Latent Embeddings for Zero-shot Classification

Yongqin Xian[1], Zeynep Akata[1], Gaurav Sharma[1,2,*], Quynh Nguyen[3], Matthias Hein[3] and Bernt Schiele[1]

[1]MPI for Informatics [2]IIT Kanpur [3]Saarland University

Abstract

We present a novel latent embedding model for learning a compatibility function between image and class embeddings, in the context of zero-shot classification. The proposed method augments the state-of-the-art bilinear compatibility model by incorporating latent variables. Instead of learning a single bilinear map, it learns a collection of maps with the selection, of which map to use, being a latent variable for the current image-class pair. We train the model with a ranking based objective function which penalizes incorrect rankings of the true class for a given image. We empirically demonstrate that our model improves the state-of-the-art for various class embeddings consistently on three challenging publicly available datasets for the zero-shot setting. Moreover, our method leads to visually highly interpretable results with clear clusters of different fine-grained object properties that correspond to different latent variable maps.

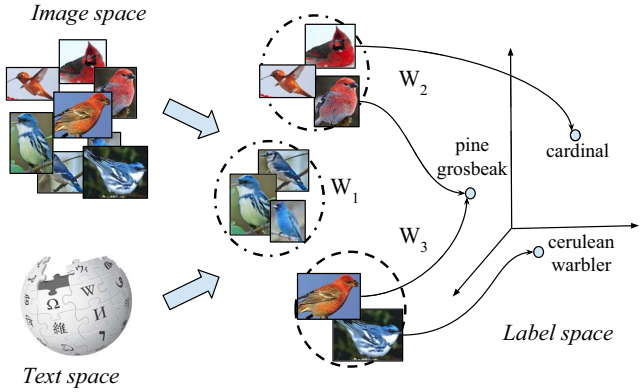

Figure 1: LatEm learns multiple W_i's that maximize the compatibility between the input embedding (image, text space) and the output embedding (label space) of all training examples. The different W_i's may capture different visual characteristics of objects, i.e. color, beak shape etc. and allow distribution of the complexity among them, enabling the model to do better classification.

1. Introduction

Zero-shot classification [14, 20, 21, 30, 41] is a challenging problem. The task is generally set as follows: training images are provided for certain visual classes and the classifier is expected to predict the presence or absence of novel classes at test time. The training and test classes are connected via some auxiliary, non visual source of information e.g. attributes.

Combining visual information with attributes [7, 8, 11, 17, 20, 28, 27] has also supported fine grained classification. In fine grained image collections, images that belong to different classes are visually similar to each other, e.g. different bird species. Image labeling for such collections is a costly process, as it requires either expert opinion or a large number of attributes. To overcome this limitation, recent works have explored distributed text representations [23, 29, 24] which are learned from general (or domain specific) text corpora.

Substantial progress has been made for image classification problem in the zero-shot setting on fine-grained image collections [2]. This progress can be attributed to (i) strong deep learning based image features [19, 36] and (ii) learning a discriminative compatibility function between the structured image and class embeddings [1, 2, 12, 32]. The focus of this work is on the latter, i.e. on improving the compatibility learning framework, in particular via unsupervised auxiliary information.

The main idea of structured embedding frameworks [1, 2, 12, 32] is to first represent both the images *and the classes* in some multi-dimensional vector spaces. Image embeddings are obtained from state-of-the-art image representations e.g. those from convolutional neural networks [19, 36]. Class embeddings can either (i) be obtained using manually specified side information e.g. attributes [20], or (ii) extracted automatically [23, 29] from an unlabeled large text corpora. A discriminative bilinear compatibility function is then learned that pulls images from the same class close to each other and pushes images from different classes away from each other. Once learned, such a compatibility

*Currently with CSE, Indian Institute of Technology Kanpur. Majority of this work was done at Max Planck Institute for Informatics.

1063-6919/16 $31.00 © 2016 IEEE
DOI 10.1109/CVPR.2016.15

function can be used to predict the class (more precisely, the embedding) of any given image (embedding). In particular, this prediction can be done for images from both seen and unseen classes, hence enabling zero-shot classification.

We address the fine-grained zero-shot classification problem while being particularly interested in more flexible unsupervised text embeddings. The state-of-the-art methods [1, 2, 12, 32] use a unique, globally linear compatibility function for all types of images. However, learning a linear compatibility function is not particularly suitable for the challenging fine-grained classification problem. For fine-grained classification, a model that can automatically group objects with similar properties together and then learn for each group a separate compatibility model is required. For instance, two different linear functions that separate blue birds with brown wings and from other blue birds with blue wings can be learned separately. To that end, we propose a novel model for zero-shot setting which incorporates latent variables to learn a piecewise linear compatibility function between image and class embeddings. The approach is inspired by many recent advances in visual recognition that utilize latent variable models, e.g. in object detection [10, 15], human pose estimation [39] and face detection [43] etc.

Our contributions are as follows. (1) We propose a novel method for zero-shot learning. By incorporating latent variables in the compatibility function our method achieves factorization over such (possibly complex combinations of) variations in pose, appearance and other factors. Instead of learning a single linear function, we propose to learn a collection of linear models while allowing each image-class pair to choose from them. This effectively makes our model non-linear, as in different local regions of the space the decision boundary, while being linear, is different. We use an efficient stochastic gradient descent (SGD) based learning method. (2) We propose a fast and effective method for model selection, i.e. through model pruning. (3) We evaluate our novel piecewise linear model for zero-shot classification on three challenging datasets. We show that incorporating latent variables in the compatibility learning framework consistently improves the state-of-the-art.

