Zero Shot Learning Based Object Classification

Prateek Shroff
Texas A&M University
College Station, TX
prateek.shroff@tamu.edu

Stuti Sakhi
Texas A&M University
College Station, TX
stuti@tamu.edu

Abstract

We aim to perform Zero Shot Learning (ZSL) which learns visual classifiers for categories with zero learning examples. The main goal of ZSL is to classify unseen or novel categories which are learn via transferring the knowledge from seen classes to unseen classes. In this project report, we provided an introduction to the Zero Shot Learning methods, its various applications, discuss the related work and the key challenges. The report describes the architecture which consists of two different feature extractors. We propose to use different semantic based feature extractors keeping the visual features the same. Our model follows the paradigm of learning a projection function which facilitates the knowledge transfer from semantic embedding to visual embedding space. Later in the report, we elaborate our training methodologies, implementation details and analyze the experimental results obtained from training all our models and compare our results with various state-of-the-art ZSL models. We conclude by discussing ways to further improve accuracy. Finally, we mention the individual contributions of each member of the team.

1. Introduction

Zero-shot learning refers to the process by which a machine learns how to perform classification without any labeled training data for certain categories. Specifically, we aim to work on the task of zero shot image classification. In case of zero shot image classification, the network learns to classify images from both the seen and the unseen classes using the semantic relation between these classes. This can be explained perfectly by an example, if given the images of a horse along with the information of how a zebra and horse differ (say, zebra is a horse with black and white stripes), the ZSL based model learns to identify both zebra and horse, without seeing a single example of zebra. With the emergence of large-scale datasets, it becomes easy to classify 100s if not 1000s of different categories of objects. But, we that also comes the problem of storing, collecting, annotated

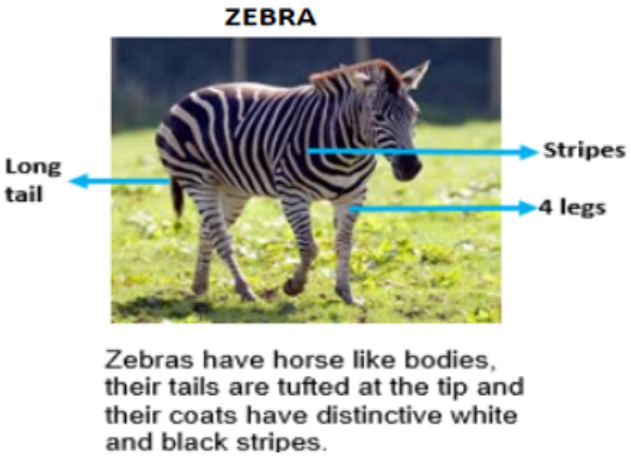

Figure 1. The image provides three different semantic information. (a) "Zebra" word can be used for word embedding. (b) Attributes are "stripes, long tail" (c) a line about the image represent the textual description.

these images. Moreover, the number of images required for a model to classify its category is limited and sometimes tough to obtain. Thus, it poses the problem of scaling in terms of collecting image samples for these rare images and rare categories. The famous Image dataset large-scale visual recognition challenge (ILSVRC) [7] consider the task of recognition 1K classes with 1M images, rather a handful of 21,814 classes with 14M images. As cited in [], many of the 21K categories have a very small number of classes with 296 classes having only one image. Zero shot learning tries to bridge the gap by using auxiliary semantic information instead of visual images to classify an image. Hence, the task of zero shot learning has many applications as it can learn even with the lack of image data by integrating semantic information in the model. This enables us to perform classification for images of rare categories. Even though humans are really good at zero shot learning, this task is really challenging for machines. The most challenging task is to encode the semantic information in such a way that it represents the relations between the seen and unseen

