Unbiased Metric Learning: On the Utilization of Multiple Datasets and Web Images for Softening Bias

Chen Fang Ye Xu Daniel N. Rockmore
Computer Science Department
Dartmouth College
Hanover, NH 03755, U.S.A.
{chenfang, ye, rockmore}@cs.dartmouth.edu

Abstract

Many standard computer vision datasets exhibit biases due to a variety of sources including illumination condition, imaging system, and preference of dataset collectors. Biases like these can have downstream effects in the use of vision datasets in the construction of generalizable techniques, especially for the goal of the creation of a classification system capable of generalizing to unseen and novel datasets. In this work we propose Unbiased Metric Learning (UML), a metric learning approach, to achieve this goal. UML operates in the following two steps: (1) By varying hyperparameters, it learns a set of less biased candidate distance metrics on training examples from multiple biased datasets. The key idea is to learn a neighborhood for each example, which consists of not only examples of the same category from the same dataset, but those from other datasets. The learning framework is based on structural SVM. (2) We do model validation on a set of weakly-labeled web images retrieved by issuing class labels as keywords to search engine. The metric with best validation performance is selected. Although the web images sometimes have noisy labels, they often tend to be less biased, which makes them suitable for the validation set in our task. Cross-dataset image classification experiments are carried out. Results show significant performance improvement on four well-known computer vision datasets.

1. Introduction

Over the last decades object recognition systems have been improved dramatically [26][19][6]. One of the forces driving the development is the availability of medium or large size high quality image datasets (e.g., Caltech 101 [7], PASCAL VOC [4], LabelMe [20] and SUN09 [3]) that enable researchers to evaluate and practice sophisticated feature designing and machine learning techniques. However, Torralba and Efros [23] point out that every dataset carries bias in its own way, which can be caused by various reasons, such as illumination condition, different imaging system and preference of database collectors. Bias inevitably leads learning algorithms to overfit for the training set to the detriment of any ability to generalize to other datasets. In other words, an object recognition system trained solely on one dataset tends to perform poorly on unseen and novel datasets at test time, because the underlying bias is incorporated into the learning algorithm. This is an important issue, because most image classification systems are required to handle all kinds of test examples, regardless of where the test examples are drawn from. For example, it's rare to see a question like "Can you classify a dog image from Caltech101?".

The new challenge now is to build a system that performs well on unseen datasets. This requires the learned model to capture the general class knowledge, while discarding the information reflective of the bias. Note that this problem is potentially more general than a challenge facing researchers performing image analysis. Microarray analysis is one context in which this issue has received significant attention (see e.g., [1]).

Failing the ability to create a vision dataset free of bias, it is of interest to address the assumption of bias directly. To this end we introduce *Unbiased Metric Learning (UML)*, which learns an unbiased metric using multiple biased datasets and web images. Our approach operates in the following two steps:

Step 1 – we learn a set of distance metrics on training examples from multiple biased datasets. The key idea is that in the learned feature spaces, the neighborhoods of training examples should consist of not only examples of the same category from the same dataset, but those examples of the same category from *other datasets*. We call this property "neighborhood diversity." In such a

way, datasets (or domains) are bridged together via these neighborhoods. By varying related hyperparameters, we learn a set of distance metrics, each of which bears a specific degree of "neighborhood diversity." In other words, the training data is distributed differently in the spaces defined by these metrics. We form a candidate set of these metrics for Step 2 (below), in which a novel validation is performed. Technically, we cast it as a learning to rank problem [11][27], and solve it with structural metric learning [15].

Step 2 – we do model validation to identify the metric with best generalization ability. Conventionally, the validation set is from the same source as the training set, for example, cross-validation. However, in our case, a good validation performance on seen datasets does not necessarily generalize to unseen datasets. Therefore conventional validation may fail to uncover the desired model. So, instead of using images from seen datasets, we propose to use a set of weakly-labeled web images retrieved from the Internet by issuing class labels as keywords to the search engine. Those images are less biased and have higher intra-class variability than human collected datasets, which makes it suitable to be used as our validation set.

We do image classification experiments, and use cross-dataset performance to measure a model's ability to generalize to unseen datasets. Experiments show the following facts: (1) In Step 1, by varying hyperparameters, our learning framework is capable of producing models with superior cross-dataset classification performance. (2) In the validation step, the model selected by our novel validation method significantly outperforms those selected by conventional validation procedures.

In the rest of the paper, we will first review related work, discuss the advantages of our framework, then cover technical details. We then follow this with experiments demonstrating the effectiveness of our approach.

2. Related Work

The issue of dataset bias in vision datasets was first raised by Torralba and Efros [23]. Since then, Khosla *et al*. [12] has proposed a solution to improve cross-dataset generalization ability for an object recognition system. The approach in [12] learns a common weight vector which is expected to work well on unseen datasets, in a way that is similar to regularized multi-task learning [5]. Our method adopts a metric learning solution to this problem.

Metric learning methods have been popular in the machine learning community as well as computer vision [25][18]. Most of them can be categorized as either learning a "global" metric or a "local" metric. Our approach is closer to the former category, where a single parametric transformation is learned to map data from the original space to a new space, where the data distribution exhibits some desired properties. Weinberger *et al*. [24] propose a large margin method to group together examples with the same label and separate the ones with different labels. In [9] a metric is learned by collapsing all examples in a class to a single point. However, these methods are not designed to utilize domain information. As shown in later experiments, this results in poor cross-dataset generalization performance.