The rest of the paper is structured as follows. In Sec. 3 we detail the bilinear compatibility learning framework that we base our method on. In Sec. 4 we present our novel Latent Embedding (LatEm) method. In Sec. 5 we present our experimental evaluation and in Sec. 6 we conclude.

2. Related Work

We are interested in the problem of zero-shot learning where the test classes are disjoint from the training classes [14, 20, 21, 30, 41, 42]. As visual information from such test classes is not available during training, zero-shot learning requires secondary information sources to make up for the missing visual information. While secondary information can come from different sources, usually they are derived from either large and unrestricted, but freely available, text corpora, e.g. word2vec [23], glove [29], or structured textual sources e.g. wordnet hierarchies [24], or costly human annotations e.g. manually specified attributes [7, 8, 11, 17, 20, 28, 27]. Attributes, such as 'furry', 'has four legs' etc. for animals, capture several characteristics of objects (visual classes) that help associate some and differentiate others. They are typically collected through costly human annotation [7, 17, 28] and have shown promising results [1, 3, 6, 20, 22, 33, 34, 40] in various computer vision problems.

The image classification problem, with a secondary stream of information, could be either solved by solving related sub-problems, e.g. attribute prediction [20, 30, 31], or by a direct approach, e.g. compatibility learning between embeddings [1, 12, 38]. One such factorization could be by building intermediate attribute classifiers and then making a class prediction using a probabilistic weight of each attribute for each sample [20]. However, these methods, based on attribute classifiers, have been shown to be suboptimal [1]. This is due to their reliance on binary mappings (by thresholding attribute scores) between attributes and images which causes loss in information. On the other hand, solving the problem directly, by learning a direct mapping between images and their classes (represented as numerical vectors) has been shown to be better suited. Such label embedding methods [1, 2, 12, 13, 25, 26, 32, 35] aim to find a mapping between two embedding spaces, one each for the two streams of information e.g. visual and textual. Among these methods, CCA [13] maximizes the correlation between these two embedding spaces, [26] learns a linear compatibility between an fMRI-based image space and the semantic space, [35] learns a deep non-linear mapping between images and tags, ConSe [25] uses the probabilities of a softmax-output layer to weight the vectors of all the classes, SJE [2] and ALE [1] learn a bilinear compatibility function using a multiclass [4] and a weighted approximate ranking loss [16] respectively. DeViSE [12] does the same, however, with an efficient ranking formulation. Most recently, [32] proposes to learn this mapping by optimizing a simple objective function which has closed form solution.

We build our work on multimodal embedding methods. However, instead of learning a linear compatibility function, we propose a nonlinear compatibility framework that learns a collection of such linear models making the overall function piecewise linear.

3. Background: Bilinear Joint Embeddings

In this section, we describe the bilinear joint embedding framework [38, 1, 2], on which we build our Latent Embedding Model (LatEm) (Sec. 4).

We work in a supervised setting where we are given an annotated training set $\mathcal{T} = \{(\mathbf{x}, \mathbf{y}) | \mathbf{x} \in \mathcal{X} \subset \mathbb{R}^{d_x}, \mathbf{y} \in \mathcal{Y} \subset \mathbb{R}^{d_y}\}$ where $\mathbf{x}$ is the image embedding defined in an image feature space $\mathcal{X}$, e.g. CNN features [19], and $\mathbf{y}$ is the class embedding defined in a label space $\mathcal{Y}$ that models the conceptual relationships between classes, e.g. attributes [9, 20]. The goal is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ to predict the correct class for the query images. In previous work [38, 1, 2], this is done via learning a function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ that measures the compatibility between a given input embedding ($\mathbf{x} \in \mathcal{X}$) and an output embedding ($\mathbf{y} \in \mathcal{Y}$). The prediction function then chooses the class with the maximum compatibility, i.e.

$$f(\mathbf{x}) = \arg\max_{\mathbf{y} \in \mathcal{Y}} F(\mathbf{x}, \mathbf{y}). \tag{1}$$

In general, the class embeddings reflect the common and distinguishing properties of different classes using side-information that is extracted independently of images. Using these embeddings, the compatibility can be computed even with those unknown classes which have no corresponding images in the training set. Therefore, this framework can be applied to zero-shot learning [1, 2, 26, 32, 35]. In previous work, the compatibility function takes a simple form,

$$F(\mathbf{x}, \mathbf{y}) = \mathbf{x}^\top W \mathbf{y} \tag{2}$$

with the matrix $W \in \mathbb{R}^{d_x \times d_y}$ being the parameter to be learnt from training data. Due to the bilinearity of F in $\mathbf{x}$ and $\mathbf{y}$, previous work [1, 2, 38] refer to this model as a bilinear model, however one can also view it as a linear one since F is linear in the parameter W. In the following, these two terminologies will be used interchangeably depending on the context.

4. Latent Embeddings Model (LatEm)

In general, the linearity of the compatibility function (Eq. (2)) is a limitation as the problem of image classification is usually a complex nonlinear decision problem. A very successful extension of linear decision functions to nonlinear ones, has been through the use of piecewise linear decision functions. This idea has been applied successfully to various computer vision tasks e.g. mixture of templates [15] and deformable parts-based model [10] for object detection, mixture of parts for pose estimation [39] and face detection [43]. The main idea in most of such models, along with modeling parts, is that of incorporating latent variables, thus making the decision function piecewise linear, e.g. the different templates in the mixture of templates [15] and the different 'components' in the deformable parts model [10]. The model then becomes a collection of linear models and the test images pick one from these linear models, with the selection being latent. Intuitively, this factorizes the decision function into components which focus on distinctive 'clusters' in the data e.g. one component may focus on the profile view while another on the frontal view of the object.

