Write a Classifier: Predicting Visual Classifiers from Unstructured Text Descriptions

Mohamed Elhoseiny, *Member, IEEE,* Ahmed Elgammal, *Senior Member, IEEE,* and Babak Saleh

Abstract—**People typically learn through exposure to visual concepts associated with linguistic descriptions. For instance, teaching visual object categories to children is often accompanied by descriptions in text or speech. In a machine learning context, these observations motivates us to ask whether this learning process could be computationally modeled to learn visual classifiers. More specifically, the main question of this work is how to utilize purely textual description of visual classes with no training images, to learn explicit visual classifiers for them. We propose and investigate two baseline formulations, based on regression and domain transfer that predict a linear classifier. Then, we propose a new constrained optimization formulation that combines a regression function and a knowledge transfer function with additional constraints to predict the linear classifier parameters for new classes. We also propose a generic kernelized models where a kernel classifier, in the form defined by the representer theorem, is predicted. The kernelized models allow defining any two RKHS kernel functions in the visual space and text space, respectively, and could be useful for other applications. We finally propose a kernel function between unstructured text descriptions that builds on distributional semantics, which shows an advantage in our setting and could be useful for other applications. We applied all the studied models to predict visual classifiers for two fine-grained categorization datasets, and the results indicate successful predictions of our final model against several baselines that we designed.**

Index Terms—**Language and Vision, Zero Shot Learning, Unstructured Text.**

I. INTRODUCTION

One of the main challenges for scaling up object recognition systems is the lack of annotated images for real-world categories. Typically there are few images available for training classifiers for most of these categories. This is reflected in the number of images per category available for training in most object categorization datasets, which, as pointed out in [1], shows a Zipf distribution. The problem of lack of training images becomes even more severe when we target recognition problems within a general category, i.e., fine-grained categorization, for example building classifiers for different bird species or flower types (there are estimated over 10000 living bird species, similar for flowers). Researchers try to exploit shared knowledge between categories to target such scalability issue. This motivated many researchers who looked into approaches that learn visual classifiers from few examples, e.g. [2], [3], [4]. This even motivated a more recent work on zero-shot learning of visual categories where there are no training images available for test categories (unseen classes), e.g. [5]. Such approaches exploit the similarity (visual or semantic) between seen classes and unseen ones, or describe unseen classes in terms of a learned vocabulary of semantic visual attributes.
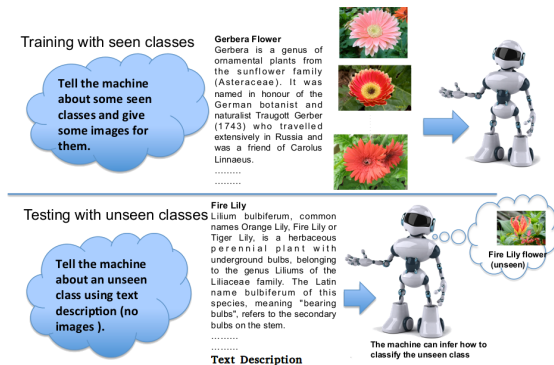

Fig. 1: Our proposed setting where machine can predict unseen class from class-level unstructured text description

In contrast to the lack of reasonable size training sets for a large number of real world categories, there are abundant of textual descriptions of these categories. This comes in the form of dictionary entries, encyclopedia articles, and various online resources. For example, it is possible to find several good descriptions of a "bobolink" in encyclopedias of birds, while there are only a few images available for that bird online.

The main question we address in this paper is how to use purely textual description of categories with no training images to learn visual classifiers for these categories. In other words, we aim at zero-shot learning of object categories where the description of unseen categories comes in the form of typical text such as an encyclopedia entry; see figure 1. We explicitly address the question of how to automatically decide which information to transfer between classes without the need of human intervention. In contrast to most related work, we go beyond the simple use of tags and image captions, and apply standard Natural Language Processing techniques to typical text to learn visual classifiers.

Fine-grained categorization refers to classification of highly similar objects. This similarity can be due to natural intrinsic characteristics of subordinates of one category of objects (e.g. different breeds of dogs) or artificial subcategories of an object class (different types of airplanes). Diverse applications of fine-grained categories range from classification of natural species [6], [7], [8], [9] to retrieval of different types of commercial products [10]. In this problem, When we learn from an expert about different species of birds, the teacher will not just give you sample images of each species and their class labels; the teacher will tell you about discriminative visual or non-visual features for each species, similarity and differences between species, hierarchal relations between species, and many other aspects. The same learning experience takes place when you read a book or a web page to learn about the dif-

2

The Bobolink (Dolichonyx oryzivorus) is a small New World blackbird and the only member of genus Dolichonyx.

Description: Adults are 16-18 cm (6-8 in) long with short finch-like bills. They weigh about 1 oz. Adult males are mostly black, although they do display creamy napes, and white scapulars, lower backs and rumps. Adult females are mostly light brown, although their coloring includes black streaks on the back and flanks, and dark stripes on the head; their wings and tails are darker. The collective name for a group of bobolinks is a chain.

Distribution and movement: These birds migrate to Argentina, Bolivia and Paraguay. One bird was tracked flying 12,000 mi over the course of the year, and up to 1,100 mi in one day. They often migrate in flocks, feeding on cultivated grains and rice, which leads to them being considered a pest by farmers in some areas. Although Bobolinks migrate long distances, they have rarely been sighted in Europe-like many vagrants from the Americas, the overwhelming majority of records are from the British Isles. Each fall, Bobolinks gather in large numbers in South American rice fields, where they are inclined to eat grain. This has earned them the name "ricebird" in these parts. However, they are called something entirely different in Jamaica (Butterbirds) where they are collected as food, being that they are very fat as they pass through on migration.

Behavior: Their breeding habitats are open grassy fields, especially hay fields, across North America. In high-quality habitats, males are often polygynous. Females lay 5 to 6 eggs in a cup-shaped nest, which is always situated on the ground and is usually well-hidden in dense vegetation. Both parents feed the young. Bobolinks forage on or near the ground, and mainly eat seeds, insects, cultivated grains and rice. Males sing bright, bubbly songs in flight; these songs gave this species its common name.
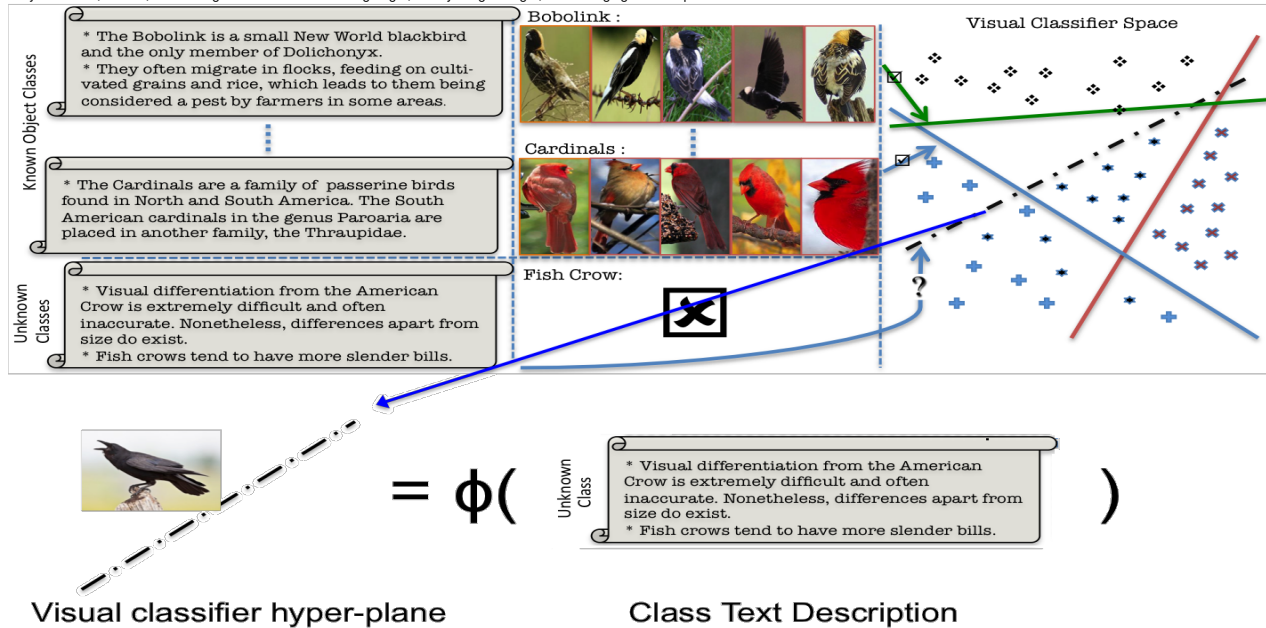
Fig. 2: Top: Example Wikipedia article about the Painted Bunting, with an example image. Bottom: The proposed learning setting. For each category we are give one (or more) textual description (only a synopsis of a larger text is shown), and a set of training images. Our goal is to be able to predict a classifier for a category based only on the narrative (zero-shot learning).

ferent species of birds; For example, Fig. 2 shows an example narrative about the Bobolink. Typically, the narrative tells you about the bird's taxonomy, highlights discriminative features about that bird and discusses similarities and differences between species, as well as within-species variations (male vs. female). The narrative might eventually show very few example images, which are often selected wisely to illustrate certain visual aspects that might be hard to explain in the narrative. This learning strategy using textual narrative and images makes the learning effective without a huge number of images that a typical visual learning algorithm would need to learn the class boundaries.

However, a narrative about a specific species does not contain only "visually" relevant information, but also gives abundant information about the species's habitat, diet, mating habits that is not relevant for visual identification. In a sense, this information might be textual clutter for that task. The same problem takes place in images. While one image can be very effective in highlighting an important feature for learning, many images might have a lot of visual clutter that makes their uses in learning not effective. Thus, a picture can be worth a thousand words, but not always, and an abundant number of pictures might not be the most effective way for learning. Similarly, one text paragraph can be worth a thousand pictures for learning a concept, but not always, and large amounts of text might not necessarily be effective.

Contributions. The contribution of the paper is on explor-

ing this new problem, which to the best of our knowledge, is firstly explored in the computer vision community in this work. We learn from an image corpus and a textual corpus, however not in the form of image-caption pairs, instead the only alignment between the corpora is at the level of the category. In particular, we address the problem of formulating a visual classifier prediction function $\Phi(\cdot)$, which predicts a classifier of unseen visual class given its text description; see figure 2. While a part of this work was published in [11], we extend the work here to study more formulations to solve the problem in Sec. V (B,E). In addition, we also propose a kernel method to explicitly predict a kernel classifier in the form defined in the representer theorem [12]. The kernelized prediction has an advantage that it opens the door for using any kind of side information about classes, as long as kernels can be used on the side information representation. The side information can be in the form of textual, parse trees, grammar, visual representations, concepts in the ontologies (adopted in NLP domain), or any form. We focus here on unstructured text descriptions. The image features also do not need to be in a vectorized format. The kernelized classifiers also facilitates combining different types of features through a multi-kernel learning (MKL) paradigm, where the fusion of different features can be effectively achieved.

Beyond the introduction and the related work sections, the paper is structured as follows: Sec III and IV details the problem definition and relation to regression and knowledge

transfer models. Sec V shows different formulations of $\Phi(\cdot)$ that we studied to predict a linear visual classifier; see figure 2. Section VI developed extends our notion where $\Phi(\cdot)$ predicts kernel classifier in the form defined by the representer theorem [12]. Sec VII presents our proposed distributional semantic kernel between unstructured text description, which is applicable to our kernel formulation and could be useful for other applications. Sec VIII presents our experiments on Flower Dataset [7] (102 classes) and Caltech-UCSD dataset [13] (200 classes) for both the linear and the kernel classifier prediction.