Because our approach utilizes domain information in training data, it is related to domain adaptation and multi-task learning. In domain adaptation, the goal is to transfer knowledge from the source domain to help perform a task in the target domain, and multi-task learning aims at good performance simultaneously in multiple domains. Among the large range of work in this area, [21] and [17] are both metric learning-based. [21] learns a metric to transfer knowledge to the target domain by randomly sampling pairs consisting of a labeled example from the source domain and another from the target domain, and constraining their distance to be no greater (less) than a bound if the labels are the same (different). In [17], the metric is decomposed into a domain specific part and a global part, which is shared among all domains. Recent work by Tommasi *et al*. [22] proposes to learn general knowledge from multiple datasets, which will be transferred to a target test dataset later with the help of a few labeled examples from the target dataset. Although related, our problem is different. In our problem, there is no target dataset where the task will be performed, not to mention labeled examples to assist in knowledge transfer. This means that incoming test examples may come from any domains, although most are unseen at training time. Therefore, the goal is to learn a transformed new space, where all the training examples are less biased. So any classification system trained on it will generalize better to novel datasets.

3. Approach Overview

Our approach tackles the problem from a learning-to-rank perspective. First, we consider each training example as a query and rank in ascending order the other training examples based on their L^2 distances to the query. We want a high precision among top-k positions. This is essentially constructing a label-coherent neighborhood. Then, beyond the neighborhood label coherence, we look at the domain information of the positive examples within the neighborhood. We encourage to include more positive examples from multiple domains, which was mentioned before as "neighborhood diversity." From a ranking perspective, this is inserting into the top-k positions, those positive examples from domains other than the current query's. Fig.1 gives a schematic comparison of the case in which "neighborhood diversity" is enforced and the case in which it is

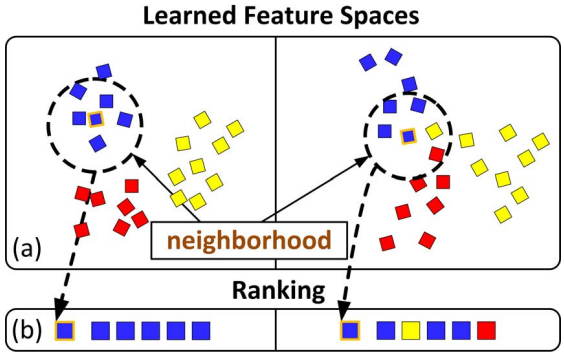

Figure 1. Schematic illustration of neighborhoods without (top-left) and with (top-right) "neighborhood diversity" (best viewed in color). (a) The distribution of data in different learned feature spaces (Left: a normal feature space. Right: a feature space learned by UML). Color and shape represent dataset and category, respectively. Squares with colored boundaries are the corresponding queries of the two local neighborhoods in circle. In the top right figure, data from different datasets are linked at circled neighborhoods. (b) A look at the local neighbors in Fig.1a from a ranking perspective. The ranking on the right, unlike the one on the left, exhibits both label coherence and neighborhood diversity.

not. As the top right figure illustrates, examples of the same category from different datasets are linked together via diversified neighborhood, which is constructed automatically in the learning procedure by discovering and grouping appropriate pairs of positive examples from different datasets. However, we still let each dataset keep its own distributional independence, since we do not want to overly "merge" the data. We let the hyperparameters vary, so as to produce a set of metrics with different levels of "neighborhood diversity" which later will be screened by validation procedure.

As for model validation, we use weakly-labeled web images as the validation set. Images returned by search engines are sometimes loosely related to textual query (keyword), compared to human-labeled datasets. Therefore, if used as training set, in order to get comparable results, extra labor is needed to remove noise and build a robust learning algorithm [8][10]. However, studies have shown that they serve well as a relatively noisy validation set [8]. In our case, we are interested in the fact that web images tend to be less biased, since they are implicitly sampled from a countless number of unknown sources and there is less human intervention during collection process. Our experiments support the claim that web images serve better as a validation set for our problem.

4. Technical Details

In this section, we will discuss the formulation of the structural metric learning (SML) framework as well as *Unbiased Metric Learning (UML)*. Technical details will be covered.

4.1. Notations and preliminaries

Let $\mathcal{D} = \{D_1, \ldots, D_Q\}$ denote the set of biased datasets that share a set of common classes $\mathcal{C} = \{c_1, \ldots, c_K\}$. The training set is denoted as $\mathcal{X}$ with $|\mathcal{X}| = n$. Each training example $i \in \mathcal{X}$ is a triplet (x_i, l_i, d_i), where $x_i \in \mathbb{R}^d$ is the feature vector of item i, $l_i \in \mathcal{C}$ is the corresponding class label and $d_i \in \mathcal{D}$ indicates the dataset that i is from. For a query q, we use $\mathcal{X}_q^+/\mathcal{X}_q^-$ to denote the set of positive/negative examples in $\mathcal{X}$. $\mathcal{Y}$ will be the set of permutations/rankings of items in $\mathcal{X}$. Similarly, $\mathcal{Y}_q^+$ is the set of permutations/rankings of items in $\mathcal{X}_q^+$. If i is ranked before (after) j in some ranking $y \in \mathcal{Y}$, we say $i \prec_y j$ ($i \succ_y j$).