Objective. We propose to construct a nonlinear, albeit piecewise linear, compatibility function. Parallel to the latent SVM formulation, we propose a non-linear compatibility function as follows,

$$F(\mathbf{x}, \mathbf{y}) = \max_{1 \leq i \leq K} \tilde{\mathbf{w}}_i^\top (\mathbf{x} \otimes \mathbf{y}), \tag{3}$$

where $i = 1, \ldots, K$, with $K \geq 2$, indexes over the latent choices and $\tilde{\mathbf{w}}_i \in \mathbb{R}^{d_x d_y}$ are the parameters of the individual linear components of the model. This can be rewritten as a mixture of bilinear compatibility functions from Eq. (2) as

$$F(\mathbf{x}, \mathbf{y}) = \max_{1 \leq i \leq K} \mathbf{x}^\top W_i \mathbf{y}. \tag{4}$$

Our main goal is to learn a set of compatibility spaces that minimizes the following empirical risk,

$$\frac{1}{N} \sum_{n=1}^{|\mathcal{T}|} L(\mathbf{x}_n, \mathbf{y}_n), \tag{5}$$

where $L : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$ is the loss function defined for a particular example $(\mathbf{x}_n, \mathbf{y}_n)$ as

$$L(\mathbf{x}_n, \mathbf{y}_n) = \sum_{\mathbf{y} \in \mathcal{Y}} \max\{0, \Delta(\mathbf{y}_n, \mathbf{y}) + F(\mathbf{x}_n, \mathbf{y}) - F(\mathbf{x}_n, \mathbf{y}_n)\} \tag{6}$$

where $\Delta(\mathbf{y}, \mathbf{y}_n) = 1$ if $\mathbf{y} \neq \mathbf{y}_n$ and 0 otherwise. This ranking-based loss function has been previously used in [12, 38] such that the model is trained to produce a higher compatibility between the image embedding and the class embedding of the correct label than between the image embedding and class embedding of other labels.

Optimization. To minimize the empirical risk in Eq. (5), one first observes that the ranking loss function L from Eq. (6) is not jointly convex in all the W_i's even though F is convex. Thus, finding a globally optimal solution as in the previous linear models [1, 2] is out of reach. To solve this problem, we propose a simple SGD-based method that works in the same fashion as in the convex setting. It turns out that our algorithm works well in practice and achieves state-of-the-art results as we empirically show in Sec. 5.

We explain the details of our Algorithm 1 as follows. We loop through all our samples for a certain number of epochs T. For each sample $(\mathbf{x}_n, \mathbf{y}_n)$ in the training set, we randomly select a $\mathbf{y}$ that is different from $\mathbf{y}_n$ (step 3 of Algorithm 1). If the randomly selected $\mathbf{y}$ violates the margin (step 4 in Algorithm 1), then we update the W_i matrices (steps $5 - 13$ in Algorithm 1). In particular, we find the W_i

Algorithm 1 SGD optimization for LatEm

$\mathcal{T} = \{(\mathbf{x}, \mathbf{y}) | \mathbf{x} \in \mathbb{R}^{d_x}, \mathbf{y} \in \mathbb{R}^{d_y}\}$

1: **for all** $t = 1$ to T **do**
2: **for all** $n = 1$ to $|\mathcal{T}|$ **do**
3: Draw $(\mathbf{x}_n, \mathbf{y}_n) \in \mathcal{T}$ and $\mathbf{y} \in \mathcal{Y} \setminus \{\mathbf{y}_n\}$
4: **if** $F(\mathbf{x}_n, \mathbf{y}) + 1 > F(\mathbf{x}_n, \mathbf{y}_n)$ **then**
5: $i^* \leftarrow \arg\max_{1 \le k \le K} \mathbf{x}_n^\top W_k \mathbf{y}$
6: $j^* \leftarrow \arg\max_{1 \le k \le K} \mathbf{x}_n^\top W_k \mathbf{y}_n$
7: **if** $i^* = j^*$ **then**
8: $W_{i^*}^{t+1} \leftarrow W_{i^*}^t - \eta_t \mathbf{x}_n (\mathbf{y} - \mathbf{y}_n)^\top$
9: **end if**
10: **if** $i^* \neq j^*$ **then**
11: $W_{i^*}^{t+1} \leftarrow W_{i^*}^t - \eta_t \mathbf{x}_n \mathbf{y}^\top$
12: $W_{j^*}^{t+1} \leftarrow W_{j^*}^t + \eta_t \mathbf{x}_n \mathbf{y}_n^\top$
13: **end if**
14: **end if**
15: **end for**
16: **end for**

that leads to the maximum score for $\mathbf{y}$ and the W_j that gives the maximum score for $\mathbf{y}$. If the same matrix gives the maximum score (step 7 in Algorithm 1), we update that matrix. If two different matrices lead to the maximum score (step 9 in Algorithm 1), we update them both using SGD.

Model selection. The number of matrices K in the model is a free parameter. We use two strategies to select the number of matrices. As the first method, we use a standard cross-validation strategy – we split the dataset randomly into disjoint parts (in a zero-shot setup) and choose the K with the best cross-validation performance. While this is a well established strategy which we find to work well experimentally, we also propose a pruning based strategy which is competitive while being faster to train. As the second method, we start with a large number of matrices and prune them as follows. As the training proceeds, each sampled training examples chooses one of the matrices for scoring – we keep track of this information and build a histogram over the number of matrices counting how many times each matrix was chosen by any training example. In particular, this is done by increasing the counter for W_{j^*} by 1 after step 6 of Algorithm 1. With this information, after five passes over the training data, we prune out the matrices which were chosen by less than 5% of the training examples, so far. This is based on the intuition that if a matrix is being chosen only by a very small number of examples, it is probably not critical for performance. With this approach we have to train only one model which adapts itself, instead of training multiple models for cross-validating K and then training a final model with the chosen K.

