T. Darrell, and K. Saenko, "Towards adapting imagenet to reality: Scalable domain adaptation with implicit low-rank transformations," *CoRR*, 2013.

[16] Q. V. Le, M. Ranzato, R. Monga, M. Devin, G. Corrado, K. Chen, J. Dean, and A. Y. Ng, "Building high-level features using large scale unsupervised learning," in *ICML*, 2012.

[17] H. Lee, R. B. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *ICML*, 2009.

[18] A. Dosovitskiy, J. T. Springenberg, M. A. Riedmiller, and T. Brox, "Discriminative unsupervised feature learning with convolutional neural networks," in *NIPS*, 2014.

[19] S. Sukhbaatar and R. Fergus, "Learning from noisy labels with deep neural networks," *arXiv*, 2014.

[20] W. Chu, F. D. la Torre, and J. F. Cohn, "Selective transfer machine for personalized facial action unit detection," in *CVPR*, 2013.

[21] L. van der Maaten, "Barnes-hut-sne," *CoRR*, vol. abs/1301.3342, 2013.

[22] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

[1] Work partially supported by the Datascale project, funded by the French Ministère de l'economie, des finances et de l'industrie and the USEMP FP7 project, funded by the EC under contract number 611596.