

A Simple Exponential Family Framework for Zero-Shot Learning

Vinay Kumar Verma[#] and Piyush Rai[#]

[#]Dept. of Computer Science & Engineering, IIT Kanpur, India
{vkverma,piyush}@cse.iitk.ac.in

Abstract. We present a simple generative framework for learning to predict previously unseen classes, based on estimating *class-attribute-gated* class-conditional distributions. We model each class-conditional distribution as an exponential family distribution and the parameters of the distribution of each seen/unseen class are defined as functions of the respective observed class attributes. These functions can be learned using only the seen class data and can be used to predict the parameters of the class-conditional distribution of each unseen class. Unlike most existing methods for zero-shot learning that represent classes as fixed embeddings in some vector space, our generative model naturally represents each class as a probability distribution. It is simple to implement and also allows leveraging additional unlabeled data from unseen classes to improve the estimates of their class-conditional distributions using transductive/semi-supervised learning. Moreover, it extends seamlessly to few-shot learning by easily updating these distributions when provided with a small number of additional labelled examples from unseen classes. Through a comprehensive set of experiments on several benchmark data sets, we demonstrate the efficacy of our framework.

1 Introduction

The problem of learning to predict unseen classes, also popularly known as Zero-Shot Learning (ZSL), is an important learning paradigm which refers to the problem of recognizing objects from classes that were not seen at training time [13,26]. ZSL is especially relevant for learning “in-the-wild” scenarios, where new concepts need to be discovered on-the-fly, without having access to labelled data from the novel classes/concepts. This has led to a tremendous amount of interest in developing ZSL methods that can learn in a robust and scalable manner, even when the amount of supervision for the classes of interest is relatively scarce.

A large body of existing prior work for ZSL is based on embedding the data into a semantic vector space, where distance based methods can be applied to find the most likely class which itself is represented as a point in the same semantic space [26,20,33]. However, a limitation of these methods is that each class is represented as a fixed point in the embedding space which does not adequately account for intra-class variability [2,18]. We provide a more detailed overview of existing work on ZSL in the Related Work section.

Another key limitation of most of the existing methods is that they usually lack a proper generative model of the data. Having a generative model has several advantages [19]. For example, (1) data of different types can be modeled in a principled way

using appropriately chosen class-conditional distributions; (2) unlabeled data can be seamlessly integrated (for both seen as well as unseen classes) during parameter estimation, leading to a transductive/semi-supervised estimation procedure, which may be useful when the amount of labeled data for the seen classes is small, or if the distributions of seen and unseen classes are different from each other [11]; and (3) a rich body of work, both frequentist and Bayesian, on learning generative models [19] can be brought to bear during the ZSL parameter estimation process.

Motivated by these desiderata, we present a generative framework for zero-shot learning. Our framework is based on modelling the class-conditional distributions of seen as well as unseen classes using exponential family distributions [3], and further conditioning the parameters of these distributions on the respective class-attribute vectors via a linear/nonlinear regression model of one’s choice. The regression model allows us to predict the parameters of the class-conditional distributions of *unseen* classes using only their class attributes, enabling us to perform zero-shot learning.

In addition to the generality and modelling flexibility of our framework, another of its appealing aspects is its simplicity. In contrast with various other state-of-the-art methods, our framework is very simple to implement and easy to extend. In particular, as we will show, parameter estimation in our framework simply reduces to solving a linear/nonlinear regression problem, for which a closed-form solution exists. Moreover, extending our framework to incorporate unlabeled data from the unseen classes, or a small number of labelled examples from the unseen classes, i.e., performing few-shot learning [23,17] is also remarkably easy under our framework which models class-conditional distributions using exponential family distributions with conjugate priors.

2 A Generative Framework For ZSL

In zero-shot learning (ZSL) we assume there is a total of S seen classes and U unseen classes. Labelled training examples are only available for the seen classes. The test data is usually assumed to come only from the unseen classes, although in our experiments, we will also evaluate our model for the setting where the test data could come from both seen and unseen classes, a setting known as generalised zero-shot learning [6].

We take a generative modeling approach to the ZSL problem and model the class-conditional distribution for an observation \mathbf{x} from a seen/unseen class c ($c = 1, \dots, S+U$) using an exponential family distribution [3] with natural parameters $\boldsymbol{\theta}_c$

$$p(\mathbf{x}|\boldsymbol{\theta}_c) = h(\mathbf{x}) \exp\left(\boldsymbol{\theta}_c^\top \phi(\mathbf{x}) - A(\boldsymbol{\theta}_c)\right) \quad (1)$$

where $\phi(\mathbf{x})$ denotes the sufficient statistics and $A(\boldsymbol{\theta}_c)$ denotes the log-partition function. We also assume that the distribution parameters $\boldsymbol{\theta}_c$ are given conjugate priors

$$p(\boldsymbol{\theta}_c|\boldsymbol{\tau}_0, \boldsymbol{\nu}_0) \propto \exp(\boldsymbol{\theta}_c^\top \boldsymbol{\tau}_0 - \boldsymbol{\nu}_0 A(\boldsymbol{\theta}_c)) \quad (2)$$

Given a test example \mathbf{x}_* , its class y_* can be predicted by finding the class under which \mathbf{x}_* is most likely (i.e., $y_* = \arg \max_c p(\mathbf{x}_*|\boldsymbol{\theta}_c)$), or finding the class that has the largest posterior probability given \mathbf{x}_* (i.e., $y_* = \arg \max_c p(\boldsymbol{\theta}_c|\mathbf{x}_*)$). However, doing this requires first estimating the parameters $\{\boldsymbol{\theta}_c\}_{c=S+1}^{S+U}$ of all the unseen classes.

Given labelled training data from any class modelled as an exponential family distribution, it is straightforward to estimate the model parameters θ_c using maximum likelihood estimation (MLE), maximum-a-posteriori (MAP) estimation, or using fully Bayesian inference [19]. However, since there are no labelled training examples from the unseen classes, we cannot estimate the parameters $\{\theta_c\}_{c=S+1}^{S+U}$ of the class-conditional distributions of the unseen classes.