Discussion. LatEm builds on the idea of Structured Joint Embeddings (SJE) [2]. We discuss below the differences between LatEm and SJE and emphasize our technical contributions.

LatEm learns a piecewise linear compatibility function through multiple W_i matrices whereas SJE [2] is linear. With multiple W_i's the compatibility function has the freedom to treat different types of images differently. Let us consider a fixed class $\hat{\mathbf{y}}$ and two substantially visually different types of images $\mathbf{x}_1, \mathbf{x}_2$, e.g. the same bird flying and swimming. In SJE [2] these images will be mapped to the class embedding space with a single mapping $W^\top \mathbf{x}_1, W^\top \mathbf{x}_2$. On the other hand, LatEm will have learned two different matrices for the mapping i.e. $W_1^\top \mathbf{x}_1, W_2^\top \mathbf{x}_2$. While in the former case a single W has to map two visually, and hence numerically, very different vectors (close) to the same point, in the latent case such two different mappings are factorized separately and hence are arguably easier to perform. Such factorization is also expected to be advantageous when two classes sharing partial visual similarity are to be discriminated e.g. while blue birds could be relative easily distinguished from red birds, to do so for different types of blue birds is harder. In such cases, one of the W_i's could focus on color while another one could focus on the beak shape (in Sec. 5.2 we show that this effect is visible). The task of discrimination against different bird species would then be handled only by the second one, which would also arguably be easier.

LatEm uses the ranking based loss [38] in Eq. (6) whereas SJE [2] uses the multiclass loss of Crammer and Singer [4] which replaces the $\sum$ in Eq. (6) with $\max$. The SGD algorithm for multiclass loss of Crammer and Singer [4] requires at each iteration a full pass over all the classes to search for the maximum violating class. Therefore it can happen that some matrices will not be updated frequently. On the other hand, the ranking based loss in Eq. (6) used by our LatEm model ensures that different latent matrices are updated frequently. Thus, the ranking based loss in Eq. (6) is better suited for our piecewise linear model.

5. Experiments

We evaluate the proposed model on three challenging publicly available datasets of Birds, Dogs and Animals. First, we describe the datasets, then give the implementation details and finally report the experimental results.

Datasets. Caltech-UCSD Birds (CUB), Stanford Dogs (Dogs) are standard benchmarks of fine-grained recognition [7, 5, 37, 18] and Animals With Attributes (AWA) is another popular and challenging benchmark dataset [20]. All these three datasets have been used for zero-shot learning [2, 30, 17, 41]. Tab. 1 gives the statistics for them.

In zero-shot setting, the dataset is divided into three dis-

	Total		train+val		test	
	imgs	cls	imgs	cls	imgs	cls
CUB	11786	200	8855	150	2931	50
AWA	30473	50	24293	40	6180	10
Dogs	19499	113	14681	85	4818	28

Table 1: The statistics of the three datasets used. CUB and Dog are fine-grained datasets whereas AWA is a more general concept dataset.

	CUB		AWA		Dogs	
	SJE	LatEm	SJE	LatEm	SJE	LatEm
att	**50.1**	45.5	66.7	**71.9**	N/A	N/A
w2v	28.4	**31.8**	51.2	**61.1**	19.6	**22.6**
glo	24.2	**32.5**	58.8	**62.9**	17.8	**20.9**
hie	20.6	**24.2**	51.2	**57.5**	24.3	**25.2**

Table 2: Comparison of Latent Embeddings (LatEm) method with the state-of-the-art SJE [2] method. We report average per-class Top-1 accuracy on unseen classes. We use the same data partitioning, same image features and same class embeddings as SJE [2]. We cross-validate the K for LatEm.

joint sets of train, val and test. For comparing with previous works, we follow the same train/val/test set split used by [2]. In zero-shot learning, where training and test classes are disjoint sets, to get a more stable estimate in our own results, we make four more splits by random sampling, while keeping the number of classes the same as before. We average results over the total of five splits. The average performance over the five splits is the default setting reported in all experiments, except where mentioned otherwise, e.g. in comparison with previous methods.

Image and class embeddings. In our latent embedding (LatEm) model, the image embeddings (image features) and class embeddings (side information) are two essential components. To facilitate direct comparison with the state-of-the-art, we use the embeddings provided by [2]. Briefly, as image embeddings we use the $1,024$ dimensional outputs of the top-layer pooling units of the pre-trained GoogleNet [36] extracted from the whole image. We do not do any task specific pre-processing on images such as cropping foreground objects.

As class embeddings we evaluate four different alternatives, i.e. attributes (att), word2vec (w2v), glove (glo) and hierarchies (hie). Attributes [20, 9] are distinguishing properties of objects that are obtained through human annotation. For fine-grained datasets such as CUB and Dogs, as objects are visually very similar to each other, a large number of attributes are needed. Among the three datasets used, CUB contains 312 attributes, AWA contains 85 attributes while Dogs does not contain annotations for attributes. Our attribute class embedding is a vector per-class measuring the strength of each attribute based on human judgment.