For a matrix $W \subset \mathbb{R}^{d\times d}$, $W \succeq 0$ means it is a symmetric and positive semidefinite matrix. The Mahalanobis distance defined by W is $d_{\mathbf{W}}(i,j) = \sqrt{(x_i - x_j)^\mathsf{T}\mathbf{W}(x_i - x_j)}$. This is equivalent to applying a transformation $L \subset \mathbb{R}^{d'\times d}$ to the original feature space and calculating the L^2 distance in the new space. Thus, $W = L^\mathsf{T} \times L$. The Frobenius inner product of two matrices $A, B \in \mathbb{R}^{d\times d}$ is denoted as $\langle A, B\rangle_F = tr(A^\mathsf{T}B)$. Finally, $\mathbf{1}(\mathbf{X})$ is the indicator function of event $\mathbf{X}$.

4.2. Structural metric learning

We now review the structural metric learning (SML) framework [15]. The goal is to learn a positive semidefinite matrix W, so that when a query q is issued, the corresponding ranking or ordering y_q, which is produced based on the Mahalanobis distance defined by W, will have some desiring properties, such as high Precision@k, Mean Average Precision or ROC area. This can be solved via structural learning and its mathematical formulation is similar to structural SVM [11]. The following objective function is minimized:

$$\min_{W\succeq 0, \xi\geq 0} \; tr\,(W) + \frac{C}{n}\sum_{q\in\mathcal{X}} \xi_q \tag{1}$$

subject to constraints as follows:

$$\forall q \in \mathcal{X}, \forall y \in \mathcal{Y}\backslash y_q^* : \tag{2}$$

$$\langle W, \psi_{po}(q, y_q^*) - \psi_{po}(q, y)\rangle_F \geq \Delta(y, y_q^*) - \xi_q.$$

in Eq.1 $tr\,(W)$ is the regularizer. In Eq.2, y_q^* is the ground truth ranking for query q. The right-hand side of Eq.2 includes the slack variable ξ_q and the structural loss function $\Delta(y, y_q^*)$, which encodes the structural information in $\mathcal{Y}$. Thus, the margin is rescaled to be $\Delta(y, y_q^*)$. The Frobenius inner product term on the left is the discriminative score difference between current ranking y and y_q^*.

$\psi(q, y)$ is the partial order feature [11] with the following form:

$$\psi_{po}(q, y) = \sum_{i \in \mathcal{X}_q^+} \sum_{j \in \mathcal{X}_q^-} y_{ij} \left(\frac{\phi(q, i) - \phi(q, j)}{|\mathcal{X}_q^+| \cdot |\mathcal{X}_q^-|} \right) \quad (3)$$

where

$$y_{ij} = \begin{cases} +1 & i \prec_y j \\ -1 & i \succ_y j \end{cases} \quad (4)$$

$\phi(q, i)$ is a feature map that captures how q and i are related. Thus, a ranking y for query q is encoded by $\psi_{po}(q, y)$ in a feature space by examining all possible relevant/irrelevant pairs. At prediction time, given a fixed W, the ranking y that maximizes the discriminative score $\langle W, \psi_{po}(q, y) \rangle_F$ is simply the collection of $i \in \mathcal{X}$ sorted by descending $\langle W, \phi(q, i) \rangle_F$. Now choose ϕ to be $\phi(q, i) = -(x_q - x_i)(x_q - x_i)^\mathsf{T}$. Because $d^2_{\mathbf{W}}(q, i) = \langle W, (x_q - x_i)(x_q - x_i)^\mathsf{T} \rangle_F$, we see that the y that maximizes $\langle W, \psi_{po}(q, y) \rangle_F$ is simply $i \in \mathcal{X}$ sorted by ascending $d^2_{\mathbf{W}}(q, i)$.

4.3. Unbiased Metric Learning

Learning model Our model extends the original SML. First, note that the neighborhood label coherence is already forced by Eq.2, if we set the loss function Δ to be the following form:

$$\Delta(y, y_q^*) = 1 - Prec@k(y) \quad (5)$$

where $Prec@k(y)$ gives the percentage of positive examples among the top-k positions of ranking y, which is equivalent to the notion of label coherence in a neighborhood. To incorporate the "neighborhood diversity" property, the following constraint is added to the SML formulation:

$$\forall q \in \mathcal{X}, \forall y^+ \in \mathcal{Y}^+ \backslash y_q^{+*} : \quad (6)$$

$$\langle W, \psi_{po}(q, y_q^{+*}) - \psi_{po}(q, y^+) \rangle_F \geq \hat{\Delta}(y^+, y_q^{+*}) - \xi_q^+$$

