Visual Concept Similarity

tuw**,cea*, uaic[†], hu[‡]

*CEA, LIST, LVIC

** Vienna University of Technology, ISIS, IMP

[†] "Al. I. Cuza" University

[‡] Hacettepe University

Contacts: pinar@cs.hacettepe.edu.tr, lupu@ifs.tuwien.ac.at, adrian.popescu@cea.fr

MUCKE Project, Deliverable 4.2

01/03/2015

Contents

Abstract

Addressing visual concept similarity as the central theme, we present in this deliverable our works of constructing visual concepts under unsupervised and weak-supervised settings, given the unlimited data availability from the Web. Three perspectives are under investigation. First, Concept Map is proposed to discover representative image groups of visual concepts from a noise-contaminated Web image collection. Second, FAME is a refinement method that iteratively builds up visual concepts of face identities from Web images. Third, deep learning is applied to learn a large amount of visual concepts from noise Web images. Promising results obtained in our experiments demonstrate the applicability of using weakly labeled data to build visual concepts as well as mechanisms to measure concept similarity.

1 INTRODUCTION

The need for manually labelled data continues to be one of the most important limitations in large scale recognition. Alternatively, images are available on the Web in huge amounts. This fact recently attracted many researchers to build (semi-)automatic methods to learn from web data collected for a given concept. However, there are several challenges that makes the data collections gathered from web different from the hand crafted datasets. Images on the web are "in the wild" inheriting all types of challenges due to variations and effects. Since usually images are gathered based on the surrounding text, the collection is very noisy with several visually irrelevant images as well as images corresponding to different characteristic properties of the concept (Figure 1).

For the queried data for automatic learning of concepts, we propose in Section 2 a novel method to obtain a representative groups with irrelevant images removed. 1Our intuition is that, given a concept category by a query, although the list of images returned include irrelevant ones, there will be common characteristics shared among subset of images. Our main idea is to obtain visually coherent subsets, that are possibly corresponding to semantic sub-categories, through clustering and to build models for each sub-category (see Figure 2). The model for each concept category is then a collection of multiple models, each representing a different aspect.

To retain only the relevant images that describe the concept category correctly, during clustering we need to remove outliers, i.e. irrelevant ones. The outliers may resemble to each other while not being similar to the correct category resulting in a **outlier cluster**. Alternatively, outlier images could be mixed with correct category images inside **salient clusters** corresponding to relevant ones. These images, that we refer to as **outlier elements**, should also be removed for the quality data for learning.

A novel method **Concept Maps (CMAP)** for which organises the data by purifying it not only from outlier clusters but also from outlier elements in salient clusters. CMAP captures category characteristics through organising the set of given instances into sub-categories pruned from irrelevant instances. In Section3 we present the application of CMAP into the problem of builing face classifiers for public faces.

The keep-growing content of Web images is also the indispensable data source to scale up scalable semi-supervised algorithms. Recent successes of deep neural networks have demonstrated the power of training process based on huge amount of data. While current deep networks are performant the supervised learning regime, harnessing Web images requires learning methods are capable of learning from weakly labeled images and noisy data. In the second part of this chapter, we propose to use convnet in order to leverage semi-supervised representation learning. We present in Section 4 the approach that uses massive amounts of unlabeled and noisy Web images to train convnets as general feature detectors despite challenges coming from data such as high level of mislabeled data, outliers, and data biases. Extensive experiments were conducted at several data scales, different network architectures, and data reranking techniques. The learned representations are evaluated on nine public datasets of various topics. The best results obtained by our convnets,

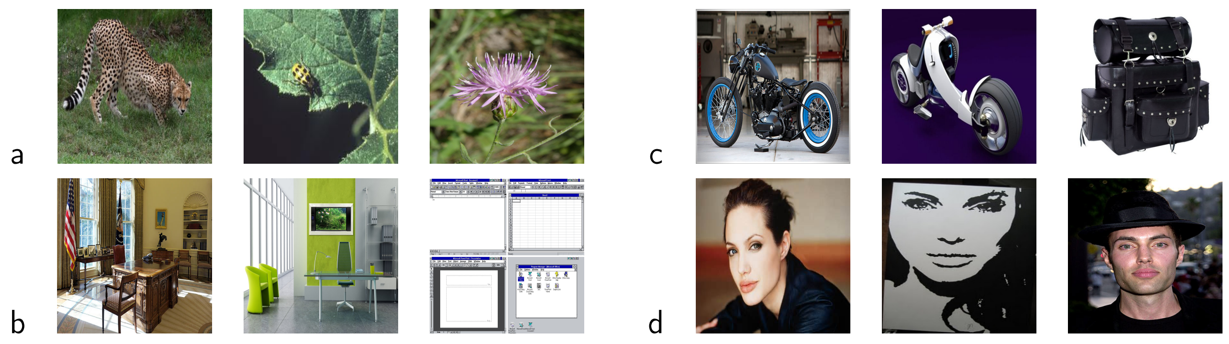


FIGURE 1: EXAMPLE WEB IMAGES FOR (A) SPOTTED, (B) OFFICE, (C) MOTORBIKES, (D) ANGELINA JOLIE.

trained on 3.14 million Web images, outperforms AlexNet [37] trained on 1.2 million clean images of ILSVRC 2012 and is closing the gap with VGG-16 [65]. These prominent results suggest a good solution to use deep learning as an efficient tool to build large-scale visual concepts.

2 CONCEPT MAP

We propose CMAP which is inspired from the well-known Self Organizing Maps (SOM) [35]. In the following, SOM will be revisited briefly, and then CMAP will be described.

Revisiting Self Organizing Maps (SOM): Intrinsic dynamics of SOM are inspired from developed animal brain where each part is known to be receptive to different sensory inputs and which has a topographically organized structure[35]. This phenomena, i.e. "receptive field" in visual neural systems [32], is simulated with SOM, where neurons are represented by weights calibrated to make neurons sensitive to different type of inputs. Elicitation of this structure is furnished by competitive learning approach.

Consider input $X = \{x_1, .., x_M\}$ with M instances. Let $N = \{n_1, ..., n_K\}$ be the locations of neuron units on the SOM map and $W = \{w_1, ..., w_K\}$ be the associated weights. The neuron whose weight vector is most similar to the input instance x_i is called as the winner and denoted by $\hat{v}$. Weights of the winner and units in the neighbourhood are adjusted towards the input at each iteration t with delta learning rule.

$$w_j^t = w_j^{t-1} + h(n_i, n_{\hat{v}} : \epsilon^t, \sigma^t)[x_i - w_j^{t-1}] \tag{1}$$

Update step is scaled by the window function $h(n_i, n_{\hat{v}} : \epsilon^t, \sigma^t)$ for each SOM unit, inversely proportional to the distance to the winner (Eq.2). Learning rate ϵ is a gradually decreasing value, resulting in larger updates at the beginning and finer updates as the algorithm evolves. σ^t defines the neighbouring effect so with the decreasing σ, neighbour update steps are getting smaller in each epoch. Note that, there are different alternatives for update and windows functions in SOM literature.

$$h(n_i, n_{\hat{v}} : \epsilon^t, \sigma^t) = \epsilon^t \exp \frac{-||n_j - n_{\hat{v}}||^2}{2{\sigma^t}^2} \tag{2}$$

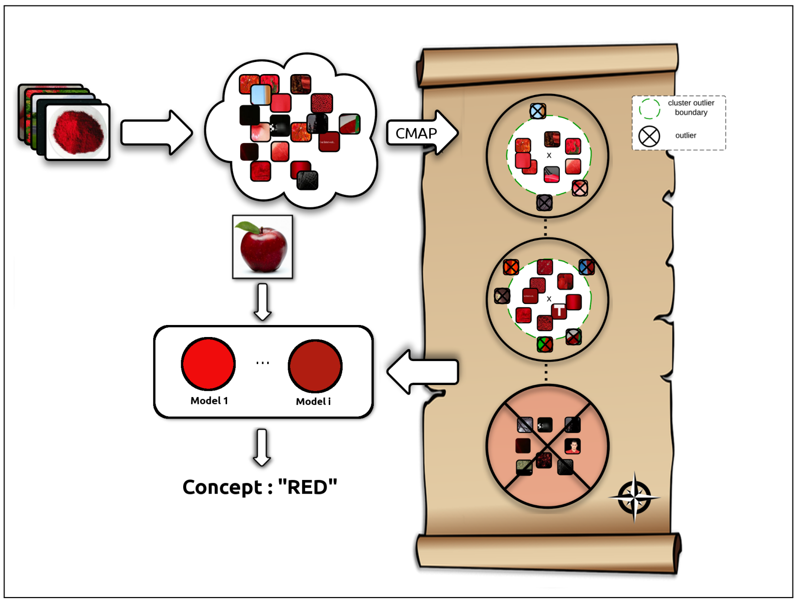


FIGURE 2: OVERVIEW OF OUR FRAMEWORK FOR CONCEPT LEARNING SHOWN ON EXAMPLE CONCEPT "RED". CONCEPT MAP (CMAP) ORGANISES THE IMAGES COLLECTED FROM WEB FOR THE GIVEN TEXT QUERY INTO CLUSTERS WHICH ARE PRUNED FROM OUTLIER ELEMENTS INSIDE SALIENT CLUSTERS AND OUTLIER CLUSTERS. EACH CLUSTER IS THEN USED AS A SUB-MODEL FOR LEARNING AND LOCALISING THE CONCEPT IN A NOVEL IMAGE.

Clustering and outlier detection with CMAP: We introduce excitation scores $E = \{e_1, e_2, \ldots, e_K\}$ where e_j, the score for neuron unit j, is updated as in Eq.3.

$$e_j^t = e_j^{t-1} + \rho^t(\beta_j + z_j) \tag{3}$$

As in SOM, window function is getting smaller with each iteration. z_j is the activation or win count for the unit j, for one epoch. ρ is learning solidity scalar that represents the decisiveness of learning with dynamically increasing value, assuming that later stages of the algorithm has more impact on the definition of salient SOM units. ρ is equal to the inverse of the learning rate ϵ. β_j is the total measure of the activation of jth unit in an epoch, caused by all the winners of the epoch but the neuron itself (Eq.4).

$$\beta_j = \sum_{\hat{v}=1}^{u} h(n_j, n_{\hat{v}}) z_{\hat{v}} \tag{4}$$

At the end of the iterations, normalized e_j is a quality value of a unit j. Higher value of e_j indicates that total amount of excitation of the unit j in whole learning period is high thus it is responsive to the given class of instances and it captures notable amount of data. Low excitation values indicate the contrary. CMAP is capable of detecting outlier units via a threshold θ in the range $[0, 1]$.

Let $C = \{c_1, c_2, \ldots, c_K\}$ be the cluster centres corresponding to each unit. c_j is considered to be a **salient cluster** if $e_j \geq \theta$, and an **outlier cluster** otherwise.

The excitation scores E are the measure for saliency of neuron units in CMAP. Given the data belonging to a category, we expect that data is composed of sub-categories that share common properties. For instance `red` images might include tones to be captured by clusters but they are supposed to share a common characteristics of being red. For the calculation of the excitation scores we use individual activations of the units as well as the neighboring activations. Individual activations measure being a salient cluster corresponding to a particular sub-category, such as `lighter red`. Neighborhood activations count the saliency in terms of the shared regularity between sub-categories. If we don't count the neighborhood effect, some unrelated clusters would be called salient, e.g. noisy white background patches in `red` images.

Outlier instances in salient clusters (**outlier elements**) should also be detected. After the detection of outlier neurons, statistics of the distances between neuron weight w_i and its corresponding instance vectors is used as a measure of instance divergence. If the distance between the instance vector x_j and its winner's weight $\hat{w}_i$ is more than the distances of other instances having the same winner, x_j is raised as an outlier element. We exploit box plot statistics, similar to [49]. If the distance of the instance to its cluster's weight is more than the upper-quartile value, then it is an outlier. The portion of the data, covered by the upper whisker is decided by τ.

CMAP provides good basis of cleansing of poor instances whereas computing cost is relatively smaller since an additional iteration after clustering phase is not required. All the necessary information (excitation scores, box plot statistics) for outliers is calculated at runtime of learning. Hence, CMAP is suitable for large scale problems.

CMAP is also able to estimate number of intrinsic clusters of the data. We use PCA as a simple heuristic for that purpose, with defined variance ν to be retained by the selected first principle components. Given data, principle components describing the data with variance ν is used as the number of clusters for the further processing of CMAP. If we increase ν, CMAP latches more clusters.

$$Num.Clusters = \max_q \big(\frac{\sum_{i=1}^{q} \lambda_i}{\sum_{j=1}^{p} \lambda_j} \leq \nu\big) \tag{5}$$

q is the number of top principle components selected after PCA and p is the dimension of instance vectors. λ is the eigenvalue of corresponding component.

2.1 CONCEPT LEARNING WITH CMAP

We utilise the clusters, that are obtained through CMAP as presented above, for learning sub-models in concepts. We exploit the proposed framework for learning of attributes, scenes, objects and faces. Each task requires the collection of data, clustering and outlier detection with CMAP, and training of sub-models from the resulting clusters. In the following, first we will describe the attribute learning, and then describe the differences in learning other concepts. Implementation details are presented in Section2.2

Learning low-level attributes: Most of the methods require learning of visual attributes from labelled data, and cannot eliminate human effort. Here, we describe our method in learning

In the real code we use vectorized implementation whereas we write down iterative pseudo-code for the favour of simplicity.

```
Input: $X$, $\theta, \tau$, $K$, $T$, $\nu$, $\sigma^{init}$, $\epsilon^{init}$
Output: $OutlierUnits, Mapping, W$
set each item $z_i$ in $Z$ to $0$
$u \leftarrow estimateUnitNumber(X, variation)$
$W \leftarrow randomInit(u)$
while $t \leq T$ do
    $\epsilon^t \leftarrow computeLearningRate(t, \epsilon^{init})$
    $\rho^t \leftarrow 1/\epsilon^t$
    set each item $\beta_i$ in $B$ to $0$
    select a batch set $X^t \subset X$ with $K$ instances
    for each $x_i \in X$ do
        $\hat{w}_i^t \leftarrow findWinner(x_i, W)$
        $\hat{v} \leftarrow min_j(||x_i - w_j||)$
        increase win count $z_{\hat{w}^t} \leftarrow z_{\hat{w}_i^t} + 1$
        increase win count $z_{\hat{v}} \leftarrow z_{\hat{v}} + 1$
        for each $w_k \in W$ do
            $\beta_k^t = \beta_k^t + h(n_k, n_{\hat{v}})$
            $w_k = w_k + h(n_k, n_{\hat{v}})||x_i - w_{\hat{v}}||$
        end
    end
    for each $w_j \in W$ do
        $e_j^t = e_j^{t-1} + \rho^t(\beta_j^t + z_j)$
    end
    $t \leftarrow t + 1$
end
$W_{out} \leftarrow thresholding(E, \theta)$
$W_{in} \leftarrow W \setminus W_{out}$
$Mapping \leftarrow findMapping(W_{in}, X)$
$Whiskers \leftarrow findUpperWhiskers(W_{in}, X)$
$X_{out} \leftarrow findOutlierIns(X, W_{in}, Whiskers, \tau)$
return $W_{out}, X_{out}, Mapping, W$
```

Algorithm 1: CMAP

attributes from web data without any supervision.

We collect web images through querying colour and texture names. The data is weakly labelled, with the labels given by queries. Hence, there are irrelevant images in the collection, as well as images with a tiny portion corresponding to the query keyword.

Each image is densely divided into non-overlapping fixed-size patches to sufficiently capture the required information. We assume that the large volume of the data itself is sufficient to provide instances at various scales and illuminations, thus we did not perform any scaling or normalization. The collection of patches extracted from all images for a single attribute is then given to CMAP to obtain clusters which are likely to capture different characteristics of the attribute as removing the irrelevant ones.

Each cluster obtained through CMAP is used to train a separate classifier. Positive examples are selected as the members of the cluster and negative instances are selected among the outliers removed by CMAP and also elements from other categories.