II. RELATED WORK

We focus our related work discussion on three related lines of research: "zero/few-shot learning", "visual knowledge transfer", and "Language and Vision".

Zero/Few-Shot Learning: Similar to the setting of zero-shot learning, we use classes with training data (seen classes) to predict classifiers for classes with no training data (unseen classes). Recent works on zero-shot learning of object categories focused on leveraging knowledge about common attributes and shared parts [5]. Typically, attributes [14], [15] are manually defined by humans and are used to transfer knowledge between seen and unseen classes. In contrast, in our work we do not use any explicit attributes. The description of a new category is purely textual and the process is totally automatic without human annotation beyond the class labels.

Motivated by the practical need to learn visual classifiers of rare categories, researchers have explored approaches for learning from a single image (one-shot learning [16], [3], [17], [4]) or even from no images (zero-shot learning). One way of recognizing object instances from previously unseen test categories (the zero-shot learning problem) is by leveraging knowledge about common attributes and shared parts. Typically an intermediate semantic layer is introduced to enable sharing knowledge between classes and facilitate describing knowledge about novel unseen classes, e.g. [18]. For instance, given adequately labeled training data, one can learn classifiers for the attributes occurring in the training object categories. These classifiers can then be used to recognize the same attributes in object instances from the novel test categories. Recognition can then proceed on the basis of these learned attributes [5], [15]. Such attribute-based "knowledge transfer" approaches use an intermediate visual attribute representation to enable describing unseen object categories. Typically attributes are manually defined by humans to describe shape, color, surface material, e.g., furry, striped, etc.Therefore, an unseen category has to be specified in terms of the used vocabulary of attributes. Rohrbach et al. [19] investigated extracting useful attributes from large text corpora. In [20], an approach was introduced for interactively defining a vocabulary of attributes that are both human understandable and visually discriminative. Huang et al. [21] relaxed the attribute independence assumption by modeling correlation between attributes to achieve better zero shot performance, as opposed to prior models. In contrast, our work does not use any explicit attributes. The description of a new category is purely textual.

Visual Knowledge Transfer: Our proposed work can be seen in the context of knowledge sharing and inductive transfer. In general, knowledge transfer aims at enhancing recognition by exploiting shared knowledge between classes. Most existing research focused on knowledge sharing within the visual domain only, e.g. [22]; or exporting semantic knowledge at the level of category similarities and hierarchies, e.g. [23], [1]. We go beyond the state-of-the-art to explore cross-domain knowledge sharing and transfer. We explore how knowledge from the visual and textual domains can be used to learn across-domain correlation, which facilitates prediction of visual classifiers from textual description.

Language and Vision: The relation between linguistic semantic representations and visual recognition have been explored. For example in [2], it was shown that there is a strong correlation between semantic similarity between classes, based on WordNet, and confusion between classes. Linguistic semantics in terms of nouns from WordNet [24] have been used in collecting large-scale image datasets such as ImageNet[25] and Tiny Images [26]. It was also shown that hierarchies based on WordNet are useful in learning visual classifiers, e.g. [1].

One of the earliest work on learning from images and text corpora is the work of Barnard et al. [27], which showed that learning a joint distribution of words and visual elements facilitates clustering the images in a semantic way, generating illustrative images from a caption, and generating annotations for novel images. There has been an increasing recent interest in the intersection between computer vision and natural language processing with researches that focus on generating textual description of images and videos, e.g. [28], [29], [30], [31]. This includes generating sentences about objects, actions, attributes, patial relation between objects, contextual information in the images, scene information, etc. In contrast, our work is different in two fundamental ways. In terms of the goal, we do not target generating textual description from images, instead we target predicting classifiers from text, in a zero-shot setting. In terms of the learning setting, the textual descriptions that we use is at the level of the category and do not come in the form of image-caption pairs, as in typical datasets used for text generation from images, e.g. [32]. Recent impressive works has been proposed for image captioning (e.g., [33], [34], [35], [36]), which are based on the success of sequence to sequence training of neural nets for translation (e.g., [37]). In contrast, our work focus on a different setting, which is transforming an Encyclopedia text description to visual classifier.

There are several recent works that involved unannotated text. In [38] and [39], word embedding language models (e.g [40]) was adopted to represent class names as vectors which requires training using a big text-corpus. Their goal is to embed images into the language space then perform classification. There are several differences between these works and our method. First, one limitation of the adopted language model is that it produces only one vector per word, which produces problems when a word has multiple meanings. Second, it can not represent a big text description of a class instead of a single or few words or others forms that could be ambiguous. Third, our goal is different which is to map the text description to a classifier in the visual domain, i.e.the

apposite direction of their goal. Forth, these models do not support non-linear classification, supported by the kernelized version proposed in this work. We finally focus on fine-grained recognition, which is very challenging.

We can summarize the features of our proposed framework in contrast to prior work as follows: 1) Our framework explicitly predicts classifiers; 2) The predicted classifiers could be linear or kernelized; 3) The approach requires the text description at the class level, not at the image level, hence, it needs only weak annotation.

III. PROBLEM DEFINITION

Fig 2 illustrates the learning setting. The information in our problem comes from two different domains: the visual domain and the textual domain, denoted by $\mathcal{V}$ and $\mathcal{T}$, respectively. Similar to traditional visual learning problems, we are given training data in the form $V = \{(x_i, l_i)\}_N$, where x_i is an image and $l_i \in \{1 \cdots N_{sc}\}$ is its class label. We denote the number of classes available at training as N_{sc}, where sc indicates "seen classes". As typically done in visual classification setting, we can learn N_{sc} binary one-vs-all classifiers, one for each of these classes.

Our goal is to be able to predict a classifier for a new category based only on the learned classes and a textual description(s) of that category. In order to achieve that, the learning process has to also include textual description of the seen classes (as shown in Fig 2). Depending on the domain we might find a few, a couple, or as little as one textual description to each class. We denote the textual training data for class j by $\{t_i \in \mathcal{T}\}^j$. In this paper we assume we are dealing with the extreme case of having only one textual description available per class, which makes the problem even more challenging. For simplicity, the text description of class j is denoted by t_j. However, the formulation we propose in this paper directly applies to the case of multiple textual descriptions per class.

In this paper, we cover predicting visual classifier $\Phi(t_*)$ from an unseen text description t_* in linear form or RKHS kernelized form, defined as follows

A. Linear Classifier

Let us consider a typical linear classifier in the feature space in the form

$$f_j(\mathbf{x}) = \mathbf{c}_j^\mathsf{T} \cdot \mathbf{x}$$

where $\mathbf{x}$ (bold) is the visual feature vector of an image x (not bold) amended with 1 and $\mathbf{c}_j \in \mathbb{R}^{d_v}$ is the linear classifier parameters for class j. Given a test image, its class is determined by

$$l^* = \arg\max_j f_j(\mathbf{x}) \tag{1}$$

Similar to the visual domain, the raw textual descriptions have to go through a feature extraction process. Let us denote the linear extracted textual feature by $T = \{\mathbf{t}_j \in \mathbb{R}^{d_t}\}_{j=1\cdots N_{sc}}$, where $\mathbf{t}_j$ is the features of text description t_j (not bold). Given a textual description t_* of a new unseen category $\mathcal{U}$ with linear feature vector representation $\mathbf{t}_*$, the problem can now be defined as predicting a one-vs-all linear classifier parameters $\Phi(t_*) = c(\mathbf{t}_*) \in \mathbb{R}^{d_v}$, such that it can be directly used to classify any test image $\mathbf{x}$ as

$$c(\mathbf{t}_*)^\mathsf{T} \cdot \mathbf{x} > 0 \quad \text{if } \mathbf{x} \text{ belongs to } \mathcal{U}$$
$$c(\mathbf{t}_*)^\mathsf{T} \cdot \mathbf{x} < 0 \qquad \text{otherwise} \tag{2}$$

B. Kernel Classifier

For kernel classifiers, we assume that each of the domains is equipped with a kernel function corresponding to a *reproducing kernel Hilbert space* (RKHS). Let us denote the kernel for $\mathcal{V}$ by $k(\cdot, \cdot)$, and the kernel for $\mathcal{T}$ by $g(\cdot, \cdot)$.

According to the generalized representer theorem [12], a minimizer of a regularized empirical risk function over an RKHS could be represented as a linear combination of kernels, evaluated on the training set. Adopting the representer theorem on classification risk function, we define a kernel-classifier of a visual class j as follows

$$f_j(x) = \sum_{i=1}^N \beta_j^i k(x, x_i) + b = \sum_{i=1}^N \beta_j^i \varphi(x_i)^\mathsf{T} \varphi(x) + b$$
$$f_j(x) = \boldsymbol{\beta}_j^\mathsf{T} \cdot \mathbf{k}(x) = \mathbf{c}_j^\mathsf{T} \cdot [\varphi(x); 1], \mathbf{c}_j = [\sum_{i=1}^N \beta_j^i \varphi(x_i); b] \tag{3}$$

where $x \in \mathcal{V}$ is the test image, x_i is the ith image in the training data V, $\mathbf{k}(x) = [k(x, x_1), \cdots, k(x, x_N), 1]^\mathsf{T}$, $\boldsymbol{\beta}_j = [\beta_j^1 \cdots \beta_j^N, b]^\mathsf{T}$. Having learned $f_j(x^*)$ for each class j (for example using SVM classifier), the class label of the test image x can be predicted by Eq. 1, similar to the linear case. Eq. 3 also shows how $\boldsymbol{\beta}_j$ is related to $\mathbf{c}_j$ in the linear classifier, where $k(x, x') = \varphi(x)^\mathsf{T} \cdot \varphi(x')$ and $\varphi(\cdot)$ is a feature map that does not have to be defined given $k(\cdot, \cdot)$ on $\mathcal{V}$. Hence, our goal in the kernel classifier prediction is to predict $\boldsymbol{\beta}(t_*)$ instead of $\mathbf{c}(t_*)$ since it is sufficient to define $f_{t_*}(x)$ for a text description t_* of an unseen class given $\mathbf{k}(x)$

It is clear that $f_j(x)$ could be learned for all classes with training data $j \in 1 \cdots N_{sc}$, since there are examples for the seen classes; we denote the kernel-classifier parameters of the seen classes as $\mathcal{B}_{sc} = \{\boldsymbol{\beta}_j\}_{N_{sc}}, \forall j$. However, it is not obvious how to predict $f_{t_*}(x)$ for an unseen class given its text description t_*. Similar to the linear classifier prediction, our main notion is to use the text description t_*, associated with unseen class, and the training data to directly predict the unseen kernel-classifier parameters. In other words, the kernel classifier parameters of the unseen class is a function of its text description t_*, the image training data V and the text training data $\{t_j\}, j \in 1 \cdots N_{sc}$; i.e.

$$f_{t_*}(x) = \boldsymbol{\beta}(t_*)^\mathsf{T} \cdot \mathbf{k}(x),$$

$f_{t_*}(x)$ could be used to classify new points that belong to an unseen class as follows: 1) one-vs-all setting $f_{t_*}(x) \gtrless 0$; or 2) in a Multi-class prediction as in Eq 1. $\Phi(t_*)$ in this case is $\boldsymbol{\beta}(t_*)$. In contrast to linear classifier prediction, there is no need to explicitly represent an image x or a text description t by features, which are denoted by the bold symbols in the previous section. Rather, only the $k(\cdot, \cdot)$ and $g(\cdot, \cdot)$ needs to be defined which is general.

IV. RELATION TO REGRESSION AND KNOWLEDGE TRANSFER MODELS

We introduce two possible frameworks for this problem and discuss potential limitations for them. In this background section, we focus on predicting linear classifiers for simplicity, which motivates the the evaluated linear classifier formulations that follow in Sec V.