Assume the current query is $q = (x_q, l_q, d_q)$. In Eq.6, $y^+ \in \mathcal{Y}_q^+$ is a subset ranking of only $i \in \mathcal{X}_q^+$. We let y_q^{+*} be the ground truth ranking in $\mathcal{Y}_q^+$ with the following property:

$$\forall i, j \in \mathcal{X}_q^+ : i \prec_{y_q^{+*}} j, \ \textbf{if} \ (d_i \neq d_q \wedge d_j = d_q) \quad (7)$$

where d_i indicates the dataset containing example i. Eq.7 means the all the positive examples not in d_q precede those in d_q. We also have a new objective function by adding the slack variable ξ_q^+ to Eq.1:

$$\min_{W \succeq 0, \xi \geq 0} \ tr(W) + \frac{C_1}{n} \sum_{q \in \mathcal{X}} \xi_q + \frac{C_2}{n} \sum_{q \in \mathcal{X}} \xi_q^+ \quad (8)$$

Now let us tentatively assume that $\hat{\Delta}(y^+, y_q^{+*})$ measures the "neighborhood diversity" score difference between y^+ and the ground truth. Then Eq.6 encourages any ranking predicted by W to have a neighborhood as diverse as the ground truth.

To measure "neighborhood diversity" we introduce a new function:

$$Div(q, y^+, k') = \frac{1}{k'} \sum_{i \in y^+, i=1}^{k'} \mathbf{1}(d_i \neq d_q) \quad (9)$$

which is the percentage of items not in d_q among the top-k' positions of y^+. Based on Eq.9, the following diversity score function is devised:

$$DivS(q, y^+, k') = \frac{1}{1 + e^{-Div(q, y^+, k')/\eta}} \quad (10)$$

where η is, mathematically, a hyperparameter controlling the shape of this sigmoid-like function. Eq.10 is needed because setting the ground truth y_q^{+*} to have the property in Eq.7 may dangerously overconnect or overmerge datasets. The degree to which we can connect the data is data dependent. If the data resists merging, we need to set η to be a small value, so that even a tiny increase of Eq.9 leads to a gigantic boost in Eq.10., resulting in a score very close to ground truth. At a higher level, η is a hyperparameter that we should vary and one that depends on the intrinsic property of training data. Finally, a natural choice for $\hat{\Delta}$ is the score difference:

$$\hat{\Delta}(y^+, y_q^{+*}) = DivS(q, y_q^{+*}, k') - DivS(q, y_q^+, k'). \quad (11)$$

Optimization When solving for the optimal W in Eq.8, we can not enumerate the entire $\mathcal{Y}$ and $\mathcal{Y}^+$ to list all the constraints. To optimize for UML, we use an efficient cutting-plane algorithm, which is slightly different from [15][11] due to the additional constraint of Eq.6. The general idea is that for each training example, maintain one set $\mathcal{S}_i$ of active constraints corresponding to Eq.2, and another set $\mathcal{S}_i^+$ of active constraints corresponding to Eq.6. The algorithm alternates between solving for W under current active constraints, and updating $\mathcal{S}_i$ and $\mathcal{S}_i^+$ by adding to them the most violated constraint $\hat{y}_i$ and $\hat{y}_i^+$ of each training example under current W. If a newly found $\hat{y}_i$ ($\hat{y}_i^+$) has a violation, which is the value of its slack variable indeed, greater than current violation ξ_i (ξ_i^+) by some threshold ϵ, then it will be added, otherwise discarded. This iteration keeps going until no new constraint needs to be added. Gradient descent is used to solve for W, and the solution at each gradient step will be projected to its positive semidefinite (PSD) cone so that W is a feasible metric. Algorithm 1 is a high level illustration of the optimization procedure for UML.

Notice that in order to further speed up the optimization process, two modification are made in our implementation.

First, as in [15][11] the problem is reformulated so that only one or two global slack variables are maintained. Second, as in [13], we use the alternating direction method of multipliers [2] to reduce the number of times eigen-decomposition is performed, which is done when projecting the solution to its PSD cone.

Algorithm 1 Optimization Algorithm

Input: training data $\mathcal{X}$, positive/negative set $\mathcal{X}_i^+/\mathcal{X}_i^-$, ranking y_i^* and y_i^{+*}, hyperparamters: $C_1 > 0$, $C_2 > 0$ and $\eta > 0$, stop threshold $\epsilon > 0$

Output: $W \succeq 0$ and slack variables ξ_i and ξ_i^+

for all $i = 1, \ldots, n$, $\mathcal{S}_i \leftarrow \emptyset, \mathcal{S}_i^+ \leftarrow \emptyset, \xi_i \leftarrow 0, \xi_i^+ \leftarrow 0$
repeat
for $i = 1, \ldots, n$ **do**
  $\hat{y}_i \leftarrow argmax_{y \in \mathcal{Y}} \langle W, \psi_{po}(i, y) \rangle_F + \Delta(y, y_i^*)$
  $\hat{y}_i^+ \leftarrow argmax_{y^+ \in \mathcal{Y}^+} \langle W, \psi_{po}(i, y^+) \rangle_F + \hat{\Delta}(y^+, y_i^{+*})$
  if the slack of $\hat{y}_i$ is greater than $\xi_i + \epsilon$ **then**
    $\mathcal{S}_i \leftarrow \mathcal{S}_i \cup \{\hat{y}_i\}$
  end if
  if the slack of $\hat{y}_i^+$ is greater than $\xi_i^+ + \epsilon$ **then**
    $\mathcal{S}_i^+ \leftarrow \mathcal{S}_i^+ \cup \{\hat{y}_i^+\}$
  end if
  Solve for Eq.8
    subject to $\mathcal{S}_i$ and $\mathcal{S}_i^+$ for all $i = 1, \ldots, n$
end for
until no $\mathcal{S}_i$ and $\mathcal{S}_i^+$ has changed during iteration

5. Experiments

In this section, we evaluate our approach on the image classification task.