Learning scene categories: To show CMAP capability on higher level concepts, we target scene categories. In this case, we use the entire images as instances, and aim to discover groups of images each representing a different property of the scene, at the same time by eliminating the images that are either spurious. These clusters are then used as models similar to the attribute learning.

Learning object categories: In the case of objects, we detect salient regions on each image via [17], to eliminate background noise. Then these salient regions are fed into CMAP framework for clustering.

Learning faces: We address the problem of learning faces associated with a name -which is generally referred to face naming in the literature-, through finding salient clusters in the set of images collected from web through querying the name. Here, the clusters are likely to correspond to different poses and possibly different hair and make-up style differences as well as ageing effects. Note that this task is not the detection of faces, but recognition of faces for a given name. We detect the faces in the images, and only use a single face with the highest confidence for each image.

2.2 EXPERIMENTS

2.2.1 QUALITATIVE EVALUATION OF CLUSTERS

As Figure 3 depicts, CMAP captures different characteristics of concepts in separate salient clusters, while eliminating outlier clusters that group irrelevant images coherent among themselves, as well as outlier elements wrongly mixed with the elements of salient clusters . On more difficult tasks of grouping objects and faces, CMAP is again successful in eliminating outlier elements and outlier clusters as shown in Figure 4.

2.2.2 ATTRIBUTE LEARNING

Datasets and representation: We collected images from Google for 11 distinct colours as in [74] and 13 textures. We included the terms "colour" and "texture" in the queries, such as ”red colour", or "wooden texture". For each attribute, 500 images are collected. In total we have 12000 web images. Each image is divided into 100x100 non-overlapping patches. Unlike [74], we didn’t apply

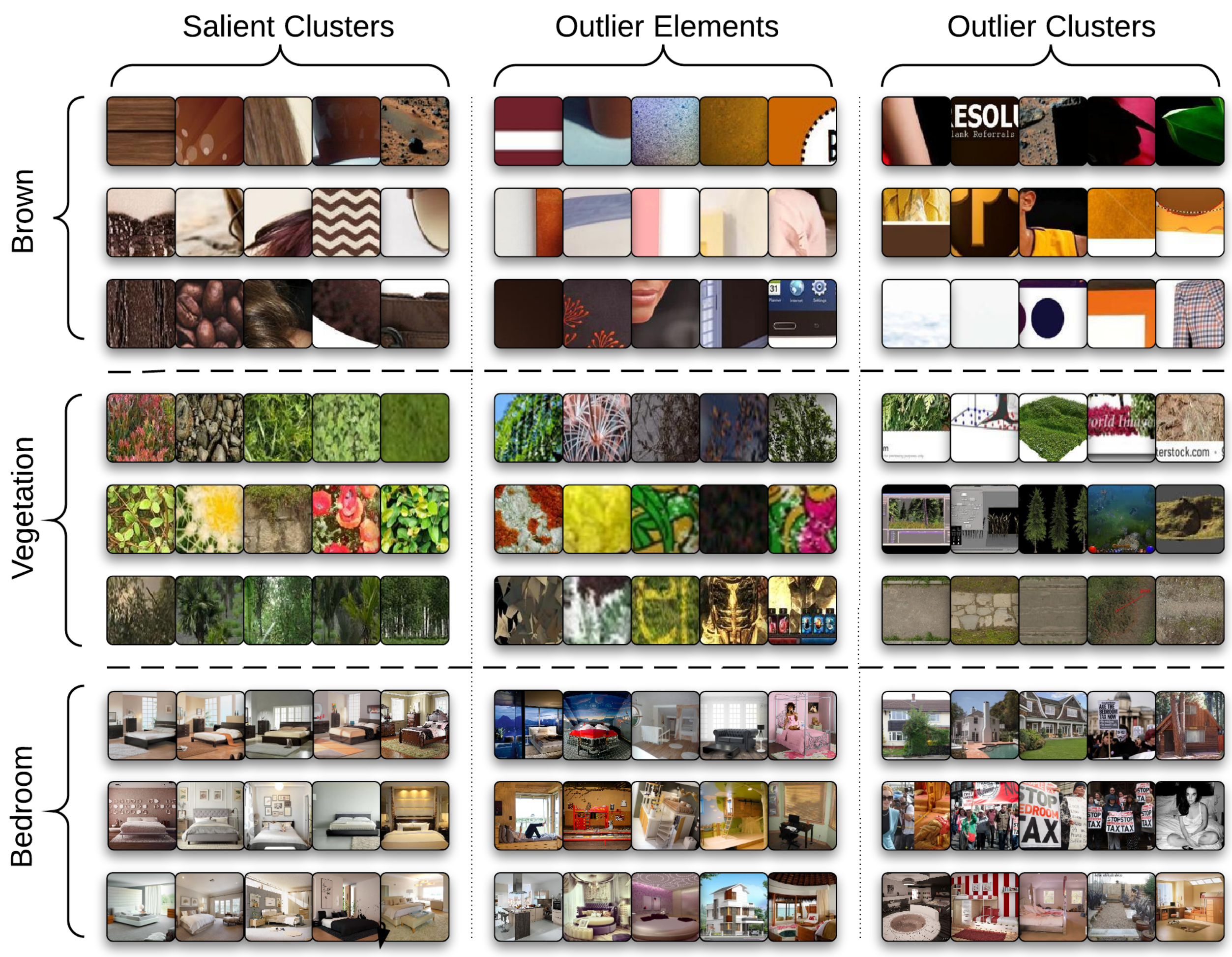


FIGURE 3: FOR COLOUR AND TEXTURE ATTRIBUTES *BROWN* AND *VEGETATION* AND SCENE CONCEPT *BEDROOM*, RANDOMLY SAMPLED IMAGES DETECTED AS (I) ELEMENTS OF **SALIENT CLUSTERS**, (II) ELEMENTS OF **OUTLIER CLUSTERS**, AND (III) **OUTLIER ELEMENTS** IN SALIENT CLUSTERS. CMAP DETECTS DIFFERENT SHADES OF "BROWN" AND ELIMINATES SOME SUPERIORS ELEMENTS BELONGING THE DIFFERENT COLORS. FOR THE "VEGETATION" AND "BEDROOM", CMAP AGAIN DIVIDES THE VISUALS ELEMENTS WITH RESPECT TO STRUCTURAL AND ANGULAR PROPERTIES. ESPECIALLY FOR "BEDROOM", EACH CLUSTER IS ABLE TO CAPTURE DIFFERENT VIEW-ANGLE OF THE IMAGES AS IT SUCCESSFULLY REMOVES OUTLIER INSTANCES WITH SOME OF LITTLE MISTAKES THAT ARE BELONGING TO THE LABEL BUT NOT REPRESENTATIVE (CIRCULAR BED IN VERY SHINY ROOM) FOR THE CONCEPT PART.

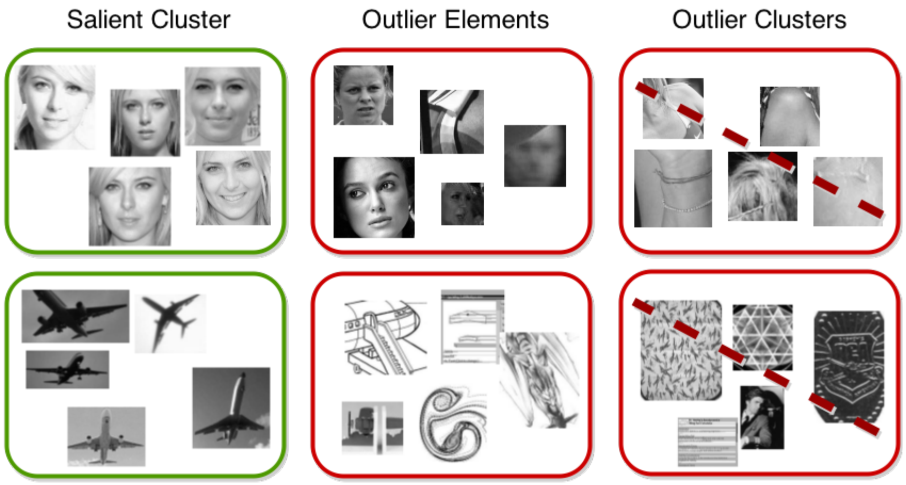


FIGURE 4: CMAP RESULTS FOR OBJECT AND FACE EXAMPLES. LEFT COLUMNS SHOWS ONE EXAMPLE OF SALIENT CLUSTER. MIDDLE COLUMN SHOWS OUTLIER INSTANCES CAPTURED FROM SALIENT CLUSTERS. RIGHT COLUMN IS THE DETECTED OUTLIER CLUSTERS.

gamma correction. For colour concepts we use 10x20x20 bins Lab colour histograms and for texture concepts we use BoW representation for densely sampled SIFT [45] features with 4000 words. We keep the feature dimensions high to utilise from the over-complete representations of the instances with L1 norm linear SVM classifier.

Attribute recognition on novel images: The goal of this task is to label a given image with a single attribute name. Although there may be multiple attributes in a single image, for being able to compare our results on benchmark data-sets we consider one attribute label per image. For this purpose, first we divide the test images into grids in three levels using spatial pyramiding [39]. Non-overlapping patches (with the same size of training patches) are extracted from each grid of all three levels. Recall that, we have multiple classifiers for each attribute trained on different salient clusters. We run all the classifiers on each grid for all patches. Then, we have a vector of confidence values for each patch, corresponding to each particular cluster classifier. We sum those confidence vectors of each patch in the same grid. Each grid at each level is labelled by the maximum confidence classifier among all the outputs for the patches. All of those confidence values are then merged with a weighted sum to a label for the entire image. $D^i = \sum_{l=1}^{3} \sum_{n=1}^{N_l} \frac{1}{2^{3-l}} h_i e^{-(\hat{x}-x)/2\sigma^2}$ Here, N_l is the grid number for level l and h_i is the confidence value for grid i. We include a Gaussian filter, where $\hat{x}$ is center of the image and x is location of the spatial pyramid grid, to give more priority to the detections around the center of the image for reducing noisy background effect.

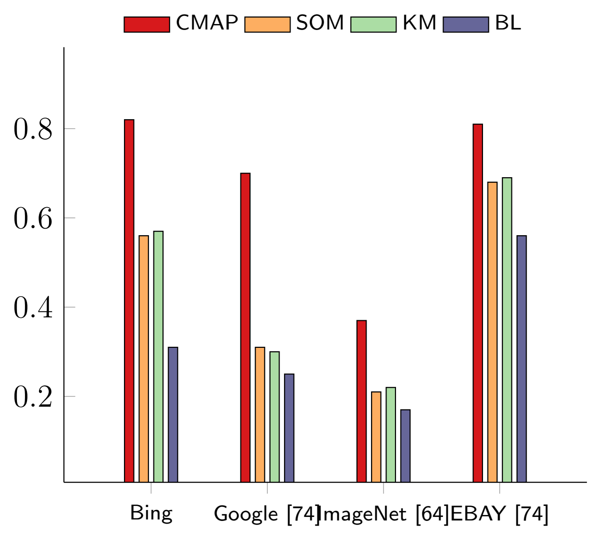
For evaluation we use three different datasets. The first dataset is Bing Search Images curated by ourselves from the top 35 images returned with the same queries we used for initial images. This set includes 840 images in total for testing. Second dataset is Google Colour Images [74] previously used by [74] for learning colour attributes. Google Colour Images includes 100 images for each color name. We used the whole data-sets only for testing of our models learned on a possibly different set that we collected from Google, contrary to [74]. The last dataset is sample annotated images from ImageNet [64] for 25 attributes. To test the results on a human labelled dataset, we use Ebay dataset provided by [74] which has labels for the pixels in cropped regions. It includes 40 images for each colour name.

Figure 5 compares the overall accuracy of the proposed method (**CMAP**) with three other methods on the task of attribute learning. As the baseline (**BL**), we use all the images returned for the concept query to train a single model. As expected, the performance is very low suggesting that a single model trained by crude noisy web images performs poorly and the data should be organised to train at least some qualified models from coherent clusters in which representative images are grouped. As two other methods for clustering the data, we used k-means (**KM**) and original SOM algorithm (**SOM**) with optimal cluster number, decided by cross-validation of whole pipeline, and again train a model for each cluster. The low results support the need for pruning of the data through outlier elimination. Results show that, CMAP's clusters are able to detect coherent and clean representative data groups so we train less number of classifiers by eliminating outlier clusters but those classifiers better in quality and also, on novel test sets with images having different characteristics than the images used in training, CMAP can still perform very well on learning of attributes.

Our method is also utilised for retrieving images on EBAY dataset as in [74]. [74] learns the models from web images and apply the models to another set so both method study a similar problem. We utilise CMAP with patches obtained from the entire images (**CMAP**) as well as from the masks provided by [74] (**CMAP-M**). As shown in Figure 5 Right, even without masks CMAP is comparable to the performance of the PLSA based method of [74], and with the same setting CMAP outperforms the PLSA based method with significant performance difference.

On ImageNet, we obtained 37.4% accuracy compared to 36.8% of [64]. It is also seen that, our models trained from different source of information are better to generalize for some of worse performance classes (rough, spotted, striped, wood) of [64]. Recall that we globally learn the attribute models from web images, not from any partition of the ImageNet. Thus, it is encouraging to observe better results in such a large data-set against [64]'s attribute models trained by a sufficiently large training subset.

Attribute based scene recognition: While the results on different datasets support the ability of our approach to be generalised to different datasets, we also perform experiments to understand the effect of the learned attributes on a different task, namely for classification of scenes using entirely different collections. Experiments are performed on MIT-indoor [59], and Scene-15 [39] datasets. MIT-indoor has 67 different indoor scene with 15620 images with at least 100 images for each category and we use 100 images from each class to test our results. Scene-15 is composed by 15 different scene categories. We use 200 images from each category for our testing. MIT-indoor is



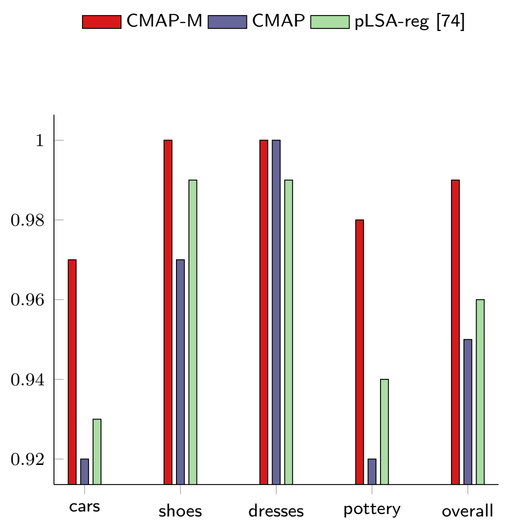


FIGURE 5: **LEFT**: ATTRIBUTE RECOGNITION PERFORMANCES ON NOVEL IMAGES COMPARED TO OTHER METHODS. **RIGHT**: EQUAL ERROR RATES ON EBAY DATASET FOR IMAGE RETRIEVAL USING THE CONFIGURATION OF [?]. CMAP DOES NOT UTILIZE THE IMAGE MASKS USED IN [?], WHILE CMAP-M DOES.

extended and even harder version of Scene-15 with many additional categories.

We again get the confidence values for each grid in three levels of the spatial pyramid on the test images. However, rather than using a single value for the maximum classifier output, we keep the confidence values for all the classifiers for each grid. We concatenate these vectors for all grids in all levels to get a single feature vector of size $3xNxK$ for the image, which is then used for scene classification. Here N is the number of grids at each level, and K is the number of different concepts. Note that, while the attributes are learned in an unsupervised way, in this experiment scene classifiers are trained on the datasets provided (see next section for automatic scene concept learning).

As shown in Table 1, our method for scene recognition with learned attributes (**CMAP-A**), performs competitively with [43] while using shorter feature vectors in relatively cheaper environment, and outperforms the others. Comparisons with [59] show that using the visual information acquired from attributes is more descriptive in the cluttered nature of MIT-indoor scenes. For instance, "bookstore" images has very similar structural layout to "clothing store" images, but they are more distinct with colour and texture information around the scene. Attribute level features do not create this much difference for Scene-15 data-set since images include some obvious statistical differences.

