EXPLORING THE LIMITS OF ZERO-SHOT LEARNING - HOW LOW CAN YOU GO ?

by

HEMANTH DANDU

(Under the Direction of Suchendra Bhandarkar)

ABSTRACT

Zero-shot learning aims to classify input data into categories with zero training examples. The classification is performed by inferring unseen categories using visual data of seen categories and their relationships with unseen categories. Relationships are determined by using auxiliary data pertaining to categories such as attributes and semantics. Standard zero-shot learning techniques use a large number of seen categories to predict very few unseen categories while maintaining unified data splits and evaluation metrics. This has enabled the research community to advance notably towards formulating a standard benchmark zero-shot learning algorithm. However, the most substantial impact of zero-shot learning lies in enabling the prediction of a large number of unseen categories from very few seen categories within a specific domain. This permits the collection of training data for only a few previously seen categories, thereby mitigating the training data collection process significantly. In this thesis, we focus on the difficult problem of predicting a large number of unseen object categories from very few previously seen categories. We propose a framework that enables us to examine the limits of inferring several unseen object categories from very few previously seen object categories, i.e., the limits of zero-shot learning. In particular, we examine the functional dependence of the classification accuracy of unseen object classes on the number of previously seen classes. We also determine the minimum number of previously seen classes required to achieve pre-specified classification accuracy for the unseen classes on three standard zero-shot learning data sets, i.e., AWA2, CUB and SUN. Additionally, we compare the proposed framework with a prominent zero-shot learning technique on the aforementioned data sets and find that we achieve 21% higher accuracy on the AWA2 data set, 6% higher accuracy on the CUB data set, and comparable performance on the SUN data set while providing valuable insights into the unseen class inference process.

INDEX WORDS: machine learning, image classification, zero-shot learning, transfer learning

EXPLORING THE LIMITS OF ZERO-SHOT LEARNING - HOW LOW CAN YOU GO ?

by

HEMANTH DANDU

B.Tech., Amrita Vishwa Vidyapeetham University, India, 2015

A Thesis Submitted to the Graduate Faculty of The University of Georgia in Partial Fulfillment of the Requirements for the Degree

MASTER OF SCIENCE

ATHENS, GEORGIA

2020

EXPLORING THE LIMITS OF ZERO-SHOT LEARNING - HOW LOW CAN YOU GO ?

by

HEMANTH DANDU

Major Professor: Suchendra Bhandarkar

Committee: Khaled Rasheed
Frederick Maier
Karan Sharma

Electronic Version Approved:
Ron Walcott
Dean of the Graduate School
The University of Georgia
December 2020

ACKNOWLEDGEMENTS

I would like to thank all my committee members for their time and invaluable suggestions throughout my research. A special thanks to Dr. Bhankarkar and Dr. Sharma for their constant guidance, support, and introduction to this project. I am extremely grateful to my family and friends who have helped me to keep moving and achieve my goals during COVID-19. I would also like to thank everyone at the *Institute for Artificial Intelligence* and the *University of Georgia* for keeping everything running smoothly during these unprecedented times.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

Advances in deep neural networks have empowered machines to achieve human level classification performance on object recognition tasks. Very powerful and robust visual classifier frameworks have been developed, and will no doubt keep improving. In typical object recognition tasks, it is necessary to establish a certain number of predetermined object categories so that classification accuracy can be improved by collecting as many training image samples as possible for each object category. Many problem domains are faced with a large and growing number of object categories. As a consequence, it is becoming increasingly difficult to collect and annotate training data for each object category. Moreover, these images need to capture different aspects of the objects under various imaging conditions to account for the natural variance in appearance for each object category. The problem thus lies in collecting and annotating training data in an efficient and reliable manner for a wide variety of object categories. In addition, trained classifiers can only classify observed object instances into the classes or categories covered by the training data; they lack the ability to deal with previously unseen classes. To address this issue, zero-shot learning (ZSL) techniques have been proposed in the research literature. ZSL frameworks are designed to tackle the problem of learning classifiers when no explicit visual training examples are provided.