5.1. Data

We used four standard datasets: Caltech101 [7], PASCAL VOC [4], LabelMe [20] and SUN09 [3]. The following five common categories were selected: bird, car, chair, dog and person.

Training set The training set contains images from the four datasets. For each category in each dataset, we randomly selected up to 100 images, so that we could vary the number of training examples per category per dataset. In cross-dataset classification experiments one dataset will be held out as unseen dataset. Therefore, we remove its images from the pool of training images to form a subset (denoted as $\mathbf{Tr}^{\mathbf{unseen}}$) and the rest of the images form another subset called $\mathbf{Tr}^{\mathbf{seen}}$. In practice, only $\mathbf{Tr}^{\mathbf{seen}}$ was used

Validation set This validation set contains images from the four datasets. The 20 images per category per dataset are randomly selected. Similar to the training set, in cross-dataset experiments one dataset will be held out, so we have two subsets as $\mathbf{Va}^{\mathbf{unseen}}$ and $\mathbf{Va}^{\mathbf{seen}}$.

Web validation set In order to carry out our novel validation method, we constructed another validation set with web images. We issued the class labels as keyword to Google and downloaded the top 50 returned images. After removing duplicates with regard to the training set, 20 images were randomly selected for each class to form the set. We will refer to it as $\mathbf{Va}^{\mathbf{web}}$.

Test set There are 20 images per class per dataset. Similar to training set, the test set was divided into $\mathbf{Te}^{\mathbf{unseen}}$ and $\mathbf{Te}^{\mathbf{seen}}$ in cross-dataset evaluation.

We generated 3 different copies of the above sets, so that all of our following experiments are carried out 3 times on different data and the mean results are reported. For each image, grayscale SIFT descriptors [14] were extracted at interest points detected with the Hessian-Affine detector [16]. Then we quantized these descriptors to bag-of-words representation using a vocabulary of size 500 at 3 spatial pyramid levels and the feature representation is of $10,500$ dimensions. Eventually, PCA was applied to reduce the dimensionality to 800.

We started with an experiment to validate the existence of bias in our datasets. Then we applied our method to cross-dataset classification task. Finally, we compare our approach with other metric learning methods. A simple k-nearest neighbor classifier is used in all classification experiments. Details and results are shown in the following sections.

5.2. Existence of bias

In this first experiment we show that our four datasets are strongly biased, in the sense that a model trained on one dataset is ineffective (due to bias) on the other datasets. We do this by using SML to learn a metric on each dataset individually and then test it on each dataset individually. For each metric, the classification accuracy on each dataset is reported. We used 20 images per category per dataset to form the training set, as well as the validation and test sets. The hyperparameter C in Eq.1 was selected from the following values $\{0.001, 0.01, 0.1, 1, 10, 100, 1000\}$.

Train	Test			
	Cal	Pas	SUN	Lab
Cal	**0.87**	0.33	0.24	0.39
Pas	0.31	**0.40**	0.32	0.30
SUN	0.11	0.23	**0.37**	0.22
Lab	0.24	0.25	0.18	**0.47**

Table 1. Classification accuracy on all datasets. Metrics are learned on each dataset individually. The left-most column specifies the training dataset where the metric is learned, while the uppermost row specifies the test dataset. Cal, Pas, SUN and Lab stand for Caltech101, PASCAL VOC, SUN09 and LabelMe respectively.

Table 1 shows clearly that our datasets are highly biased. With the same test example, the performance of models trained on different datasets varies widely (read Table 1 vertically). For example, when tested on Caltech101 (see column 1), the best classification performance is achieved by the model trained on the same dataset, while the lowest accuracy comes from the model trained on SUN09 images.

5.3. Cross-dataset classification

In this experiment, our goal is to measure the true generalizability of the model learned by UML with cross-dataset performance. We should point out that cross-dataset classification does not completely measure a model's true generalizability, but it is a reasonable indicator.

Since the performance of UML is subject to model training as well as validation, our experiment is composed of two stages, where we evaluate them separately. Finally, we will do full evaluation and compare UML to other metric learning methods.

5.3.1 Training stage

In this experiment, we want to test the effectiveness of the UML learning framework, so the question for this stage is whether UML can produce models with better generalizability. Recall that by varying hyperparameters we will learn a set of metrics with different levels of "neighborhood label coherence" and "neighborhood diversity". These metrics will be validated by the validation procedure. However, if the validation is carried out improperly (e.g., using a validation set that is not related to the target task) then a bad metric can be selected. For example, in our case, it would be bad to use $\mathbf{Va}^{\mathbf{seen}}$, since the goal is to generalize to every possible unseen datasets/domains, not only the three seen datasets. The corresponding consequence is that it is impossible to tell the quality of models produced by UML training step. Clearly, in this stage, we need a validation set that is closely related to the target task, which is, in the cross-dataset setting, generalizing to the unseen dataset. The choice is obvious: using $\mathbf{Va}^{\mathbf{unseen}}$ as the validation set at this stage. In this way, the effect of the validation procedure on the final performance is minimized, thus resulting in a better evaluation of UML's learning step.