In addition to human annotation the class embeddings can be constructed automatically from either a large unlabeled text corpora or through hierarchical relationship between classes. This has certain advantages such as we do not need any costly human annotation, however as a drawback, they tend not to perform as well as supervised attributes. One of our motivations for this work is that the class embeddings captured from a large text corpora contains latent relationships between classes and we would like to automatically learn these. Therefore, we evaluate three common methods for building unsupervised text embeddings. Word2Vec [23] is a two-layer neural network which predicts words given the context within a skip window slided through a text document. It builds a vector for each word in a learned vocabulary. Glove [29] is another distributed text representation method which uses co-occurrence statistics of words within a document. We use the pre-extracted word2vec and glove vectors from wikipedia provided by [2]. Finally, another way of building a vectorial structure for our classes is to use a hierarchy such as WordNet [24]. Our hierarchy vectors are based on the hierarchical distance between child and ancestor nodes, in WordNet, corresponding to our class names. For a direct comparison, we again use the hierarchy vectors provided by [2]. In terms of size, w2v and glo are 400 dimensional whereas hie is ≈ 200 dimensional.

Implementation details. Our image features are z-score normalized such that each dimension has zero mean and unit variance. All the class embeddings are ℓ_2 normalized. The matrices W_i are initialized at random with zero mean and standard deviation $\frac{1}{\sqrt{d_x}}$ [1]. The number of epochs is fixed to be 150. The learning rates for the CUB, AWA and Dog datasets are chosen as $\eta_t = 0.1, 0.001, 0.01$, respectively, and kept constant over iterations. For each dataset, these parameters are tuned on the validation set of the default dataset split and kept constant for all other dataset folds and for all class embeddings. As discussed in Sec 4, we perform two strategies for selecting the number of latent matrices K: cross-validation and pruning. When using cross-validation, K is varied in $\{2, 4, 6, 8, 10\}$ and the optimal K is chosen based the accuracy on a validation set. For pruning, K is initially set to be 16, and then at every fifth epoch during training, we prune all those matrices that support less than 5% of the data points.

5.1. Comparison with State-of-the-Art

We now provide a direct comparison between our LatEm and the state-of-the-art SJE [2] method. SJE (Sec. 3) learns

	CUB		AWA		Dogs	
	SJE	LatEm	SJE	LatEm	SJE	LatEm
w	**51.7**	47.4	73.9	**76.1**	N/A	N/A
w/o	29.9	**34.9**	60.1	**66.2**	35.1	**36.3**

Table 3: Combining embeddings either including or not including supervision in the combination. w: the combination includes attributes, w/o: the combination does not include attributes.

	CUB		AWA		Dogs	
	SJE	LatEm	SJE	LatEm	SJE	LatEm
att	**49.5**	45.6	70.7	**72.5**	N/A	N/A
w2v	27.7	**33.1**	49.3	**52.3**	23.0	**24.5**
glo	24.8	**30.7**	50.1	**50.7**	14.8	**20.2**
hie	21.4	**23.7**	43.4	**46.2**	24.6	**25.6**

Table 4: Average per-class top-1 accuracy on unseen classes (the results are averaged on five folds). SJE: [2], LatEm: Latent embedding model (K is cross-validated).

a bilinear function that maximizes the compatibility between image and class embeddings. Our LatEm on the other hand learns a nonlinear, i.e. piece-wise linear function, through multiple compatibility functions defined between image and class embeddings.

The results are presented in Tab. 2. Using the text embeddings obtained through human annotation, i.e. attributes (`att`), LatEm improves over SJE on AWA (71.9% vs. 66.7%) significantly. However, as our aim is to reduce the accuracy gap between supervised and unsupervised class embeddings, we focus on unsupervised embeddings, i.e. `w2v`, `glo` and `hie`. On all datasets, LatEm with `w2v`, `glo` and `hie` improves the state-of-the-art SJE [2] significantly. With `w2v`, LatEm achieves 31.8% accuracy (vs 28.4%) on CUB, 61.1% accuracy (vs 51.2%) on AWA and finally 22.6% (vs 19.6%) on Dogs. Similarly, using `glo`, LatEm achieves 32.5% accuracy (vs 24.2%) on CUB, 62.9% accuracy (vs. 58.8%) on AWA and 20.9% accuracy (vs. 17.8%) on Dogs. Finally, while LatEm with `hie` on Dogs improves the result to 25.2% from 24.3%, the improvement is more significant on CUB (24.2% from 20.6%) and on AWA (57.5% from 51.2%). These results establish our novel Latent Embeddings (LatEm) as the new state-of-the-art method for zero-shot learning on three datasets in ten out of eleven test settings. They are encouraging, as they quantitatively show that learning piecewise linear latent embeddings indeed capture latent semantics on the class embedding space.

Following [2] we also include a comparison when combining supervised and unsupervised embeddings. The results are given in Tab 3. First, we combine all the embeddings, i.e. `att,w2v,glo,hie` for AWA and CUB. LatEm improves the results over SJE significantly on AWA (76.1% vs 73.9%). Second, we combine the unsupervised class embeddings, i.e. `w2v,glo,hie`, for all datasets. LatEm consistently improves over the combined embeddings obtained with SJE in this setting. On CUB combining `w2v,glo,hie` achieves 34.9% (vs 29.9%), on AWA, it achieves 66.2% (vs 60.1%) and on Dogs, it obtains 36.3% (vs 35.1%). These experiments show that the embeddings contain non-redundant information, therefore the results tend to improve by combining them.

Stability evaluation of zero-shot learning. Zero-shot learning is a challenging problem due to the lack of labeled training data. In other words, during training time, neither images nor class relationships of test classes are seen. As a consequence, zero-shot learning suffers from the difficulty in parameter selection on a zero-shot set-up, i.e. `train`, `val` and `test` classes belong to disjoint sets. In order to get stable estimates of our predictions, we experimented on additional (in our case four) independently and randomly chosen data splits in addition to the standard one. Both with our LatEm and the publicly available implementation of SJE [2] we repeated the experiments five times.