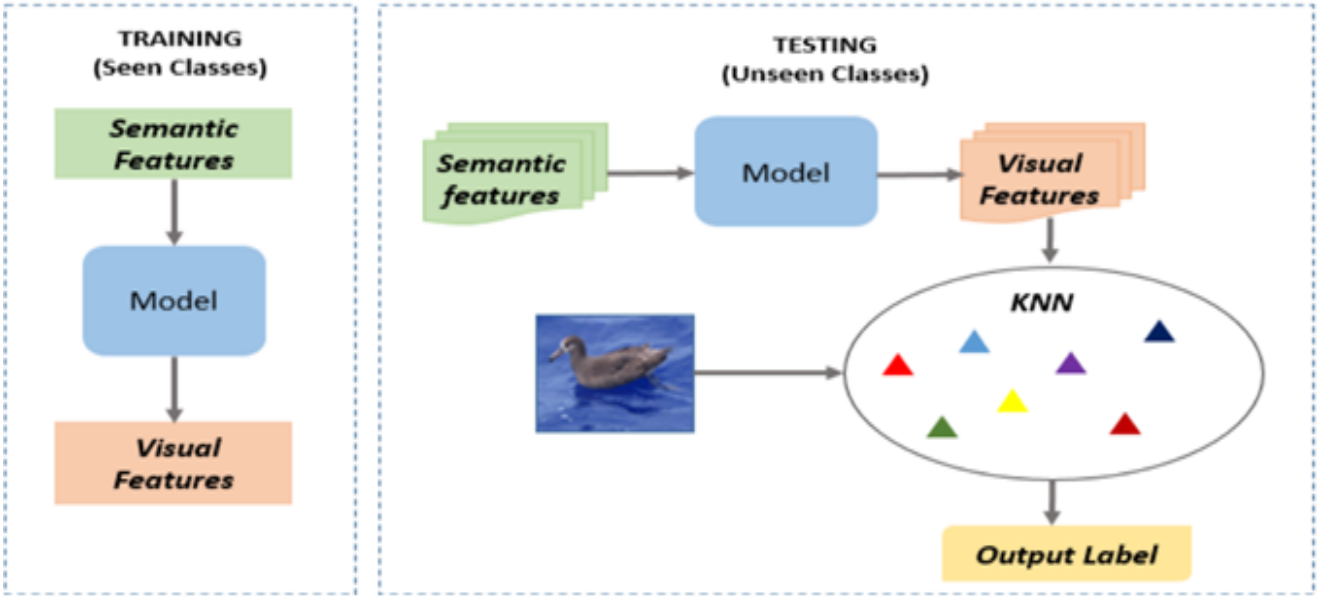

Figure 2. Our two-stage approach to solving ZSL based image classification

classes accurately. This relation via semantic information usually exists in high dimensional space often referred to as semantic space. Depending upon the semantic information being used, such space can be an attributed based [20] or word-vector space [9, 35] or textual description based [43]. Please refer to figure 1 to get an idea about various semantic information. In these spaces, the vector representation is closely related for which semantic information is transferable or close like in our example vectors of zebra and horse.

Many Zero shot learning models learn a joint embedding space between the semantic space and the visual space which helps to relate the semantic features to the visual content of the images. There has been a lot of work in having better joint embedding space (also called visual-semantic space) [27, 31]. Although, there is another paradigm which is less explored. It uses explicit knowledge graphs to represents knowledge, rules, or relationship between objects. Using graphs to model the complex relationship among seen and unseen categories have shown to greatly enhance the performance of the model. This paradigm based paper [40] achieves a new state-of-art performance. While there is a great improvement over the accuracy (at some place over 18%), we found significant difficulty in training such models. Knowledge uses GCN (Graph Convolution Network) to model the relationship. Hence, as categories grow so does the network and its weights. The weights take around 1.5TB space and difficult to transfer. Apart from this factor and the unavailability of the related dataset, we have to drop this paradigm (which we first decided and mentioned in our proposal).

With visual-semantic joint embedding paradigm, ZSL classifies unseen categories in following two steps. 1) A joint embedding is learned when features are projected and 2) during test time a neighbor search algorithm (in our case K-Nearest Neighbour) is used to model to match the unseen class prototypes [] with that of the image features. These two phases are shown in figure **??**. Image features are extracted from a pretrained state-of-the-art CNN based feature extractor. On the other hand, semantic based features extraction is quite different and depends on the type of semantic information. For attributes, the continuous labels are usually hand-annotated and the number of attributes forms the dimension of embedding. For Word-based, neural network based embedding can be extracted using Word2Vec [26] and GloVE [28]. This word embedding can be further fine-tuned depending on dataset and its class label. Finally, the attributes or words do not provide the fine grained subtle difference between very similar classes. Hence, the textual descriptions of the images are used to provide more semantic information. A language modeler network can be used to extract the features from the textual information. In the testing phase, the term 'near' in the k-nearest neighbour can be defined differently in various metrics like Euclidean space, cosine similarity, or learn a metric like Mahalanobis distance.

With the groundwork laid, we would to like to present our contributions: 1) We tried to get semantic feature representation for the class/category by using attributes-based, textual-sentence based and a fusion of both. 2) We also tried to work with different metric (mainly euclidean, cosine similarity, and Mahalanobis) in our embedding space. Much of our work is based on [43] but we made significant

changes to our model structure for the fusion of different semantic metrics. Model [43] uses only mean squared loss while we used a different metric learning technique called Mahalanobis distance which gave us better performance results.

2. Related Works

With the advent of state-of-art deep learning neural networks [14, 36] and large-scale image dataset [7], the task of image classification has become tremendously accurate. However, as the number of classes grows, it becomes increasingly difficult to obtain sufficient numbers of training images for rare objects. Hence, the focus has now shifted to scaling these systems in terms of categories. So, the importance of zero-shot recognition has been gaining traction and over a period of time, a lot of approaches with the aim of recognizing novel objects (zero training examples) has been purposed. Many approaches use some sort of shareable information is required so that knowledge from seen classes can be transferred to unseen classes. Most cases [4, 9, 44] uses semantic embedding to establish this shareable information. The semantic embedding can be formed using attributes of images as done by [19]. However, annotating attributes for images is manual and laborious therefore, class word-based embedding [35] is more popular. Also, SAE [18] proposed a semantic autoencoder to make use of both of the mapping function to enforce the embedding as representative as possible.