In detail, when a dataset was marked as unseen, the corresponding $\mathbf{Tr}^{\mathbf{seen}}$ and $\mathbf{Te}^{\mathbf{unseen}}$ denote the training set and test set respectively. There were 20 images per class per dataset in $\mathbf{Va}^{\mathbf{unseen}}$ and we let the number of training examples per class per dataset in $\mathbf{Tr}^{\mathbf{seen}}$ to be the following values $\{15, 20, 30, 50, 70, 100\}$. The validation set, as explained above, was $\mathbf{Va}^{\mathbf{unseen}}$, which was of the same size as $\mathbf{Te}^{\mathbf{unseen}}$. The k in Eq.5 and the k' in Eq.11 was set to be the same as current number of training examples. The neighborhood size of the k-nearest neighbor classifier was determined via validation. We let each dataset be unseen once, and report in Fig.2 the test accuracy on the corresponding $\mathbf{Te}^{\mathbf{unseen}}$, as well as the average performance of these individual experiments. From the top figure in Fig.2 we can tell that the models produced by UML have better cross-dataset generalizability than SML. This suggest that utilizing domain information (such as the source of the example dataset), in learning can be helpful to extract more general knowledge. The bottom figures report the results of individual experiments. As shown, UML still gives better results most times. One observation we found is that UML has fewer advantages over baseline when fewer (e.g., 15) training examples are used, but benefits more from more training examples. This is because UML can not enforce "neighborhood diversity" due to the lack of good pairs to pull together, and with more examples at hand, better pairs can be found and this in turn give rise to "better" neighborhoods can be constructed. There are certain points at which both methods do not benefit from more training examples. This is due to the fact that newly added examples can bring in more bias, which will then further bias the derived learning algorithm.

5.3.2 Validation stage

The goal of the second stage is to evaluate how different validation sets affect the final performance. Two validation sets were used, $\mathbf{Va}^{\mathbf{web}}$ and the corresponding $\mathbf{Va}^{\mathbf{seen}}$ in individual experiments. The set of metrics produced by UML in the training stage was the input, thus with the same input, our expectation was that our validation procedure with $\mathbf{Va}^{\mathbf{web}}$ would consistently outperform the other. This is proved by results reported in Fig.3, in which metrics selected by $\mathbf{Va}^{\mathbf{web}}$ not only outperform metrics selected by $\mathbf{Va}^{\mathbf{seen}}$, but almost matches the ones selected by $\mathbf{Va}^{\mathbf{unseen}}$.

An additional question we ask is whether web images can help conventional metric learning method to generalize. To answer it, we applied $\mathbf{Va}^{\mathbf{seen}}$ and $\mathbf{Va}^{\mathbf{web}}$ to metrics produced by SML, and found that metrics selected by $\mathbf{Va}^{\mathbf{web}}$ outperform those of $\mathbf{Va}^{\mathbf{seen}}$. Due to the lack of space, we leave the corresponding figure in the supplementary material.

5.3.3 Full evaluation

In this experiment, we compared our approach, UML, to the following baseline methods: Structural Metric Learning (SML) [15], Large Margin Nearest Neighbor (LMNN) [24], Maximally Collapsing Metric Learning (MCML) [9] and Large Margin Multi-Task Metric Learning (mtLMNN) [17]. In mtLMNN, a task was to classify on one of seen training sets, and the learned shared metric, which encodes the knowledge shared among different tasks, was applied to unseen test set. We vary hyperparameters for all baseline