A. Regression Models

A straightforward way to solve this problem is to pose it as a regression problem where the goal is to use the textual data and the learned classifiers, $\{(\mathbf{t}_j, \mathbf{c}_j)\}_{j=1\cdots N_{sc}}$ to learn a regression function from the textual feature domain to the visual classifier domain, i.e., a function $c(\cdot) : \mathbb{R}^{d_t} \to \mathbb{R}^{d_v}$. The question is which regression model would be suitable for this problem? and would posing the problem this way give reasonable results?

A typical regression model, such as ridge regression [41] or Gaussian Process (GP) Regression [42], learns the regressor to each dimension of the output domain (the parameters of a linear classifier) separately, i.e., a set of functions $c^i(\cdot) : \mathbb{R}^{d_t} \to \mathbb{R}$. Clearly this will not capture the correlation between the visual classifier dimensions. Instead, a structured prediction regressor would be more suitable since it would learn the correlation between the input and output domain. However, even a structured prediction model, will only learn the correlation between the textual and visual domain through the information available in the input-output pairs $(\mathbf{t}_j, \mathbf{c}_j)$. Here the visual domain information is encapsulated in the pre-learned classifiers and prediction does not have access to the original data in the visual domain. Instead we need to directly learn the correlation between the visual and textual domain and use that for prediction.

Another fundamental problem that a regressor would face, is the sparsity of the data; the data points are the textual description-classifier pairs, and typically the number of classes can be very small compared to the dimension of the classifier space (i.e. $N_{sc} \ll d_v$). In a setting like that, any regression model is bound to suffer from an under fitting problem. This can be best explained in terms of GP regression, where the predictive variance increases in the regions of the input space where there are no data points. This will result in poor prediction of classifiers at these regions.

B. Knowledge Transfer Models

An alternative formulation is to pose the problem as domain adaptation from the textual to the visual domain. In the computer vision context, domain adaptation work has focused on transferring categories learned from a source domain, with a given distribution of images, to a target domain with different distribution, e.g., images or videos from different sources [43], [44], [45], [46]. What we need is an approach that learns the correlation between the textual domain features and the visual domain features, and uses that correlation to predict new visual classifier given textual features.

In particular, in [45] an approach for learning cross domain transformation was introduced. In that work a regularized asymmetric transformation between points in two domains were learned. The approach was applied to transfer learned categories between different data distributions, both in the visual domain. A particular attractive characteristic of [45], over other domain adaptation models, is that the source and target domains do not have to share the same feature spaces or the same dimensionality.

While a totally different setting is studied in [45], it inspired us to formulate the zero-shot learning problem as a domain transfer problem. This can be achieved by learning a linear transfer function $\mathbf{W}$ between $\mathcal{T}$ and $\mathcal{V}$. The transformation matrix $\mathbf{W}$ can be learned by optimizing, with a suitable regularizer, over constraints of the form $\mathbf{t}^\mathsf{T}\mathbf{W}\mathbf{x} \geq l$ if $\mathbf{t} \in \mathcal{T}$ and $\mathbf{x} \in \mathcal{V}$ belong to the same class, and $\mathbf{t}^\mathsf{T}\mathbf{W}\mathbf{x} \leq u$ otherwise. Here l and u are model parameters. This transfer function acts as a compatibility function between the textual features and visual features, which gives high values if they are from the same class and a low value if they are from different classes.

It is not hard to see that this transfer function can act as a classifier. Given a textual feature $\mathbf{t}^*$ and a test image, represented by $\mathbf{x}$, a classification decision can be obtained by $\mathbf{t}_*^\mathsf{T}\mathbf{W}\mathbf{x} \gtrless b$ where b is a decision boundary which can be set to $(l + u)/2$. Hence, our desired predicted classifier in Eq 2 can be obtained as $c(\mathbf{t}_*) = \mathbf{t}_*^\mathsf{T}\mathbf{W}$ (note that the features vectors are amended with ones). However, since learning $\mathbf{W}$ was done over seen classes only, it is not clear how the predicted classifier $c(\mathbf{t}_*)$ will behave for unseen classes. There is no guarantee that such a classifier will put all the seen data on one side and the new unseen class on the other side of that hyperplane.

V. FORMULATIONS FOR PREDICTING A LINEAR CLASSIFIER FORM OF $\Phi(t_*)$

The proposed formulations in this section aims at predicting a linear hyperplane parameter $\mathbf{c}$ of a one-vs-all classifier for a new unseen class given a textual description, encoded as a feature vector $\mathbf{t}_*$ and the knowledge learned at the training phase from seen classes[1]. We start by defining the learning components that were used by the formulations described in this section:

Classifiers:
    a set of linear one-vs-all classifiers $\{\mathbf{c}_j\}$ are learned, one for each seen class.

Probabilistic Regressor:
    Given $\{(\mathbf{t}_j, \mathbf{c}_j)\}$ a regressor is learned that can be used to give a prior estimate for $p_{reg}(\mathbf{c}|\mathbf{t})$ (Details in Sec V-A).

Domain Transfer:
    Given T and V a domain transfer function, encoded in the matrix $\mathbf{W}$ is learned, which captures the correlation between the textual and visual domains (Details in Sec V-C).

Each of the following subsections show a different approach to predict a linear classifier from t_* as $\Phi(t_*) = \mathbf{c}(\mathbf{t}_*)$;

---

[1]The notations follow from Subsection III-A

see Sec III-A. The final approach (E) combines regression, domain transfer, and additional constraints, which achieves the best performance. We compare between these alternative formulations in our experiments. Hyper-parameter selection is detailed in the supplementary materials for all the approaches.

A. Probabilistic Regressor

There are different regressors that can be used, however we need a regressor that provide a probabilistic estimate $p_{reg}(\mathbf{c}|(t))$. For the reasons explained in Sec III, we also need a structure prediction approach that is able to predict all the dimensions of the classifiers together. For these reasons, we use the Twin Gaussian Process (TGP) [47]. TGP encodes the relations between both the inputs and structured outputs using Gaussian Process priors. This is achieved by minimizing the Kullback-Leibler divergence between the marginal GP of the outputs (i.e. classifiers in our case) and observations (i.e. textual features). The estimated regressor output ($\tilde{c}(\mathbf{t}_*)$) in TGP is given by the solution of the following non-linear optimization problem [47] [2].

$$\Phi(t_*) = \tilde{c}(\mathbf{t}_*) = \underset{\mathbf{c}}{\mathrm{argmin}}[K_C(\mathbf{c}, \mathbf{c}) - 2k_c(\mathbf{c})^\mathsf{T}\mathbf{u} - \eta \log(\\ K_C(\mathbf{c}, \mathbf{c} - k_c(\mathbf{c})^\mathsf{T}(\mathbf{K}_C + \lambda_c \mathbf{I})^{-1} k_c(c))] \tag{4}$$

where $\mathbf{u} = (\mathbf{K}_\mathsf{T} + \lambda_t \mathbf{I})^{-1} k_t(\mathbf{t}_*)$, $\eta = K_T(\mathbf{t}_*, \mathbf{t}_*) - k(\mathbf{t}_*)^\mathsf{T}\mathbf{u}$, $K_T(\mathbf{t}_l, \mathbf{t}_m)$ and $K_C(\mathbf{c}_l, \mathbf{c}_m)$ are Gaussian kernel for input feature $\mathbf{t}$ and output vector $\mathbf{c}$. $k_c(\mathbf{c}) = [K_C(\mathbf{c}, \mathbf{c}_1), \cdots, K_C(\mathbf{c}, \mathbf{c}_{N_{sc}})]^\mathsf{T}$. $k_t(\mathbf{t}_*) = [K_T(\mathbf{t}_*, \mathbf{t}_1), \cdots, K_T(\mathbf{t}_*, \mathbf{t}_{N_{sc}})]^\mathsf{T}$. λ_t and λ_c are regularization parameters to avoid overfitting. This optimization problem can be solved using a second order, BFGS quasi-Newton optimizer with cubic polynomial line search for optimal step size selection [47]. In this case the classifier dimension are predicted jointly. In this case $p_{reg}(\mathbf{c}|\mathbf{t}_*)$ is defined as a normal distribution.

$$p_{reg}(\mathbf{c}|\mathbf{t}_*) = \mathcal{N}(\mu_c = \tilde{c}(\mathbf{t}_*), \Sigma_c = \mathbf{I}) \tag{5}$$

The reason that $\Sigma_c = \mathbf{I}$ is that TGP does not provide predictive variance, unlike Gaussian Process Regression. However, it has the advantage of handling the dependency between the dimensions of the classifiers $\mathbf{c}$ given the textual features $\mathbf{t}$.

B. Constrained Probabilistic Regressor

We also investigated formulations that use regression to predict an initial hyperplane $\tilde{c}(\mathbf{t}_*)$ as described in section V-A, which is then optimized to put all seen data in one side, i.e.

$$\Phi(t_*) = \hat{c}(\mathbf{t}_*) = \underset{\mathbf{c}, \zeta_i}{\mathrm{argmin}}[\mathbf{c}^\mathsf{T}\mathbf{c} + \alpha\,\psi(\mathbf{c}, \tilde{c}(\mathbf{t}_*)) + C\sum_{i=1}^{N}\zeta_i]$$
$$s.t. : -\mathbf{c}^\mathsf{T}\mathbf{x}_i \geq \zeta_i, \ \zeta_i \geq 0, i = 1, \cdots, N$$

where $\psi(\cdot, \cdot)$ is a similarity function between hyperplanes, e.g.a dot product used in this work, α is its constant weight, and C is the weight to the soft constraints of existing images as negative examples (inspired by linear SVM formulation). We call this class of methods *constrained GPR/TGP*, since $\tilde{c}(\mathbf{t}_*)$ is initially predicted through GPR or TGP.

---

[2]notice we are using $\tilde{\mathbf{c}}$ to denote the output of the regressor, while using $\hat{\mathbf{c}}$ to denote the output of the final optimization problem in Eq 8

C. Domain Transfer (DT)

To learn the domain transfer function $\mathbf{W}$ we adapted the approach in [45] as follows. Let $\mathbf{T}$ be the textual feature data matrix and $\mathbf{X}$ be the visual feature data matrix where each feature vector is amended with a 1. Notice that amending the feature vectors with a 1 is essential in our formulation since we need $\mathbf{t}^\mathsf{T}\mathbf{W}$ to act as a classifier. We need to solve the following optimization problem

$$\min_{\mathbf{W}} r(\mathbf{W}) + \lambda \sum_i c_i(\mathbf{TWX}^\mathsf{T}) \tag{6}$$

where c_i's are loss functions over the constraints and $r(\cdot)$ is a matrix regularizer. It was shown in [45], under condition on the regularizer, that the optimal $\mathbf{W}$ is in the form of $\mathbf{W}^* = \mathbf{TK}_T^{-\frac{1}{2}}\mathbf{L}^*\mathbf{K}_X^{-\frac{1}{2}}\mathbf{X}^\mathsf{T}$, where $\mathbf{K}_T = \mathbf{TT}^\mathsf{T}$, $\mathbf{K}_X = \mathbf{XX}^\mathsf{T}$. $\mathbf{L}^*$ is computed by minimizing the following minimization problem

$$\min_{\mathbf{L}}[r(\mathbf{L}) + \lambda \sum_p c_p(\mathbf{K}_T^{\frac{1}{2}}\mathbf{LK}_X^{\frac{1}{2}})], \tag{7}$$

where $c_p(\mathbf{K}_T^{\frac{1}{2}}\mathbf{LK}_X^{\frac{1}{2}}) = (max(0, (l - e_i\mathbf{K}_T^{\frac{1}{2}}\mathbf{LK}_X^{\frac{1}{2}}e_j)))^2$ for same class pairs of index i,j, or $= (max(0, (e_i\mathbf{K}_T^{\frac{1}{2}}\mathbf{LK}_X^{\frac{1}{2}}e_j - u)))^2$ otherwise, where e_k is a one-hot vector of zeros except a one at the kth element, and $u > l$ (note any appropriate l, u could work. In our case, we used $l = 2$, $u = -2$). We used a Frobenius norm regularizer. This energy is minimized using a second order BFGS quasi-Newton optimizer. Once L is computed $\mathbf{W}^*$ is computed using the transformation above. Finally $\Phi(t_*) = c(t_*) = \mathbf{t}_*^\mathsf{T}\mathbf{W}$, simplifying $\mathbf{W}^*$ as $\mathbf{W}$.