There has been a lot of work on type and direction of projection from/to visual embedding to/from semantic embedding. These can be groups into three different type: the First group of models learns a projection function from visual feature space to a semantic space (visual $\rightarrow$ semantic) via using regression or ranking models. Previously, these models [2, 20] were handcrafted but with the rise of the deep neural network, deep neural network regression or ranking [22, 30] is being used. The second group reverses the direction of projection from semantic space to visual feature space (semantic $\rightarrow$ visual) [33]. The third group of projection methods learns a latent space where both the visual and semantic features are projected to [5, 23, 45]. Thus this intermediate space is neither the semantic space nor the visual space. In our method, we went with projecting from semantic space to visual space. As per [43], it is shown that it helps to alleviate the hubness problem [29] in the high dimensional space during K-nearest Neighbour search.

Apart from using only one semantic space, we could fusion multiple semantic spaces to form one representative semantic space. Taking a cue from [43], multiple semantic spaces are often complementary to each other. Fusing them could enrich the semantic information in the space which could improve the model performance. There has been some work on fusion multiple spaces; Akal et [2] fused multiple spaces between attribute, text and their hierarchical relationship. Another form of score level fusion is also applied and tried in [13]. Different from these, our approach jointly learns the space between attributes and textual information and can be trained with visual embedding for end-to-end learning.

The stage to map from one embedding space to another embedding space suffers from domain shift problems [12]. Recently, another popular approach is to establish a knowledge graph. These graphs share statistical relationships among the classes which allows better generalization when faced with a lack of data. Several approaches have been proposed to use knowledge graph for image classifications [24, 25] The approach we are taking is adopted from [40] which uses Graph Convolutions Network (GCN) based on [17] to learn the knowledge graph representation and semantic relationship among the classes. Unlike standard neural networks, graph neural networks retain a state that can represent information from its neighborhood with arbitrary depth. We aim to exploit this property of GCN to draw semantic relations between the classes to identify novel image categories.

3. Methodology

3.1. Problem Statement

Consider a set of 'seen' labelled training samples given by S = $\{(v_n, s_n, y_n), n = 1, 2, 3...., N\}$ and the associated training class label set χ_{tr}, where v_n represents the visual information for the n^{th} training sample, $s_n \in R^L$ denotes the L dimensional semantic information about the n^{th} training sample, and y_n represents the class label of the training sample. The objective now is given a new test image or particularly the visual features (v_t) of the test image, the network should able to estimate a class label belong to test label set ('unseen') χ_{te}. Note that the training class label set χ_{tr} and testing class label set χ_{te} are disjoint i.e., $\chi_{tr} \cap \chi_{te} = \emptyset$.

Basically, we seek to learn a mapping function $f_v : v_n \rightarrow y_n$ and $v_n \rightarrow y_n$ that minimize the empirical risk as given in [30]

$$1/N \sum_{n=1}^{N} \Delta(y_n, f_v(v_n)) + \Delta(y_n, f_s(s_n))$$

where Δ represents 0-1 loss and N represents number of image in training set.

3.2. Model Architecture

3.2.1 Architecture Overview

Our proposed deep-learning network uses CUB dataset [39] for the training. Before feeding into the model, the training data is subjected to pre-processing techniques which have

been discussed in later sections. The whole architecture of the network is composed of two different branches. The first branch is the visual encoding branch. This branch uses a CNN based feature extractor to get the visual embedding of the training image. It inputs an image I_n and after passing through the feature extractor outputs a D dimensional feature vector $\in \mathbb{R}^{D\times 1}$. The second branch is the semantic encoding branch. This branch uses the semantic information associated with the class or image in the form of attributes or textual descriptions. This branch inputs an associated semantic information s_n and outputs a L dimensional feature vector $\in \mathbb{R}^{L\times 1}$.

The output of the visual branch spans a D dimensional feature space where both the visual and the representative semantic features of the training sample will be embedded. To facilitate this, the output of the semantic branch is fed to two fully connected layers with ReLU activation function which converts the L dimensional feature to D dimensional feature. Finally, an appropriate loss function is used to minimize the difference between the visual embedding and its semantic embedding in visual space. This difference is quantified by different metrics which is explained in details in loss section 3.2.4.

3.2.2 Visual Encoding Branch

The task of visual encoding branch is to extract visual features from the images, for which standard state of the art models designed for image classification like Inception-v4 [38], Resnet50 [15] can be used. We tried our network with a number of models individually to check which yields better results. In the following sections, we briefly discuss these models and the architectural changes we employ:

Resnet50[15] Residual networks ease the training of the networks that are substantially deeper by utilizing the skip connections or short-cuts to skip some layers. These residual networks are easier to optimize and can gain accuracy from considerably increased depth. Hence this model was initially used for feature extraction. We removed the final fully connected layer of the model. Also, we replaced the penultimate average pooling layer with global average pooling (GAP) layer. This results in a visual dimension of 2048.