To address this issue, we learn a model that allows us to predict the parameters θ_c for any class c using the *attribute vector* of that class via a *gating* scheme, which is basically defined as a linear/nonlinear regression model from the attribute vector to the parameters. As is the common practice in ZSL, the attribute vector of each class may be derived from a human-provide description of the class or may be obtained from an external source such as Wikipedia in form of word-embedding of each class. We assume that the class-attribute of each class is a vector of size K . The class-attribute of all the classes are denoted as $\{\mathbf{a}_c\}_{c=1}^{S+U}$, $\mathbf{a}_c \in \mathbb{R}^K$.

2.1 Gating via Class-Attributes

We assume a regression model from the class-attribute vector \mathbf{a}_c to the parameters θ_c of each class c . In particular, we assume that the class-attribute vector \mathbf{a}_c is mapped via a function f to generate the parameters θ_c of the class-conditional distribution of class c , as follows

$$\theta_c = f_\theta(\mathbf{a}_c) \quad (3)$$

Note that the function f_θ itself could consist of multiple functions if θ_c consists of multiple parameters. For concreteness, and also to simplify the rest of the exposition, we will focus on the case when the class-conditional distribution is a D dimensional Gaussian, for which θ_c is defined by the mean vector $\mu_c \in \mathbb{R}^D$ and a p.s.d. covariance matrix $\Sigma_c \in \mathcal{S}_+^{D \times D}$. Further, we will assume Σ_c to be a diagonal matrix defined as $\Sigma_c = \text{diag}(\sigma_c^2)$ where $\sigma_c^2 = [\sigma_{c1}^2, \dots, \sigma_{cD}^2]$. Note that one can also assume a full covariance matrix but it will significantly increase the number of parameters to be estimated. We model μ_c and σ_c^2 as functions of the attribute vector \mathbf{a}_c

$$\mu_c = f_\mu(\mathbf{a}_c) \quad (4)$$

$$\sigma_c^2 = f_{\sigma^2}(\mathbf{a}_c) \quad (5)$$

Note that the above equations define two regression models. The first regression model defined by the function f_μ has \mathbf{a}_c as the input and μ_c as the output. The second regression model defined by f_{σ^2} has \mathbf{a}_c as the input and σ^2 as the output. The goal is to learn the functions f_μ and f_{σ^2} from the available training data. Note that the form of these functions is a modelling choice and can be chosen appropriately. We will consider both linear as well as nonlinear functions.

2.2 Learning The Regression Functions

Using the available training data from all the seen classes $c = 1, \dots, S$, we can form empirical estimates of the parameters $\{\hat{\mu}_c, \hat{\sigma}_c^2\}_{c=1}^S$ of respective class-conditional distributions using MLE/MAP estimation. Note that, since our framework is generative,

both labeled as well as unlabeled data from the seen classes can be used to form the empirical estimates $\{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=1}^S$. This makes our estimates of $\{\boldsymbol{\mu}_c, \boldsymbol{\sigma}_c^2\}_{c=1}^S$ reliable even if each seen class has very small number of labeled examples. Given these estimates for the seen classes

$$\hat{\boldsymbol{\mu}}_c = f_{\boldsymbol{\mu}}(\mathbf{a}_c) \quad c = 1, \dots, S \quad (6)$$

$$\hat{\boldsymbol{\sigma}}_c^2 = f_{\boldsymbol{\sigma}^2}(\mathbf{a}_c) \quad c = 1, \dots, S \quad (7)$$

We can now learn $f_{\boldsymbol{\mu}}$ using ‘‘training’’ data $\{\mathbf{a}_c, \hat{\boldsymbol{\mu}}_c\}_{c=1}^S$ and learn $f_{\boldsymbol{\sigma}^2}$ using training data $\{\mathbf{a}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=1}^S$. We consider both linear and nonlinear regression models for learning these.

The Linear Model For the linear model, we assume $\hat{\boldsymbol{\mu}}_c$ and $\hat{\boldsymbol{\sigma}}_c^2$ to be linear functions of the class-attribute vector \mathbf{a}_c , defined as

$$\hat{\boldsymbol{\mu}}_c = \mathbf{W}_{\boldsymbol{\mu}} \mathbf{a}_c \quad c = 1, \dots, S \quad (8)$$

$$\hat{\rho}_c = \log \hat{\boldsymbol{\sigma}}_c^2 = \mathbf{W}_{\boldsymbol{\sigma}^2} \mathbf{a}_c \quad c = 1, \dots, S \quad (9)$$

where the regression weights $\mathbf{W}_{\boldsymbol{\mu}} \in \mathbb{R}^{D \times K}$, $\mathbf{W}_{\boldsymbol{\sigma}^2} \in \mathbb{R}^{D \times K}$, and we have re-parameterized $\hat{\boldsymbol{\sigma}}_c^2 \in \mathbb{R}_+^D$ to $\hat{\rho}_c \in \mathbb{R}^D$ as $\hat{\rho}_c = \log \hat{\boldsymbol{\sigma}}_c^2$.

We use this re-parameterization to map the output space of the second regression model $f_{\boldsymbol{\sigma}^2}$ (defined by $\mathbf{W}_{\boldsymbol{\sigma}^2}$) to real-valued vectors, so that a standard regression model can be applied (note that $\hat{\boldsymbol{\sigma}}_c^2$ is positive-valued vector).

Estimating Regression Weights of Linear Model: We will denote $\mathbf{M} = [\hat{\boldsymbol{\mu}}_1, \dots, \hat{\boldsymbol{\mu}}_S] \in \mathbb{R}^{D \times S}$, $\mathbf{R} = [\hat{\rho}_1, \dots, \hat{\rho}_S] \in \mathbb{R}^{D \times S}$, and $\mathbf{A} = [\mathbf{a}_1, \dots, \mathbf{a}_S] \in \mathbb{R}^{K \times S}$. We can then write the estimation of the regression weights $\mathbf{W}_{\boldsymbol{\mu}}$ as the following problem

$$\hat{\mathbf{W}}_{\boldsymbol{\mu}} = \arg \min_{\mathbf{W}_{\boldsymbol{\mu}}} \|\mathbf{M} - \mathbf{W}_{\boldsymbol{\mu}} \mathbf{A}\|_2^2 + \lambda_{\boldsymbol{\mu}} \|\mathbf{W}_{\boldsymbol{\mu}}\|_2^2 \quad (10)$$