Human beings perform ZSL naturally, enabling recognition of at least 30,000 object classes [3]. When faced with a new unfamiliar object, we are, after a while, able to state what it resembles: "A New York City hot dog cart, with the large block being the central food storage and cooking area, the rounded part underneath as a wheel, the large arc on the right as a handle, the funnel as an orange juice squeezer and the various vertical pipes as vents or umbrella supports." It is not a good cart, but we can see how it might be related to one [3]. For humans, it is as easy as recognizing a 10-letter word with 3 wrong letters. However, in the case of machines, we need a vast number of training images for each type of cart to learn to adapt to the naturally occurring variations in cart appearances. In humans, the ability to understand natural variations comes from our existing and ever evolving language knowledge base, which enables us to connect unseen categories with seen

categories using high-level descriptions.

To emulate the ZSL process in machines, previously unseen object categories are recognized by leveraging auxiliary information related to categories. Auxiliary information are derived from external data sources such Wikipedia, WordNet [19] etc. which make it analogous to the human (natural language) knowledge base. As the auxiliary inputs usually carry semantic information, they constitute a *semantic space*. A typical source of semantic information used in ZSL is *attribute spaces*. Attribute spaces are semantic spaces that are engineered manually for each domain or data set. Attributes are a list of terms that describe various properties of each object class or category. For example, an attribute could be *hair color* with values "black", "brown", "white" etc. The attributes can be either discrete or continuous. Label-embedding spaces are also often used as a source of semantic information, where the word/label representations are obtained by employing information retrieval techniques on large digital text corpora. Examples of widely used label-embedding models include Word2Vec [18], GloVe [24], and FastText [4]. Hierarchical information is another source of semantic information that can be derived from a pre-existing ontology such as WordNet [19]. All sources of auxiliary information when combined together comprise the semantic space.

In typical ZSL frameworks, a set of previously observed classes is used to train the visual classifier. These classes are termed as *seen classes*. The framework is then evaluated on another set of previously not observed classes termed as *unseen classes*. While training, the classifier has access to auxiliary information of both the seen and unseen classes. A formal definition of the ZSL task is provided in Definition 1.1. Conventional ZSL is restrictive in its formulation since it assumes that the input images at the time of prediction or inference can only come from the unseen classes. In contrast, generalized ZSL addresses the more general setting where the input images at the time of prediction or inference can come from both, the seen and unseen classes [27]. Generalized ZSL is formally defined in Definition 1.2.

Definition 1.1 (Conventional Zero-shot Learning)**.** Given labelled training instances X_s belonging to seen classes Y_s, zero-shot learning aims to learn a classifier that can classify testing instances X_u belonging to the unseen classes Y_u

Definition 1.2 (Generalised Zero-shot Learning)**.** Given labelled training instances X_s belonging to seen classes Y_s, generalised zero-shot learning aims to learn a classifier that can classify testing instances $X_{u \cup s}$ belonging to the classes $Y_u \cup Y_s$

Several ZSL frameworks have been proposed in the literature, however, all of these frameworks use a proposed split [33] of standard ZSL data sets [14, 32, 23] into seen and unseen classes. This split is formulated in to aid uniform research towards finding a universal ZSL framework that outperforms the existing ones. Altogether, the zero-shot learning problem has been formally framed for each standard data set using specific categories as seen classes and the remaining as unseen classes in a race for attaining maximum classification accuracy. Of the total object categories present in each data set, the number of seen classes has always been significantly higher than the number of unseen classes in most ZSL frameworks. For example, the *Animals-with-Attributes* (AWA2) data set [14] has a proposed 40:10 seen:unseen class split, the *Caltech-USCD-Birds* (CUB) data set [32] has a 150:50 seen:unseen class split, and the large-scale *Scene Understanding* (SUN) database [23] has a 645:72 seen:unseen class split. While this formulation has helped to formulate several benchmark approaches to ZSL tasks, we notice that the original intent of mitigating the data collection process has been skirted at a very early stage. Therefore, we aim to infer larger number of unseen object categories using very few seen object categories. We believe this addresses the original problem of obtaining annotated images to a greater extent.