D. Constrained-DT

We also investigated constrained-DT formulations that learns a transfer matrix $\mathbf{W}$ and enforce $\mathbf{t}_j^\mathsf{T}\mathbf{W}$ to be close to the classifiers learned on seen data, $\{\mathbf{c}_j\}$,i.e.

$$\min_{\mathbf{W}} r(\mathbf{W}) + \lambda_1 \sum_i c_i(\mathbf{TWX}^\mathsf{T}) + \lambda_2 \sum_k (\mathbf{c}_j - \mathbf{t}_j^\mathsf{T}\mathbf{W})^T(\mathbf{c}_j - \mathbf{t}_j^\mathsf{T}\mathbf{W})$$

A classifier can be then obtained by $\Phi(t_*) = c(t_*) = \mathbf{t}_*^\mathsf{T}\mathbf{W}$.

E. Constrained Regression and Domain Transfer for classifier prediction

Fig 3 illustrates our final framework which combines regression (formulation A (using TGP)) and domain transfer (formulation C) with additional constraints. This formulation combines the three learning components described in the beginning of this section. Each of these components contains partial knowledge about the problem. The question is how to combine such knowledge to predict a new classifier given a textual description. The new classifier has to be consistent with the seen classes. The new classifier has to put all the seen instances at one side of the hyperplane, and has to be
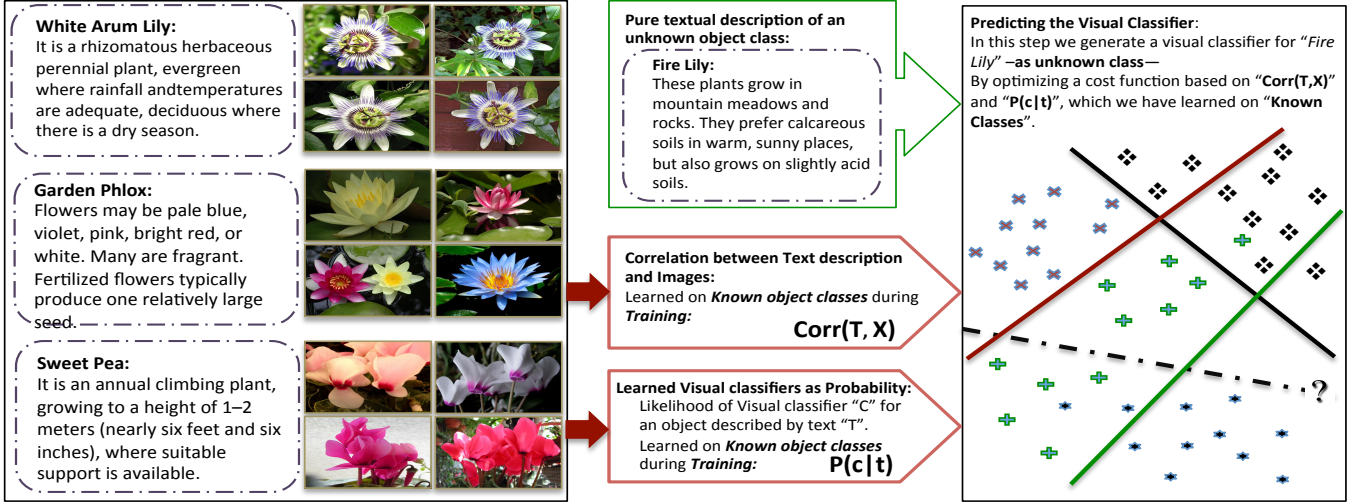
Fig. 3: Illustration of the Proposed Linear Prediction Framework (Constrained Regression and Domain Transfer) for the task Zero-shot learning from textual description (Linear Formulation (E))

consistent with the learned domain transfer function. This leads to the following constrained optimization problem

$$\Phi(t_*) = \hat{c}(\mathbf{t}_*) = \underset{\mathbf{c}, \zeta_i}{\arg\min} \big[\mathbf{c}^\mathsf{T}\mathbf{c} - \alpha\mathbf{t_*}^\mathsf{T}\mathbf{W}\mathbf{c} - \gamma\ln(p_{reg}(\mathbf{c}|\mathbf{t}_*))$$
$$+ C\sum\zeta_i \big]$$
$$s.t. : -(\mathbf{c}^\mathsf{T}\mathbf{x}_i) \geq \zeta_i, \ \zeta_i \geq 0, \ i = 1 \cdots N$$
$$\mathbf{t_*}^\mathsf{T}\mathbf{W}\mathbf{c} \geq l$$
$$\alpha, \gamma, C, l : \text{hyperparameters}$$
(8)

The first term is a regularizer over the classifier $\mathbf{c}$. The second term enforces that the predicted classifier has high correlation with $\mathbf{t_*}^\mathsf{T}\mathbf{W}$; $\mathbf{W}$ is learnt by Eq 10. The third term favors a classifier that has high probability given the prediction of the regressor. The constraints $-\mathbf{c}^\mathsf{T}\mathbf{x}_i \geq \zeta_i$ enforce all the seen data instances to be at the negative side of the predicted classifier hyperplane with some missclassification allowed through the slack variables ζ_i. The constraint $\mathbf{t_*}^\mathsf{T}\mathbf{W}\mathbf{c} \geq l$ enforces that the correlation between the predicted classifier and $\mathbf{t_*}^\mathsf{T}\mathbf{W}$ is no less than l, this is to enforce a minimum correlation between the text and visual features.

Solving for $\hat{c}$ as a quadratic program: According to the definition of $p_{reg}(\mathbf{c}|\mathbf{t}_*)$ for TGP, $\ln p(\mathbf{c}|\mathbf{t}_*)$ is a quadratic term in c in the form

$$-\ln p(\mathbf{c}|\mathbf{t}_*) \propto (\mathbf{c} - \tilde{c}(\mathbf{t}_*))^\mathsf{T}(\mathbf{c} - \tilde{c}(\mathbf{t}_*))$$
$$= \mathbf{c}^\mathsf{T}\mathbf{c} - 2\mathbf{c}^\mathsf{T}\tilde{c}(\mathbf{t}_*) + \tilde{c}(\mathbf{t}_*)^\mathsf{T}\tilde{c}(\mathbf{t}_*)$$
(9)

We reduce $-\ln p(\mathbf{c}|\mathbf{t}_*)$ to $-2\mathbf{c}^\mathsf{T}\tilde{c}(\mathbf{t}_*))$, since 1) $\tilde{c}(\mathbf{t}_*)^\mathsf{T}\tilde{c}(\mathbf{t}_*)$ is a constant (i.e.does not affect the optimization), 2) $\mathbf{c}^\mathsf{T}\mathbf{c}$ is already included as regularizer in equation 8. In our setting, the dot product is a better similarity measure between two hyperplanes. Hence, $-2\mathbf{c}^\mathsf{T}\tilde{c}(\mathbf{t}_*)$ is minimized. Given $-\ln p(\mathbf{c}|\mathbf{t}_*)$ from the TGP and $\mathbf{W}$, Eq 8 reduces to a quadratic program on $\mathbf{c}$ with linear constraints. We tried different quadratic solvers, however the IBM CPLEX solver [3] gives the best performance in speed and optimization for our problem.

[3] http://www-01.ibm.com/software/integration/optimization/cplex-optimizer

VI. FORMULATIONS FOR PREDICTING A KERNEL CLASSIFIER FORM OF $\Phi(\mathbf{t}_*)$

Prediction of $\Phi(t_*) = \boldsymbol{\beta}(t_*)$ (Sec. III-B), is decomposed into training (domain transfer) and prediction phases, detailed as follows

A. Kernelized Domain Transfer

During training, we firstly learn $\mathcal{B}_{sc} = \{\boldsymbol{\beta}_j\}, j = 1 \rightarrow N_s c$ as SVM-kernel classifiers based on the training data and defined by $k(\cdot, \cdot)$ visual kernel. Then, we learn a kernel domain transfer function to transfer the text description information $t_* \in \mathcal{T}$ to kernel-classifier parameters $\boldsymbol{\beta} \in \mathbb{R}^{N+1}$ in $\mathcal{V}$ domain. We call this domain transfer function $\boldsymbol{\beta}_{DA}(t_*)$, which has the form as $\boldsymbol{\Psi}^\mathsf{T}\mathbf{g}(t_*)$, where $\mathbf{g}(t_*) = [g(t_*, t_1) \cdots g(t_*, t_{N_{sc}})]^\mathsf{T}$; $\boldsymbol{\Psi}$ is an $N_{sc} \times N+1$ matrix, which transforms t to kernel classifier parameters for the class that t_* represents.

We aim to learn $\boldsymbol{\Psi}$ from V and $\{t_j\}, j = 1 \cdots N_{sc}$, such that $\mathbf{g}(t)^\mathsf{T}\boldsymbol{\Psi}\mathbf{k}(x) > l$ if t and x correspond to the same class, $\mathbf{g}(t)^\mathsf{T}\boldsymbol{\Psi}\mathbf{k}(x) < u$ otherwise. Here l controls similarity lower-bound if t and x correspond to same class, and u controls similarity upper-bound if t and x belong to different classes. In our setting, the term $\boldsymbol{\Psi}^\mathsf{T}\mathbf{g}(t_j)$ should act as a classifier parameter for class j in the training data. Therefore, we introduce penalization constraints to our minimization function if $\boldsymbol{\Psi}^\mathsf{T}\mathbf{g}(t_j)$ is distant from $\boldsymbol{\beta}_j \in \mathcal{B}_{sc}$, where t_i corresponds to the class that $\boldsymbol{\beta}_i$ classifies. we model the kernel domain transfer function as follows

$$\boldsymbol{\Psi}^* = \arg\min_{\boldsymbol{\Psi}} L(\boldsymbol{\Psi}) = [\frac{1}{2}r(\boldsymbol{\Psi}) + \lambda_1 \sum_k c_k(\mathbf{G}\,\boldsymbol{\Psi}\,\mathbf{K}) +$$
$$\lambda_2 \sum_{i=1}^{N_{sc}} \|\boldsymbol{\beta}_i - \boldsymbol{\Psi}^\mathsf{T}\mathbf{g}(t_i)\|^2$$
(10)

where, $\mathbf{G}$ is an $N_{sc} \times N_{sc}$ symmetric matrix, such that both the ith row and the ith column are equal to $\mathbf{g}(t_i)$, $i = 1 : N_{sc}$; $\mathbf{K}$ is an $N+1 \times N$ matrix, such that the ith column is equal to $\mathbf{k}(x_i)$, $x_i, i = 1 : N$. c_k's are loss functions over the constraints defined as $c_k(\mathbf{G}\,\boldsymbol{\Psi}\,\mathbf{K}) = (max(0, (l - \mathbf{1}_i^\mathsf{T}\mathbf{G}\,\boldsymbol{\Psi}\,\mathbf{K}\mathbf{1}_j)))^2$ for same class pairs of index i and j, or $= r \cdot (max(0, (\mathbf{1}_i^\mathsf{T}\mathbf{G}\,\boldsymbol{\Psi}\,\mathbf{K}\mathbf{1}_j -$

$u)))^2$ otherwise, where $\mathbf{1}_i$ is an $N_{sc} \times 1$ vector with all zeros except at index i, $\mathbf{1}_j$ is an $N \times 1$ vector with all zeros except at index j. This leads to that $c_k(\mathbf{G} \boldsymbol{\Psi} \mathbf{K})) = (max(0, (l - \mathbf{g}(t_i)^\mathsf{T} \boldsymbol{\Psi} \mathbf{k}(x_j)))^2$ for same class pairs of index i and j, or $= r \cdot (max(0, (\mathbf{g}(t_i)^\mathsf{T} \boldsymbol{\Psi} \mathbf{k}(x_j) - u)))^2$ otherwise, where $u > l$, $r = \frac{nd}{ns}$ such that nd and ns are the number of pairs (i, j) of different classes and similar pairs respectively. Finally, we used a Frobenius norm regularizer for $r(\boldsymbol{\Psi})$.