This is essentially a multi-output regression [7] problem $\mathbf{W}_{\boldsymbol{\mu}} : \mathbf{a}_s \mapsto \hat{\boldsymbol{\mu}}_s$ with least squares loss and an ℓ_2 regularizer. The solution to this problem is given by

$$\hat{\mathbf{W}}_{\boldsymbol{\mu}} = \mathbf{M} \mathbf{A}^{\top} (\mathbf{A} \mathbf{A}^{\top} + \lambda_{\boldsymbol{\mu}} \mathbf{I}_K)^{-1} \quad (11)$$

Likewise, we can then write the estimation of the regression weights $\mathbf{W}_{\boldsymbol{\sigma}^2}$ as the following problem

$$\hat{\mathbf{W}}_{\boldsymbol{\sigma}^2} = \arg \min_{\mathbf{W}_{\boldsymbol{\sigma}^2}} \|\mathbf{R} - \mathbf{W}_{\boldsymbol{\sigma}^2} \mathbf{A}\|_2^2 + \lambda_{\boldsymbol{\sigma}^2} \|\mathbf{W}_{\boldsymbol{\sigma}^2}\|_2^2 \quad (12)$$

The solution of the above problem is given by

$$\hat{\mathbf{W}}_{\boldsymbol{\sigma}^2} = \mathbf{R} \mathbf{A}^{\top} (\mathbf{A} \mathbf{A}^{\top} + \lambda_{\boldsymbol{\sigma}^2} \mathbf{I}_K)^{-1} \quad (13)$$

Given $\hat{\mathbf{W}}_{\boldsymbol{\mu}}$ and $\hat{\mathbf{W}}_{\boldsymbol{\sigma}^2}$, parameters of the class-conditional distribution of each unseen class $c = S + 1, \dots, S + U$ can be easily computed as follows

$$\hat{\boldsymbol{\mu}}_c = \hat{\mathbf{W}}_{\boldsymbol{\mu}} \mathbf{a}_c \quad (14)$$

$$\hat{\boldsymbol{\sigma}}_c^2 = \exp(\hat{\rho}_c) = \exp(\hat{\mathbf{W}}_{\boldsymbol{\sigma}^2} \mathbf{a}_c) \quad (15)$$

The Nonlinear Model For the nonlinear case, we assume that the inputs $\{\mathbf{a}_c\}_{c=1}^S$ are mapped to a kernel induced space via a kernel function k with an associated nonlinear mapping ϕ . In this case, using the representer theorem [24], the solution for the two regression models f_μ and f_{σ^2} can be written as the spans of the inputs $\{\phi(\mathbf{a}_c)\}_{c=1}^S$. Note that mappings $\phi(\mathbf{a}_c)$ do not have to be computed explicitly since learning the nonlinear regression model only requires dot products $\phi(\mathbf{a}_c)^\top \phi(\mathbf{a}_{c'}) = k(\mathbf{a}_c, \mathbf{a}_{c'})$ between the nonlinear mappings of two classes c and c' .

Estimating Regression Weights of Nonlinear Model: Denoting \mathbf{K} to be the $S \times S$ kernel matrix of the pairwise similarities of the attributes of the seen classes, the nonlinear model f_μ is obtained by

$$\hat{\boldsymbol{\alpha}}_\mu = \arg \min_{\boldsymbol{\alpha}_\mu} \|\mathbf{M} - \boldsymbol{\alpha}_\mu \mathbf{K}\|_2^2 + \lambda_\mu \|\boldsymbol{\alpha}_\mu\|_2^2 \quad (16)$$

where $\hat{\boldsymbol{\alpha}}_\mu$ is a $D \times S$ matrix consists of the coefficients of the span of $\{\phi(\mathbf{a}_c)\}_{c=1}^S$ defining the nonlinear function f_μ .

Note that the problem in Equation 16 is essentially a multi-output *kernel* ridge regression [7] problem, which has a closed form solution. The solution for $\hat{\boldsymbol{\alpha}}_\mu$ is given by

$$\hat{\boldsymbol{\alpha}}_\mu = \mathbf{M}(\mathbf{K} + \lambda_\mu \mathbf{I}_S)^{-1} \quad (17)$$

Likewise, the nonlinear model f_{σ^2} is obtained by solving

$$\hat{\boldsymbol{\alpha}}_{\sigma^2} = \arg \min_{\boldsymbol{\alpha}_{\sigma^2}} \|\mathbf{M} - \boldsymbol{\alpha}_{\sigma^2} \mathbf{K}\|_2^2 + \lambda_{\sigma^2} \|\boldsymbol{\alpha}_{\sigma^2}\|_2^2 \quad (18)$$

where $\hat{\boldsymbol{\alpha}}_{\sigma^2}$ is a $D \times S$ matrix consists of the coefficients of the span of $\{\phi(\mathbf{a}_c)\}_{c=1}^S$ defining the nonlinear function f_{σ^2} . The solution for $\hat{\boldsymbol{\alpha}}_{\sigma^2}$ is given by

$$\hat{\boldsymbol{\alpha}}_{\sigma^2} = \mathbf{R}(\mathbf{K} + \lambda_{\sigma^2} \mathbf{I}_S)^{-1} \quad (19)$$

Given $\hat{\boldsymbol{\alpha}}_\mu$, $\hat{\boldsymbol{\alpha}}_{\sigma^2}$, parameters of class-conditional distribution of each unseen class $c = S + 1, \dots, S + U$ will be

$$\hat{\boldsymbol{\mu}}_c = \hat{\boldsymbol{\alpha}}_\mu \mathbf{k}_c \quad (20)$$

$$\hat{\boldsymbol{\sigma}}_c^2 = \exp(\hat{\boldsymbol{\rho}}_c) = \exp(\hat{\boldsymbol{\alpha}}_{\sigma^2} \mathbf{k}_c) \quad (21)$$

where $\mathbf{k}_c = [k(\mathbf{a}_c, \mathbf{a}_1), \dots, k(\mathbf{a}_c, \mathbf{a}_S)]^\top$ denotes an $S \times 1$ vector of kernel-based similarities of the class-attribute of unseen class c with the class-attributes of all the seen classes.