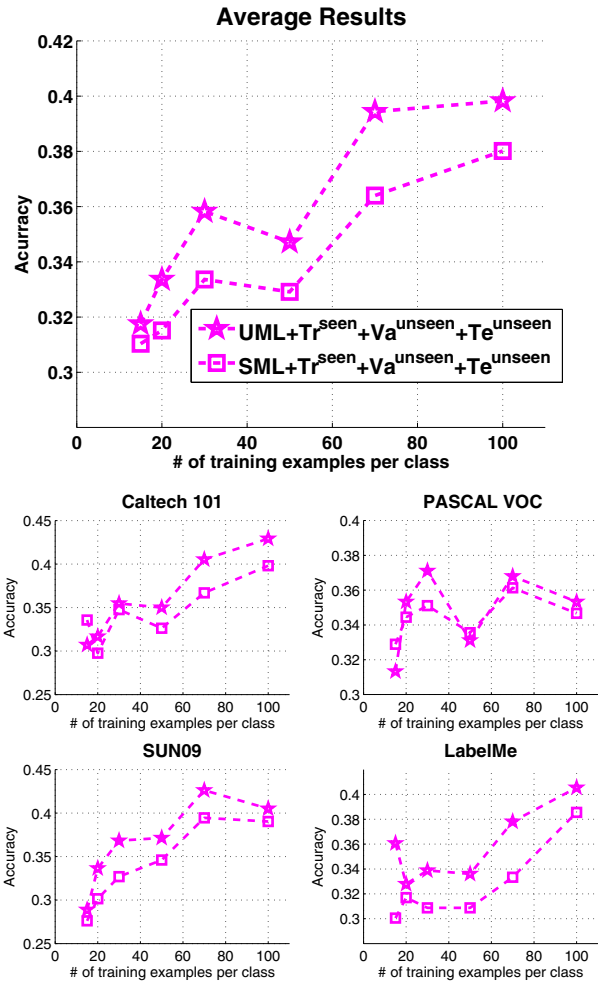

Figure 2. Cross-dataset classification accuracy on $\mathbf{Te}^{\mathbf{unseen}}$ with validation on $\mathbf{Va}^{\mathbf{unseen}}$. Top: the figure reports the average accuracy over four individual experiments in the bottom. Bottom: individual classification accuracy on hold-out dataset.

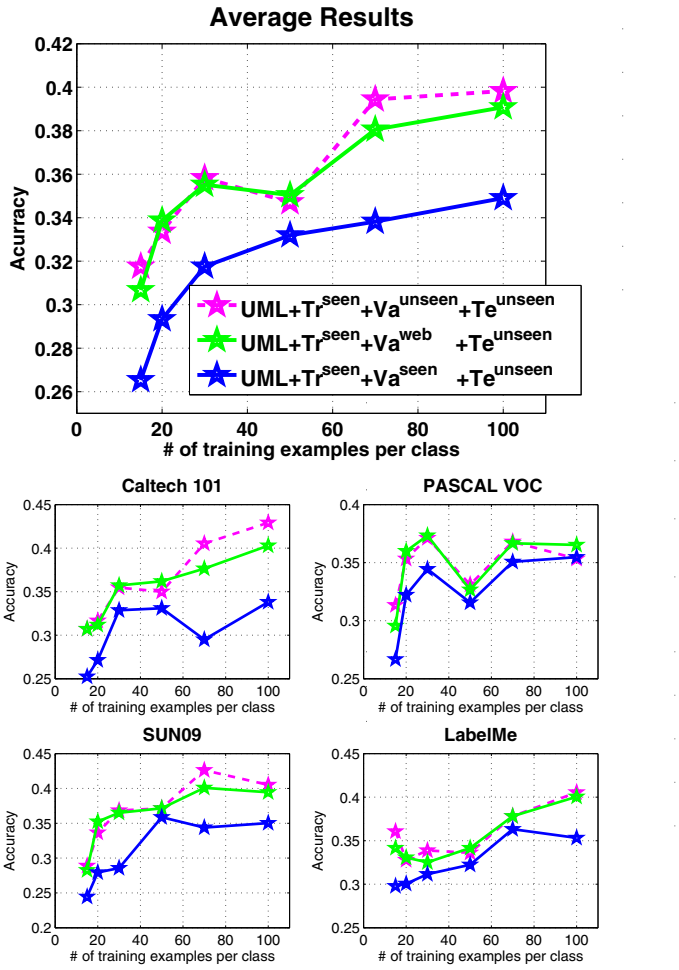

Figure 3. Cross-dataset classification accuracy on unseen test sets. Different validation methods are compared. Top: the figure reports the average accuracy over four individual experiments in the bottom. Bottom: individual classification accuracy on hold-out dataset. The green curve is our validation method using $\mathbf{Va}^{\mathbf{web}}$ and the blue one is conventional validation using $\mathbf{Va}^{\mathbf{seen}}$. The magenta dash line is validated on $\mathbf{Va}^{\mathbf{unseen}}$ (the same as in Fig.2).

methods and validate the produced metrics on $\mathbf{Va}^{\mathbf{seen}}$. As Fig.4 demonstrates our approach significantly outperforms all the baseline methods.

6. Discussion

The goal of this paper is to learn a less biased distance metric that generalizes better to unseen datasets. First, we introduced the notion of "neighborhood diversity" to better connect examples from different datasets and extract general knowledge. We then proposed to use web images as a validation set to select metrics with better generalizability. We have shown that our approach is able to produce superior performance in cross-dataset image classification experiments on four popular datasets.

The contributions of this paper are two-fold. We demonstrated that by tuning the distribution of data from different domains, more generalizable models can be produced. We also showed the advantage of using weakly-labeled web images as validation set to select model with better generalization ability. Web images are known to be easy and cheap to obtain. Our work uncovers another nice property: less biased.

Acknowledgements

We are grateful to Aditya Khosla for sharing data. Thanks to Alessandro Bergamo and Yuting Sun for proofreading drafts. The authors were partly supported by AFOSR Award FA9550-11-1-0166 and the Neukom Institute for Computational Science.