2.2.3 LEARNING CONCEPTS FOR SCENE CATEGORIES

Alternative to recognising scenes through the learned attributes, we directly learn higher level concepts for scene categories. We call this method as **CMAP-S**. Specifically, we perform testing for scene classification for 15 scene categories on [39] and MIT-indoor [59] data-sets, but learn the scene concepts directly from the images collected from Web through querying for the names of the

-	CMAP-A	CMAP-S	CMAP-S+HM	Li et al. [43] VQ	Pandey et al. [56]	Kwitt et al. [38]	Lazebnik et al. [39]	Singh et al. [67]
MIT-indoor [59]	46.2%	40.8%	41.7%	47.6%	43.1%	44%	-	38%
Scene-15 [39]	82.7%	80.7%	81.3%	82.1%	-	82.3%	81%	77%

TABLE 1: COMPARISON OF OUR METHODS ON SCENE RECOGNITION IN RELATION TO STATE-OF-THE-ART STUDIES ON MIT-INDOOR AND SCENE-15 DATASETS.

scene concepts used in these datasets. That is, we do not use any manually labelled training set (or training subset of the benchmark data-sets), but directly the crude web images which are pruned and organised by CMAP, in contrast to comparable fully supervised methods. As shown in Table 1, our method is competitive with the state-of-the-art studies without requiring any supervised training.

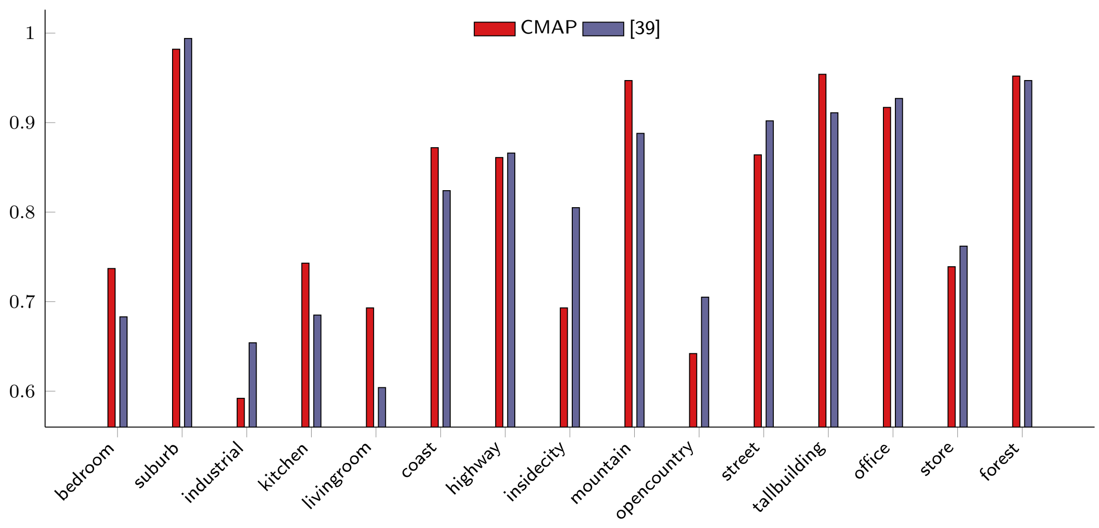
We then made a slight change on our original CMAP-S implementation by using the hard-negatives of previous iteration as a negative set of next iteration (we refer to this new method as **CMAP-S-HM**). We relax the memory needs with less but strong negative instances. As the results in Table 1 and Figure 6 show, we achieve better performances in Scene-15 than the state-of-the-art studies with this simple addition, still without requiring any supervisory input. However, on a harder MIT-indoor dataset, without using attribute information, low-level features are not very distinctive.

In order to understand the effect of discriminative visual features, which aim to capture representative and discriminative mid-level features, we also compare our method with the work of Singh et al. [67]. As seen in Table 1, our performances are better than both their reported results on MIT-indoor [59], and our implementation on Scene-15 [39].

2.2.4 LEARNING CONCEPTS OF OBJECT CATEGORIES

We learn object concepts from Google web images [21] and compare our results with [21] and [42] (Figure 6 Right). [21] provides a data-set from Google with 7 classes and total 4088 gray scale images, 584 images in average for each class with many "junk" images in each class as they indicated. They test their results in a manually selected subset of Caltech Object data-set. Because of its raw nature of the Google images and adaptation to the Caltech subset, it is a good experimental ground for our pipeline.

Salient regions extracted from images are represented with 500 word quantized SIFT [45] vector with additional 256 dimension LBP [53] vector. In total we aggregated a 756 dimension vector representation for each salient region. At the final stage of learning with CMAP, we learn L2 norm, linear SVM classifiers for each cluster with negatives are gathered from other classes and the global outliers. For each learning iteration, we also apply hard mining to cull highest rank negative instances in the amount 10 times of salient instances in the cluster. All pipeline hyper-parameters are tuned via the validation set provided by [21]. Given a novel image, learned classifiers are passed over the image with gradually increasing scales, up to a point where the maximum class confidences are stable. Among class confidences, maximum confidence indicates the final prediction for that image. We observe 6.3 salient clusters in average for all classes and 69.4 instances for each salient clusters. That is, CMAP eliminates 147 instances for each class as supposedly outlier instances. Results



	CMAP	[21]	[42]		CMAP	[21]	[42]
airplane	0.63	0.51	0.76	car	0.97	0.98	0.94
face	0.67	0.52	0.82	guitar	**0.89**	0.81	0.60
leopard	0.76	0.74	0.89	motorbike	**0.98**	0.98	0.67
watch	**0.55**	0.48	0.53	overall	**0.78**	0.72	0.75

FIGURE 6: **TOP**: COMPARISONS ON SCENE-15 DATASET. OVERALL ACCURACY IS 81.3% FOR CMAP-S+HM , VERSUS 81% FOR LAZEBNIK ET AL [39]. CLASSES "INDUSTRIAL", "INSIDECITY", "OPENCOUNTRY" RESULTS VERY NOISY SET OF WEB IMAGES, HENCE TRAINED MODELS ARE NOT STRONG ENOUGH AS MIGHT BE OBSERVED FROM THE CHART. **BOTTOM:** CLASSIFICATION ACCURACIES OF OUR METHOD IN RELATION TO FERGUS ET AL. [21] AND LI ET AL [42].

support that elimination of "junk" images gives significant improvements, especially for the noisy classes in [21].

2.2.5 LEARNING FACES

We use FAN-large [55] face data-set for testing our method in face recognition problem. We use Easy and Hard subsets with the names accommodating more than 100 images (to have fair testing results). Our models are trained over web images queried from Bing Image search engine for the same names. All the data preprocessing and the feature extraction flow follow the same line of [55], that is owned from [20]. However, [55] trains the models and evaluates the results at the same collection.

We retrieve the top 1000 images from Bing results. Face are detected and face with the highest confidence is extracted from each image to be fed into CMAP. Face instances are clustered and spurious face instances are pruned. Salient clusters are used for learning SVM models for each cluster in the same settings of the object categories. For our experiments we used two different face detectors. One is cascade classifier of [76] implemented in OpenCV library [7] and another is [87] with more precise detection results, even the OpenCV implementation is very fast relatively. Results

Method	GBC+CF(half)[55]	CMAP-1	CMAP-2	BaseLine
Easy	0.58	0.63	0.66	0.31
Hard	0.32	0.34	0.38	0.18

TABLE 2: FACE LEARNING RESULTS WITH DETECTING FACES USING OPENCV(CMAP-1) AND ZHU ET AL. [87](CMAP-2).

are depicted at Table 2 with two different face detection method and baseline result with models trained on raw Bing images for each person.

3 ASSOCIATION AS MODEL EVOLUTION

As a special case for learning concepts, we attack the problem of building classifiers for public faces from web images collected through querying a name.Constituting an important portion of the queries, searching for people requires to manage large number of face images piling up on the web. With the recent advances, especially for celebrities and politicians, the returned results -even with a query based on the textual content- provide a large pool of positive instances. This suggests the use of returned results for building models automatically in developing large-scale systems and eliminating the human effort.

Although queries for the popular people are likely to provide more promising results compared to the others (see Figure 7), famous people tend to change their make-up, hair style/color, and accessories more often, and they are photographed in unconstrained environments and conditions with a diverse set of sources, resulting in a large variety in their visual appearances. They are also likely to be captured with others, and their names are mentioned in stories related to others, causing irrelevant faces to be retrieved.

For the query results to be helpful in building models, faces corresponding to other people should be eliminated and discriminative as well as diverse set of faces for the queried individuals should be selected.

In this study, we address the problem of building models for identification of faces through exploiting the weakly labeled web data. We propose a new method, **Face Association through Model Evolution (FAME)**, that utilizes the noisy results obtained through a name query to construct models. Our models evolve through consecutive iterations to associate the query name with the correct set of faces. These models are then used to label faces on novel datasets. FAME removes the outlier faces in constructing models, while retaining the diversity as much as possible.

Figure 8 depicts the overview of FAME. Details will follow the review of the relevant studies and the benchmark datasets used in the experiments.

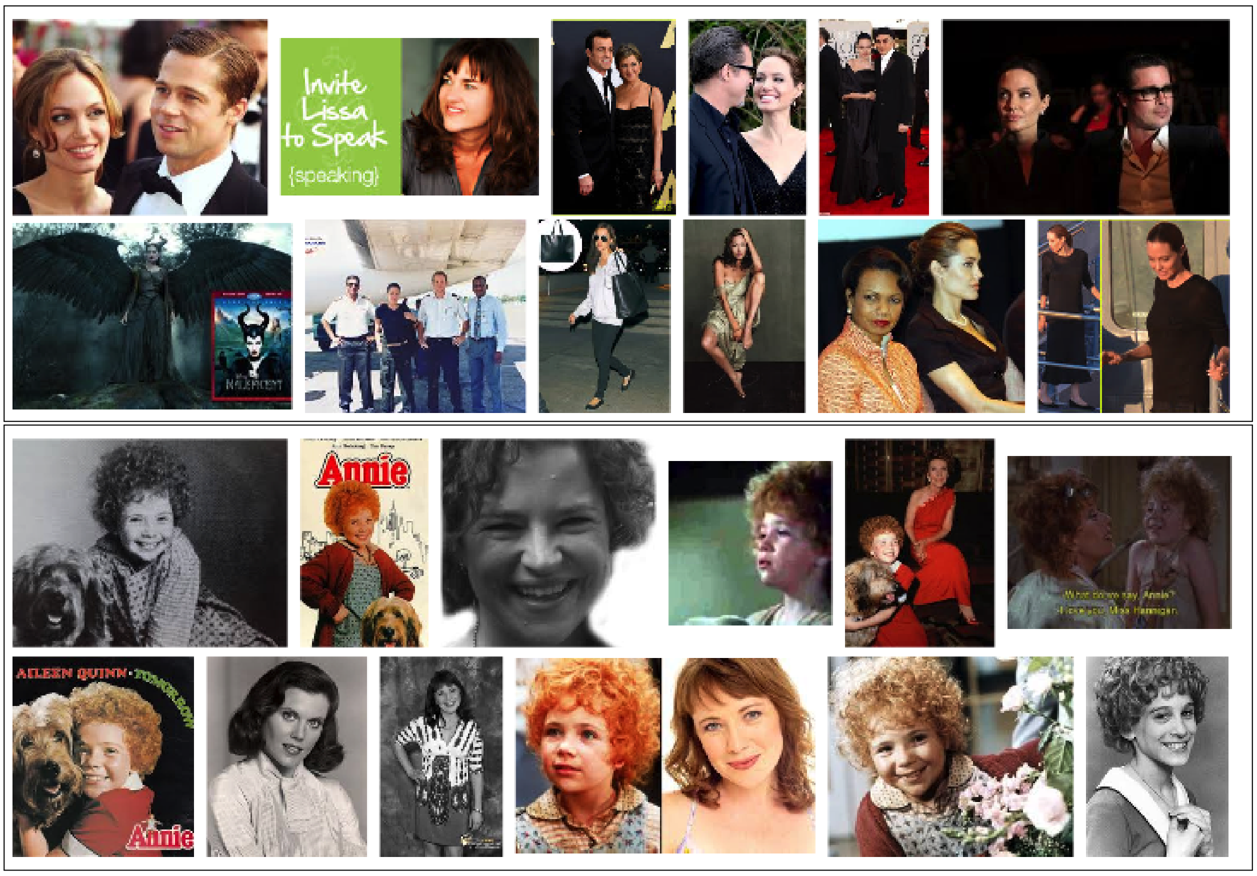


FIGURE 7: THE SEARCH RESULTS FOR *ANGELINA JOLIE* ARE MORE SATISFACTORY THAN THE ONES FOR *AILEEN QUINN* SINCE THERE ARE MORE INSTANCES ENCOUNTERED ON THE WEB. ON THE OTHER HAND, *ANGELINA JOLIE* IS PICTURED MORE OFTEN IN A DIVERSE SET OF CONDITIONS AND OUTFITS, CAUSING LARGER VARIETY IN HER LOOKS COMPARED TO *AILEEN QUINN* WHOSE PICTURES ARE MOSTLY TAKEN FROM THE MOVIE *ANNIE*. IN BOTH CASES, IT IS LIKELY FOR THE RETURNED IMAGES TO INCLUDE MORE THAN A SINGLE PERSON: EITHER IRRELEVANT PEOPLE DUE TO THE TEXT MENTIONING THE NAME IN A DIFFERENT STORY, OR THE OTHERS IN RELATION WITH THE QUERIED PERSON.

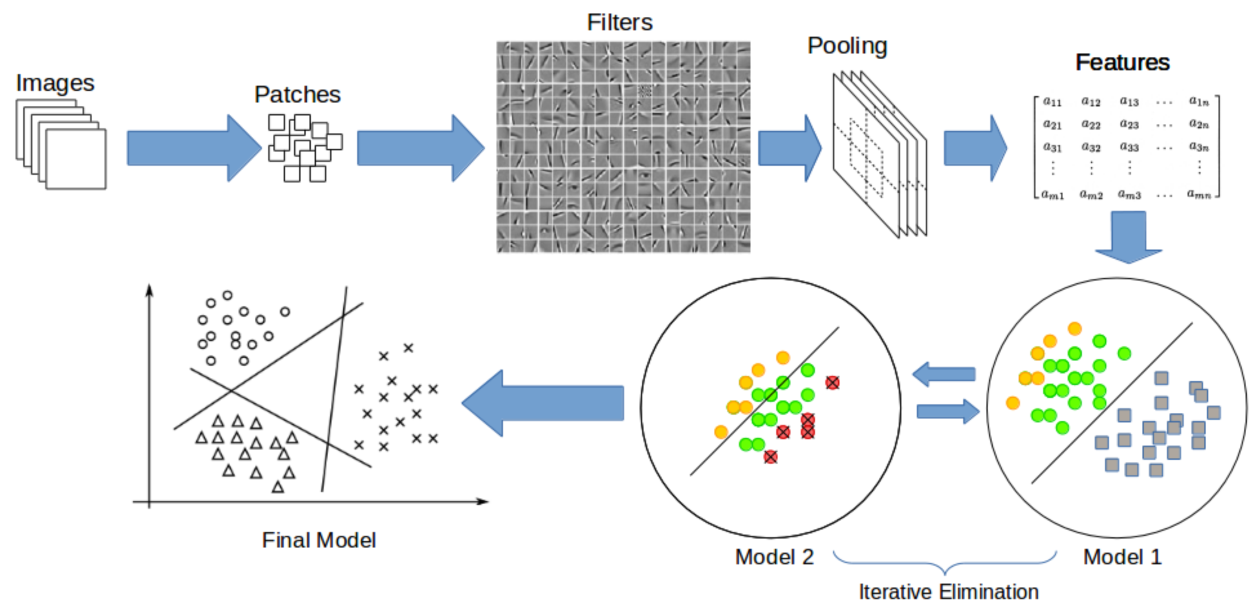


FIGURE 8: OVERVIEW OF THE PROPOSED METHOD. THE DATA IS PRUNED FROM SPURIOUS INSTANCES THROUGH ELIMINATING THE OUTLIERS. THEN, THE MOST CONFIDENT IN-CLASS EXAMPLES ARE UTILIZED TO BUILD THE MODELS. THESE SUCCESSIVE STEPS ARE REPEATED TO CONSTRUCT THE FINAL MODEL.