Other Exponential Family Distributions Although we illustrated our framework taking the example of Gaussian class-conditional distributions, our framework readily generalizes to the case when these distributions are modelled using any exponential family distribution. The estimation problems can be solved in a similar way as the Gaussian case with the basic recipe remaining the same: Form empirical estimates of the parameters $\boldsymbol{\Theta} = \{\boldsymbol{\theta}_c\}_{c=1}^S$ for the seen classes using all the available seen class data and then

learn a linear/nonlinear regression model from the class-attributes \mathbf{A} (or their kernel representation \mathbf{K} in the nonlinear case) to Θ .

In addition to its modeling flexibility, an especially remarkable aspect of our generative framework is that it is very easy to implement, since both the linear model as well as the nonlinear model have closed-form solutions given by Eq. 11 and Eq. 13, and Eq. 17 and Eq. 19, respectively (the solutions will be available in similar closed-forms in the case of other exponential family distributions). A block-diagram describing our framework is shown in Figure 1. Note that another appealing aspect of our framework is its modular architecture where each of the blocks in Figure 1 can make use of a suitable method of one’s choice.

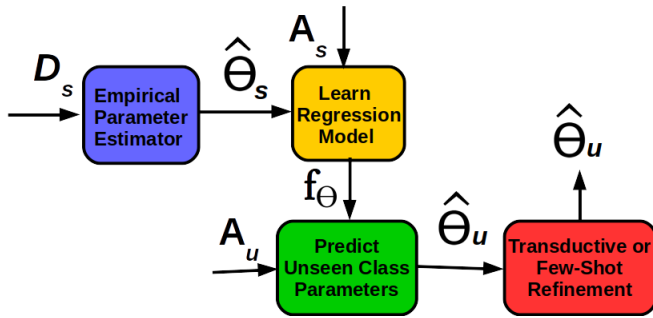


Fig. 1. Block-diagram of our framework. \mathcal{D}_s denotes the seen class data (can be labeled (and optionally also unlabeled)); \mathbf{A}_s denotes seen class attributes; \mathbf{A}_u denotes unseen class attributes; $\hat{\Theta}_s$ denotes the estimated seen class parameters; $\hat{\Theta}_u$ denotes the estimated unseen class parameters. The last stage - transductive/few-shot refinement - is optional (Section 2.3 and 4.2)

2.3 Transductive/Semi-Supervised Setting

The procedure described in Section 2.2 relies only on the seen class data (labeled and, optionally, also unlabeled). As we saw for the Gaussian case, the seen class data is used to form empirical estimates of the parameters $\{\hat{\mu}_c, \hat{\sigma}_c^2\}_{c=1}^S$ of the class-conditional distributions of seen classes, and then these estimates are used to learn the linear/nonlinear regression functions f_μ and f_{σ^2} . These functions are finally used to compute the parameters $\{\hat{\mu}_c, \hat{\sigma}_c^2\}_{c=S+1}^{S+U}$ of class-conditionals of unseen classes. We call this setting the *inductive* setting. Note that this procedure does not make use of any data from the unseen classes. Sometimes, we may have access to unlabeled data from the unseen classes.

Our generative framework makes it easy to leverage such *unlabeled* data from the *unseen* classes to further improve upon the estimates $\{\hat{\mu}_c, \hat{\sigma}_c^2\}_{c=S+1}^{S+U}$ of their class-conditional distributions. In our framework, this can be done in two settings, *transductive* and *semi-supervised*, both of which leverage unlabeled data from unseen classes, but in slightly different ways. If the unlabeled data is the unseen class test data itself, we call it the *transductive* setting. If this unlabeled data from the unseen classes is different from the actual unseen class test data, we call it the *semi-supervised* setting.

In either setting, we can use an Expectation-Maximization (EM) based procedure that alternates between inferring the labels of unlabeled examples of unseen classes and using the inferred labels to update the estimates of the parameters $\{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=S+1}^{S+U}$ of the distributions of unseen classes.

For the case when each class-conditional distribution is a Gaussian, this procedure is equivalent to estimating a Gaussian Mixture Model (GMM) using the unlabeled data $\{\mathbf{x}_n\}_{n=1}^{N_u}$ from the unseen classes. The GMM is initialized using the estimates $\{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=S+1}^{S+U}$ obtained from the inductive procedure of Section 2.2. Note that each of the U mixture components of this GMM corresponds to an unseen class.

The EM algorithm for the Gaussian case is summarized next

1. Initialize mixing proportions $\boldsymbol{\pi} = [\pi_1, \dots, \pi_U]$ uniformly set mixture parameters as $\boldsymbol{\Theta} = \{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=S+1}^{S+U}$
2. **E Step:** Infer the probabilities for each \mathbf{x}_n belonging to each of the unseen classes $c = S + 1, \dots, S + U$ as

$$p(y_n = c | \mathbf{x}_n, \boldsymbol{\pi}, \boldsymbol{\Theta}) = \frac{\pi_c \mathcal{N}(\mathbf{x}_n | \hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2)}{\sum_c \pi_c \mathcal{N}(\mathbf{x}_n | \hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2)}$$

3. **M Step:** Use to inferred class labels to re-estimate $\boldsymbol{\pi}$ and $\boldsymbol{\Theta} = \{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=S+1}^{S+U}$.
4. Go to step 2 if not converged.

Note that the same procedure can be applied even when each class-conditional distribution is some exponential family distribution other than Gaussian. The E and M steps in the resulting mixture model are straightforward in that case as well. The E step will simply require the Gaussian likelihood to be replaced by the corresponding exponential family distribution's likelihood. The M step will require doing MLE of the exponential family distribution's parameters, which has closed-form solutions.