We propose a new framework that helps us to examine the limits of inferring unseen object categories from very few seen object categories, i.e., test the limits of ZSL. We note the functional dependence of the classification accuracy on the number of previously seen classes across the spectrum of the classes on three widely used object classification data sets [14, 32, 23]. An important contribution of the proposed approach is its ability to determine the optimal set of representative classes using which one could infer a large number of previously unseen classes with a prespecified measure of accuracy. We explore intuitive techniques to select a few seen classes which would enable us to predict a larger number of unseen classes. The proposed approach also aids the training data collection process significantly by identifying the key object categories from which the training data collection process can be initiated and determining which object categories to stop at,

based on an expected or pre-specified classification accuracy measure for a specific problem. We evaluate the proposed approach in the generalized ZSL setting, thus making it very practical. We present valuable insights into the inference process for general and specific cases where the proposed approach performs exceptionally well, and also for cases where we fail to infer the correct unseen category. We also compare the proposed approach with the well known Attribute Label Embedding (ALE) [1] procedure, which has been shown to perform very well on the aforementioned three standard data sets as published in [33]. In comparison to ALE, we observe that the proposed approach achieves 21% higher accuracy on the AWA2 data set, 6% higher accuracy on the CUB data set and comparable performance on the SUN data set. We also establish the minimum number of previously seen classes needed to obtain reasonable (or above average) generalized ZSL performance on the AWA2 data set as 20 seen classes out of a total of 50 classes, on the CUB data set as 80 seen classes out of a total of 200 classes and on the SUN data set as 360 seen classes out of a total of 717 classes.

This thesis is organized into six chapters. Chapter 1 introduces the concept of zero-shot learning (ZSL) and summarizes the work done in this thesis. Chapter 2 reviews the related work in ZSL and position our work in the overall ZSL research literature. Chapter 3 describes the various data sets used for experiments carried out in this thesis. Chapter 4 discusses the overall methodology underlying the proposed approach and also explains the finer details about the methods used. Chapter 5 presents the experimental results of the proposed approach on the aforementioned three data sets. Chapter 5 compares the proposed approach with the Attribute Label Embedding (ALE) scheme and shows how well the proposed approach fares in comparison to the widely used ALE-based ZSL framework. Finally, in Chapter 6, we conclude this thesis and discuss directions for future work.

CHAPTER 2

LITERATURE REVIEW

Zero-shot learning (ZSL) approaches can be broadly classified into two categories based on the unseen class information the model has during the training process. In the inductive ZSL framework, we have access to to labeled image data from the seen classes as well as auxiliary information (i.e., semantic attributes/descriptions) about both, seen and unseen classes during the training phase. In the transductive ZSL framework, we have access to auxiliary information (i.e., semantic attributes/descriptions) about both, seen and unseen classes during the training phase as in the case of the inductive ZSL framework. The major difference is that in the case of transductive ZSL, we have access to labeled image data from the seen classes *and* unlabeled image data from the unseen classes during the training phase which is a departure from inductive ZSL. Within both frameworks, one can make an distinction based on the type of setting used to evaluate the model during testing. i.e. the conventional ZSL setting and generalized ZSL setting as described in Chapter 1. In this section we review work on both the inductive and transductive ZSL frameworks and place the proposed approach within the ZSL taxonomy.

Preliminary work in inductive ZSL uses a two-stage approach to infer unseen class labels. In the first stage, the attributes of an image are predicted. In the next stage, the class label is inferred by searching for the class label with the most similar set of attributes. Lampert et al [13] introduced the Directed Attribute Prediction (DAP) and Indirect Attribute Prediction (IAP) models which use the aforementioned two-stage approach. In DAP [13], a probabilistic attribute classifier is first learned. The class posteriors are then computed and class labels predicted via a maximum a posteriori (MAP) estimate. In IAP [13], a multi-class classifier is first used to predict the class posterior. The probability of each class is then used to compute the attribute posteriors of an image. While the DAP and IAP frameworks have historically been some of the most widely cited ZSL methods in the literature, they suffer from the problem of domain shift [9] where the intermediate functions learned from the auxiliary information without any adaptation to the target domain introduce an unknown bias.

Subsequent ZSL frameworks attempt to learn a compatibility function from image feature space to the semantic or auxiliary space. These frameworks can be further categorized based on the type of compatibility function that they learn. The first set of methods learn linear compatibility functions whereas the next set of methods learn non-linear compatibility functions.