3.1 MODEL EVOLUTION

An important caveat in learning models from weakly-labeled data is the impurity of the collection. Spurious instances in the collection should be eliminated before generating models for the categories. We present an approach for learning better models through iteratively pruning the data (see Figure 2). The proposed method allows the models to evolve through eliminating the outlier instances and separating the most confident instances from the others with successive linear classifiers.

First, we learn a hyperplane that separates the initial set of **candidate class instances** from the large set of global negatives representing the rest of the world against the class of interest. Then, we select some fraction of the class instances distant from the separating hyperplane and use them as the **category references** as they are confidently classified against the rest of the world. We consider the rest of the class instances as **possible spurious instances**.

We then learn another model to capture in-class dissimilarities between the category references and possible spurious instances. We combine the confidence scores of the first and the second models as a measure of instance saliency. This combination allows us to benefit both from being different from the rest of the world, and in-class affinity of the instance. We detect instances with the lowest confidence scores as the outliers for that iteration. These steps are iterated multiple times up to a desired level of pruning (see Figure 9).

The large dimensional representation used (see Section 3.1.2) allows the diverse set of positive examples to be kept in the final model, but might cause computational burden with complicated learning models. Therefore, we leverage simple linear regression (LR) models with L1 norm regular-

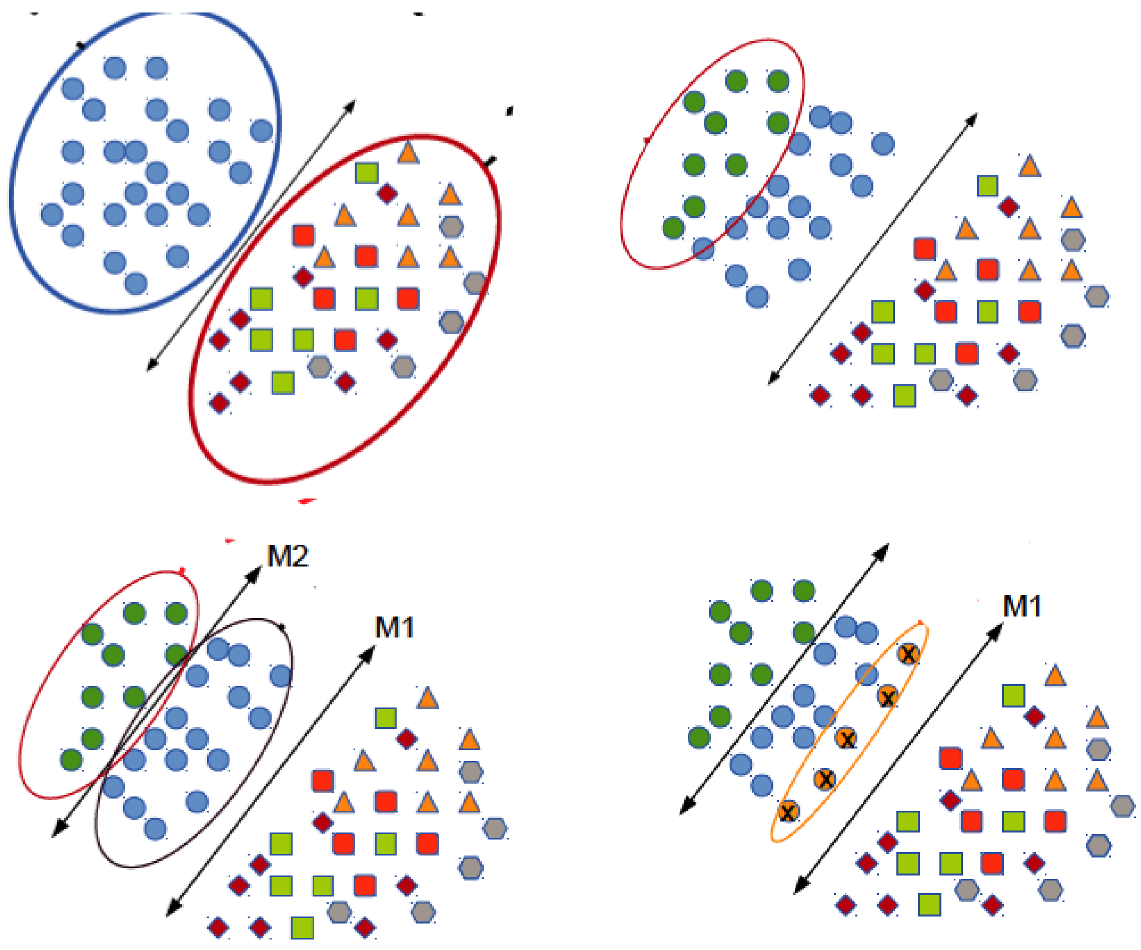


FIGURE 9: ONE STEP OF MODEL EVOLUTION. FIRST, A MODEL $M1$ IS LEARNED TO SEPARATE CANDIDATE CATEGORY INSTANCES FROM GLOBAL NEGATIVES. THEN, THE MOST CONFIDENTLY CLASSIFIED EXAMPLES ARE CONSIDERED AS CATEGORY REFERENCES. ANOTHER MODEL $M2$ IS LEARNED TO SEPARATE THE CATEGORY REFERENCES FROM THE OTHER CANDIDATES. SPURIOUS INSTANCES THAT LIE FARTHEST FROM THE HYPERPLANE ARE ELIMINATED.

ization performing sparse feature selection as the learning evolves.

Note that, our focus is to eliminate the outliers and purify the data while keeping most of the positives, and therefore it is not sufficient to only select most confident instances.

3.1.1 ITERATIVE DATA ELIMINATION

Algorithm 2 summarizes our data elimination procedure. Here, $C = \{c_1, c_2, \dots c_m\}$ refers to the example face images collected for a class and $N = \{n_1, n_2, ..., n_l\}$ refers to the vast numbers of global negatives. Each vector is a d dimensional representation of a single face image.

At each iteration t, the first LR model M^1 learns a hyperplane between the candidate class instances C and global negatives N. Then, C is divided into two subsets: p instances in C that are farthest from the hyperplane are kept as the candidate positive set (C^+) and the rest is considered as the negative set (C^-) for the next model. C^+ is the set of salient instances representing the

```
$C_0 \leftarrow C$
$t \leftarrow 1$
while $stoppingConditionNotSatisfied()$ do
    $M_t^1 \leftarrow LogisticRegression(C_{t-1}, N)$
    $C_t^+ \leftarrow selectTopPositives(C_{t-1}, M_t^1, p)$
    $C_t^- \leftarrow C_{t-1} - C_t^+$
    $M_t^2 \leftarrow LogisticRegresstion(C_t^+, C_t^-)$
    $[S_1^-, S_2^-] \leftarrow getConfidenceScores(C_t^-, M_t^1 M_t^2)$
    $O_t \leftarrow selectOutliers(C_t^-, S_1^-, S_2^-, o)$
    $C_t \leftarrow C_{t-1} - O_t$
    $t \leftarrow t + 1$
end
$C \leftarrow C_t$
return $C$
```

Algorithm 2: FAME

category references for the class and C^- is the set of possible spurious instances.

The second LR model M^2 uses C^+ as positive and C^- as the negative set to learn the best possible hyperplane separating them. For each instance in C^-, by aggregating the confidence values of both models, o instances with the lowest scores are eliminated as the outliers.

This iterative procedure continues until it satisfies a stopping condition. We use M^1's objective as the measure of data quality. As we incrementally remove poor instances, we expect to have better separation against the negative instances. If it saturates after a small number of iterations, we guarantee that at least 0.1 of the initial data is removed.

3.1.2 REPRESENTATION

Being effective, variants of Locally Binary Patterns (LBP) have been heavily utilized in the literature [1, 80, 10, 62].

Following the same direction we exploit LBP features.

To represent face images we learn two distinct set of filters by an unsupervised method as in [12] (Figure 10). First set is learned from the raw-pixel random patches extracted from grey-scale images. The second set is learned from LBP encoded images [1].

First set is receptive to edge- and corner-like structural points and the second set is sensitive to textural commonalities of the LBP histogram statistics. LBP encoded images are invariant to illumination since the intensity relations between pixels are considered instead of pixel values. We use rotation invariant LBP encoding [52] that gives binary codes for each pixel. We convert these binary codes into corresponding integer values. A Gaussian filter is used to smooth out the heavy-tailed locations.

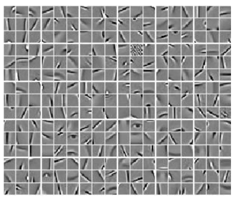
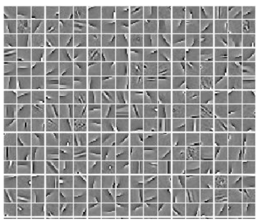
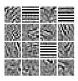
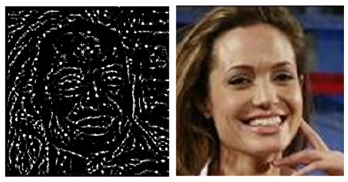
(a)  (b)  (c)  (d) 

FIGURE 10: RANDOM SET OF FILTERS LEARNED FROM (A) WHITENED RAW IMAGE PIXELS, (B) LBP ENCODED IMAGES. (C) OUTLIERS FOR RAW-IMAGE FILTERS. (D) LBP ENCODING FOR AN RGB IMAGE. WE MIGHT OBSERVE EYE OR MOUNT SHAPED FILTERS FROM THE RAW IMAGE FILTERS AND MORE TEXTURAL INFORMATION FROM THE LBP ENCODED FILTERS. OUTLIER FILTERS ARE VERY CLUTTERED AND OBSERVE LOW NUMBER OF ACTIVATIONS MOSTLY FROM BACKGROUND PATCHES.

First, we extract a set of randomly sampled patches in the size of predefined receptive field. Then, contrast normalization is applied to each patch (for only raw-image filters) and patches are whitened to reduce the correlations among dimensions. These patches are clustered into K groups using k-means. We perform thresholding to centroids with box-plot statistics over the activations counts to remove the outlier centroids. After the learning phase, centroid activations are collected from receptive fields with small striding. We applied spatial average pooling onto five different grids (center and four quadrants). This yields a 5xK dimensional representation for each face, for each different set of filters. We use triangular activation function to map each receptive field to learned centroids. Assuming the patches assigned to outlier centroids are not relevant, we avoid them in pooling.

3.2 EXPERIMENTS

For web-scale face verification, the task of given two face images deciding whether both belong to the same person, performances are closely approaching to human level [71]. We are interested in face identification, i.e. inferring the identity of people from their face images, and thus the setup is different than face verification.

3.2.1 DATASETS

Training images are collected from Bing, and benchmark datasets FANlarge [54] and PubFig83 [57] are used to test.

Bing collection: For each name, 500 images are gathered using Bing image search[1]. Categories are chosen as the people having more than 50 annotated face images in FAN-large or PubFig83 datasets. In total, 22,6691 images are collected corresponding to 365 names in FAN-large, and 83 names in PubFig83. Additional 2,500 face images for queries “female face”, “male face”,‘ ‘face

[1]*https : //www.bing.com/*

images" are collected to construct the global negatives. Face detector of [87] is used for detecting faces. Only the most confident detection is selected from each image to be put into the initial pool of faces associated with the name. This process results in 450 faces on the average per category. Other detections are added to global negatives. Note that, this process assumes that the queried person appears as the largest and most visible face in the image, although this is not true in most of the cases and it may result in additional noise.

Test collections: EASY and ALL sets from FAN-large face dataset are used [54]. EASY subset includes faces larger than 60x70 pixels. ALL includes all faces without any size constraint. There are 138 names from EASY, and 365 from ALL subsets, with 23,952 and 199,295 images respectively. On the average there are 541 images for each name. PubFig83 [57] dataset, the subset of well-known PugFig dataset with 83 different celebrities having at least 100 images, is also used in testing. In this set, near-duplicates and the ones that are no longer available at Internet are removed [4].

3.2.2 EVALUATIONS

As seen in Figure 11, at each iteration of model evolution, dataset is divided into candidate positives (the most representative class instances), and possible negatives (where outliers are likely to be found). As Figure 12 shows, FAME is able to learn models from noisy datasets, while eliminating the outliers at successive steps for a variety of people.

We evaluate the performance of FAME on PubFig83 dataset to test the effectiveness of some implementation details. As Figure 13-(a) shows with the increasing number of iterations, more outliers are eliminated. Although some correct instances are also eliminated, the ratio is very low compared to the spurious instances. Moreover, observations show that the eliminated positive examples are usually not in good quality and thus their elimination from the final model is not harmful but rather helpful as supported with the results in Figure 13-(b). As seen in Figure 13-(c) , we can achieve accuracies up to 75.2 on FAN-Large (EASY) and 79.8 on PubFig83 by removing one outlier at each iteration while we prefer to eliminate five outliers for the efficiency.

We also compared the performances obtained with different features on PubFig83 dataset with the models learned from web. While LBP filters alone have the accuracy 60.7 and raw-pixel filters reach up to 71.6, the combination of both gives the highest performance of 79.3. As the results suggests, although LBP filters are not competitive with raw-pixel filters, its textural information is subsidiary to raw-pixel filters with increasing performance.

3.2.3 COMPARISONS

We compare FAME with the baseline method that learns models from the raw collection gathered through querying the name without any pruning. As seen in Table 3, with one versus all L1 norm Linear SVM model on the raw data, the performance is very low on all datasets.

We learn the models from web images and test them on the novel datasets (FAN-large and

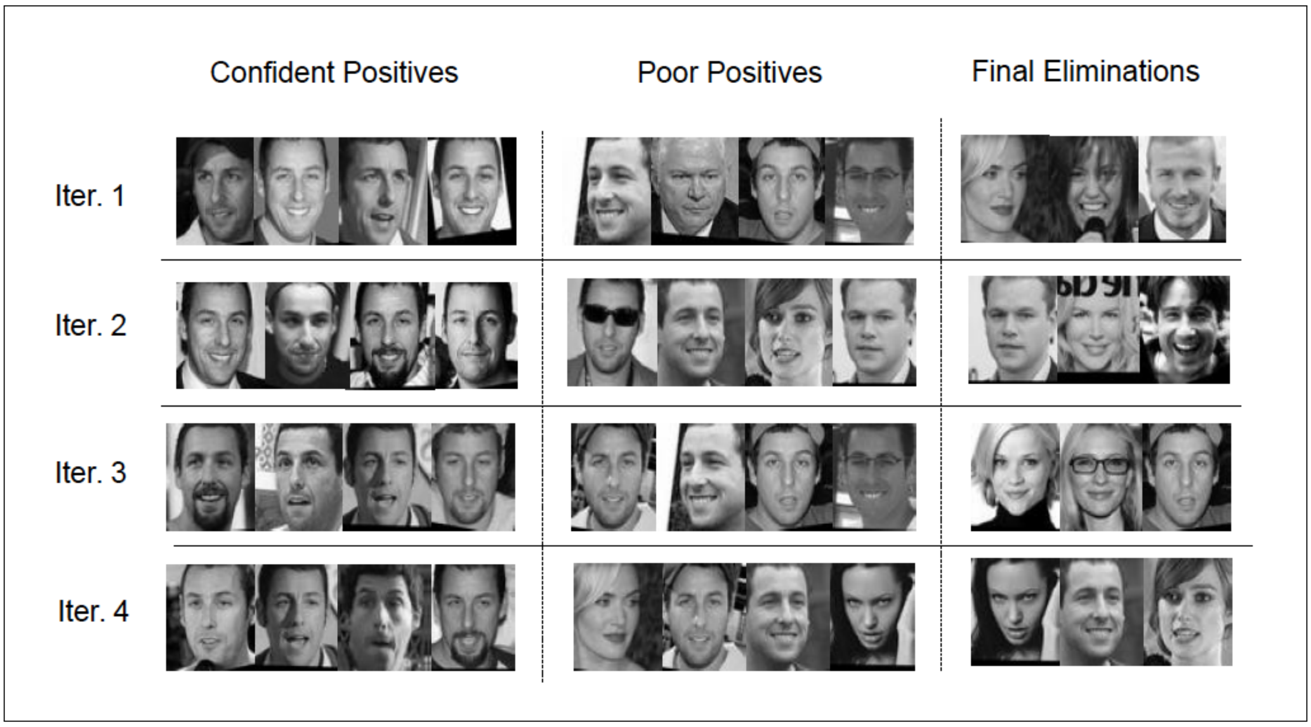


FIGURE 11: SOME OF THE INSTANCES SELECTED FOR CONFIDENT POSITIVES C^+, POOR POSITIVES C^- AND OUTLIERS O FOR ITERATIONS $T = 1 \ldots 4$.