GoogLeNet[37] GoogLeNet architecture is another state of arts model for feature extraction. At the heart, the architecture consists of several inception modules which consist of 11 conv, 33 conv, 55 conv, and 33 max pooling stacked together. And the fully connected layer is replaced by the global average pooling (GAP) layer. This model outputs a visual space of 1024 dimension.

Our model follows the standard practice of initializing weights with pre-trained weights on ImageNet [8]. GoogLeNet gave us better performance as well as better convergence on dataset. Hence all our results are tabulated with GoogLeNet as feature extractor in visual embedding branch.

3.2.3 Semantic Encoding Branch

The aim of semantic encoding branch is to extract the features which were derived from the semantic information. This semantic information can be provided in the form of attributes of the class, word label of class, the fine-grained textual descriptions of the image or fusion of these embedding. Since we required this space to be embedded into visual space we used two fully connected layers. Hence, the whole semantic unit consists of semantic features extractor and two fully connected layers followed by ReLU activation function. In our project, we worked with three semantic spaces (Attributes, Textual Descriptions, and Fusion) which have been described briefly below.

3.2.3.1 Attributes

One of the semantic space consists of the discriminative attributes which are based on different parts representing a class. Some examples of attributes available in CUB dataset are has_upper_tail_color::red , has_bill_shape::dagger. The cross-category property of attributes makes it possible to transfer knowledge from the seen classes to the unseen classes. The corresponding embedding is provided with the CUB dataset. We used a fully connected to convert it to $L^{'}$ and then another fully connected layer to convert $L^{'}$ to D dimension which is the dimension of visual space. Both of the fully connected layers is followed by ReLU activation function. Specifically,

$$s_n = f(W_2(f(W_1 A_n)))$$

where $W_1 \in \mathbb{R}^{L\times L^{'}}$ and $W_2 \in \mathbb{R}^{L^{'}\times D}$ represents weights of first and second fully connected layer respectively. $s_n \in \mathbb{R}^{1\times D}$ is the representative semantic space of attributes and $A_n \in \mathbb{R}^{1\times L}$ is attribute embedding. Also, f represents the ReLU activation function.

3.2.3.2 Text Description

A line or two about the image serves as fine-grained textual information which can provide discriminative features among the seen classes and also the overlapping descriptions words between seen and unseen classes helps to map these two type of classes effectively. For extracting semantic embedding from the descriptions, we first passed the text words through a word embedding layer. This layer uses the GloVe [28] embedding to get features from words. The weights of embedding layers are only initialized with the GloVe embedding but remain trainable. The output of this layer is fed to two Bidirectional Long-Shot Term Memory (BiLSTM) [16]. LSTM improves language modeling