2.4 Extension for Few-Shot Learning

In few-shot learning, we assume that a very small number of labeled examples may also be available for the unseen classes [23,17]. The generative aspect of our framework, along with the fact the the data distribution is an exponential family distribution with a conjugate prior on its parameters, makes it very convenient for our model to be extended to this setting. The outputs $\{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=S+1}^{S+U}$ of our generative zero-shot learning model can naturally serve as the hyper-parameters of a conjugate prior on parameters of class-conditional distributions of unseen classes, which can then be updated given a small number of labeled examples from the unseen classes. For example, in the Gaussian case, due to conjugacy, we are able to update the estimates $\{\hat{\boldsymbol{\mu}}_c, \hat{\boldsymbol{\sigma}}_c^2\}_{c=S+1}^{S+U}$ in a straightforward manner when provided with such labeled data. In particular, given a small number of labeled examples $\{\mathbf{x}_n\}_{n=1}^{N_c}$ from an unseen class c , $\hat{\boldsymbol{\mu}}_c$ and $\hat{\boldsymbol{\sigma}}_c^2$ can be easily updated as

$$\boldsymbol{\mu}_c^{(FS)} = \frac{\hat{\boldsymbol{\mu}}_c + \sum_{n=1}^{N_c} \mathbf{x}_n}{1 + N_c} \quad (22)$$

$$\boldsymbol{\sigma}_c^{2(FS)} = \left(\frac{1}{\hat{\boldsymbol{\sigma}}_c^2} + \frac{N_c}{\boldsymbol{\sigma}^2} \right)^{-1} \quad (23)$$

where $\sigma^2 = \frac{1}{N_c} \sum_{n=1}^{N_c} (\mathbf{x}_n - \hat{\boldsymbol{\mu}}_c)^2$ denotes the empirical variance of the N_c observations from the unseen class c .

A particularly appealing aspect of our few-shot learning model outlined above is that it can also be updated in an online manner as more and more labelled examples become available from the unseen classes, without having to re-train the model from scratch using all the data.

3 Related Work

Some of the earliest works on ZSL are based on predicting attributes for each example [13]. This was followed by a related line of work based on models that assume that the data from each class can be mapped to the class-attribute space (a shared semantic space) in which each seen/unseen class is also represented as a point [26,1,33]. The mapping can be learned using various ways, such as linear models or feed forward neural networks or convolutional neural networks. Predicting the label for a novel unseen class example then involves mapping it to this space and finding the “closest” unseen class. Some of the work on ZSL is aimed at improving the semantic embeddings of concepts/classes. For example, [29] proposed a ZSL model to incorporate relational information about concepts. In another recent work, [4] proposed a model to improve the semantic embeddings using a metric learning formulation. A complementary line of work to the semantic embedding methods is based on a “reverse” mapping, i.e., mapping the class-attribute to the observed feature space [32,37].

In contrast to such semantic embedding methods that assume that the classes are collapsed onto a single point, our framework offers considerably more flexibility by modelling each class using its own distribution. This makes our model more suitable for capturing the intra-class variability, which the simple point-based embedding models are incapable of handling.

Another popular approach for ZSL is based on modelling each unseen class as a linear/convex combination of seen classes [20] or of a set of “abstract” or “basis” classes [22,5]. The latter class of methods, in particular, can be seen as a special case of our framework since, for our linear model, we can view the columns of the $D \times K$ regression weights as representing a set of K basis classes. Note however that our model has such regression weights for each parameter of the class-conditional distribution, allowing it to be considerably more flexible. Moreover, our framework is also significantly different in other ways due to its fully generative framework, due to its ability to incorporate unlabeled data, performing few-shot learning, and its ability to model different types of data using an appropriate exponential family distribution.

A very important issue in ZSL is the *domain shift* problem which may arise if the seen and unseen class come from very different domains. In these situations, standard ZSL models tend to perform badly. This can be somewhat alleviated using some additional unlabeled data from the unseen classes. To this end, [11] provide a dictionary learning based approach for learning unseen class classifiers in which the dictionary is adapted to the unseen class domain. The dictionary adaptation is facilitated using unlabeled data from the unseen classes. In another related work, [8] leverage unlabeled data in a transductive ZSL framework to handle the domain shift problem. Note that

our framework is robust to the domain shift problem due to its ability to incorporate unlabeled data from the unseen classes (the transductive setting). Our experimental results corroborate this.

Semi-supervised learning for ZSL can also be used to improve the semantic embedding based methods. [16] provide a semi-supervised method that leverages prior knowledge for improving the learned embeddings. In another recent work, [37] present a model to incorporate unlabeled unseen class data in a setting where each unseen class is represented as a linear combination of seen classes. [34] provide another approach, motivated by applications in computer vision, that jointly facilitates the domain adaptation of attribute space and the visual space. Another semi-supervised approach presented in [15] combines a semisupervised classification model over the observed classes with an unsupervised clustering model over unseen classes together to address the zero-shot multi-class classification.

In contrast to these models for which the mechanism for incorporating unlabeled data is model-specific, our framework offers a general approach for doing this, while also being simple to implement. Moreover, for large-scale problems, it can also leverage more efficient solvers (e.g., gradient methods) for estimating the regression coefficients associated with class-conditional distributions.

4 Experiments

We evaluate our generative framework for zero-shot learning (hereafter referred to as **GFZSL**) on several benchmark data sets and compare it with a number of state-of-the-art baselines. We conduct our experiments on various problem settings, including standard *inductive* zero-shot learning (only using seen class labeled examples), *transductive* zero-shot learning (using seen class labeled examples and unseen class unlabeled examples), and few-shot learning (using seen class labeled examples and a very small number of unseen class labeled examples). We report our experimental results on the following benchmark data sets:

- **Animal with Attribute (AwA)**: The AwA data set contains 30475 images with 40 seen classes (training set) and 10 unseen classes (test set). Each class has a human-provided binary/continuous 85-dimensional class-attribute vector [12]. We use continuous class-attributes since prior works have found these to have more discriminative power.
- **Caltech-UCSD Birds-200-2011 (CUB-200)**: The CUB-200 data set contains 11788 images with 150 seen classes (training set) and 50 unseen class (test set). Each image has a binary 312-dimensional class-attribute vector, specifying the presence or absence of various attribute of that image [28]. The attribute vectors for all images in a class are averaged to construct its continuous class-attribute vector [2]. We use the same train/test split for this data set as used in [2].
- **SUN attribute (SUN)**: The SUN data set contains 14340 images with 707 seen classes (training set) and 10 unseen classes (test set). Each image is described by a 102-dimensional binary class-attribute vector. Just like the CUB-200 data set, we average the attribute vectors of all images in each class to get its continuous attribute vector [10]. We use the same train/test split for this data set as used in [10].