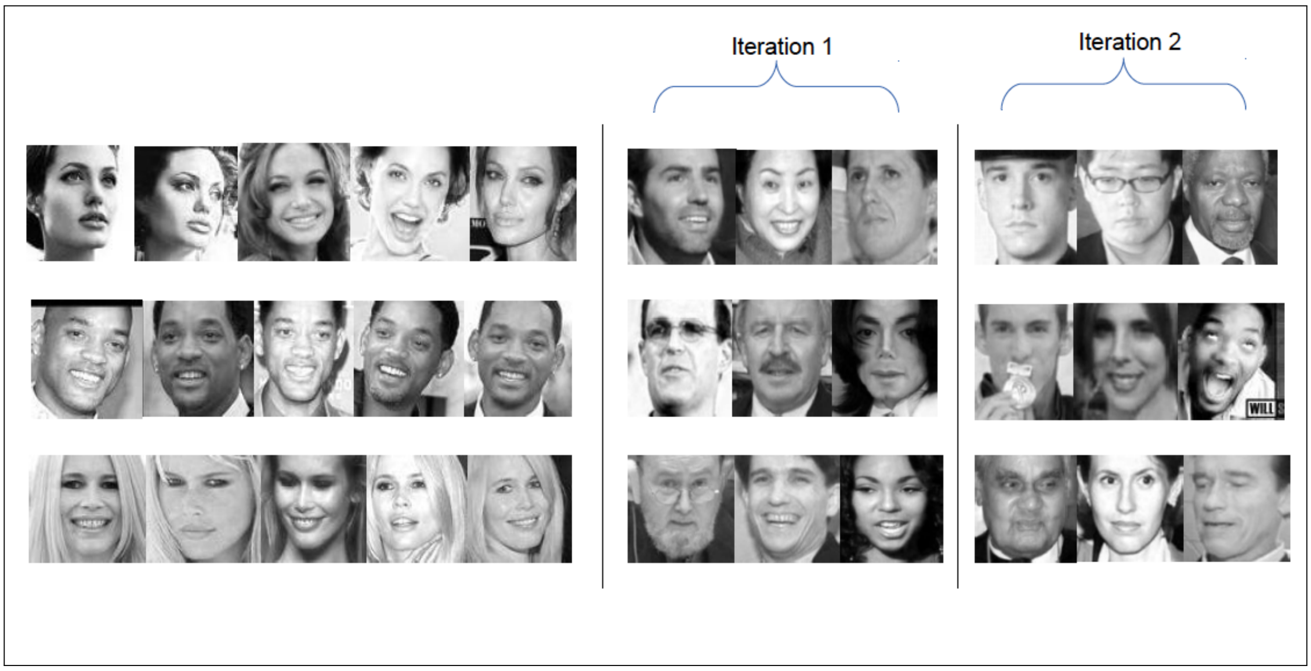
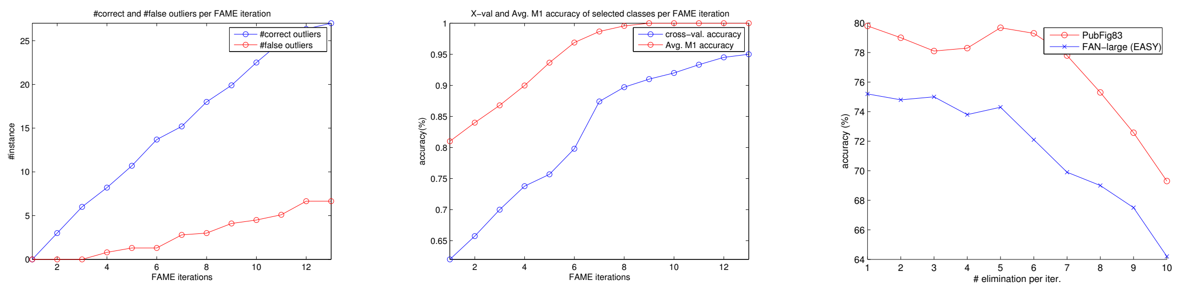


FIGURE 12: FINAL MODEL FACES AND OUTLIERS IN THE FIRST TWO ITERATIONS.



(a) (b) (c)

FIGURE 13: (A) CORRECT VERSUS FALSE OUTLIER DETECTIONS UNTIL ALL THE OUTLIERS ARE FOUND FOR ALL CLASSES. AT EACH ITERATION VALUES ARE AGGREGATED WITH THOSE OF THE PREVIOUS ONE. (B) CROSS-VALIDATION AND M1 ACCURACIES AS THE ALGORITHM PROCEEDS. THERE IS A CORRELATION BETWEEN CROSS-VALIDATION AND M1 MODELS, WITHOUT M1 MODELS INCURRING OVER-FITTING. (C) NUMBER OF OUTLIERS REMOVED AT EACH ITERATION VERSUS ACCURACY. ELIMINATION AFTER SOME LIMIT IMPOSES DEGRADATION OF FINAL PERFORMANCE AND ELIMINATING ONE INSTANCE PER ITERATION IS THE SALIENT SELECTION WITHOUT ANY SANITY CHECK.

PubFig83) for the same categories. To test the effect of training and testing on the same type of dataset, we perform an experiment by dividing the collected Bing images into two subsets. As expected results are better on the same type of data, but FAME leads encouraging results even in the case of domain shift.

As the most similar data handling approach to ours, we compare FAME with the method of Singh et. al. [?] (Table 3). [?] clusters the data to capture intra-cluster variance and uncover the representative instances. They require to decide the optimal cluster number in advance and divide the problem into multiple homologous pieces to be solved separately, increasing the complexity of the proposed system. We solve intra-class modularity by using large dimensional representations that supposedly make different classes linearly separable, even if classes include different modularities. We also find the representative instances by a supervised model which separates representative ones from the rest. Another difference lies in the philosophy. They aim to discover representative and discriminative set of instances whereas we aim to prune spurious ones. Hence, they need to keep all vast negative instances on memory but we can sample different subsets of global negatives and find corresponding outlier instances. It provides faster and easier way of data pruning. They divide each class into two sets and apply their scheme by interchanging data after each iteration like in the case of co-training learning procedure which demands large number of instances for reliable results. We prefer to use all the class data at once in our particular scheme. Comparisons with [?] show the superiority of FAME. We use the released code with up-limit settings of our resources.

To test the effectiveness of the proposed linear regression based model learning, we compare our results by using only the M^1 model (FAME-M1) and using SVM for classification (FAME-SVM). As shown in Table 3, all FAME models outperform the baseline method as well as the method of [?]

TABLE 3: ACCURACIES ON FAN-LARGE OZCAN ET AL. [54] EASY AND ALL (FANL-E AND FANL-A), AND PUBFIG83 AS WELL AS ON THE HELD-OUT SET OF BING COLLECTION. FAME-M1 USES ONLY THE MODEL M1 THAT REMOVES INSTANCES REGARDING GLOBAL NEGATIVES. FAME-SVM USES SVM IN TRAINING AND FAME-LR IS THE PROPOSED METHOD USING LINEAR REGRESSION. BASELINE METHOD LEARNS MODELS DIRECTLY FROM THE ORIGINAL RESULTS WITHOUT PRUNING. COMPARISONS ARE ALSO GIVEN FOR [68].

-	Bing	FANL-E	FANL-A	PubFig83
Baseline	62.5	56.5	52.7	52.8
Singh et. al.[68]	74.7	65.9	62.3	71.4
FAME-M1	78.6	68.3	60.2	71.7
FAME-SVM	81.4	73.1	65.4	76.8
FAME-LR	**83.7**	**74.3**	**67.1**	**79.3**

TABLE 4: COMPARISONS WITH OTHER METHODS ON PUBFIG83. [?] HAS SINGLE LAYER (S) AND MULTI-LAYER (M) ARCHITECTURES. FACE.COM API IS ALSO EXPERIENCED IN [?]. FAME IS TRAINED ON PUBFIG83.

method	[57]-S	face.com [57]	[4]	[57]-M	FAME
acc.	75.6	82.1	85.9	87.1	**90.75**

with a large improvement using the LR model.

Finally, we compare the performance of FAME on the PubFig83 dataset with the other state-of-the-art studies on face identification. In this case, unlike the previous experiments where we learned the models from web images, in order to make a fair comparison we learned the models from the same dataset. As seen in Table 4 FAME achieves the best accuracy in this setting. Referring back to Table 3, even with the domain adaptation setting where the model is learned from the noisy web images our results are comparable to the most recent studies on face identification that train and test on the same dataset. Note that, the method of Pinto et. al. [57] is similar to our classification pipeline but we prefer to learn the filters in an unsupervised way with the method of Coastes et. al. [12]. In this setting, we also test the effect of number of centroids K. The accuracy for K=1500, 2000, 2400 are 84.90, 88.60, 90.75 respectively. Even for K=2000, FAME is better than the other methods.

4 DEEP REPRESENTATION LEARNING

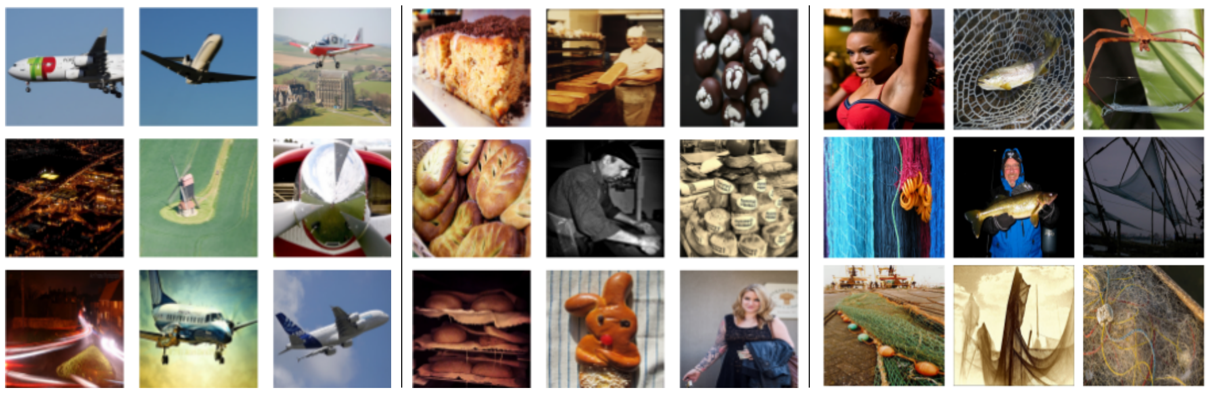
For a long time the vision community has been striving for the quest of creating human-like intelligent vision systems. Recently the resurgence of neural networks [29, 28] has first led to a revolution in computer vision, for example [11, 37], and then quickly provoked to other areas including reinforcement learning [47], speech recognition [25], and natural language processing [46]. The most successful models are the supervised ones that require lots of labeled data, which is costly to obtain. In this section we present an alternative to train deep neural networks for representation learning using massive amounts of unannotated Web images.

The most successful deep learning model is probably convolutional neural networks (i.e. convnet) [22, 41] where its hidden layers transform raw input images into highly discriminative features once the network is trained from millions of labeled images. Importantly, the generalization of convnet goes beyond the general i.i.d assumption of training and test data. For example the AlexNet [37] trained on 1.2 million ImageNet images [63] has been used as an off-the-shelf feature detector for object detection [18], image segmentation [44], and image retrieval [3, 79].

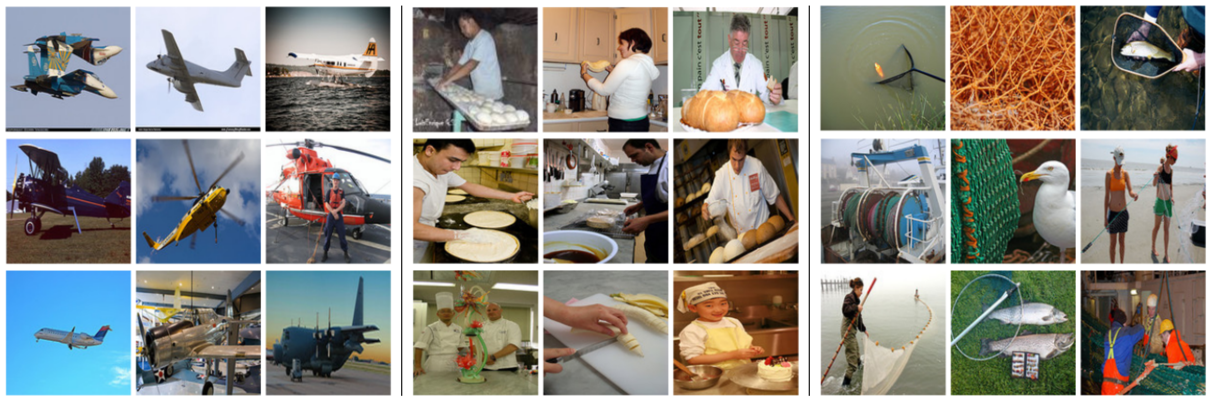
Prior to the success of convnet, a pioneer is [28], where a deep autoencoder can be trained based on unsupervised layer-wise fashion. This coarsely pretrained model could be converted to a classifier afterward by adapting it on a labeled dataset. This sophisticated learning process did not appear in a convnet because linear-rectifer activation [24] replaces conventional activation functions such as sigmoid and tanh. Model adaptation however is still useful for a convnet if there is a need to adapt the pretrained convnet to a specific task. This adaptive method (i.e. fine-tuning), is particularly suitable for fine-grained classification tasks [83, 84, 8].

Does it exists an universal pretrained convnet on which fine-tuning for any problem can be built? A handful of a convnet architectures [37, 65, 70, 27] have been popularly used but they are far from the state of universal models for any vision task to based on. As reported in the new Places database [86], a freshly trained convnet always performs better than a fine-tuned version. In other

(a) IMAGES FROM BING



(b) IMAGES FROM FLICKR



(c) IMAGES FROM IMAGENET

FIGURE 14: IMAGE EXAMPLES OF THREE SYNSETS *AIRFRAME*, *BAKER*, *FISHNET* OF WEB IMAGE COLLECTIONS (A-B) AND IMAGENET (C).

words, whenever abundant amount of labeled data is available, training a convnet from scratch is preferred.

What if labeled data is expensive or scarce? Fine-tuning of a fixed pretrained convnet is still the best choice? It does not seem so, given that recent work [69] found that fine-tuning becomes less helpful when the source domain is loosely related to target domains. A fixed recipe for representation extraction is therefore suboptimal. We re-think an alternative way of representation learning in which a convnet can be inexpensively trained while retaining most of its expressive power. Such an economical model can even serve as an efficient weight initialization for fine-tuning because an initialization does not need to be a perfect one.