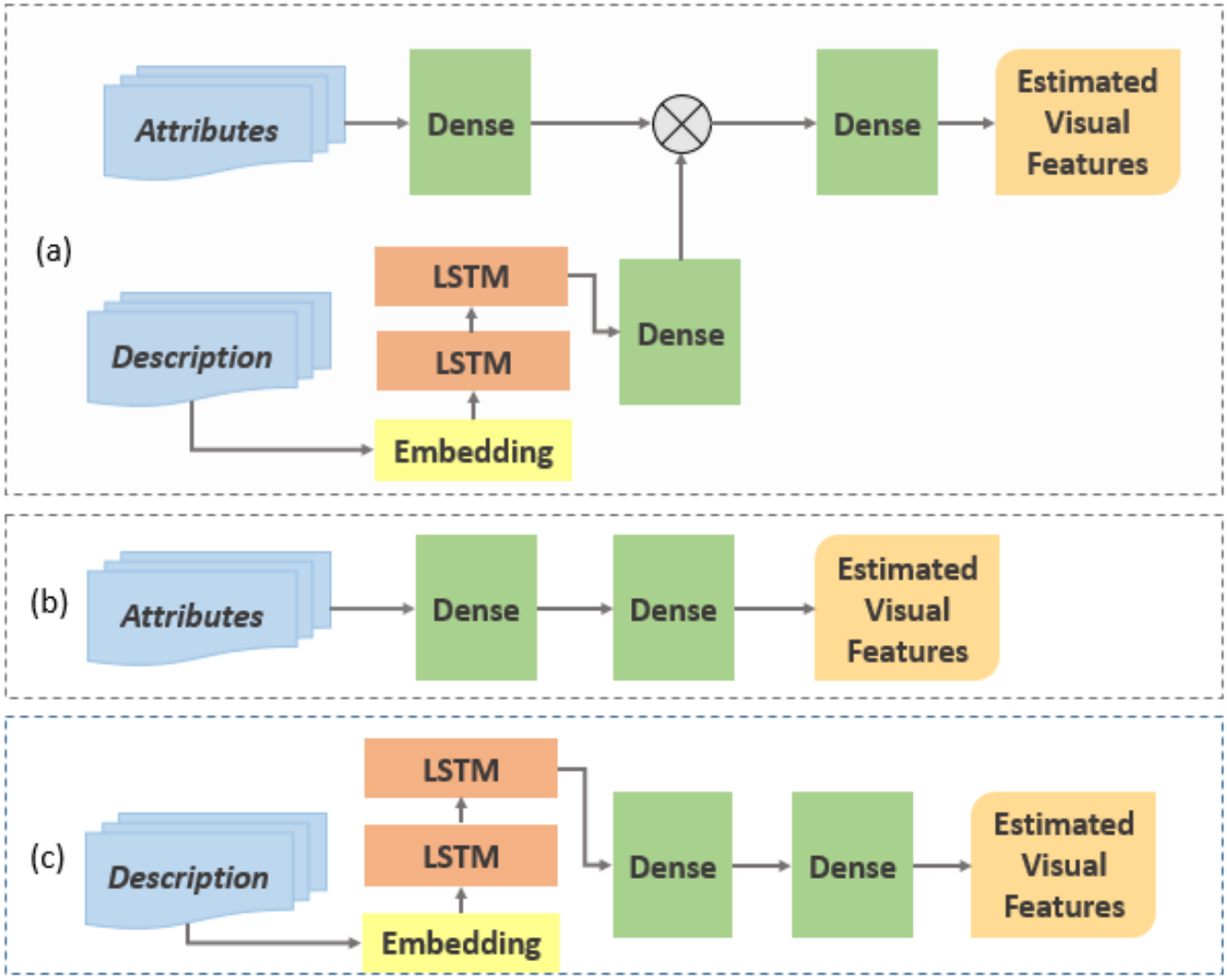

Figure 3. Architecture for fusion network(a), Attribute only network(b), Description only network(c)

over long term dependencies. LSTM makes use of different states: a cell state **c**, a hidden state **h** and different gates: input gate i_t, forget gate f_t, output gate o_t to facilitate flow of information from time step **t** to **t+1**. Finally, the output is taken by averaging the outputs of hidden states of the final BiLSTM layer. Specifically,

$$h_t = W_f * i_t + W_b * i_t$$

where W_f and w_b represents the forward and backward weights of Bidirectional LSTM. h_t and i_t represents output/hidden state and input at time step t. In the testing phase, we create batches of description. Each batch consists of all the descriptions belonging to one class. When this batch is passed through the semantic layer results and averaged at the final layer outputs a feature vector that is the representative class vector. These representative class vectors are called test-prototypes as per [30] and are used to form the underlying visual space for K-Nearest Neighbour (KNN).

3.2.3.3 Fusion of Attribute and Text Description

We argue that the fusion of two different semantic embedding space enriches the semantic information of the class providing better features. Moreover, our architecture optimizes both spaces jointly by training together in an end-to-end manner. But this requires small changes when compared to single semantic space architecture. To be precise, we map different semantic space (specifically, semantic and sentence description) to space where they are added and then the fusion features are converted to visual features. Mathematically,

$$s^{'} = f(W_1 \times s_1) + f(W_2 \times s_2) \quad (1)$$

$$s = f(W_3 \times s^{'}) \quad (2)$$

where $W_1 \in \mathbb{R}^{L_1 \times L^{'}}$ and $W_2 \in \mathbb{R}^{L_2 \times L^{'}}$ represents the weights to project both of the semantic space $s_1 \in \mathbb{R}_1^L$ and $s_2 \in \mathbb{R}_2^L$ onto an intermediate space $s^{'} \in \mathbb{R}^{L^{'}}$. Finally this intermediate space is projected onto visual space $s \in \mathbb{R}^D$. Here again, we use f as ReLU activation function.

3.2.4 Loss functions and KNN metrics

As per fig. , the loss during training is actually the discrepancy in semantic representation and visual feature in visual embedding space. The term 'discrepancy' can be measured in a number of different metrics. Traditionally, the fixed metrics like Euclidean distance, Cosine similarity were used to calculate the distance between two high dimensional vectors which is the same as calculating the distance between two vectors in low - dimension space. But, it turns out to be shallow and is not suitable for every problem. Metric Learning favors learning a suitable metric based on the information provided in the training set. As per the survey in [42], a learned metric can greatly improve the performance of the model. We used the training data to supervisely learn a Mahalanobis distance such that visual features and the representative semantic features in visual space are close to it. In the following subsections, we describe three different metrics we used with to measure the 'discrepancy' in the visual and equivalent semantic embedding.

Euclidean Distance is one of the most popular and widely used metrics to measure the distance between two features in the space. Euclidean distance can be seen as measuring the shortest path between two data point which is defined by the straight line joining the two points. Hence, this distance effectively measures the length of the path connecting them. In our case, given our visual/target feature and the corresponding semantic feature resides in D dimensional feature space. Mathematically,

$$L(V, S) = \sqrt{\sum_{i=1}^{D} (v_i - s_i)^2}$$

where $V \in \mathbb{R}^D$ and $S \in \mathbb{R}^D$ are the features in D dimensional space. $L(V, S)$ is the loss function which we used during the training. In the testing phase, the same metric was used to assign the visual features of test images to test prototypes defined in 3.2.3.2.