The results are presented in Tab 4. For all datasets, all the result comparisons between SJE and LatEm hold and therefore the conclusions are the same. Although the SJE outperforms LatEm with supervised attributes on CUB, LatEm outperforms the SJE results with supervised attributes on AWA and consistently outperforms all the SJE results obtained with unsupervised class embeddings. The details of our results are as follows. Using supervised class embeddings, i.e. attributes, on AWA, LatEm obtains an impressive 72.5% (vs 70.5%) and using unsupervised embeddings the highest accuracy is observed with `w2v` with 52.3% (vs 49.3%). On CUB, LatEm with `w2v` obtains the highest accuracy among the unsupervised class embeddings with 33.1% (vs 27.7%) On Dogs, LatEm with `hie` obtains the highest accuracy among all the class embeddings, i.e. 25.6% (vs 24.6%). These results insure that our accuracy improvements reported in Tab 2 were not due to a dataset bias. By augmenting the datasets with four more splits, our LatEm obtains a consistent improvement on all the class embeddings on all datasets over the state-of-the-art.

Note that, for completion, in this section we provided a full comparison with the state-of-the-art on all class embeddings, including supervised attributes. However, there are two disadvantages of using attributes. First, since fine-grained object classes share many common properties, we need a large number of attributes which is costly to obtain. Second, attribute annotations need to be done on a dataset basis, i.e. the attributes collected for birds do not work with dogs. Consequently, attribute based methods are not gen-

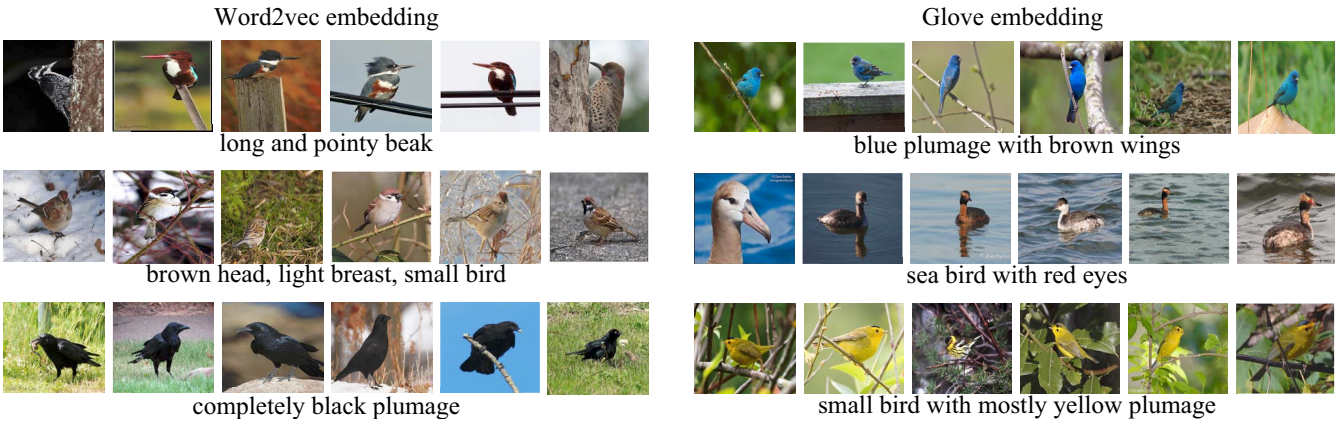

Figure 2: Top images ranked by the matrices using word2vec and glove on CUB dataset, each row corresponds to different matrix in the model. Qualitative examples support our intuition – each latent variable captures certain visual aspects of the bird. Note that, while the images may not belong to the same fine-grained class, they share common visual properties.

eralizable across datasets. Therefore, we are interested in the unsupervised text embeddings settings, i.e. `w2v`, `glo`, `hie`. Moreover, with these unsupervised embeddings, our LatEm outperforms the SJE on nine out of nine cases in all our datasets. For the following sections, we will present results only with `w2v`, `glo` and `hie`.

5.2. Interpretability of latent embeddings

In Sec. 5.1, we have demonstrated that our novel latent embedding method improves the state-of-the-art of zero-shot classification on two fine-grained datasets of birds and dogs, i.e. CUB and Dogs, and one dataset of Animals, i.e. AWA. In this section, we zoom into the challenging CUB dataset and aim to investigate if individual W_i's learn visually consistent and interpretable latent relationships between images and classes. We use word2vec and glove as text embeddings. Fig 2 shows the top scoring images retrieved by three different W_i for the two embeddings i.e. `w2v` and `glo`.

For `w2v`, we observe that the images scored highly by the same W_i (each row) share some visual aspect. The images in the first row are consistently of birds which have long and pointy beaks. Note that they belong to different classes; having a long and pointy beak is one of the shared aspect of birds of these classes. Similarly, for the second row, the retrieved images are of small birds with brown heads and light colored breasts and the last row contains large birds with completely black plumage. These results are interesting because although `w2v` is trained on wikipedia in an unsupervised manner with no notion of attributes, our LatEm is able to (1) infer hidden common properties of classes and (2) support them with visual evidence, leading to a data clustering which is optimized for classification, however also performs well in retrieval.

For `glo`, similar to the results with `w2v`, the top-scoring images using the same W_i consistently show distinguishing visual properties of classes. The first row shows blue birds although belonging to different species, are clustered together which indicates that this matrix captures the "blue"ness of the objects. The second row has exclusively aquatic birds, surrounded by water. Finally, the third row has yellow birds only. Similar to `w2v`, although `glo` is trained in an unsupervised manner, our LatEm is able to bring out the latent information that reflect object attributes and support this with its visual counterpart.

These results clearly demonstrate that our model factorizes the information with visually interpretable relations between classes.