By not prioritizing the task of training an image classifier as the premier goal, we exploit convnet as a medium to learn a representational mapping with high reusability. We accomplish this task by optimizing the classification loss of a convnet on massive amounts of unannotated Web images. This data is not completely unlabeled since some of them were annotated manually or automatically by either users or search engines. The data is nevertheless heavily corrupted by noises so that the overall certainty of labeling is low. By training Convnet on this noisy data, the proposed approach benefits both from the generalization capabilities of convnet and the inexpensive data usage of unsupervised learning.

Prior to deep learning, there have been some works [72, 78] that use images harvested from Internet and photo sharing sites such as Flickr to train scalable image classifiers. However there is a lack of a thoroughly empirical analysis on the effectiveness of using noisy Web images to train deep networks. This is where our work comes into the context. Working with Web images comes with both pros and cons. The advantage of Web images easily satisfies the data-hungry property of convnet, our only concern being the tolerance of convnet against noise.

Lately, we learned that a convnet is surprisingly noise tolerable. The studies presented in [69] and soon followed by [82] proposed solutions to train deep convolutional networks as classifiers under noisy condition. In their works, training data is assumed to contain mislabeled images so that probabilistic frameworks are proposed to estimate conditional mislabeling probabilities. Finally those probabilities are integrated as an extra label noise layer placed at the top of convnet in order to improve posterior predictions. Different from [69, 82], we are rather interested in building a robust representation for general purposes from noisy data. Our experiments show that even without any of special treatment of noisy images, a convnet already performs pretty well. We aim to further improve this performance, not just limited to a specific problem.

At a smaller scale, the contribution of [16] is related to our approach. In this work, a robust image representation is learned via the classification of each input image patch into its own class. A good generalization of inner representation can be obtained by training a convnet using unlabeled images. However, the method is limited to small-size images and can thus not be applied to the problem scale we tackle.

The contribution of our approach is threefold. First, we train convnets using noisy and unlabeled Web images retrieved from the image search engine Bing and the photo sharing network Flickr. Experiments are scaled from a small image collection of a hundred concepts and 400K images to a larger collection with a thousand concepts, and 3.14 million images. According to both scales, the learned representations provide good generalized features that lead to promising accuracy on many classification datasets. Second, we use image reranking techniques to remove noise from training data and train convnets with a deeper architectures. Results show that the proposed techniques help improving classification results significantly. The best of our configuration outperforms AlexNet [37] and closes the gap with VGG-16 [65]. Third, our approach offers an inexpensive, yet effective way to initialize refinement training on fine-grained category datasets.

In the remainder of this section, we present related works in Section 4.1, data collection procedures in Section 4.2 and methods in Section 4.3. Section 4.4 presents experiment results.

4.1 RELATED WORK

Sharing many of its aspects to dimensionality reduction and manifold learning, representation learning means to find a mapping of high dimensional input data into some low dimensional representation such that the new representation exposes meaningful data dynamics. Different from manifold learning, where its resulting embedding are best used in data visualization [75, 5], representation learning is interested in learning discriminating features that exploit categorical properties of data in order to classify them. The representation to be learned should be not only robust against data perplexity but also sensitive to intrinsic dimensions of data.

In the previous section, we mentioned that convnets have been the *de facto* representation learning method thanks to their excellent generalization. At the heart of a convnet is an end-to-end feature mapping: starting from raw pixel intensities, then through many hidden layers of different types, a robust representation can be learned. Giryes et al. [23] proved that deep neural networks are distance-preserving mappings with a special treatment for intra- and inter-class data.

The distinguishing point of a convnet is that it learns distributed representations [6], the property that is absent in shallow networks. Having a distributed representation is indeed much more expressive than a local representation due to lesser hidden units [15] and much more regions of linearity [48]. There have been theoretical justifications of deep networks as a class of universal approximates [14, 30]. A more recent work, [2] proved that a two-layer rectifier network can make any disjoint data linearly separable. While a distributed representation is common in many deep networks, shift-invariance [9] and locality preservation, the properties of convolutional and pooling layers, add to the uniqueness of convnets. In fact, convolution is crucial for convnets to obtain better representation than other deep networks such as stacked auto-encoders (e.g. when comparing results in [40] and [37]).

Representation learning has been conventionally pursued by unsupervised methods such as auto-encoders [29], deep belief nets [28], and sparse encoding [58]. From our viewpoint, representation learning should combine advantages of both supervised and unsupervised paradigms. Unsupervised learning alone lacks a strong data prior. Since label information of training data is unavailable in an unsupervised setting, the objective function of an unsupervised network [29] uses reconstruction loss. This loss relies too much on redundant image details (i.e. it tries to reconstruct as much as possible input images at pixel level), so that an unsupervised model is less capable of generalizing distinctive features, which are necessary for classification.

In supervised learning, on the contrary, there is access to the labels of training data, thus providing better guidance. By minimizing the classification loss that penalizes the difference between predicted labels versus groundtruth, supervised training helps pruning unnecessary details and magnifying discriminating features. This perspective is also shared in [16, 73, 61]. We notice that this perspective is different from the known way [28] of combining unsupervised and supervised learning to train multi-layer networks.

Our method can be classified as semi-supervised learning, where there exists in training data a non-negligible amount of properly labeled images among lots of noise (i.e. images whose contents are irrelevant to their labels). While these both unlabeled and labeled images are useful in learning strong low-level features, properly labeled images could sharpen on the distinctiveness of the representation to be learned in top layers.

4.2 WEB IMAGE COLLECTIONS

In this section, we present steps to collect and organize training collections. We draw randomly a subset of synsets from Wordnet. Given a synset, its synonyms are used as keywords to retrieve images. Thanks to Bing and Flickr APIs, we can retrieve images with zero cost. Data imbalance is managed by setting a threshold on the total number of images per synset. Downloaded images are not subjected to manual screening except the removal of duplicating images.

As seen in Fig. 14, Web images pose several challenges. First of all, each data source has its own bias. Bing seems to have more documentary images and diagrams, while Flickr has many personal photos with better aesthetic quality. This difference is perhaps rooted from the way images are contributed to and indexed by those platforms. In particular, Bing Image Search has an underlying mechanism of text-based search so that images with rich accompanying text are better indexed, thus appear more in top retrieved results. On the contrary Flickr images are uploaded by users and are strongly oriented toward a consumer or advanced photographic usage (pro or semi-pro), including specific groups of interest.

Using images from multiple sources raises both pros and cons. On the positive side, mixing images leads to better data diversification. On the negative one, mixing images may reduce intra-class consistency. Experiments have shown us that mixing Flickr and Bing sources always leads to better results.

There are two image collections used in our experiments. The small one tests our approach in small-scale problems and also takes less training time, which is convenient to study the influence of various settings. The large collection validates the proposed hypotheses at large scale and can be used to compare with state-of-the-art methods.

Flickr-Bing 100 (FB100) consists of 100 synsets. Out of its 416K images, 67% are from Flickr and 33% from Bing.

Flickr-Bing 1K (FB1K) consists of 1000 synsets used by the classification challenge ILSVRC'12 [63]. Using the same synsets allows us to compare our approach versus state-of-the-art. Out of its 3.14 million images 30% are from Bing and 70% are from Flickr.

4.3 PROPOSED METHOD

4.3.1 IMAGE RERANKING TECHNIQUES

Will a convnet perform better if it is trained with cleaner data? We convey three reranking techniques to answer this question. Given that reranking is just a preprocessing step, it is regarded as helpful if the learned representation produces better classification performance.

In the following, let us denote $\mathcal{L} = \{(\mathbf{x}_i, y_i)\}_{i=1}^m$ the set of labeled examples and $\mathcal{U} = \{\mathbf{z}_j\}_{j=1}^n$ the set of retrieved Web images with $m \ll n$. Here, $\mathbf{x}_i$ and $\mathbf{z}_j$ are the vectorial representations of labeled example i and unlabeled image j. A reranking algorithm aims to select a subset $\mathcal{S} \subset \mathcal{U}$ such that elements $\mathbf{x}_i \in \mathcal{S}$ are more relevant to at least one of the elements in $\mathcal{L}$ than to those in $\mathcal{U}\backslash\mathcal{S}$.

Cross-Validation (CV) splits $\mathcal{U}$ into K equal disjoint subsets. Each subject is scored by a binary SVM classifier [13] trained on the rest $(K-1)$ subsets as positive data and 10K irrelevant images as negative data. Every data point in $\mathcal{U}$ is predicted once. Negative-scored data are considered as noise and rejected. A larger K tends to reject less images.

Kernel Mean Matching (KMM) is a semi-supervised technique [31] that reweights unlabeled data $\mathbf{z}_i \in \mathcal{U}$ w.r.t labeled data $\mathbf{x}_i \in \mathcal{L}$ such that the (weighted) arithmetic means of the two sets are approximately equal, i.e. $\sum \mathbf{x}_i/m \approx \sum \alpha_i \mathbf{z}_i / (\sum \alpha_j)$. If $\alpha_i \approx 0$ then $\mathbf{z}_i$ is considered as noise. The optimal $\boldsymbol{\alpha}^*$ is the solution of the following convex quadratic program

$$\arg \min_{\substack{\boldsymbol{\alpha}\succeq 0 \\ \boldsymbol{\alpha}'\mathbf{1}\approx n}} \frac{1}{2} \left\| \frac{1}{m}\sum_{i=1}^m \mathbf{x}_i - \frac{1}{n}\sum_{j=1}^n \alpha_j \mathbf{z}_j \right\|^2 . \tag{6}$$

Transductive Support Vector Machine (TSVM) The general assumption of TSVM[66] is that $|\mathcal{L}|$ could be much smaller than $|\mathcal{U}|$, which fits to our reranking problem. The inference of TSVM is optimized w.r.t both training $\mathcal{L}$ and test data $\mathcal{U}$. To rerank $\mathcal{U}$ the following non-convex program must be iteratively solved

$$\arg \min_{\mathbf{w}} \frac{1}{2} \|\mathbf{w}\|_2^2 + \frac{\alpha}{m}\sum_{i=1}^m \ell(y_i \mathbf{w}'\mathbf{x}_i) + \frac{\beta}{n}\sum_{j=1}^n \ell(t_j \mathbf{w}'\mathbf{z}_j), \tag{7}$$

where α and β control the influences of labeled data $\{\mathbf{x}_i\}$ and unlabeled data $\{\mathbf{z}_i\}$ on the classifier $\mathbf{w}$; the loss $\ell(\cdot)$ penalizes the predicted labels $\hat{t}_j = \text{sign}(\mathbf{w}'\mathbf{z}_j)$ and $\hat{x}_i = \text{sign}(\mathbf{w}'\mathbf{x}_i)$ w.r.t temporary label t_i and groundtruth y_i, respectively. Here, $\{t_j\}$ is the set of temporary labels of $\{\mathbf{z}_j\}$ during optimization, i.e. $\{t_j\}^{(\tau)}$ is assigned by $\mathbf{w}^{(\tau)}$ at iteration τ-th. The optimal $\mathbf{w}$ is found when $\{t_j\}^{(\tau)} \equiv \{t_j\}^{(\tau+1)}$. Unlabeled point $\mathbf{z}_j$ is considered as noise if $t_j = -1$. To avoid a trivial solution, the ratio of positive labels $\{t_j\}_+$ versus negative labels $\{t_j\}_-$ is fixed.

4.3.2 CONVNET ARCHITECTURES

With millions of trainable parameters, dozens of hyperparameters and close to infinite arrangements of network topology, architecture design of deep neural networks is more an art than science. Fortunately the architecture of convnet is constrained by some factors such as feedforward topology, particularities of layers and their relative orders. So far, just several architectures are performing seamlessly, for example AlexNet [37], VGG [65], PreLU-nets [27], GoogLeNet [70]. We adopt AlexNet [37] as the default setting in our experiments due to its popularity, modest time complexity and lesser memory demanding than others.

In most cases, increasing the depth of convnet leads to performance gain [65, 27, 70]. However, we encounter some problems when adding more layers. The first one is related to the size of the minibatch in stochastic gradient descent (SGD), the optimization algorithm used to train convnet. Given fixed memory resource, reducing the batch size will release memory space enough to add some extra convolutional layers. While small batch sizes, for example 32 images per patch, still guarantee the convergence of SGD, this may no longer hold if training data are heavily corrupted by noise. In practice, we found that with a too small batch size, our convnet could not learn anything after many thousands of iterations. This is probably the consequence of several factors in combination such as fluctuating gradient directions, too small gradient magnitude caused by noise-contaminated minibatch, and random states of many hidden layers at the beginning.

The second problems relates to the size of the convolution filter. Recent works, for example [65], suggest that small filter sizes such as 3×3 and even 1×1 tend to improve classification performance. However, in our experiments small filter sizes, especially at very first convolutional layers, are not helpful. It is probably due to overcomplex instances of Web images, where objects in images may appear under any style (e.g. cartoon, diagram, sketch, artistic, etc.), scales and orientations.

Taking into account all the factors above we propose the architecture FB-13 as described in Table 5. FB-13 consists of 13 layers, where the first convolutional layer has filter size 7×7. To train FB-13 we use minibatch of size 196; the training uses up to 11GB RAM on a single GTX Titan-Z card and takes about 80 minutes for 1000 iterations; the training stops after 350K iterations.

We also test with GoogLeNet [70], denoted FB-GLN, which is very deep. On average it takes 25 minutes for a 1000 iterations with the minibatch size 128. The training stops after 1.2 million iterations.

4.4 EXPERIMENTS

Learned representations are evaluated by classification tasks on datasets coming from several topics: indoor scenes - MIT67 [60], outdoor and street scenes - SUN397 [81], human actions - Action40 [85], object categories - Caltech256 [26] and VOC07 [19]. We are also interested in fine-grained category datasets: Flowers102 [51], Dogs120 [34], Cars196 [36] and Birds200 [77]. Classification accuracy is mostly used in evaluations with the exception of mean average precision (MAP), that is used to evaluate VOC07.

Layer	**AlexNet**[37]	**VGG-16**[65]	**FB-13 (Ours)**
conv	$11 \times 11, 96, /4$	$3 \times 3, 64$	$7 \times 7, 96, /2$
		$3 \times 3, 64$	
maxpool	$3 \times 3, /2$	$2 \times 2, /2$	
conv	$5 \times 5, 256$	$3 \times 3, 128$	
		$3 \times 3, 128$	
maxpool	$3 \times 3, /2$	$2 \times 2, /2$	$2 \times 2, /2$
conv	$3 \times 3, 384$	$3 \times 3, 256$	$3 \times 3, 256$
		$3 \times 3, 256$	$3 \times 3, 256$
		$3 \times 3, 256$	$3 \times 3, 256$
maxpool		$2 \times 2, /2$	$2 \times 2, /2$
conv	$3 \times 3, 384$	$3 \times 3, 512$	$3 \times 3, 512$
		$3 \times 3, 512$	$3 \times 3, 512$
		$3 \times 3, 512$	$3 \times 3, 512$
maxpool		$2 \times 2, /2$	$2 \times 2, /2$
conv	$3 \times 3, 256$	$3 \times 3, 512$	$3 \times 3, 512$
		$3 \times 3, 512$	$3 \times 3, 512$
		$3 \times 3, 512$	$3 \times 3, 512$
maxpool	$3 \times 3, /2$	$2 \times 2, /2$	$2 \times 2, /2$
fc6	4096	4096	4096
fc7	4096	4096	4096
fc8	1000	1000	1000

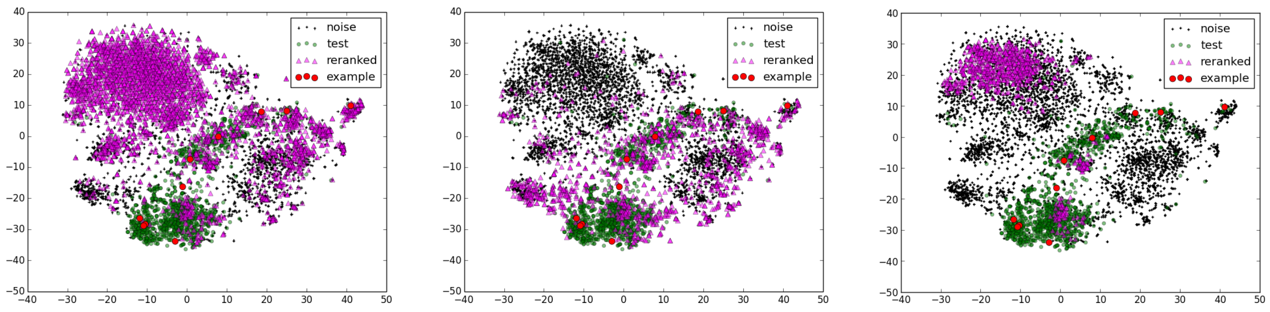
TABLE 5: THE FEEDFORWARD ARCHITECTURES OF TWO REFERENCES ARCHITECTURES (ALEXNET AND VGG-16) VERSUS OURS (FB-13).