Cosine similarity is another fixed metric to measure the similarity between the vectors in multidimensional space. Cosine similarity is actually normalized dot product of two features, hence we are more concern about the orientation of feature rather than their magnitude. By determining the cosine similarity, we would effectively try to find the cosine of the angle between the two vectors. The value ranges from [-1,1] indicating how close the features are; the '1' denotes if the features are in the same direction, '0' denotes the features are orthogonal, and '-1' denotes features in a different direction. In our case, we measure the similarity between the visual vector and the semantic vector in D dimensional embedded space. Mathematically,

$$L(\boldsymbol{s}, \boldsymbol{v}) = \frac{\boldsymbol{s} \cdot \boldsymbol{v}}{||\boldsymbol{s}|| \cdot ||\boldsymbol{v}||}$$

Cosine similarity helps to consider the directional aspect between the two features so that instead of reducing the distance in magnitude we tried to reduce the angle between them making them more align with each other thereby making the semantic vectors close to the visual representation.

Mahalanobis distance is a type of metric learning method which is used to find the distance between two vectors in multivariate space. Basically, it finds the Euclidean distance but also takes into account the covariance of two data distribution. We use it to find the distance between the batch of visual features and corresponding semantic features in visual space. Mathematically,

$$L(s, v) = \sqrt{((s, v) S^{-1} (s, v)^T)}$$

where s and v are semantic-¿ visual feature and visual feature respectively.And S denotes the covariance matrix of s and v. During training, the semantic and visual features are used to calculate the loss using equation 3.2.4 loss. During backpropagation, it doesn't have any parameters, so the covariance matrix is updated with the via weighted average between previous covariance matrix and current covariance calculated based on the current difference in features.

4. Training

In this section, we discuss the training process followed by us. Firstly, we discuss the dataset we have used and then we talk about the parameters of our model

4.1. Dataset

We used the Caltech-UCSD Birds 200 dataset for our project. This dataset contains 6,033 images for 200 categories of birds with attribute information for each class. Apart from the attribute information, we also use the descriptions for each image in our project. We obtained the image features and descriptions as mentioned in [11]. We split the data into groups of 100, 50 and 50 categories for training, validation and test respectively as in [11].

4.2. Model Setting

As in [11], images features were extracted using GoogLeNet[34] pre-trained on ImageNet. Each image feature had a dimension of 1024. The word embedding layer was initialized by glove vectors [7] of dimension 50. The attribute feature for each class had a size of 312. We

used a batch size of 100 to train all the networks and the networks were trained for 25K iterations. The complete code was written using pytorch in python. Now, we mention the model setting for each individual network type.

Attribute Network : Two fully connected layers converted the attribute feature from 312 to 700 and then finally to 1024. Adam optimizer with a learning rate of 0.0005 was used along with a weight decay of 0.02. The gradients for parameters at both the layers were clipped at 1.

Description Network : Two Bidirectional LSTM layers were applied to the output from embedding layer. The final output from LSTM was taken as an average of all the outputs. Two fully connected layers converted the output of LSTM layers from 512 to 700 and then finally to 1024. Adam optimizer with a learning rate of 0.01 was used along with a weight decay of 0.02. The gradients for parameters at both the fully connected layers were clipped at 1.

Fusion Network : A fully connected layer converted the output from the LSTM layers (same as the description only network) from 512 to 700. A fully connected layer converted attribute features from 312 to 700. Finally, the features from both these fully connected layers were averaged together and input into another fully connected layer which gave the final output of size 1024. Adam optimizer with a learning rate of 0.0001 was used along with a weight decay of 0.02. The gradients for parameters at all the fully connected layers were clipped at 1. Also, a scheduler was used which reduced the learning rate by 10 times after 2000 iterations. Note that the network setting remains the same irrespective of the loss function used.

5. Testing

During testing, the description and attribute features for each test class were input into the trained network and the output was averaged to obtain the representative image feature for that class. These representative visual features were used to build the K- Nearest Neighbour Classifier for the test classes. Each new image was classified by first extracting the 1024 dimension feature from pre-trained GoogLeNet and then finding the feature's nearest neighbor in the KNN classifier. The class of the nearest neighbour was output as the class of the test image.

6. Results

In this section, we provide the results we have obtained for zero shot learning on the Caltech-UCSD Birds 200 dataset. Firstly, we show the accuracy we obtained on the test set for different semantic encoding and loss combinations. Then, we compare the performance of our network with the state of the art methods for zero shot learning.