5.3. Pruning vs. cross-validation for model selection

In this section we evaluate the performances obtained with the number of matrices in the model is fixed with pruning vs. cross-validation.

Tab. 5 presents the number of matrices selected by two methods along with their performances on three datasets. In terms of performance, both methods are competitive. Pruning outperforms cross validation on five cases and is outperformed on the remaining six cases. The performance gaps are usually within 1-2% absolute, with the exception of AWA dataset with `att` and `w2v` with 72.5% vs. 70.7% and 52.3% vs. 49.3%, respectively for cross validation and pruning. Hence neither of the methods has a clear advantage in terms of performance, however cross validation is slightly better.

In terms of the model size, cross validation seems to have a slight advantage. It selects a smaller model, hence more space and time efficient one, seven cases out of eleven. The trend is consistent for all class embeddings for the AwA dataset but is mixed for CUB and Dogs. The advantage of pruning over cross-validation is that it is much faster to train – while cross validation requires training and testing with multiple models (once each per every possible choice

	CUB		AWA		Dogs	
	PR	CV	PR	CV	PR	CV
att	3	4	7	2	N/A	N/A
w2v	8	10	8	4	6	8
glo	6	10	7	6	9	4
hie	8	2	7	2	11	10

	CUB		AWA		Dogs	
	PR	CV	PR	CV	PR	CV
att	43.8	**45.6**	63.2	**72.5**	N/A	N/A
w2v	**33.9**	33.1	48.9	**52.3**	**25.0**	24.5
glo	**31.5**	30.7	**51.6**	50.7	18.8	**20.2**
hie	**23.8**	23.7	45.5	**46.2**	25.2	**25.6**

Table 5: (Left) Number of matrices selected (on the original split) and (right) average per-class top-1 accuracy on unseen classes (averaged over five splits). PR: proposed model learnt with pruning, CV: with cross validation.

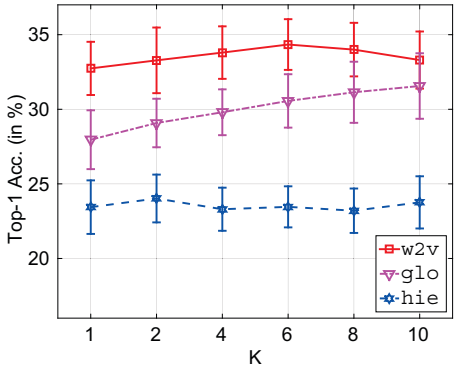

Figure 3: Effect of latent variable K, with unsupervised class embeddings (on CUB dataset with five splits).

of K), pruning just requires training once. There is however another free parameter in pruning i.e. choice of the amount of training data supporting a matrix for it to survive pruning. Arguably, it is more intuitive than setting directly the number of matrices to use instead of cross validating.

5.4. Evaluating the number of latent embeddings

In Sec. 5.1, when we use multiple splits of the data, although the relative performance difference between the state of the art and our method has not changed, for some cases we observe a certain increase or decrease in accuracy. In this section, we investigate the experiments performed with five-folds on the CUB dataset and provide further analysis for a varying number of K. For completeness of the analysis, we also evaluate the single matrix case, namely $K \in \{1, 2, 4, 6, 8, 10\}$ using unsupervised embeddings, i.e. `w2v`, `glo`, `hie`.

Fig. 3 shows the performance of the model with a different number of matrices. We observe that the performance generally increases with increasing K, initially, and then the patterns differ with different embeddings. With `w2v` the performance keeps increasing until $K = 6$ and then starts decreasing, probably due to model overfitting. With `glo` the performance increases until $K = 10$ where the final accuracy is $\approx 5\%$ higher than with $K = 1$. With the `hie` embedding the standard errors do not increase significantly in any of the cases, are similar for all values of K and there is no clear trend in the performance. In conclusion, the variation in performance with K seems to depend of the embeddings used, however, in the zero-shot setting, depending on the data distribution the results may vary up to 5%.

6. Conclusions

We presented a novel latent variable based model, Latent Embeddings (LatEm), for learning a nonlinear (piecewise linear) compatibility function for the task of zero-shot classification. LatEm is a multi-modal method: it uses images and class-level side-information either collected through human annotation or in an unsupervised way from a large text corpus. LatEm incorporates multiple linear compatibility units and allows each image to choose one of them – such choices being the latent variables. We proposed a ranking based objective to learn the model using an efficient and scalable SGD based solver.

We empirically validated our model on three challenging benchmark datasets for zero-shot classification of Birds, Dogs and Animals. We improved the state-of-the-art for zero-shot learning using unsupervised class embeddings on AWA up to 66.2% (vs 60.1%)and on two fine-grained datasets, achieving 34.9% accuracy (vs 29.9%) on CUB as well as achieving 36.3% accuracy (vs 35.1%) on Dogs with word2vec. On AWA, we also improve the accuracy obtained with supervised class embeddings, obtaining 76.1% (vs 73.9%). This demonstrates quantitatively that our method learns a latent structure in the embedding space through multiple matrices. Moreover, we made a qualitative analysis on our results and showed that the latent embeddings learned with our method leads to visual consistencies. Our stability analysis on five dataset folds for all three benchmark datasets showed that our method can generalize well and does not overfit to the current dataset splits. We proposed a new method for selecting the number of latent variables automatically from the data. Such pruning based method speeds the training up and leads to models with competitive space-time complexities cf. the cross-validation based method.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *IEEE TPAMI*, 2015. 1, 2, 3, 5

[2] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of Output Embeddings for Fine-Grained Image Classification. In *CVPR*, 2015. 1, 2, 3, 4, 5, 6

[3] H. Chen, A. Gallagher, and B. Girod. What's in a name? first names as facial attributes. In *CVPR*, 2013. 2