The objective function in Eq 10 controls the involvement of the constraints c_k by the term multiplied by λ_1, which controls its importance; we call it $C_{l,u}(\boldsymbol{\Psi})$. While, the trained classifiers penalty is captured by the term multiplied by λ_2; we call it $C_\beta(\boldsymbol{\Psi})$. One important observation on $C_\beta(\boldsymbol{\Psi})$, is that it reaches zero when $\boldsymbol{\Psi} = \mathbf{G}^{-1}\mathbf{B}^\mathsf{T}$, where $\mathbf{B} = [\boldsymbol{\beta}_1 \cdots \boldsymbol{\beta}_{N_{sc}}]$, since it could be rewritten as $C_\beta(\boldsymbol{\Psi}) = \|\mathbf{B}^\mathsf{T} - \mathbf{G} \boldsymbol{\Psi}\|_F^2$.

We minimize $L(\boldsymbol{\Psi})$ is by gradient-based optimization using a quasi-Newton optimizer. Our gradient derivation of $L(\boldsymbol{\Psi})$ leads to the following form

$$\frac{\delta L(\boldsymbol{\Psi})}{\delta \boldsymbol{\Psi}} = \boldsymbol{\Psi} + \lambda_1 \cdot \sum_{i,j} \mathbf{g}(t_i)\mathbf{k}(x_j)^\mathsf{T} v_{ij} + r \cdot \lambda_2 \cdot (\mathbf{G}^2 \boldsymbol{\Psi} - \mathbf{GB}) \quad (11)$$

where $v_{ij} = -2 \cdot max(0, (l - \mathbf{g}(t_i)^\mathsf{T} \boldsymbol{\Psi} \mathbf{k}(x_j))$ if i and j correspond to the same class, $2 \cdot max(0, (\mathbf{g}(t_i)^\mathsf{T} \boldsymbol{\Psi} \mathbf{k}(x_j) - u)$ otherwise. Another approach that can be used to minimize $L(\boldsymbol{\Psi})$ is through alternating projection using Bregman algorithm [48], where $\boldsymbol{\Psi}$ is updated by a single constraint every iteration.

B. Kernel Classifier Prediction

We study two ways to infer the final kernel-classifier prediction. (1) Direct Kernel Domain Transfer Prediction, denoted by "DT-kernel", (2) One-class SVM adjusted DT Prediction, denoted by "SVM-DT kernel". Hyper-parameter selection is attached in the supplementary materials. The source code is available here https://sites.google.com/site/mhelhoseiny/computer-vision-projects/write_kernel_classifier.

Direct Domain Transfer (DT) Prediction: By construction a classifier of an unseen class can be directly computed from our trained domain transfer model as follows

$$\Phi(t_*) = \tilde{\boldsymbol{\beta}}_{DT}(t_*) = \boldsymbol{\Psi}^{*\mathsf{T}} \mathbf{g}(t_*) \quad (12)$$

One-class-SVM adjusted DT (SVM-DT) Prediction: In order to increase separability against seen classes, we adopted the inverse of the idea of the one class kernel-svm, whose main idea is to build a confidence function that takes only positive examples of the class. Our setting is the opposite scenario; seen examples are negative examples of the unseen class. In order introduce our proposed adjustment method, we start by presenting the one-class SVM objective function. The Lagrangian dual of the one-class SVM [49] can be written as

$$\boldsymbol{\beta}_+^* = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} [\boldsymbol{\beta}^\mathsf{T}\mathbf{K}'\boldsymbol{\beta} - \boldsymbol{\beta}^T\mathbf{a}]$$
$$s.t. : \boldsymbol{\beta}^T \mathbf{1} = 1, 0 \le \boldsymbol{\beta}_i \le C; i = 1 \cdots N \quad (13)$$

where $\mathbf{K}'$ is an $N \times N$ matrix, $\mathbf{K}'(i,j) = k(x_i, x_j)$, $\forall x_i, x_j \in \mathcal{S}_x$ (i.e. in the training data), $\mathbf{a}$ is an $N \times 1$ vector, $\mathbf{a}_i = k(x_i, x_i)$, C is a hyper-parameter. It is straightforward to see that, if β is aimed to be a negative decision function instead, the objective function becomes in the form

$$\boldsymbol{\beta}_-^* = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} [\boldsymbol{\beta}^\mathsf{T}\mathbf{K}'\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{a}]$$
$$s.t. : \boldsymbol{\beta}^T \mathbf{1} = -1, -C \le \boldsymbol{\beta}_i \le 0; i = 1 \cdots N \quad (14)$$

While $\boldsymbol{\beta}_-^* = -\boldsymbol{\beta}_+^*$, the objective function in Eq 14 of the one-negative class SVM inspires us with the idea to adjust the kernel-classifier parameters to increase separability of the unseen kernel-classifier against the points of the seen classes, which leads to the following objective function

$$\Phi(t_*) = \hat{\boldsymbol{\beta}}(t_*) = \underset{\boldsymbol{\beta}}{\operatorname{argmin}} [\boldsymbol{\beta}^\mathsf{T}\mathbf{K}'\boldsymbol{\beta} - \zeta\hat{\boldsymbol{\beta}}_{DT}(t_*)^\mathsf{T}\mathbf{K}'\boldsymbol{\beta} + \boldsymbol{\beta}^T\mathbf{a}]$$
$$s.t. : \boldsymbol{\beta}^T\mathbf{1} = -1, \hat{\boldsymbol{\beta}}_{DT}^\mathsf{T}\mathbf{K}'\boldsymbol{\beta} > l, -C \le \boldsymbol{\beta}_i \le 0; \forall i$$
$$C, \zeta, l : \text{hyper-parameters}, \quad (15)$$

where $\hat{\boldsymbol{\beta}}_{DT}$ is the first N elements in $\tilde{\boldsymbol{\beta}}_{DT}(t^*) \in \mathbb{R}^{N+1}$, $\mathbf{1}$ is an $N \times 1$ vector of ones. The objective function, in Eq 8, pushes the classifier of the unseen class to be highly correlated with the domain transfer prediction of the kernel classifier, while putting the points of the seen classes as negative examples. It is not hard to see that Eq 15 is a quadratic program in β, which could be solved using any quadratic solver. It is worth to mention that, linear classifier prediction in Eq 8 (best Linear formulation in our results) predicts classifiers by solving an optimization problem of size $N + d_v + 1$ variables, $d_v + 1$ linear-classifier parameters and N slack variables. In contrast, the kernelized objective function (Eq 15) solves a quadratic program of only N variables, and predicts a kernel-classifier instead with fewer parameters. Using very high-dimensional features will not affect the optimization complexity.

VII. DISTRIBUTIONAL SEMANTIC (DS) KERNEL FOR TEXT DESCRIPTIONS

We propose a distributional semantic kernel $g(\cdot, \cdot) = g_{DS}(\cdot, \cdot)$ to define the similarity between two text descriptions in $\mathcal{T}$ domain. While this kernel is applicable to kernel classifier predictors presented in Sec VI, it could be used for other applications. We start by distributional semantic models by [50], [51] to represent the semantic manifold $\mathcal{M}_s$, and a function $vec(\cdot)$ that maps a word to a $K \times 1$ vector in $\mathcal{M}_s$. The main assumption behind this class of distributional semantic model is that similar words share similar context. Mathematically speaking, these models learn a vector for each word w_n, such that $p(w_n | (w_{n-L}, w_{n-L+1}, \cdots, w_{n+L-1}, w_{n+L})$ is maximized over the training corpus, where $2 \times L$ is the context window size. Hence similarity between $vec(w_i)$ and $vec(w_j)$ is high if they co-occurred a lot in context of size $2 \times L$ in the training text-corpus. We normalize all the word vectors to length 1 under L2 norm, i.e., $\|vec(\cdot)\|^2 = 1$.

Let us assume a text description D that we represent by a set of triplets $D = \{(w_l, f_l, vec(w_l)), l = 1 \cdots M\}$, where w_l is a word that occurs in D with frequency f_l and its corresponding word vector is $vec(w_l)$ in $\mathcal{M}_s$. We drop the stop words from D. We define $\mathbf{F} = [f_1, \cdots, f_M]^\mathsf{T}$ and $\mathbf{P} = [vec(w_1), \cdots, vec(w_M)]^\mathsf{T}$, where $\mathbf{F}$ is an $M \times 1$ vector of term frequencies and $\mathbf{P}$ is an $M \times K$ matrix of the corresponding term vectors.

Given two text descriptions D_i and D_j which contains M_i and M_j terms respectively. We compute $\mathbf{P}_i$ ($M_i \times 1$) and $\mathbf{V}_i$ ($M_i \times K$) for D_i and $\mathbf{P}_j$ ($M_j \times 1$) and $\mathbf{V}_j$ ($M_j \times K$) for D_j. Finally $g_{DS}(D_i, D_j)$ is defined as

$$g_{DS}(D_i, D_j) = \mathbf{F}_i^\mathsf{T}\mathbf{P}_i\mathbf{P}_j^\mathsf{T}\mathbf{F}_j \quad (16)$$

One advantage of this similarity measure is that it captures semantically related terms. It is not hard to see that the standard Term Frequency (TF) similarity could be thought as a special case of this kernel where $vec(w_l)^\mathsf{T} vec(w_m) = 1$ if $w_l = w_m$, 0 otherwise, i.e., different terms are orthogonal. However, in our case the word vectors are learnt through a distributional semantic model which makes semantically related terms have higher dot product $(vec(w_l)^\mathsf{T} vec(w_m))$.

VIII. Experiments

A. Datasets and Features

Datasets: We used the CU200 Birds [13] (200 classes - 6033 images) and the Oxford Flower-102 [7] (102 classes - 8189 images) image dataset to test our methods, since they are among the largest and widely used fine-grained datasets. We created textual descriptions for each class in both datasets. The CUB200 Birds image dataset was created based on birds that have a corresponding Wikipedia article, so we have developed a tool to automatically extract Wikipedia articles given the class name. The tool succeeded to automatically generate 178 articles, and the remaining 22 articles was extracted manually from Wikipedia. These mismatches happens only when article title is a different synonym of the same bird class. On the other hand, Flower image dataset was not created using the same criteria as the Bird dataset, so classes of the Flower dataset classes does not necessarily have corresponding Wikipedia article. The tool managed to generate only 16 classes from Wikipedia out of 102, the remaining 86 articles was generated manually for each class from Wikipedia, Plant Database [4], Plant Encyclopedia [5], and BBC articles [6]. The collected textual description for FLower and Birds dataset are available here https://sites.google.com/site/mhelhoseiny/1202-Elhoseiny-sup.zip .