For image features, we considered both GoogleNet features [27] and VGG-19(4096) fc7 features [25] and found that our approach works better with VGG-19. All of the state-of-the-art baselines we compare with in our experiments use VGG-19 fc7 features or GoogleNet features [27]. For the nonlinear (kernel) variant of our model, we use a quadratic kernel. Our set of experiments include:

- **Zero-Shot Learning:** We consider both inductive ZSL as well as transductive ZSL.
 - **Inductive ZSL:** This is the standard ZSL setting where the unseen class parameters are learned using only seen class data.
 - **Transductive ZSL:** In this setting [34], we also use the unlabeled test data while learning the unseen class parameters. Note that this setting has access to more information about the unseen class; however, it is only through unlabeled data.
- **Few-Shot Learning:** In this setting [23,17], we also use a small number of labelled examples from each unseen class.
- **Generalized ZSL:** Whereas standard ZSL (as well as few-shot learning) assumes that the test data can only be from the unseen classes, generalized ZSL assumes that the test data can be from unseen as well as seen classes. This is usually a more challenging setting [6] and most of the existing methods are known to be biased towards predicting the seen classes.

We use the standard train/test split as given in the data description section. For selecting the hyperparameters, we further divide the train set further into train and validation set. In our model, we have two hyper-parameter λ_μ and λ_{σ^2} , which we tune using the validation dataset. For AWA, from the 40 seen classes, a random selection of 30 classes are used for the training set and 10 classes are used for the validation set. For CUB-200, from the 150 seen classes, 100 are used for the training set and rest 50 are used for the validation set. Similarly, for the SUN dataset from the 707 seen classes, 697 are used for the training set and rest 10 is used for the validation set. We use cross-validation on the validation set to choose the best hyperparameter $[\lambda_\mu, \lambda_{\sigma^2}]$ for the each data set and use these for testing on the unseen classes.

4.1 Zero-Shot Learning

In our first set of experiments, we evaluate our model for zero-shot learning and compare with a number of state-of-the-art methods, for the inductive setting (which uses only the seen class labelled data) as well as the transductive setting (which uses the seen class data and the unseen class unlabeled data).

Inductive ZSL Table-1 shows our results for the inductive ZSL setting. The results of the various baselines are taken from the corresponding papers. As shown in the Table-1, on CUB-200 and SUN, both of our models (linear and nonlinear) perform better than all of the other state-of-the-art methods. On AWA, our model has only a marginally lower test accuracy as compared to the best performing baseline [34]. However, we also have an average improvement 5.67% on all the 3 data sets as compared to the overall best baseline [34]. Among baselines using VGG-19 features (bottom half of Table-1), our

Method	AwA	CUB-200	SUN	Average
Akata et al. [2]	66.70	50.1	–	–
Qiao et al. [21]	66.46±0.42	29±0.28	–	–
Xian et al. [31]	71.9	45.5	–	–
Changpimyo et al.[5]	72.9	54.7	62.7	63.43
Wang et al.[29]	75.99	33.48	–	–
Lampert et al.[14]	57.23	–	72.00	–
Romera and Torr[22]	75.32±2.28	–	82.10± 0.32	–
Bucher et al.[4]	77.32±1.03	43.29±0.38	84.41±0.71	68.34
Z. Zhang et al.[35]	79.12±0.53	41.78±0.52	83.83±.29	68.24
Wang et al.[30]	79.2±0.0	46.7±0.0	–	–
Z. Zhang et al.[34]	81.03±0.88	46.48±1.67	84.10±1.51	70.53
GFZSL: Linear	79.90	52.09	86.50	72.23
GFZSL: Nonlinear	80.83	56.53	86.50	74.59

Table 1. Accuracy(%) of different type of images features. **Top:** Deep features like AlexNet, GoogleNet, etc. **Bottom:** Deep VGG-19 features. The ‘-’ indicates that this result was not reported.

model achieves a 21.05% relative improvement over the best baseline on the CUB-200 data, which is considered to be a difficult data set with many fine-grained classes.

In contrast to other models that embed the test examples in the semantic space and then find the most similar class by doing a Euclidean distance based nearest neighbor search, or models that are based on computing the similarity scores between seen and unseen classes [33], for our models, finding the “most probable class” corresponds to computing the distance of each test example from a *distribution*. This naturally takes into account the shape and spread of the class-conditional distribution. This explains the favourable performance of our model as compared to the other methods.

Transductive Setting For transductive ZSL setting [9,35,36], we follow the procedure described in Section 2.3 to estimate parameters of the class-conditional distribution of each unseen class. After learning the parameters, we find the most probable class for each test example by evaluating its probability under each unseen class distribution and assign it to the class under which it has the largest probability. Table-2 and 3 compare our results from the transductive setting with other state-of-the-art baselines designed for the transductive setting. In addition to accuracy, we also report precision and recall results of our model and the other baselines (wherever available). As we can see from Table-2 and 3, both of our models (linear and kernel) outperform the other baselines on all the 3 data sets. Also comparing with the inductive setting results presented in Table-1, we observe that our generative framework is able to very effectively leverage unlabeled data and significantly improve upon the results of a purely inductive setting

4.2 Few-shot Learning (FSL)

We next perform an experiment with the few-shot learning setting [23,17] where we provide each model with a small number of labelled examples from each of the unseen classes. For this experiment, we follow the procedure described in Section 4.2 to learn the parameters of the class-conditional distributions of the unseen classes. In particular, we train the inductive ZSL model (using only the seen class training data) and the

Method	AwA	CUB-200	SUN	Average
Guo et al.[9]	78.47	–	82.00	–
Romera et al.[22]+ Zhang et al. [36]	84.30	–	37.50	–
Zhang et al.[35]+Zhang et al. [36]	92.08±0.14	55.34±0.77	86.12±0.99	77.85
Zhang et al.[34]+Zhang et al. [36]	88.04±0.69	55.81±1.37	85.35±1.56	76.40
GFZSL: Linear	94.20	57.14	87.00	79.45
GFZSL: Kernel	94.25	63.66	87.00	80.63

Table 2. ZSL accuracy(%) obtained in the transductive setting: results reported using the VGG-19 feature. Average Precision and recall for the all dataset with its standard deviation over the 100 iteration. The ‘-’ indicates that this result was not reported in the original paper.