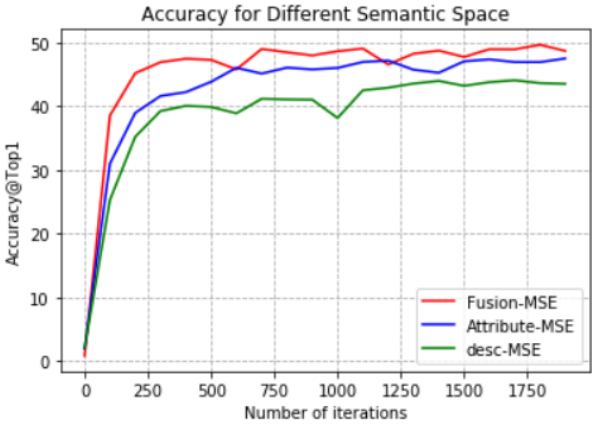

Figure 4. Accuracy plot for different Semantic Encoding types with mean squared loss

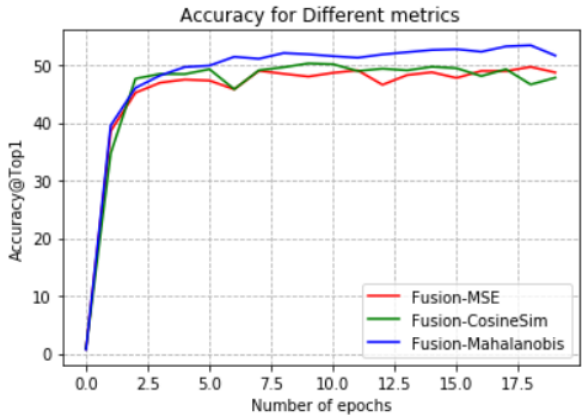

Figure 5. Accuracy plot for different loss function types with fusion semantic encoding

Figure 4 shows the plot for accuracy of test set with increasing iterations for different semantic types - attribute, description, and fusion with Mean Square Loss. The plot clearly indicates that the fusion i.e, a combination of attribute and description performs the best. We take this fusion network ahead and work on finding the best loss function for it. Figure 5 shows the plot for accuracy of test set with increasing iterations for different loss types with fusion semantic encoding. The Mahalanobis distance gives the best performance with an accuracy of 54.2% on the test set. Table 1 gives a summary of the final accuracy obtained for all semantic encoding methods we implemented. Table 2 shows the accuracy for the different loss functions we compared.

Table 2 shows the performances of various state of the art methods for zero shot learning on Caltech-UCSD Birds 200 dataset as compared with our model's performance. We perform better than many states of the art models. Our model

Semantic Encoding	Accuracy(%)
Description	44.6
Attribute	48.9
Fusion	51.1

Table 1. table shows the final accuracy for different semantic types with mean square error loss

Loss Function	Accuracy(%)
Cosine Similarity	50.2
Mean Square Error	51.1
Mahalanobis	54.1

Table 2. table shows the final accuracy for different loss types with fusion semantic encoding

Model	Accuracy(%)
DAP [21]	40.0
DEVISE [10]	52.0
SSE [28]	43.9
SJE [3]	53.9
LATEM [41]	49.3
ESZSL [32]	53.9
ALE [1]	54.9
SYNC [6]	**55.6**
OURS	54.1

Table 3. table compares the performance of our model with the state of the art models in zero shot learning for Caltech-UCSD Birds 200 dataset

which uses the fusion of attributes and description as semantic encoding and the Mahalanobis loss gives the highest performance of 54.1% which is comparable to the state of the art models.

7. Conclusion

In this project, we implemented a model to perform zero shot learning which gives comparable results to various state of the art models. Specifically, we use the attribute and description information of images to perform the classification task. The network learns to predict the representative visual features of a class given the attribute and description features. The architecture of the model provides the flexibility to use various semantic spaces as well as visual feature extractors. At the same time, the model is capable of end-to-end training. We explore various combinations of semantic spaces and loss functions to improve performance. Our implementation gives the best performance with the fusion (attribute and description) semantic space and the Mahalanobis loss function.

8. Future Work

We have explored using only implicit knowledge representations - attributes and description. Models using this information rely on the generalization power of semantic models and mapping models. We could also use explicit knowledge representations like knowledge graphs which represent the relationship between the objects. Using both the implicit and explicit knowledge representations could significantly improve performance.

9. Individual Contribution

Prateek Worked on attribute semantic type and Mahalanobis loss. Stuti Worked on description semantic type and Cosine loss. Both of us together worked on the fusion layer. We have an equal contribution to the project and report.

References

[1] Z. Akata, F. Perronnin, Z. Harchaoui, and C. Schmid. Label-embedding for image classification. *IEEE transactions on pattern analysis and machine intelligence*, 38(7):1425–1438, 2016. 8

[2] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2927–2936, 2015. 3

[3] Z. Akata, S. Reed, D. Walter, H. Lee, and B. Schiele. Evaluation of output embeddings for fine-grained image classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2927–2936, 2015. 8

[4] M. Bucher, S. Herbin, and F. Jurie. Improving semantic embedding consistency by metric learning for zero-shot classiffication. In *European Conference on Computer Vision*, pages 730–746. Springer, 2016. 3

[5] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5327–5336, 2016. 3

[6] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha. Synthesized classifiers for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5327–5336, 2016. 8