Textual Feature Extraction: The textual features were extracted in two phases, which are typical in document retrieval literature. The first phase is an indexing phase that generates textual features with tf-idf (Term Frequency-Inverse Document Frequency) configuration (Term frequency as local weighting while inverse document frequency as a global weighting). The tf-idf is a measure of how important is a word to a text corpus. The tf-idf value increases proportionally to the number of times a word appears in the document, but is offset by the frequency of the word in the corpus, which helps to control for the fact that some words are generally more common than others. We used the normalized frequency of a term in the given textual description [52]. The inverse document frequency is a measure of whether the term is common; in this work we used the standard logarithmic idf [52]. The second phase is a dimensionality reduction step, in which Clustered Latent Semantic Indexing (CLSI) algorithm [53] is used. CLSI is a low-rank approximation approach for dimensionality reduction, used for document retrieval. In the Flower Dataset, tf-idf features $\in \mathbb{R}^{8875}$ and after CLSI the final textual features

$\in \mathbb{R}^{102}$. In the Birds Dataset, tf-idf features is in $\mathbb{R}^{7086}$ and after CLSI the final textual features is in $\mathbb{R}^{200}$.

Visual features Extraction: We used the Classeme features [54] as the visual feature for our experiments since they provide an intermediate semantic representation of the input image. Classeme features are output of a set of classifiers corresponding to a set of C category labels, which are drawn from an appropriate term list defined in [54], and not related to our textual features. For each category $c \in \{1 \cdots C\}$, a set of training images is gathered by issuing a query on the category label to an image search engine. After a set of coarse feature descriptors (Pyramid HOG, GIST, etc.) is extracted, a subset of feature dimensions was selected [54], and a one-versus-all classifier φ_c is trained for each category. The classifier output is real-valued, and is such that $\varphi_c(x) > \varphi_c(y)$ implies that x is more similar to class c than y is. Given an image x, the feature vector (descriptor) used to represent it is the classeme vector $[\varphi_1(x), \cdots, \varphi_{d_v}(x)]$, $d_v = 2569$.

For Kernel classifier prediction, we evaluated these features and also additional representations for text descriptions (by the proposed distributional semantic kernel using word embedding) and images ((a) CNN features and (b) combined kernel over different features learnt by MKL (multiple kernel learning)), discussed later in Subsection VIII-C.

B. Experimental Results for Linear Classifier Prediction

Evaluation Methodology: Similar to zero-shot learning literature, we evaluated the performance of an unseen classifier in a one-vs-all setting where the test images of unseen classes are considered to be the positives and the test images from the seen classes are considered to be the negatives. We computed the ROC curve and report the area under that curve (AUC) as a comparative measure of different approaches. In zero-shot learning setting the test data from the seen class are typically very large compared to those from unseen classes. This makes other measures, such as accuracy, useless since high accuracy can be obtained even if all the unseen class test data are wrongly classified; hence we used ROC curves, which are independent of this problem. Five-fold cross validation over the classes were performed, where in each fold 4/5 of the classes are considered as "seen classes" and are used for training and 1/5th of the classes were considered as "unseen classes" where their classifiers are predicted and tested. Within each of these class-folds, the data of the seen classes are further split into training and test sets. The hyper-parameters for the approach were selected through another five-fold cross validation within the class-folds (i.e. the 80% training classes are further split into 5 folds to select the hyper-parameters). We made the seen-unseen folds used in our experiments available here https://sites.google.com/site/mhelhoseiny/computer-vision-projects/Write_a_Classifier.

Baselines: Since our work is the first to predict classifiers based on pure textual description, there are no other reported results to compare against. However, we designed three state-of-the-art baselines to compare against, which are designed to be inline with our argument in Sec III. Namely we used: 1) A Gaussian Process Regressor (GPR) [42], 2) Twin Gaussian

---

[4]http://plants.usda.gov/java/
[5]http://www.theplantencyclopedia.org/wiki/Main_Page
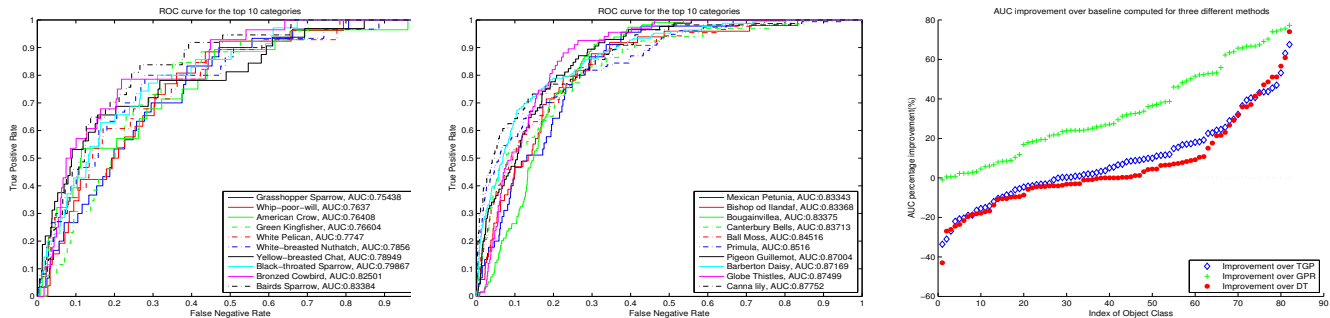[6]http://www.bbc.co.uk/science/0/

Fig. 4: Linear : **Left and Middle**: ROC curves of best 10 predicted classes by the final formulation (E) for Bird and Flower datasets respectively, **Right**: AUC improvement over the three baselines on Flower dataset (Formulations A (GPR), A (TGP), C). The improvement is sorted in an increasing order for each baseline separately (best seen in color)

TABLE II: Linear: Percentage of classes that the final pr[...] approach (formulation (E)) makes an improvement in predictin[...] the baselines (relative to the total number of classes in each d[...]

baseline	Flowers (102) % improvement	Birds (200) % improvement
(A) GPR	100 %	98.31 %
(A) TGP	66 %	51.81 %
(C) DT	54%	56.5%

Process (TGP) [47] as a structured regression method, 3) Domain Transfer (DT) [45]. The TGP and DT baselines are of particular importance since our formulation utilizes them, so we need to test if the formulation is making any improvement over them. It has to be noted that we also evaluate TGP and DT as alternative formulations that we are proposing for the problem, none of them was used in the same context before.

Results: Table I shows the average AUCs for the final linear approach in comparison to the three baselines on both datasets. GPR performed poorly in all classes in both data sets, which was expected since it is not a structure prediction approach. The DT formulation outperformed TGP in the flower dataset but slightly underperformed on the Bird dataset. The proposed approach outperformed all the baselines on both datasets, with significant difference on the flower dataset. It is also clear that the TGP performance was improved on the Bird dataset since it has more classes (more points are used for prediction). Fig 4 shows the ROC curves for our approach on best predicted unseen classes from the Birds dataset on the Left and Flower dataset on the middle. Fig 5 shows the AUC for all the classes on Flower dataset.

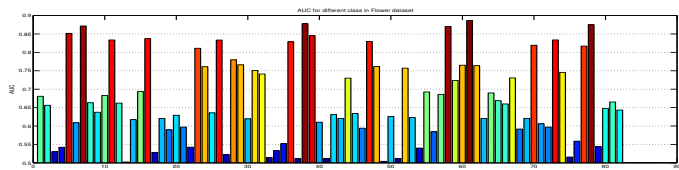Fig 4, on the right, shows the improvement over (A) GPR,

Fig. 5: Linear: AUC of the predicated classifiers for all classes of the flower datasets (Formulation E)

TABLE III: Linear: Top-5 classes with highest combined improvement in Flower dataset

class	(A) TGP (AUC)	(C) DT (AUC)	(E) Our (AUC)	% Improv.
2	0.51	0.55	0.83	57%
28	0.52	0.54	0.76	43.5%
26	0.54	0.53	0.76	41.7%
81	0.52	0.82	0.87	37%
37	0.72	0.53	0.83	35.7 %

A(TGP), and (C) DT for each class, where the improvement is calculated as (our AUC- baseline AUC)/ baseline AUC %. Table II shows the percentage of the classes which our approach makes a prediction improvement for each of the three baselines. Table III shows the five classes in Flower dataset where our approach made the best average improvement. The point of that table is to show that in these cases both TGP and DT did poorly while our formulation that is based on both of them did significantly better. This shows that our formulation does not simply combine the best of the two approaches but can significantly improve the prediction performance.

To evaluate the effect of the constraints in the objective function, we removed the constraints $-(\mathbf{c}^\mathsf{T}\mathbf{x}_i) \geq \zeta_i$ which try to enforces all the seen examples to be on the negative side of the predicted classifier hyperplane and evaluated the

TABLE I: Linear: Comparative Evaluation of Different Formulations on the Flower and Bird Datasets

Approach	Oxford Flowers Avg AUC (+/- std)	UC-UCSD Birds Avg AUC (+/- std)
(A) Regression - GPR	0.54 (+/- 0.02)	0.52 (+/- 0.001)
(A) Structured Regression - TGP	0.58 (+/- 0.02)	0.61 (+/- 0.02)
(C) Domain Transfer(DT)	0.62(+/- 0.03)	0.59 (+/- 0.01)
(B) Constrained GPR	0.62(+/- 0.005)	-
(B) Constrained TGP	0.63(+/- 0.007)	-
(D) Constrained Domain Adaptation (CDT) on Eq 8	0.64 (+/- 0.006)	-
(E) Regression+DT + constraints (final best linear approach)	0.68 (+/- 0.01)	0.62 (+/- 0.02)

approach. The result on the flower dataset (using one fold) was reduced to average AUC=0.59 compared to AUC=0.65 with the constraints. Similarly, we evaluated the effect of the constraint $\mathbf{t}_*^\mathsf{T}\mathbf{W}\mathbf{c} \geq l$. The result was reduced to average AUC=0.58 compared to AUC=0.65 with the constraint. This illustrates the importance of this constraint in the formulation. **Constrained Baselines:** Table I (bottom three lines) also shows the average AUCs for the constrained baselines formulations, namely Constrained GPR, TGP and DT; see section V. Even though the visual features and textual features were independently extracted, by learning correlation between them, we can predict classifiers for new categories. As shown previously, GPR performed poorly, while, as expected, TGP performed better. Adding constraints to GPR/TGP improved their performance. Combining regression and DT gave significantly better results for classes where both approaches individually perform poorly, as can be seen in Table I-right. We performed an additional experiment, where $\mathbf{W}$ is firstly computed using Constrained Domain Transfer (CDT). Then, the unseen classifier is predicted using equation 8 with $\gamma = 0$, which performs worse. This indicates that adding constraints to align to seen classifiers hurts the learnt domain transfer function on unseen classes. In conclusion, the final formulation that combines TGP and DT with additional constraints performs the best in both Birds and Flower datasets, where the effect of TGP is very limited since it was trained on sparse points.

C. Experimental Results for Kernel Classifier Prediction

1) Additional Evaluation Metrics: In addition to the the AUC, we already studied in the previous section. We report two additional metrics while evaluating and comparing the kernel classifier prediction to the linear classifier prediction, detailed as follows.

$|N_{sc}|$ *to* $|N_{sc}+1|$ *Recall:* Under this metric, we aim to check how the learned classifiers of the seen classes confuse the predicted classifiers, when they are involved in a multi-class classification problem of $N_{sc}+1$ classes. We use Eq 1 to predict label l^* with the maximum confidence of an image x^*, such that $l^* \in L_{sc} \cup l_{us}$, l_{us} is the label of the ground truth unseen class, and L_{sc} is the set of seen class labels. We compute the recall under this setting. This metric is computed for each predicted unseen classifier and the average is reported.