Method	Average Precision			Average Recall		
	AwA	CUB-200	SUN	AwA	CUB-200	SUN
Zhang et al.[35]+Zhang et al. [36]	91.37±14.75	57.09±27.91	85.96±10.15	90.28±8.08	55.73±31.80	86.00±13.19
Zhang et al.[34]+Zhang et al. [36]	89.19±15.09	57.20±25.96	86.06± 12.36	86.04±9.82	55.77±26.54	85.50±13.68
GFZSL: Linear	93.70	57.90	87.40	92.20	57.40	87.00
GFZSL: Kernel	93.80	64.09	87.40	92.30	63.96	0.87

Table 3. ZSL precision and recall scores obtained in the transductive setting: results reported using the VGG-19 features. Average Precision and recall for the all dataset with its standard deviation over the 100 iteration. Note: Precision and recall scores not available for Guo et al.[9] and Romera et al.[22]+ Zhang et al. [36]

Dataset	Method	2	5	10	15	20
AwA	GFZSL	87.96±1.47	91.64±0.81	93.31±0.50	94.01±.36	94.30±0.33
	SVM	74.81	83.19	90.44	91.22	92.04
CUB-200	GFZSL	60.84±1.39	64.81±1.14	68.44±1.21	70.11±0.93	71.23±0.87
	SVM	46.19	59.33	68.75	73.87	75.42
SUN	GFZSL	75.57±4.79	83.05±3.60	82.09±3.30	–	–
	SVM	56.00	77.00	78.00	–	–

Table 4. Accuracy(%) in the few-shot learning setting: For each data set, the accuracies are reported using 2, 5, 10, 15, 20 labeled examples for each unseen class

refine the learned model further using a very small number of labelled examples from the unseen classes (i.e., the few-shot learning setting).

To see the effect of knowledge transfer from the seen classes, we use a multiclass SVM as a baseline that is provided with the same number of labelled examples from each unseen class. In this experiment, we vary the number of labelled examples of unseen classes from 2 to 20 (for SUN we only use 2, 5, and 10 due to the small number of labelled examples). In Figure-2, we also compare with standard (inductive) ZSL which does not have access to the labelled examples from the unseen classes. Our results are shown Table-4 and Figure-2.

As shown in Table-4 (all data sets) and Figure-2, the classification accuracy on the unseen classes shows a significant improvement over the standard inductive ZSL, even with as few as 2 or 5 additional labelled examples per class. We also observe that the few-shot learning method outperform multiclass SVM which only relies on the labelled data from the unseen classes. This demonstrates the advantage of the knowledge transfer from the seen class data.

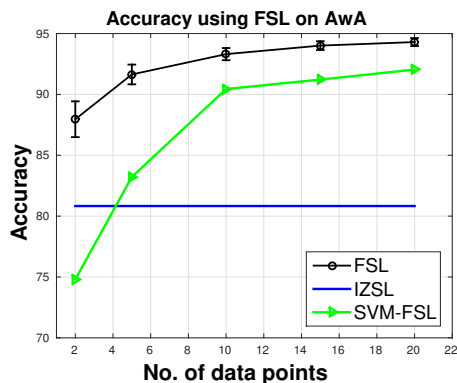


Fig. 2. (On Awa data): A comparison on classification accuracies of the few-shot learning variant of our model with multi-class SVM (training on labeled examples from seen classes) and the inductive ZSL

4.3 Generalized Few-Shot Learning (GFSL)

We finally perform an experiment on the more challenging generalized few-shot learning setting [6]. This setting assumes that test examples can come from seen as well as unseen classes. This setting is known to be notoriously hard [6]. In this setting, although the ZSL models tend to do well on predicting test examples from seen classes, the performance on correctly predicting the unseen class example is poor [6] since the trained models are heavily biased towards predicting the seen classes.

One way to mitigate this issue could be to use some labelled examples from the unseen classes (akin to what is done in few-shot learning). We, therefore, perform a

similar experiment as in Section 4.2. In Table-5, we show the results of our model on classifying the unseen class test examples in this setting.

As shown in Table-5, our model’s accuracies on the generalized FSL task improve as it gets to see labelled examples from unseen classes. However, it is still outperformed by a standard multiclass SVM. The better performance of SVM can be attributed to the fact that it is not biased towards the seen classes since the classifier for each class (seen/unseen) is learned independently.

Our findings are also corroborated by other recent work on generalized FSL [6] and suggest the need of finding more robust ways to handle this setting. We leave this direction of investigation as a possible future work.

Dataset	Method	2	5	10	15	20
AwA	GFZSL	25.32 ± 2.43	37.42 ± 1.60	43.20 ± 1.39	45.09 ± 1.17	45.96 ± 1.09
	SVM	40.84	60.81	75.36	77.00	77.10
CUB-200	GFZSL	6.64 ± 0.87	15.12 ± 1.17	22.02 ± 0.76	25.03 ± 0.71	26.47 ± 0.83
	SVM	25.97	37.98	47.10	53.87	54.42
SUN	GFZSL	1.17 ± 1.16	4.20 ± 1.77	9.48 ± 2.22	–	–
	SVM	9.94	20.00	27.00	–	–

Table 5. Accuracies (%) in the generalized few-shot learning setting.

5 Conclusion

We have presented a flexible generative framework for zero-shot learning, which is based on modelling each seen/unseen class using an exponential family class-conditional distribution. In contrast to the semantic embedding based methods for zero-shot learning which model each class as a point in a latent space, our approach models each class as a distribution, where the parameters of each class-conditional distribution are functions of the respective class-attribute vectors. Our generative framework allows learning these functions easily using seen class training data (and optionally leveraging additional unlabeled data from seen/unseen classes).

An especially appealing aspect of our framework is its simplicity and modular architecture (cf., Figure 1) which allows using a variety of algorithms for each of its building blocks. As we showed, our generative framework admits natural extensions to other related problems, such as transductive zero-shot learning and few-shot learning. It is particularly easy to implement and scale to a large number of classes, using advances in large-scale regression. Our generative framework can also be extended to jointly learn the class attributes from an external source of data (e.g., by learning an additional embedding model with our original model). This can be an interesting direction of future work. Finally, although we considered a point estimation of the parameters of class-conditional distributions, it is also possible to take a fully Bayesian approach for learning these distributions. We leave this possibility as a direction for future work.