Reranking	n/a	CV	KMM	TSVM
FB100 (10^3)	406.5	291	81.3	98.9
FB1K (10^6)	3.14	2.52	1.44	2.03

TABLE 6: TRAINING SIZES BEFORE AND AFTER RERANKING.

4.4.1 EXPERIMENTAL SETUP

Reranking We use the pretrained model AlexNet [37] delivered with Caffe [33] as the feature extractor. Images forwarded at the input layer will produce 4096-dimensional feature vectors at `fc7` layer; L_2-normalization is applied before running reranking methods. For CV, we use the linear kernel with $C = 1$ and the number of folds $K = 5$. For KMM, hyperparameter B is set to 5. For TSVM the ratio $|\{t_j\}_+|/|\{t_j\}_-|$ is set to $1000/n$ so that about 1000 images from each synset will be selected as clean; hyperparameters $\alpha = 1$ and $\beta = 10^{-4}$. Since both KMM and TSVM are semi-supervised, $m = 10$ labeled examples are provided for each synset.



(a) CROSS-VALIDATION **(b)** KERNEL MEAN MATCHING **(c)** TRANSDUCTIVE SVM

FIGURE 15: THE VISUALIZATIONS OF THREE RERANKING TECHNIQUES, CV (A), KMM (B), AND TSVM (C) ON THE SYNSET *SALMON*. RED AND GREEN DENOTE IMAGENET IMAGES WHILE BLACK AND PINK DENOTE WEB IMAGES.

Visualizing Reranking Results Behaviors of reranking algorithms may be partly perceived through visualization. To do so, we extract `fc7` features of images from synset "salmon", run the dimensionality reduction method t-SNE [75] and visualize in Fig. 15 the resulting 2D embedding. From the visualization, we see that both CV and TSVM are better KMM in preserving bias of input data while KMM is better the others in selecting relevant images w.r.t given examples. This fact helps explaining later results, where KMM often leads to improved classification results when the test set is drawn from the same distribution of examples.

Convnet Training We denote FB the convnet with AlexNet architecture trained on original Web images; similarly FB$_{cv}$, FB$_{kmm}$ and FB$_{tsvm}$ denote the convnets trained on reranked data of methods CV, KMM and TSVM, respectively. Since the size of training sets may be severely reduced after reranking (see Table 6) which may cause overfitting, we use convnets trained on FB and FB$_{cv}$ to additionally do fine-tuning on reranked data. To evaluate representations learned by our convnets we compare them to a reference, denoted as REF, which is a convnet with AlexNet architecture trained on equivalent synset sets using ImageNet labeled data. To compare with our convnets on FB100, REF is trained on ImageNet100 with 100K labeled images drawn from ImageNet. To compare on FB1K, REF is trained on ILSVRC'12 training set [63].

The Caffe library [33] is used to train convnets. The learning rate of a fresh training starts at $\eta = 10^{-2}$ and decreases by a magnitude of $1/\gamma = 10$ after each of 10^4 iterations; the maximum number of iterations is 450K but an earlier stop could be made if validation accuracy does not improve anymore. For fine-tuning, the learning rate starts at $\eta = 10^{-3}$ and is reduced after each 30K or 50K iterations, depending on the size of data. A fine-tuning process may stop after 150K iterations.

4.4.2 RESULTS OF FLICKRBING-100

End-to-End Classifiers We show in Table 7a the classification accuracies of our convnets and the reference configuration on ImageNet100 test set with 100 images per synset. With no surprise, REF model outperforms at 67.8% accuracy. What is nevertheless surprising is that our best results for FB$^{\dagger}_{kmm}$ is 58.7% (the fine-tuned version of FB on reranked data FB$_{kmm}$) and FB$_{cv}$ is 54.3% (the convnet trained on reranked data FB$_{cv}$). Although creating end-to-end classifiers is not our premier goal, these results show that convnet is very capable of achieving good generalization from Web images, at least on small scale data like FB100.

Results also indicate that image reranking in general improves classification accuracy, given that the number of images after reranking must be sufficient to avoid overfitting. The convnets trained on reranked data of KMM and TSVM gives worse results than others due to not satisfying this requirement (see Table 6). Nevertheless, reranked data are still useful if they are used with fine-tuning, as done in this experiment.

Representation Learning To evaluate the generalization of the learned representations on new datasets, we compute L_2-norm `fc7` image features for each of dataset and train one-vs-rest linear SVM classifiers. The evaluation protocol for each dataset is strictly followed. Table 7b shows good generalization performances of our convnets on all of six datasets where our results outperforms REF. Though interesting, these results need to be verified at larger scales because in this experiment FB100 outnumbers ImageNet100 so that the former may cover better image diversity, which is important for good generalization of convnets trained on FB.

Remark 1 *Data abundance is crucial in training convnets and even more important if the data are noisy.*

4.4.3 RESULTS OF FLICKRBING-1K

This time, we redo experiments in previous section on the large collection FB1K. Since it takes more training time on FB1K than FB100, we just select the best configurations to evaluate. Furthermore, image reranking will run on image features produced by the convnet trained on FB1K to make a fair comparison with the state-of-the-art. To evaluate our results, we compare against the reference

	REF	(†)FB	(‡)FB$_{cv}$	FB$_{kmm}$	FB$_{tsvm}$	FB$^{\dagger}_{cv}$	FB$^{\dagger}_{kmm}$	FB$^{\dagger}_{tsvm}$	FB$^{\ddagger}_{kmm}$	FB$^{\ddagger}_{tsvm}$
ImageNet100	68.8	53.6	54.3	48.9	51	55.3	58.7	57.7	58.4	57.6

(a) CONVNETS AS END-TO-END CLASSIFIERS. THE NOTATION FB$^{\dagger}_{CV}$ MEANS IT IS THE FINE-TUNED VERSION OF FB W.R.T RERANKED DATA FB$_{CV}$.

	REF	(†)FB	(‡)FB$_{cv}$	FB$_{kmm}$	FB$_{tsvm}$	FB$^{\dagger}_{cv}$	FB$^{\dagger}_{kmm}$	FB$^{\dagger}_{tsvm}$	FB$^{\ddagger}_{kmm}$	FB$^{\ddagger}_{tsvm}$
VOC07	54.7	57.9	57.9	48.5	54.3	58.2	57.5	58.8	58.2	58.6
Flowers102	72.4	74.2	74.4	64.3	72.4	74.3	75.4	77.7	75.3	76.1
MIT67	31.9	34.1	34	27.2	31.4	34.6	33	38.3	34.9	36.3
Action40	36.8	37.9	38.2	29.1	35.6	39.2	38.9	41.4	38.3	40
Caltech256	42.5	44.7	44.6	36.4	42	44.6	44.5	46.7	44.7	46.7
SUN397	26.9	28.6	28.6	20.8	26.2	28.7	29.1	31.1	28.7	30.7

(b) CONVNETS AS TRANSFERABLE FEATURE DETECTORS

TABLE 7: EVALUATION OF REPRESENTATION LEARNING ON FB100. THE REFERENCE MODEL REF IS TRAINED ON IMAGENET DATA.

model REF which is the AlexNet trained on ILSVRC12 training set.

End-to-end Classifiers Shown in Table 8a are the results of our convnets on the ILSVRC12 validation set with 100 images per synset. Compared to results obtained with FB100 in Table. 7a, this time our convnets perform much worse than the reference. The results do not contradict, however faithfully reflect the nature of Web images, which is noisy and highly biased. While the impact of noise is less seen with FB100, it severely hurts performance in the case of FB1K. The results pose a great challenge for unsupervised methods that aim to learn large-scale classifiers from unlabeled data.

Representation Learning Table 8b shows that our best results are comparable to REF. In particular, our results on VOC07, Caltech256, and SUN 297 obtained by FB$^{\dagger}_{cv}$ are competitive against REF. Interestingly, our results on Flower102 slightly outperform REF's. Results on MIT67 and Action40 of our convnets are not so good as other cases but still promising. Compared with the small problem FB100, it is harder for our method to obtain competitive results with REF. Still, representation learning with good generalization could be achieved by training a deep convolutional net on unlabeled and noisy Web images.

Image reranking methods also add incremental values to the approach. Among reranking methods, the unsupervised CV is the best and consistently improves results both on FB100 and FB1K. Weak-supervised methods, KMM and TSVM, contribute little in this experiment and in some cases even decrease the performance. This is due to several reasons, such as inconsistent biases between examples given to reranking algorithms and unlabeled Web images to be ranked, or the number

	REF	(†)FB	FB$^\dagger_{cv}$	FB$^\dagger_{kmm}$	FB$^\dagger_{tsvm}$
ILSVRC12	80.4	23.4	23.8	23.1	22.8

(a) CONVNETS AS END-TO-END CLASSIFIERS. THE NOTATION FB$^\dagger_{CV}$ MEANS IT IS THE FINE-TUNED VERSION OF FB W.R.T RERANKED DATA FB$_{CV}$.

	REF	(†)FB	FB$^\dagger_{cv}$	FB$^\dagger_{kmm}$	FB$^\dagger_{tsvm}$
VOC07	71.7	70.8	71	70.6	70.4
Flowers102	87	87.6	88.4	88.6	89.4
MIT67	56	52.5	53.1	53.7	51.9
Action40	60.2	55.9	56.5	56.1	56.3
Caltech256	70.3	68.6	69.5	68.9	69.7
SUN397	46.1	45.8	46.1	45.4	45.4

(b) CONVNETS AS TRANSFERABLE FEATURE DETECTORS

TABLE 8: EVALUATION OF REPRESENTATION LEARNING ON FB1K. THE REFERENCE MODEL REF IS TRAINED ON IMAGENET DATA.

of images after reranking is insufficient. KMM and TSVM would be more useful if the provided examples are also drawn from Web collections and we consider this case as future work.

4.4.4 RESULTS OF DEEPER ARCHITECTURES

As the power of deep models relies on their depth, we are curious if a deeper convnet trained on noisy data could learn better representations. We train a 13-layer convnet FB-13 (see Section 4.3.2) on FB1K data and compare it against the two other architectures: AlexNet (denoted as FB) and GoogLeNet (denoted as FB-GLN), which are also trained on FB1K. As before, we compare our results to the state-of-the-art results of AlexNet (denoted as REF) and VGG-16 trained on ImageNet data.

The common problem in training a very deep convnet is the impossibility of SGD convergence. This is caused by gradient vanishing, i.e. gradients backpropagated from the top loss layer vanish before they reach bottom layers. Rectified linear activation seems to solve this problem so that [37] we can train AlexNet in a single pass without using layer-wise pretraining. However this problem reappears in training convnets which are deeper than AlexNet. To resolve this, we use the solution of [27] that initializes the Gaussian random weights w.r.t the formula $std = \sqrt{2/(k_l^2 c_l)}$ in which k_l and c_l are the filter dimension and input channels of l-th convolutional layer.

Once convnets are trained, their `fc7` features are used to train linear SVM classifiers. Because the `fc7` layer is absent in GoogLeNet, we replace it by concatenating last layers `loss1/fc`, `loss2/fc` and `pool5/7x7_s1`. Results are presented in Table 9.

	REF	VGG-16	FB	FB-13	FB-GLN
VOC07	71.7	79.9	70.8	76.6	76.1
Flowers102	87	87.5	87.6	88.8	86.8
MIT67	56	67.1	52.5	61.6	57.2
Action40	60.2	72.6	55.9	63.3	63.8
Caltech256	70.3	77.9	68.6	75.2	71.9
SUN397	46.1	53.8	45.8	51	47.1

TABLE 9: EVALUATION OF REPRESENTATIONS LEARNED FROM FB1K OF CONVNET ARCHITECTURES WITH VARYING DEPTHS.

FB-13 outperforms REF on all of the six datasets. This means greater depth leads to good generalization even when training data are noisy. In other words, unlabeled images are underestimated (due to their lack of labels) but our experiments indicate that their content actually bring more valuable information than conventionally thought. Also in Table 9 FB-13 is inferior to VGG-16; however this gap may be narrowed down by increasing FB-13's depth up to 16 or even more.

Recent theoretical results [48, 2] have proved that the use of rectified linear units (ReLU) is crucial to the excellent performance of convnets. [2] proved that a neural net of more than two ReLU layers is capable of linearly separate any disjoint data. According to [48], a deep network represent a compositional mapping where individual map at each layer shatters input space into a number of linear regions. The compositional map therefore is able to map portions of each layer's input-space to the same output. As the number of layers increases, the compositonal map is capable of computing more complex functions (i.e. it maps more linear regions to the same output). This means the convnet may identify better intra-class versus inter-class instances. This explanation seems appropriate to justify the results of this experiment.

Remark 2 *The good generalization of convnets to unseen data is due to their deep architecture with rectified activators.*

Results in Table 8b also reveals that GoogLeNet is not so competitive as conventional architectures where convolutional layers are followed by three fully connected layers. However this may depend on the way features are extracted so that other feature extraction techniques, for example [50], should be attempted.

4.4.5 WEB IMAGES AND FINE-TUNING

Recall that in section 1, we mentioned that fine-tuning requires a sensible weight initialization. However, it is not clear whether this initialization must come from a well-trained convnet, i.e. using labeled images. In this section, we answer that question by comparing classification accuracy of convnets refined based on two initialization points: i) the pretrained AlexNet model of ILSVRC'12 training data and ii) the pretrained AlexNet model of FB1K. Comparisons are made on four fine-grained category datasets of 200 bird species, 196 car models, 120 dog species and 102 flower

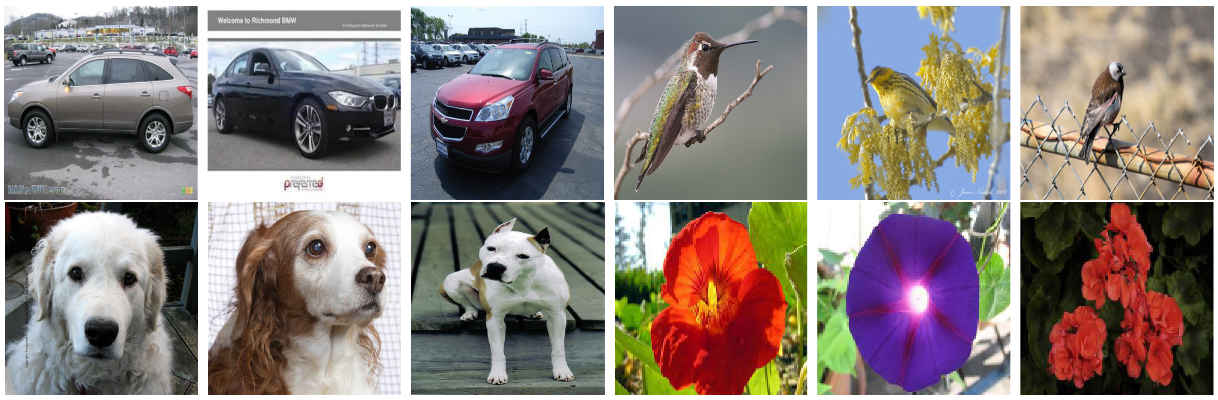


FIGURE 16: EXAMPLES FROM FINE-GRAINED CATEGORY DATASETS.