Multiclass Accuracy of Unseen classes (MAU): Under this setting, we aim to evaluate the performance of the unseen classifiers against each others. Firstly, the classifiers of all unseen categories are predicted. Then, we use Eq 1 to predict the label with the maximum confidence of a test image x, such that its label $l_{us}^* \in L_{us}$, where L_{us} is the set of all unseen class labels that only have text descriptions.

2) Comparisons to Linear Classifier Prediction: We compare the kernel methods to the linear prediction discussed earlier, which predicts a linear classifier from textual descriptions ($\mathcal{T}$ space in our framework). The aspects of the comparison includes 1) whether the predicted kernelized classifier outperforms the predicted linear classifier 2) whether this behavior is consistent on multiple datasets. We performed the comparison on both Birds and Flower dataset. For these experiments,

in our setting, domain $\mathcal{V}$ is the visual domain and domain $\mathcal{T}$ is the textual domain, i.e., the goal is to predict classifiers from pure textual description. We used the same features on the visual domain and the textual domains detailed in subsection VIII-A. That is, classeme features [54] for images, extracted from images of the Bird and the Flower datasets and tf-idf [52] features for text articles followed by a CLSI [53] dimensionality reduction phase.

We denote our kernel Domain Transfer prediction and one class SVM adjust DT prediction by "DT-kernel" and "SVM-DT-kernel" respectively. We compared against linear classifier prediction (Linear Formulation (E) approach, denoted by just Linear Classifier). We also compared against the linear direct domain transfer (Linear Formulation (C), denoted by DT-linear). In our kernel approaches, we used Gaussian rbf-kernel as a similarity measure in $\mathcal{T}$ and $\mathcal{V}$ spaces (i.e. $k(d, d') = exp(-\lambda||d - d'||)$).

Recall metric : The recall of the SVM-DT kernel approach is 44.05% for Birds and 40.34% for Flower, while it is 36.56% for Birds and 31.33% for Flower by best Linear Classifier prediction (E). This indicates that the predicted classifier is less confused by the classifiers of the seen compared with Linear Classifier prediction; see table IV (top part)

MAU metric: It is worth to mention that the multiclass accuracy for the trained seen classifiers is 51.3% and 15.4%, using the classeme features, on Flower dataset and Birds dataset[7], respectively. Table IV (middle part) shows the average MAU metric over three seen/unseen splits for Flower dataset and one split on Birds dataset, respectively. Furthermore, the relative improvements of our SVM-DT-kernel approach is reported against the baselines. On Flower dataset, it is interesting to see that our approach achieved 9.1% MAU, 182% the random guess performance, by predicting the unseen classifiers using just textual features as privileged information (i.e. $\mathcal{T}$ domain). It is important to mention that we achieved also 13.4%, 268% the random guess performance, in one of the splits (the 9.1% is the average over 3 seen/unseen splits). Similarity on Birds dataset, we achieved 3.4% MAU from text features, 132% the random guess performance (further improved up to 224% in next experiments).

AUC metric: Fig 6 (top part) shows the ROC curves for our approach on the best predicted unseen classes from the Flower dataset. Fig 6 (bottom part) shows the AUC for all the classes on Flower dataset (over three different splits). Table IV (bottom part) shows the average AUC on the two datasets, compared to the baselines. More results and figures for our kernel approach are attached in the supplementary materials.

Looking at table IV, we can notice that the proposed approach performs marginally similar to the baselines from AUC perspective. However, there is a clear improvement in MAU and Recall metrics. These results show the advantage of predicting classifiers in kernel space. Furthermore, the table shows that our SVM-DT-kernel approach outperforms our DT-kernel model. This indicates the advantage of the class separation, which is adjusted by the SVM-DT-kernel model.

---

[7]Birds dataset is known to be a challenging dataset for fine-grained, even when applied in a regular multiclass setting as it is clear from the 15.4% performance on seen classes
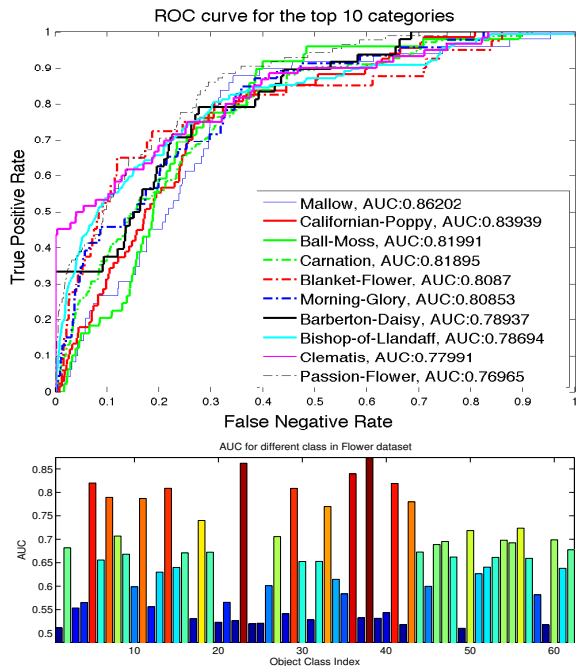
Fig. 6: Kernel: AUC of the 62 unseen classifiers the flower data-sets over three different splits (bottom part) and their Top 10 ROC-curves (top part)

More details on the hyper-parameter selection are attached in the supplementary materials.

3) Multiple Kernel Learning (MKL) Experiment: This experiment shows the added value of proposing a kernelized zero-shot learning approach. We conducted an experiment where the final kernel on the visual domain is produced by Multiple Kernel Learning [56]. For the visual domain, we extracted kernel descriptors for Birds dataset. Kernel descriptors provide a principled way to turn any pixel attribute to patch-level features, and are able to generate rich features from various recognition cues. We specifically used four types of kernels introduced by [57] as follows: *Gradient Match Kernels* that captures image variation based on predefined kernels on image gradients. *Color Match Kernel* that describes patch appearance using two kernels on top of RGB and normalized RGB for regular images and intensity for grey images. These kernels capture image variation and visual apperances. For modeling the local shape, *Local Binary Pattern* kernels

have been applied. We computed these kernel descriptors on local image patches with fixed size 16 x 16 sampled densely over a grid with step size 8 in a spatial pyramid setting with four layers. The dense features are vectorized using codebooks of size 1000. This process ended up with a 120,000 dimensional feature for each image (30,000 for each type). Having extracted the four types of descriptors, we compute an rbf kernel matrix for each type separately. We learn the bandwidth parameters for each rbf kernel by cross validation on the seen classes. Then, we generate a new kernel $k_{mkl}(d, d') = \sum_{i=1}^{4} w_i k_i(d, d')$, such that w_i is a weight assigned to each kernel. We learn these weights by applying Bucak's Multiple Kernel Learning algorithm [58]. Then, we applied our approach where the MKL-kernel is used in the visual domain and rbf kernel in the text domain similar to the previous experiments.

To compare the kernel prediction approach to the linear prediction approach (formulation (E)) under this setting, we concatenated all kernel descriptors to end up with 120,000 dimensional feature vector in the visual domain. As highlighted in the kernel approach section, the linear prediction approach solves a quadratic program of $N + d_v + 1$ variables for each unseen class. Due to the large dimensionality of data ($d_v = 120,000$), this is not tractable. To make this setting applicable, we reduced the dimensionality of the feature vector into 4000 using PCA. This highlights the benefit of our approach since our quadratic program does not depend on the dimensionality of the data. Table V shows MAU for the kernel prediction approaches under this setting against linear prediction. The results show the benefits of having a kernel prediction for zero shot learning where kernel methods to construct an arbitrary kernel that improves the performance.

D. Multiple Representation Experiment and Distributional Semantic(DS) Kernel

The aim of this experiment is to show that the kernel approach perform on different representations of text $\mathcal{T}$ and visual domains $\mathcal{V}$. In this experiment, we extracted Convolutional Neureal Network(CNN) image features for the Visual domain. We used caffe [59] implementation of [60]. Then, we extracted the sixth activation feature of the CNN (FC6) since we found it works the best on the standard classification setting. We found this consistent with the results of [61] over different CNN layers. While using TFIDF feature of

TABLE IV: Kernel: Recall, MAU, and average AUC on three seen/unseen splits on Flower Dataset and a seen/unseen split on Birds dataset

	Recall-Flower	improvement	Recall-Birds	improvement
SVM-DT kernel-rbf	**40.34% (+/- 1.2) %**		**44.05 %**	
Linear Classifier	31.33 (+/- 2.22)%	27.8 %	36.56 %	20.4 %
	MAU-Flower	improvement	MAU-Birds	improvement
SVM-DT kernel-rbf	**9.1 (+/- 2.77) %**		**3.4 %**	
DT kernel-rbf	**6.64 (+/- 4.1) %**	37.93 %	**2.95 %**	15.25 %
Linear Classifier	5.93 (+/- 1.48)%	54.36 %	2.62 %	29.77 %
DT-linear	5.79 (+/- 2.59)%	58.46 %	2.47 %	37.65 %
	AUC-Flower	improvement	AUC-Birds	improvement
SVM-DT kernel-rbf	0.653 (+/- 0.009)		0.61	
DT kernel-rbf	0.623 (+/- 0.01) %	4.7 %	0.57	7.02 %
Linear Classifier	0.658 (+/- 0.034)	- 0.7 %	0.62	-1.61%
Domain Transfer	0.644 (+/- 0.008)	1.28 %	0.56	8.93%

TABLE V: Kernel: MAU on a seen-unseen split-Birds Dataset (MKL)

	MAU	improvement
SVM-DT kernel-rbf (text)	**4.10 %**	
Linear Classifier	2.74 %	49.6 %

TABLE VI: Kernel: MAU on a seen-unseen split-Birds Dataset (CNN image features, text description)

	MAU	improvement
SVM-DT kernel ($\mathcal{V}$-rbf, $\mathcal{T}$-DS kernel)	**5.35 %**	
SVM-DT kernel ($\mathcal{V}$-rbf, $\mathcal{T}$-rbf on TFIDF)	**4.20 %**	27.3%
Linear Classifier (TFIDF text)	2.65 %	102.0%
[55]	2.3%	132.6%

TABLE VII: Kernel: Recall and MAU on a seen-unseen split-Birds Dataset (Attributes)

	Recall	improvement
SVM-DT kernel-rbf	**76.7 %**	
Lampert DAP [5]	68.1 %	12.6 %
	MAU	improvement
SVM-DT kernel-rbf	**5.6 %**	
DT kernel-rbf	4.03 %	32.7 %
Lampert DAP [5]	4.8 %	16.6 %

text description and CNN features for images, we achieved 2.65% for the linear version and 4.2% for the rbf kernel on both text and images. We further improved the performance to 5.35% by using our proposed Distributional Semantic (DS) kernel in the text domain and rbf kernel for images. In this DS experiment, we used the distributional semantic model by [50] trained on GoogleNews corpus (100 billion words) resulting in a vocabulary of size 3 million words, and word vectors of $K = 300$ dimensions. This experiment shows both the value of having a kernel version and also the value of the proposed kernel in our setting. We also applied the zero shot learning approach in [55] which performs worse in our settings; see Table VI.

1) Attributes Experiment: We emphasis that our goal from this experiment is not attribute prediction. However, it was interesting for us to see the behavior of our method where $\mathcal{T}$ space is defined from attributes instead of text. In contrast to attribute-based models, which fully utilize attribute information to build attribute classifiers, we do not learn attribute classifiers. In this experiment, our method uses only the first moment of information of the attributes (i.e. the average attribute vector). We decided to compare to an attribute-based approach from this perspective. In particular, we applied the (DAP) attribute-based model [62], [5], widely adopted in many applications (*e.g.,* [63], [64]), to the Birds dataset.