Linear Compatibility. Attribute Label Embedding (ALE) [1] learns a bi-linear compatibility function between the image space and auxiliary space using a weighted approximate ranking objective. ALE improves upon DAP [13] significantly since it can use multiple sources of auxiliary information such as word embeddings and class taxonomies. ALE also overcomes the drawbacks of the two-step process used in DAP by directly predicting the class label without the need for an intermediate step. The Deep Visual-Semantic Embedding (DEVISE) model [8] also learns a linear mapping between the image space and semantic space and has been shown to perform well on the large-scale ImageNet data set. The Structured Joint Embedding (SJE) scheme [2] uses an unregularized structured SVM to learn the compatibility function coupled with the Stochastic Gradient Descent (SGD) algorithm for optimization. The Embarrassingly Simple Zero-Shot Learning (ESZSL) scheme [25] uses an additional regularization term to suppress noise in the auxiliary space.

Non-Linear Compatibility. The Latent Embedding (LATEM) scheme [35] extends linear compatibility approaches by learning multiple mappings thereby finding a piece-wise linear compatibility function using every image-class pair. LATEM shows improved accuracy over the state-of-art the linear compatibility-based SJE scheme [2]. The Cross-Modal Transfer (CMT) scheme [28] uses a neural network with two hidden layers to learn non-linear projections from image space to Word2Vec [18] space. CMT exploits only information from word embeddings and does not use other sources of auxiliary information such as class attributes used by other methods.

Drawing from linear and non-linear compatibility approaches, hybrid models [33] learn a joint embedding of both the image and semantic features into a combined intermediate space. The Semantic Similarity Embedding (SSE) scheme [36] uses a max-margin framework to jointly optimize domain data and semantic data. The Convex Combination of Semantic Embeddings (CONSE) scheme [22] is inspired by DEVISE [8] and maps images into the semantic embedding space via convex combination of the class label embedding vectors without the need for additional training.

Synthesized Classifiers (SYNC) [5] introduces a set of "phantom" object classes whose coordinates exist in both the semantic space and the model space which are then optimized using labeled data such that the synthesized real object classifiers achieve optimal discriminating performance. Wang et al. [31] use a Graph Convolution Network (GCN) and the GLoVe text embedding model [24] to generate a knowledge graph embedding. The knowledge graph embedding exploits both, semantic embeddings and domain relationships to predict the object classifiers.

Recently, there has been a rise in the use of generative models for ZSL that represent each class as a probability distribution. Generative Framework for Zero-Shot Learning (GFZSL) [29] models the class-conditional distributions of seen as well as unseen classes using a multi-variate Gaussian distribution. Generative models such as GFZSL exhibit a significant performance boost in the transductive ZSL setting. The Feature Generating Network (FGN) [34] introduces a novel Generative Adversarial Network (GAN) that synthesizes Convolutional Neural Network (CNN) features conditioned on class-level semantic information, offering a direct shortcut from a semantic descriptor of a class to a class-conditional feature distribution. The Leveraging the Invariant Side GAN (Lis-GAN) approach [16] generates unseen features from random noise functions which are conditioned by the semantic descriptions. They train a conditional Wasserstein GAN in which the generator synthesizes fake unseen features from noises and the discriminator distinguishes the fake from real via a minimax game. The recent approach based on leveraging the semantic relationships between the seen and unseen object categories termed as LsrGAN [30] performs explicit knowledge transfer by incorporating a novel Semantic Regularized Loss (SR-Loss) function. A Tensorflow-based combination of a Variable Auto-Encoder (VAE) and GAN, termed as TF-vaegan [21], introduces a feedback loop from a semantic embedding decoder, that iteratively refines the generated features during both the training and feature synthesis stages. The synthesized features together with their corresponding latent embeddings from the decoder are then transformed into discriminative features and exploited during classification to reduce the ambiguities amongst the categories. The TF-vaegan framework [21] is currently regarded as the benchmark in inductive and transductive ZSL settings on the AWA2 [14], CUB [32], and SUN [23] data sets on the zero-shot learning and generalized zero-shot learning settings.

The proposed framework draws from the two-stage approach used by the DAP and IAP approaches [13] but the problem being addressed is substantially different from the standard ZSL problem that the aforementioned ZSL frameworks are designed for. Note that the proposed approach aims to predict a large number of unseen classes from few seen classes, and consequently, we have far less training data compared to the conventional ZSL settings. Hence, generative-adversarial-based and compatibility learning-based ZSL frameworks which require a lot of training data would be expected to perform poorly in this situation. The proposed ZSL framework represents a first step towards a novel ZSL problem formulation that strives to understand how classification accuracy measures change with change in number and type of seen classes. The proposed framework also allows for selection of an optimal number and type of seen classes based on an expected overall classification accuracy measure which aids in the training data collection process.