Acknowledgements: This work is supported by a grant from Tower Research CSR, Dr. Deep Singh and Daljeet Kaur Fellowship, and Research-I Foundation, IIT Kanpur. Vinay Verma acknowledges support from Visvesvaraya Ph.D. fellowship.

References

1. AKATA, Z., PERRONNIN, F., HARCHAOU, Z., AND SCHMID, C. Label-embedding for attribute-based classification. In *CVPR* (2013).
2. AKATA, Z., REED, S., WALTER, D., LEE, H., AND SCHIELE, B. Evaluation of output embeddings for fine-grained image classification. In *CVPR* (2015).
3. BROWN, L. D. Fundamentals of statistical exponential families. *Institute of Mathematical Statistics* (1986).
4. BUCHER, M., HERBIN, S., AND JURIE, F. Improving semantic embedding consistency by metric learning for zero-shot classification. *arXiv preprint arXiv:1607.08085* (2016).
5. CHANGPINYO, S., CHAO, W.-L., GONG, B., AND SHA, F. Synthesized classifiers for zero-shot learning. In *CVPR* (2016).
6. CHAO, W.-L., CHANGPINYO, S., GONG, B., AND SHA, F. An empirical study and analysis of generalized zero-shot learning for object recognition in the wild. In *ECCV* (2016).
7. FRIEDMAN, J., HASTIE, T., AND TIBSHIRANI, R. *The elements of statistical learning*, vol. 1. Springer series in statistics Springer, Berlin, 2001.
8. FU, Y., HOSPEDALES, T. M., XIANG, T., AND GONG, S. Transductive multi-view zero-shot learning. *PAMI* (2015).
9. GUO, Y., DING, G., JIN, X., AND WANG, J. Transductive zero-shot recognition via shared model space learning. In *AAAI* (2016).
10. JAYARAMAN, D., AND GRAUMAN, K. Zero-shot recognition with unreliable attributes. In *NIPS* (2014).
11. KODIROV, E., XIANG, T., FU, Z., AND GONG, S. Unsupervised domain adaptation for zero-shot learning. In *ICCV* (2015).
12. KRIZHEVSKY, A., AND HINTON, G. Learning multiple layers of features from tiny images.
13. LAMPERT, C. H., NICKISCH, H., AND HARMELING, S. Learning to detect unseen object classes by between-class attribute transfer. In *CVPR* (2009).
14. LAMPERT, C. H., NICKISCH, H., AND HARMELING, S. Attribute-based classification for zero-shot visual object categorization. *PAMI* (2014).
15. LI, X., AND GUO, Y. Max-margin zero-shot learning for multi-class classification. In *AISTATS* (2015).
16. LI, X., GUO, Y., AND SCHUURMANS, D. Semi-supervised zero-shot classification with label representation learning. In *CVPR* (2015).
17. MENSINK, T., GAVVES, E., AND SNOEK, C. G. Costa: Co-occurrence statistics for zero-shot classification. In *CVPR* (2014).
18. MUKHERJEE, T., AND HOSPEDALES, T. Gaussian visual-linguistic embedding for zero-shot recognition. In *EMNLP* (2016).
19. MURPHY, K. P. *Machine learning: a probabilistic perspective*. MIT press, 2012.
20. NOROUZI, M., MIKOLOV, T., BENGIO, S., SINGER, Y., SHLENS, J., FROME, A., CORRADO, G. S., AND DEAN, J. Zero-shot learning by convex combination of semantic embeddings. *ICLR* (2014).
21. QIAO, R., LIU, L., SHEN, C., AND VAN DEN HENGEL, A. Less is more: zero-shot learning from online textual documents with noise suppression. In *CVPR* (2016).
22. ROMERA-PAREDES, B. An embarrassingly simple approach to zero-shot learning. In *ICML* (2015).
23. SALAKHUTDINOV, R., TENENBAUM, J. B., AND TORRALBA, A. Learning with hierarchical-deep models. *PAMI* (2013).
24. SCHÖLKOPF, B., AND SMOLA, A. J. *Learning with kernels: support vector machines, regularization, optimization, and beyond*. MIT press, 2001.

25. SIMONYAN, K., AND ZISSERMAN, A. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).
26. SOCHER, R., GANJOO, M., MANNING, C. D., AND NG, A. Zero-shot learning through cross-modal transfer. In *NIPS* (2013).
27. SZEGEDY, C., LIU, W., JIA, Y., Sermanet, P., REED, S., ANGUELOV, D., ERHAN, D., VANHOUCHE, V., AND RABINOVICH, A. Going deeper with convolutions. In *CVPR* (2015).
28. WAH, C., BRANSON, S., WELINDER, P., PERONA, P., AND BELONGIE, S. The caltech-ucsd birds-200-2011 dataset.
29. WANG, D., LI, Y., LIN, Y., AND ZHUANG, Y. Relational knowledge transfer for zero-shot learning. In *AAAI* (2016).
30. WANG, Q., AND CHEN, K. Zero-shot visual recognition via bidirectional latent embedding. *arXiv preprint arXiv:1607.02104* (2016).
31. XIAN, Y., AKATA, Z., SHARMA, G., NGUYEN, Q., HEIN, M., AND SCHIELE, B. Latent embeddings for zero-shot classification. In *CVPR* (2016).
32. ZHANG, L., XIANG, T., AND GONG, S. Learning a deep embedding model for zero-shot learning. *arXiv preprint arXiv:1611.05088* (2016).
33. ZHANG, Z., AND SALIGRAMA, V. Zero-shot learning via semantic similarity embedding. In *ICCV* (2015).
34. ZHANG, Z., AND SALIGRAMA, V. Learning joint feature adaptation for zero-shot recognition. *arXiv preprint arXiv:1611.07593* (2016).
35. ZHANG, Z., AND SALIGRAMA, V. Zero-shot learning via joint latent similarity embedding. In *CVPR* (2016).
36. ZHANG, Z., AND SALIGRAMA, V. Zero-shot recognition via structured prediction. In *ECCV* (2016).
37. ZHAO, B., WU, B., WU, T., AND WANG, Y. Zero-shot learning via revealing data distribution. *arXiv preprint arXiv:1612.00560* (2016).