	FB	REF
Birds200	52.7 (*47.1*)	52.1 (*47.1*)
Cars196	50.9 (*31.4*)	51.9 (*34.7*)
Flowers102	79.6 (*87.6*)	80.2 (*87.0*)
Dogs120	62.5 (*61.7*)	70.5 (*73.1*)

TABLE 10: RESULTS OF FINE-TUNING BY OUR MODEL (FB) AND THE REFERENCE (REF). IN BRACKETS ARE RESULTS BEFORE FINE-TUNING.

species. Some example images are illustrated in Fig. 16. The numbers of training images per category vary from 10, 30, 40 (Flowers, Birds, Cars respectively) to a hundred (Dogs).

Similarly to previous experiments, evaluation of a pretrained convnet is based on the classification task on third-party datasets in which image features are computed at the `fc7` layer of the convnet. Fine-tuning will discard the last layer of the pretrained convnet, which is `fc8` in our case, in order to learn the new fc8 layer whose outputs match defined classes of the new data. Evaluation of a fine-tuned model therefore become straightforward: test it as an end-to-end classifier.

Fine-tuning results are listed in Table 10. For three out of four datasets, either initialized by FB or REF the convnets perform comparably. Their comparative results before fine-tuning are also very close. This means performance of fine-tuning on fine-grained category problems does not depend on the quality of weight initialization. However, the result on Dogs120 dataset does not align to our conclusion, where REF gives superior accuracy. Behind this exception is the fact that i) Dogs120 is drawn from ImageNet, the domain used to train REF, and ii) the training data of REF already contains a lot of labeled dogs images.

Remark 3 *Convnets trained on massive amount of noisy Web images may establish a good initialization for model refinement on fine-grained category data.*

Notice that the sufficiency of fine-tuning data must be respected, otherwise the convnet to be tuned gets easily overfitted. Flowers102 is an example of this case where there are just 10 training images per category, which makes the accuracy after fine-tuning worse than the pretrained state. Under such data scarcity of circumstances, the max-margin SVM classifier [13] is more likely to perform better than a logistic classifier. Also, fine-tuning is not recommended if the pretrained convnet was already well trained for the classes to be fine-tuned. This is the case when refining REF on Dogs120 gives worse results.

5 CONCLUSION

In this deliverable we present the construction of visual concepts from several points of view. First, Concept Map proposed in Section 2 addresses challenges in acquiring image data from Web; as a result, a unsupervised Kohonen neural network is proposed to remove outliers and improve the consistency of image groups that represent visual concepts. However it turns out that a more sophisticated method is necessary to differentiate subtle changes between fined-grained categories. Hence the second work in Section 3 proposed the method FAME in order to build face concepts of celebrities and politicians. In this method, we use LBP features, which is widely used in face recognition problems, to represent face images. A data elimination algorithm then runs on each of identity and iteratively removes irrelevant images, thus builds up a stronger identity representation. In Section 4, we proposed the methodology of deep learning to build a large amount of visual concepts based on noisy Web images. At some extent, this is similar to what Concept Map does; however deep representation learning harvests instance images and builds representation for given visual concepts while Concept Map discovers visual concepts from a given image collection. Instead, the image acquisition stage of deep representation learning shares objectives with the method FAME. Benefiting from abundant amount of Web images and the efficiency of image reranking algorithms, we can collect enough data for deep convolutional networks, the state of the art method for image classification. Despite of difficulties in training deep networks from weakly labeled and noisy images, the proposed method shows promising results in most of public datasets with different themes ranging from ordinary objects, to indoor and outdoor scenes, and even fine-grained categories such as flowers, dogs, and cars. The works presented in this deliverable can be integrated to build up a flexible and expandable visual similarity framework in the MUCKE system.

References

[1] Timo Ahonen, Abdenour Hadid, and Matti Pietikainen. Face description with local binary patterns: Application to face recognition. *PAMI*, 28(12), 2006.

[2] Senjian An, Farid Boussaïd, and Mohammed Bennamoun. How can deep rectifier networks achieve linear separability and preserve distances? In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, pages 514–523, 2015.

[3] Artem Babenko, Anton Slesarev, Alexander Chigorin, and Victor S. Lempitsky. Neural codes for image retrieval. In *ECCV*, 2014.

[4] B.C. Becker and E.G. Ortiz. Evaluating open-universe face identification on the web. In *CVPR Workshop on Analysis and Modeling of Faces and Gestures*, 2013.

[5] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps and spectral techniques for embedding and clustering. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 585–591, 2001.

[6] Yoshua Bengio, Aaron C. Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE TPAMI*, 35(8):1798–1828, 2013.

[7] G. Bradski. *Dr. Dobb's Journal of Software Tools*.

[8] Steve Branson, Grant Van Horn, Serge Belongie, and Pietro Perona. Bird species categorization using pose normalized deep convolutional nets. *CoRR*, abs/1406.2952, 2014.

[9] Joan Bruna, Arthur Szlam, and Yann LeCun. Learning stable group invariant representations with convolutional networks. *CoRR*, abs/1301.3537, 2013.

[10] Dong Chen, Xudong Cao, Fang Wen, and Jian Sun. Blessing of dimensionality: High-dimensional feature and its efficient compression for face verification. In *CVPR*, 2013.

[11] Dan C. Ciresan, Ueli Meier, Jonathan Masci, and Jürgen Schmidhuber. Multi-column deep neural network for traffic sign classification. *Neural Networks*, 32:333–338, 2012.

[12] Adam Coates, Andrew Y Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *International Conference on Artificial Intelligence and Statistics*, pages 215–223, 2011.

[13] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*.

[14] George Cybenko. Approximation by superpositions of a sigmoidal function. *MCSS*, 5(4):455, 1992.

[15] Olivier Delalleau and Yoshua Bengio. Shallow vs. deep sum-product networks. In *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011. Proceedings of a meeting held 12-14 December 2011, Granada, Spain.*, pages 666–674, 2011.

[16] Alexey Dosovitskiy, Jost Tobias Springenberg, Martin A. Riedmiller, and Thomas Brox. Discriminative unsupervised feature learning with convolutional neural networks. In *NIPS*, 2014.

[17] Erkut Erdem and Aykut Erdem. Visual saliency estimation by nonlinearly integrating features using region covariances. *Journal of Vision*, 13(4):1–20, 2013.

[18] Dumitru Erhan, Christian Szegedy, Alexander Toshev, and Dragomir Anguelov. Scalable object detection using deep neural networks. In *CVPR*, 2014.

[19] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *IJCV*, 88(2):303–338, June 2010.

[20] Mark Everingham, Josef Sivic, and Andrew Zisserman. Hello! my name is... buffy–automatic naming of characters in tv video. 2006.

[21] Robert Fergus, Li Fei-Fei, Pietro Perona, and Andrew Zisserman. Learning object categories from google's image search. In *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, pages 1816–1823. IEEE, 2005.

[22] Kunihiko Fukushima. Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biological Cybernetics*, 36(4):193–202, 1980.

[23] Raja Giryes, Guillermo Sapiro, and Alexander M. Bronstein. Deep neural networks with random gaussian weights: A universal classification strategy? *CoRR*, abs/1504.08291, 2015.

[24] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 315–323, 2011.

[25] Alex Graves, Abdel-rahman Mohamed, and Geoffrey E. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, pages 6645–6649, 2013.

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

[27] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *arXiv preprint arXiv:1502.01852*, 2015.

[28] Geoffrey E. Hinton, Simon Osindero, and Yee Whye Teh. A fast learning algorithm for deep belief nets. *Neural Computation*, 18(7):1527–1554, 2006.

[29] Geoffrey E Hinton and Ruslan R Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 313(5786):504–507, 2006.

[30] Kurt Hornik, Maxwell B. Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural Networks*, 2(5):359–366, 1989.

[31] Jiayuan Huang, Alexander J. Smola, Arthur Gretton, Karsten M. Borgwardt, and Bernhard Schölkopf. Correcting sample selection bias by unlabeled data. In *NIPS*, 2006.

[32] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106, 1962.

[33] Yangqing Jia, Evan Shelhamer, Jeff Donahue, Sergey Karayev, Jonathan Long, Ross Girshick, Sergio Guadarrama, and Trevor Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[34] Aditya Khosla, Nityananda Jayadevaprakash, Bangpeng Yao, and Li Fei-Fei. Novel dataset for fine-grained image categorization. In *1st Workshop on FGVC, CVPR*, June 2011.

[35] Teuvo Kohonen. *Self-organizing maps*. Springer, 1997.

[36] Jan Krause, Michael Stark, Jia Deng, and Li Fei-Fei. 3d object representations for fine-grained categorization. In *ICCVW*, pages 554–561. IEEE, 2013.

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

[38] Roland Kwitt, Nuno Vasconcelos, and Nikhil Rasiwasia. Scene recognition on the semantic manifold. *ECCV*, 2012.

[39] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 2169–2178. IEEE, 2006.

[40] Quoc V. Le, Marc'Aurelio Ranzato, Rajat Monga, Matthieu Devin, Greg Corrado, Kai Chen, Jeffrey Dean, and Andrew Y. Ng. Building high-level features using large scale unsupervised learning. In *ICML*, 2012.

[41] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, Nov 1998.

[42] Li-Jia Li and Li Fei-Fei. Optimol: automatic online picture collection via incremental model learning. *International journal of computer vision*, 88(2):147–168, 2010.

[43] Quannan Li, Jiajun Wu, and Zhuowen Tu. Harvesting mid-level visual concepts from large-scale internet images. *CVPR*, 2013.

[44] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, 2014.

[45] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.

[46] Tomas Mikolov, Ilya Sutskever, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, pages 3111–3119, 2013.

[47] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Alex Graves, Ioannis Antonoglou, Daan Wierstra, and Martin A. Riedmiller. Playing atari with deep reinforcement learning. *CoRR*, abs/1312.5602, 2013.

[48] Guido F. Montúfar, Razvan Pascanu, KyungHyun Cho, and Yoshua Bengio. On the number of linear regions of deep neural networks. In *Advances in Neural Information Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada*, pages 2924–2932, 2014.

[49] Alberto Muñoz and Jorge Muruzábal. Self-organizing maps for outlier detection. *Neurocomputing*, 18(1):33–60, 1998.

[50] Joe Yue-Hei Ng, Fan Yang, and Larry S. Davis. Exploiting local features from deep networks for image retrieval. *CoRR*, abs/1504.05133, 2015.

[51] M-E. Nilsback and A. Zisserman. Automated flower classification over a large number of classes. In *Proceedings of the Indian Conference on Computer Vision, Graphics and Image Processing*, Dec 2008.

[52] Timo Ojala, Matti Pietikäinen, and Topi Mäenpää. Gray scale and rotation invariant texture classification with local binary patterns. In *ECCV*. 2000.

[53] Timo Ojala, Matti Pietikainen, and Topi Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 24(7):971–987, 2002.

[54] M. Ozcan, L. Jie, V. Ferrari, and B. Caputo. A Large-Scale Database of Images and Captions for Automatic Face Naming. In *British Machine Vision Conference (BMVC)*, 2011.

[55] Mert Özcan, Jie Luo, Vittorio Ferrari, and Barbara Caputo. A large-scale database of images and captions for automatic face naming. In *BMVC*, pages 1–11, 2011.

[56] Megha Pandey and Svetlana Lazebnik. Scene recognition and weakly supervised object localization with deformable part-based models. *ICCV*, 2011.

[57] Nicolas Pinto, Zak Stone, Todd Zickler, and David Cox. Scaling up biologically-inspired computer vision: A case study in unconstrained face recognition on facebook. In *CVPR Workshops*, 2011.

[58] Christopher Poultney, Sumit Chopra, Yann L Cun, et al. Efficient learning of sparse representations with an energy-based model. In *NIPS*, pages 1137–1144, 2006.

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

[60] Ariadna Quattoni and Antonio Torralba. Recognizing indoor scenes. In *CVPR*, pages 413–420, 2009.

[61] Antti Rasmus, Harri Valpola, Mikko Honkala, Mathias Berglund, and Tapani Raiko. Semi-supervised learning with ladder network. *CoRR*, abs/1507.02672, 2015.

[62] Shaoqing Ren, Xudong Cao, Yichen Wei, and Jian Sun. Face alignment at 3000 fps by regressing local binary features. In *CVPR*, 2014.

[63] 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*, pages 1–42, April 2015.

[64] Olga Russakovsky and Li Fei-Fei. Attribute learning in large-scale datasets. In *Trends and Topics in Computer Vision*, pages 1–14. Springer, 2012.

[65] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*.

[66] Vikas Sindhwani and S. Sathiya Keerthi. Large scale semi-supervised linear svms. In *SIGIR*, 2006.

[67] Saurabh Singh, Abhinav Gupta, and Alexei A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*. 2012.

[68] Saurabh Singh, Abhinav Gupta, and Alexei A. Efros. Unsupervised discovery of mid-level discriminative patches. In *ECCV*, 2012.

[69] Sainbayar Sukhbaatar and Rob Fergus. Learning from noisy labels with deep neural networks. *arXiv*, 2014.

[70] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, 2014.

[71] Y. Taigman, Ming Yang, M. Ranzato, and L. Wolf. Deepface: Closing the gap to human-level performance in face verification. In *CVPR*, 2014.

[72] Adrian Ulges, Marcel Worring, and Thomas M. Breuel. Learning visual contexts for image annotation from flickr groups. *IEEE Transactions on Multimedia*, 2011.

[73] Harri Valpola. From neural PCA to deep unsupervised learning. *CoRR*, abs/1411.7783, 2014.

[74] Joost Van De Weijer, Cordelia Schmid, Jakob Verbeek, and Diane Larlus. Learning color names for real-world applications. *Image Processing, IEEE*, 2009.

[75] Laurens Van der Maaten and Geoffrey Hinton. Visualizing data using t-sne. *JMLR*, 9(2579-2605):85, 2008.

[76] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. In *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, pages I–511. IEEE, 2001.

[77] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie. The Caltech-UCSD Birds-200-2011 Dataset. Technical report, 2011.

[78] Gang Wang, Derek Hoiem, and David A. Forsyth. Learning image similarity from flickr groups using fast kernel machines. *IEEE TPAMI*, 34(11):2177–2188, 2012.

[79] Jiang Wang, Yang Song, Thomas Leung, Chuck Rosenberg, Jingbin Wang, James Philbin, Bo Chen, and Ying Wu. Learning fine-grained image similarity with deep ranking. In *CVPR*, 2014.

[80] L. Wolf, T. Hassner, and Y. Taigman. Descriptor based methods in the wild. In *ECCV Workshop on Faces in Real-Life Images*, 2008.

[81] Jianxiong Xiao, James Hays, Krista A. Ehinger, Aude Oliva, and Antonio Torralba. SUN database: Large-scale scene recognition from abbey to zoo. In *CVPR*, pages 3485–3492, 2010.

[82] Tong Xiao, Tian Xia, Yi Yang, Chang Huang, and Xiaogang Wang. Learning from massive noisy labeled data for image classification. In *CVPR*, pages 2691–2699, 2015.

[83] Saining Xie, Tianbao Yang, Xiaoyu Wang, and Yuanqing Lin. Hyper-class augmented and regularized deep learning for fine-grained image classification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 2645–2654, 2015.

[84] Linjie Yang, Ping Luo, Chen Change Loy, and Xiaoou Tang. A large-scale car dataset for fine-grained categorization and verification. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3973–3981, 2015.

[85] Bangpeng Yao, Xiaoye Jiang, Aditya Khosla, Andy Lai Lin, Leonidas J. Guibas, and Fei-Fei Li. Human action recognition by learning bases of action attributes and parts. In *ICCV*, pages 1331–1338, 2011.

[86] Bolei Zhou, Àgata Lapedriza, Jianxiong Xiao, Antonio Torralba, and Aude Oliva. Learning deep features for scene recognition using places database. In *Advances in Neural Information*

Processing Systems 27: Annual Conference on Neural Information Processing Systems 2014, December 8-13 2014, Montreal, Quebec, Canada, pages 487–495, 2014.

[87] Xiangxin Zhu and Deva Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 2879–2886. IEEE, 2012.