For this experiment, we compute the average attributes provided with images for each category, without learning any binary classifiers. We computed this as $\mathcal{T}$ domain representation for the Birds dataset only, since the Flower dataset does not have attributes. Attribute annotation of Birds dataset is provided in terms of: "Visibility" and "Certainty". The first term is 1 if the attribute is visible and 0 otherwise. The second term indicates how certain the annotator was about his decision with three levels of "Certain", "Guessing" and "probable". We assigned probability of a visible attribute when the annotator is sure about his decision to 1 and when he is sure of not seeing the attribute to 0. All other combination of decisions fall in the range of $(0, 1)$. As this attribute annotation is provided for each images, we averaged these scores across all the samples of a class to get a single attribute descriptor for each class, to be consistent with our learning setting. For visual domain, we used classeme features in this experiment (as table IV experiment).

An interesting result is that our approach achieved 5.6% MAU (224% the random guess performance); see Table VII. In contrast, we get 4.8% multiclass accuracy using DAP approach [62]. In this setting, we also measured the N_{sc} to $N_{sc} + 1$ average recall. We found the recall measure is 76.7% for our SVM-DT-kernel, while it is 68.1% on the DAP

approach, which reflects better true positive rate (positive class is the unseen one). We find these results interesting, since we achieved it without learning any attribute classifiers, as in [62]. When comparing the results of our approach using attributes (Table VII) vs. textual description (Table IV)[8] as the $\mathcal{T}$ space used for prediction, it is clear that the attribute features gives better prediction. This support our hypothesis that the more meaningful the $\mathcal{T}$ domain, the better the performance on $\mathcal{V}$ domain. This indicates that if a better textual representation is used, a better performance can be achieved. Attributes are a good semantic representation of a class but it is difficult to define attributes for an arbitrary class and further measure the confidence of each one. In contrast, it is much easier to find an unstructured text description for visual class.

IX. CONCLUSION

We explored the problem of predicting visual classifiers from textual description of classes with no training images. We investigated and experimented with different formulations for the problem within the fine-grained categorization context. We first proposed a novel formulation that captures information between the visual and textual domains by involving knowledge transfer from textual features to visual features, which indirectly leads to predicting a linear visual classifier described by the text. We also proposed a new zero-shot learning technique to predict kernel-classifiers of unseen categories using information from a privilege space. We formulated the problem as domain transfer function from text description to the visual classification space, while supporting kernels in both domains. We proposed a one-class SVM adjustment to our domain transfer function in order to improve the prediction. We validated the performance of our model by several experiments. We also showed that our approach using with weak-attributes. We illustrated the value of proposing a kernelized version by applying kernels generated by Multiple Kernel Learning (MKL) and achieved better results. In the future, we aim to improve this model by learning the unseen classes jointly and on a larger scale.

REFERENCES

[1] R. Salakhutdinov, A. Torralba, and J. B. Tenenbaum, "Learning to share visual appearance for multiclass object detection," in *CVPR*, 2011.
[2] J. Deng, A. C. Berg, K. Li, and L. Fei-Fei, "What does classifying more than 10,000 image categories tell us?" in *ECCV*, 2010.
[3] L. Fe-Fei, R. Fergus, and P. Perona, "A bayesian approach to unsupervised one-shot learning of object categories," in *CVPR*. IEEE, 2003, pp. 1134–1141.
[4] E. Bart and S. Ullman, "Cross-generalization: Learning novel classes from a single example by feature replacement," in *CVPR*, 2005.
[5] C. H. Lampert, H. Nickisch, and S. Harmeling, "Learning to detect unseen object classes by betweenclass attribute transfer," in *In CVPR*, 2009.
[6] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," 2011.
[7] M.-E. Nilsback and A. Zisserman, "Automated flower classification over large number of classes," in *ICVGIP*, 2008.

[8]We are refering to the experiment that uses classeme as visual features to have a consistent comparison to here

[8] J. Wang, K. Markert, and M. Everingham, "Learning models for object recognition from natural language descriptions." in *BMVC*, 2009.

[9] J. Liu, A. Kanazawa, D. Jacobs, and P. Belhumeur, "Dog breed classification using part localization," in *ECCV*, 2012.

[10] S. Maji, E. Rahtu, J. Kannala, M. Blaschko, and A. Vedaldi, "Fine-grained visual classification of aircraft," *arXiv preprint arXiv:1306.5151*, 2013.

[11] M. Elhoseiny, B. Saleh, and A. Elgammal, "Write a classifier: Zero shot learning using purely text descriptions," in *ICCV*, 2013.

[12] B. Schölkopf, R. Herbrich, and A. J. Smola, "A generalized representer theorem," in *COLT*, 2001.

[13] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona, "Caltech-UCSD Birds 200," California Institute of Technology, Tech. Rep., 2010.

[14] B. Saleh, A. Farhadi, and A. Elgammal, "Object-centric anomaly detection by attribute-based reasoning," in *CVPR*, 2013.

[15] A. Farhadi, I. Endres, D. Hoiem, and D. A. Forsyth, "Describing objects by their attributes," in *CVPR*, 2009.

[16] E. G. Miller, N. E. Matsakis, and P. A. Viola, "Learning from one example through shared densities on transforms," in *CVPR*, 2000.

[17] M. Fink, "Object classification from a single example utilizing class relevance metrics," in *NIPS*, 2004.

[18] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *NIPS*, 2009.

[19] M. Rohrbach, M. Stark, G. Szarvas, and B. Schiele, "Combining language sources and robust semantic relatedness for attribute-based knowledge transfer," in *Parts and Attributes Workshop at ECCV*, 2010.

[20] D. Parikh and K. Grauman, "Interactively building a discriminative vocabulary of nameable attributes," in *CVPR*, 2011.

[21] S. Huang, M. Elhoseiny, A. Elgammal, and D. Yang, "Learning hypergraph-regularized attribute predictors," in *CVPR*, 2015.

[22] G. Griffin and P. Perona, "Learning and using taxonomies for fast visual categorization," in *CVPR*. IEEE, 2008, pp. 1–8.

[23] R. Fergus, H. Bernal, Y. Weiss, and A. Torralba, "Semantic label sharing for learning with many categories," in *Computer Vision–ECCV 2010*. Springer, 2010, pp. 762–775.

[24] G. A. Miller, "Wordnet: A lexical database for english," *COMMUNICATIONS OF THE ACM*, 1995.

[25] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *CVPR*, 2009.

[26] A. Torralba, R. Fergus, and W. T. Freeman, "80 million tiny images: A large data set for nonparametric object and scene recognition," *PAMI*, 2008.

[27] K. Barnard, P. Duygulu, and D. Forsyth, "Clustering art," in *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, vol. 2. IEEE, 2001, pp. II–434.

[28] A. Farhadi, M. Hejrati, M. A. Sadeghi, P. Young, C. Rashtchian, J. Hockenmaier, and D. Forsyth, "Every picture tells a story: Generating sentences from images," in *ECCV*, 2010.

[29] G. Kulkarni, V. Premraj, S. Dhar, S. Li, Y. Choi, A. C. Berg, and T. L. Berg, "Baby talk: Understanding and generating simple image descriptions," in *CVPR*, 2011.

[30] Y. Yang, C. L. Teo, H. Daumé III, and Y. Aloimonos, "Corpus-guided sentence generation of natural images," in *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2011, pp. 444–454.

[31] N. Krishnamoorthy, G. Malkarnenkar, R. Mooney, K. Saenko, U. Lowell, and S. Guadarrama, "Generating natural-language video descriptions using text-mined knowledge," *NAACL HLT 2013*, p. 10, 2013.

[32] V. Ordonez, G. Kulkarni, and T. L. Berg, "Im2text: Describing images using 1 million captioned photographs," in *NIPS*, 2011.

[33] A. Karpathy and L. Fei-Fei, "Deep visual-semantic alignments for generating image descriptions," in *CVPR*, 2015.

[34] O. Vinyals, A. Toshev, S. Bengio, and D. Erhan, "Show and tell: A neural image caption generator," 2015.

[35] K. Xu, J. Ba, R. Kiros, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, attend and tell: Neural image caption generation with visual attention," in *ICML*, 2015.

[36] J. Mao, W. Xu, Y. Yang, J. Wang, and A. Yuille, "Deep captioning with multimodal recurrent neural networks (m-rnn)," *ICLR*, 2015.

[37] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," in *EMNLP*, 2014.

[38] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, M. Ranzato, and T. Mikolov, "Devise: A deep visual-semantic embedding model." in *NIPS*, 2013.

[39] R. Socher, M. Ganjoo, H. Sridhar, O. Bastani, C. D. Manning, and A. Y. Ng, "Zero shot learning through cross-modal transfer," in *NIPS*, 2013.

[40] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[41] A. E. Hoerl and R. W. Kennard, "Ridge Regression: Biased Estimation for Nonorthogonal Problems," *Technometrics*, 1970.

[42] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. The MIT Press, 2005.

[43] J. Yang, R. Yan, and A. G. Hauptmann, "Cross-domain video concept detection using adaptive svms," in *MULTIMEDIA*, 2007.

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

[45] B. Kulis, K. Saenko, and T. Darrell, "What you saw is not what you get: Domain adaptation using asymmetric kernel transforms," in *CVPR*, 2011.

[46] L. Duan, D. Xu, I. W.-H. Tsang, and J. Luo, "Visual event recognition in videos by learning from web data," *TPAMI*, 2012.

[47] L. Bo and C. Sminchisescu, "Twin gaussian processes for structured prediction," *IJCV*, 2010.

[48] Y. Censor and S. Zenios, *Parallel Optimization: Theory, Algorithms, and Applications*. Oxford University Press, USA, 1997.

[49] P. F. Evangelista, M. J. Embrechts, and B. K. Szymanski, "Some properties of the gaussian kernel for one class learning," in *ICANN*, 2007.

[50] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in Neural Information Processing Systems*, 2013, pp. 3111–3119.

[51] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *ICLR*, 2013.

[52] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *IPM*, 1988.

[53] D. Zeimpekis and E. Gallopoulos, "Clsi: A flexible approximation scheme from clustered term-document matrices," in *SDM*, 2005.

[54] L. Torresani, M. Szummer, and A. Fitzgibbon, "Efficient object category recognition using classemes," in *ECCV*, 2010.

[55] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. S. Corrado, and J. Dean, "Zero-shot learning by convex combination of semantic embeddings," in *ICLR*, 2014.

[56] M. Gonen and E. Alpaydin, "Multiple kernel learning algorithms," *JMLR*, 2011.

[57] L. Bo, X. Ren, and D. Fox, "Kernel descriptors for visual recognition," in *NIPS*, 2010.

[58] S. S. Bucak, R. Jin, and A. K. Jain, "Multi-label multiple kernel learning by stochastic approximation: Application to visual object recognition." in *NIPS*, 2010.

[59] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell, "Caffe: Convolutional architecture for fast feature embedding," in *ACM Multimedia*, 2014.

[60] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems (NIPS)*, 2012.

[61] J. Donahue, Y. Jia, O. Vinyals, J. Hoffman, N. Zhang, E. Tzeng, and T. Darrell, "Decaf: A deep convolutional activation feature for generic visual recognition," in *ICML*, 2014.

[62] C. Lampert, H. Nickisch, and S. Harmeling, "Attribute-based classification for zero-shot visual object categorization," *TPAMI*, vol. 36, no. 3, pp. 453–465, March 2014.

[63] J. Liu, Q. Yu, O. Javed, S. Ali, A. Tamrakar, A. Divakaran, H. Cheng, and H. Sawhney, "Video event recognition using concept attributes," in *WACV*, 2013.

[64] M. Rohrbach, M. Stark, and B. Schiele, "Evaluating knowledge transfer and zero-shot learning in a large-scale setting," in *CVPR*, 2011.