[7] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. 2009. 1, 3, 6

[8] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on computer vision and pattern recognition*, pages 248–255. Ieee, 2009. 4

[9] A. Frome, G. S. Corrado, J. Shlens, S. Bengio,

J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013. 2, 3

[10] A. Frome, G. S. Corrado, J. Shlens, S. Bengio, J. Dean, T. Mikolov, et al. Devise: A deep visual-semantic embedding model. In *Advances in neural information processing systems*, pages 2121–2129, 2013. 8

[11] S. Fu, X. Yang, and W. Liu. The comparison of different graph convolutional neural networks for image recognition. In *Proceedings of the 10th International Conference on Internet Multimedia Computing and Service*, page 12. ACM, 2018. 6

[12] Y. Fu, T. M. Hospedales, T. Xiang, and S. Gong. Transductive multi-view zero-shot learning. *IEEE transactions on pattern analysis and machine intelligence*, 37(11):2332–2345, 2015. 3

[13] Z. Fu, T. Xiang, E. Kodirov, and S. Gong. Zero-shot object recognition by semantic manifold distance. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2635–2644, 2015. 3

[14] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 3

[15] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 4

[16] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997. 4

[17] T. N. Kipf and M. Welling. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*, 2016. 3

[18] E. Kodirov, T. Xiang, and S. Gong. Semantic autoencoder for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3174–3183, 2017. 3

[19] C. H. Lampert, H. Nickisch, and S. Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 951–958. IEEE, 2009. 3

[20] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2014. 2, 3

[21] C. H. Lampert, H. Nickisch, and S. Harmeling. Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465, 2014. 8

[22] J. Lei Ba, K. Swersky, S. Fidler, et al. Predicting deep zero-shot convolutional neural networks using textual descriptions. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4247–4255, 2015. 3

[23] Y. Lu. Unsupervised learning on neural network outputs: with application in zero-shot learning. *arXiv preprint arXiv:1506.00990*, 2015. 3

[24] Y. Lu. Unsupervised learning on neural network outputs: with application in zero-shot learning. *arXiv preprint arXiv:1506.00990*, 2015. 3

[25] K. Marino, R. Salakhutdinov, and A. Gupta. The more you know: Using knowledge graphs for image classification. *arXiv preprint arXiv:1612.04844*, 2016. 3

[26] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119, 2013. 2

[27] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell. Zero-shot learning with semantic output codes. In *Advances in neural information processing systems*, pages 1410–1418, 2009. 2

[28] J. Pennington, R. Socher, and C. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543, 2014. 2, 4, 8

[29] M. Radovanović, A. Nanopoulos, and M. Ivanović. Hubs in space: Popular nearest neighbors in high-dimensional data. *Journal of Machine Learning Research*, 11(Sep):2487–2531, 2010. 3

[30] S. Reed, Z. Akata, H. Lee, and B. Schiele. Learning deep representations of fine-grained visual descriptions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 49–58, 2016. 3, 5

[31] M. Rohrbach, M. Stark, and B. Schiele. Evaluating knowledge transfer and zero-shot learning in a large-scale setting. In *CVPR 2011*, pages 1641–1648. IEEE, 2011. 2

[32] B. Romera-Paredes and P. Torr. An embarrassingly simple approach to zero-shot learning. In *International Conference on Machine Learning*, pages 2152–2161, 2015. 8

[33] Y. Shigeto, I. Suzuki, K. Hara, M. Shimbo, and Y. Matsumoto. Ridge regression, hubness, and zero-shot learning. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, pages 135–151. Springer, 2015. 3

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition.

arXiv preprint arXiv:1409.1556, 2014. 6

[35] R. Socher, M. Ganjoo, C. D. Manning, and A. Ng. Zero-shot learning through cross-modal transfer. In *Advances in neural information processing systems*, pages 935–943, 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 *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 3

[37] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. 4

[38] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016. 4

[39] C. Wah, S. Branson, P. Perona, and S. Belongie. Multiclass recognition and part localization with humans in the loop. In *2011 International Conference on Computer Vision*, pages 2524–2531. IEEE, 2011. 3

[40] X. Wang, Y. Ye, and A. Gupta. Zero-shot recognition via semantic embeddings and knowledge graphs. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6857–6866, 2018. 2, 3

[41] Y. Xian, Z. Akata, G. Sharma, Q. Nguyen, M. Hein, and B. Schiele. Latent embeddings for zero-shot classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 69–77, 2016. 8

[42] L. Yang and R. Jin. Distance metric learning: A comprehensive survey. *Michigan State Universiy*, 2(2):4, 2006. 6

[43] L. Zhang, T. Xiang, and S. Gong. Learning a deep embedding model for zero-shot learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2021–2030, 2017. 2, 3

[44] Z. Zhang and V. Saligrama. Zero-shot learning via semantic similarity embedding. In *Proceedings of the IEEE international conference on computer vision*, pages 4166–4174, 2015. 3

[45] Z. Zhang and V. Saligrama. Zero-shot learning via joint latent similarity embedding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6034–6042, 2016. 3