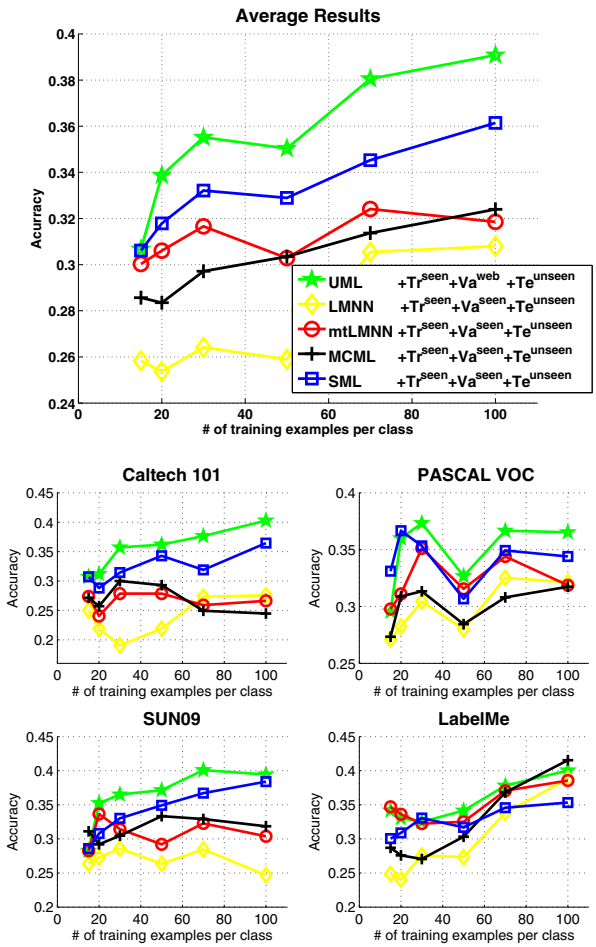

Figure 4. Cross-dataset classification accuracy on unseen test sets. UML (green curve) is compared with baselines. Top: the average accuracy over four individual experiments in the bottom. Bottom: individual classification accuracy on hold-out dataset.

References

[1] M. Benito, J. Parker, Q. Du, J. Wu, D. Xiang, C. M. Perou, and J. S. Marron. Adjustment of systematic microarray data biases. *Bioinformatics*, 20(1):105–114, 2004.

[2] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Found. Trends Mach. Learn.*, 3(1):1–122, Jan. 2011.

[3] M. J. Choi, J. J. Lim, A. Torralba, and A. S. Willsky. Exploiting hierarchical context on a large database of object categories. In *CVPR*, 2010.

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

[5] T. Evgeniou and M. Pontil. Regularized multi-task learning. In *ACM SIGKDD*, pages 109–117, 2004.

[6] C. Fang and L. Torresani. Measuring image distances via embedding in a semantic manifold. In *ECCV*, 2012.

[7] L. Fei-Fei, R. Fergus, and P. Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Comput. Vis. Image Underst.*, 106(1):59–70, Apr. 2007.

[8] R. Fergus, L. Fei-Fei, P. Perona, and A. Zisserman. Learning object categories from google's image search. In *ICCV*, 2005.

[9] A. Globerson and S. Roweis. Metric learning by collapsing classes. *NIPS*, 2006.

[10] L. jia Li, G. Wang, and L. Fei-fei. Optimol: automatic online picture collection via incremental model learning. In *CVPR*, 2007.

[11] T. Joachims. A support vector method for multivariate performance measures. In *ICML*, 2005.

[12] A. Khosla, T. Zhou, T. Malisiewicz, A. Efros, and A. Torralba. Undoing the damage of dataset bias. In *ECCV*, 2012.

[13] D. Lim, B. Mcfee, and G. R. Lanckriet. Robust structural metric learning. In *ICML*, 2013.

[14] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 60(2):91–110, Nov. 2004.

[15] B. Mcfee and G. Lanckriet. Metric learning to rank. In *ICML*, 2010.

[16] K. Mikolajczyk and C. Schmid. Scale and affine invariant interest point detectors. *IJCV*, 60(1):63–86, 2004.

[17] S. Parameswaran and K. Weinberger. Large margin multi-task metric learning. In *NIPS*. 2010.

[18] W. Ping, Y. Xu, J. Wang, and X.-S. Hua. Famer: Making multi-instance learning better and faster. In *SDM*, 2011.

[19] M. Rastegari, C. Fang, and L. Torresani. Scalable object-class retrieval with approximate and top-k ranking. In *ICCV*, 2011.

[20] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman. Labelme: A database and web-based tool for image annotation. *IJCV*, 77(1-3):157–173, May 2008.

[21] K. Saenko, B. Kulis, M. Fritz, and T. Darrell. Adapting visual category models to new domains. In *ECCV*, 2010.

[22] T. Tommasi, N. Quadrianto, B. Caputo, and C. H. Lampert. Beyond dataset bias: Multi-task unaligned shared knowledge transfer. In *ACCV*, 2012.

[23] A. Torralba and A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

[24] K. Q. Weinberger and L. K. Saul. Distance metric learning for large margin nearest neighbor classification. *J. Mach. Learn. Res.*, 10:207–244, June 2009.

[25] Y. Xu, W. Ping, and A. Campbell. Multi-instance metric learning. In *ICDM*, 2011.

[26] J. Yang, K. Yu, Y. Gong, and T. Huang. Linear spatial pyramid matching using sparse coding for image classification. In *CVPR*, 2009.

[27] Y. Yue and T. Finley. A support vector method for optimizing average precision. In *SIGIR*, 2007.