[4] K. Crammer and Y. Singer. On the learnability and design of output codes for multiclass problems. *ML*, 2002. 2, 4

[5] J. Deng, J. Krause, and L. Fei-Fei. Fine-grained crowdsourcing for fine-grained recognition. In *CVPR*, 2013. 4

[6] M. Douze, A. Ramisa, and C. Schmid. Combining attributes and Fisher vectors for efficient image retrieval. In *CVPR*, 2011. 2

[7] K. Duan, D. Parikh, D. J. Crandall, and K. Grauman. Discovering localized attributes for fine-grained recognition. In *CVPR*, 2012. 1, 2, 4

[8] A. Farhadi, I. Endres, and D. Hoiem. Attribute-centric recognition for cross-category generalization. In *CVPR*, 2010. 1, 2

[9] A. Farhadi, I. Endres, D. Hoiem, and D. Forsyth. Describing objects by their attributes. *CVPR*, 2009. 3, 5

[10] P. Felzenszwalb, R. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part based models. *PAMI*, 32(9):1627–1645, 2010. 2, 3

[11] V. Ferrari and A. Zisserman. Learning visual attributes. In *NIPS*, 2007. 1, 2

[12] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, and T. Mikolov. Devise: A deep visual-semantic embedding model. In *NIPS*, 2013. 1, 2, 3

[13] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning (2nd Ed.)*. Springer Series in Statistics. Springer, 2008. 2

[14] S. Huang, M. Elhoseiny, A. M. Elgammal, and D. Yang. Learning hypergraph-regularized attribute predictors. In *CVPR*, 2015. 1, 2

[15] S. Hussain and B. Triggs. Feature sets and dimensionality reduction for visual object detection. In *BMVC*, 2010. 2, 3

[16] T. Joachims. Training linear svms in linear time. In *ACM SIGKDD*, 2006. 2

[17] P. Kankuekul, A. Kawewong, S. Tangruamsub, and O. Hasegawa. Online incremental attribute-based zero-shot learning. In *CVPR*, 2012. 1, 2, 4

[18] A. Khosla, N. Jayadevaprakash, B. Yao, and L. Fei-Fei. Stanford dogs dataset. `http://vision.stanford.edu/aditya86/ImageNetDogs/`. 4

[19] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 3

[20] C. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. In *TPAMI*, 2013. 1, 2, 3, 4, 5

[21] H. Larochelle, D. Erhan, and Y. Bengio. Zero-data learning of new tasks. In *AAAI*, 2008. 1, 2

[22] J. Liu, B. Kuipers, and S. Savarese. Recognizing human actions by attributes. In *CVPR*, 2011. 2

[23] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *NIPS*, 2013. 1, 2, 5

[24] G. A. Miller. Wordnet: a lexical database for english. *CACM*, 38:39–41, 1995. 1, 2, 5

[25] M. Norouzi, T. Mikolov, S. Bengio, Y. Singer, J. Shlens, A. Frome, G. Corrado, and J. Dean. Zero-shot learning by convex combination of semantic embeddings. *arXiv:1312.5650*, 2013. 2

[26] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell. Zero-shot learning with semantic output codes. In *NIPS*, 2009. 2, 3

[27] D. Papadopoulos, A. Clarke, F. Keller, and V. Ferrari. Training object class detectors from eye tracking data. In *ECCV*, 2014. 1, 2

[28] D. Parikh and K. Grauman. Relative attributes. In *ICCV*, 2011. 1, 2

[29] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *EMNLP*, 2014. 1, 2, 5

[30] M. Rohrbach, M. Stark, and B.Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR*, 2011. 1, 2, 4

[31] M. Rohrbach, M. Stark, G. Szarvas, I. Gurevych, and B. Schiele. What helps here – and why? Semantic relatedness for knowledge transfer. In *CVPR*, 2010. 2

[32] B. Romera-Paredes, E. OX, and P. H. Torr. An embarrassingly simple approach to zero-shot learning. In *ICML*, 2015. 1, 2, 3

[33] W. J. Scheirer, N. Kumar, P. N. Belhumeur, and T. E. Boult. Multi-attribute spaces: Calibration for attribute fusion and similarity search. In *CVPR*, 2012. 2

[34] B. Siddiquie, R. Feris, and L. Davis. Image ranking and retrieval based on multi-attribute queries. In *CVPR*, 2011. 2

[35] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *NIPS*, 2013. 2, 3

[36] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *CVPR*, 2015. 1, 5

[37] P. Welinder, S. Branson, T. Mita, C. Wah, F. Schroff, S. Belongie, and P. Perona. Caltech-UCSD Birds 200. Technical Report CNS-TR-2010-001, Caltech, 2010. 4

[38] J. Weston, S. Bengio, and N. Usunier. Wsabie: Scaling up to large vocabulary image annotation. In *IJCAI*, 2011. 2, 3, 4

[39] Y. Yang and D. Ramanan. Articulated pose estimation with flexible mixtures-of-parts. In *CVPR*, 2011. 2, 3

[40] B. Yao, X. Jiang, A. Khosla, A. L. Lin, L. J. Guibas, and F.-F. Li. Human action recognition by learning bases of action attributes and parts. In *ICCV*, 2011. 2

[41] X. Yu and Y. Aloimonos. Attribute-based transfer learning for object categorization with zero or one training example. In *ECCV*, 2010. 1, 2, 4

[42] Z. Zhang and V. Saligrama. Zero-shot learning via joint latent similarity embedding. In *CVPR*, 2016. 2

[43] X. Zhu and D. Ramanan. Face detection, pose estimation, and landmark localization in the wild. In *CVPR*, 2012. 2, 3