CHAPTER 3

DATA SETS

This chapter describes the three data sets used in the experiments in this thesis. Table 1 shows the summary statistics for all three data sets.

3.1 ANIMALS WITH ATTRIBUTES-2 (AWA2)

The Animals with Attributes (AWA) data set was originally introduced by Lampert et al. [13]. Since the original images from AWA were not publicly available, Xian et al. [33] enhanced the AWA data set and termed it as Animals with Attributes-2 (AWA2) [14], while retaining the same animal classes and attributes as AWA. AWA2 has a total of 37,322 images compared to 30,475 images in AWA. AWA2 is considered a coarse-grained data set that is medium-scale in terms of images and small-scale in terms of number of classes, i.e. 50 animal classes. The data also provides 85 numeric attribute values for each class. Using the shared attributes, it is possible to transfer information between different classes. The AWA2 data set permits evaluation of the proposed approach in a coarse-grained domain with a small number of classes. A few examples of animal classes in AWA2 are *polar bear*, *zebra*, *otter*, and *tiger*. Figure 1 shows a few images from the AWA2 data set where the image data was collected from public sources, such as Flickr in 2016. Further information about the data set can be found here [1].

3.2 CALTECH-UCSD BIRDS (CUB)

The Caltech-UCSD Birds-200-2011 (CUB) data set [32] is an image data set with 11,788 photographs of 200 bird species. Each species is associated with a Wikipedia article and organized by its scientific classification, i.e., (order, family, genus, species). The list of species names was obtained using an online field guide. Each image is annotated with a bounding box, part location, and attribute labels, thereby providing 312 numeric attribute values for each class. CUB is consid-

[1] https://cvml.ist.ac.at/AwA2/

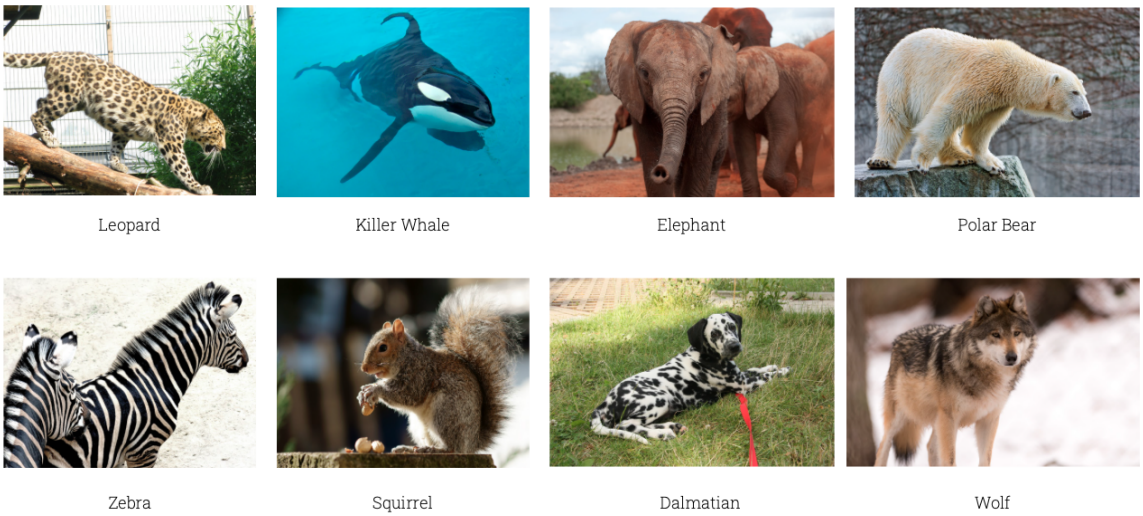

Figure 1: A few labelled images from the Animals with Attributes-2 (AWA2) data set.

ered a fine-grained data set that is medium-scale in terms of both images and classes. This data set allows evaluation of the proposed approach in a fine-grained domain with a moderate number of classes. A few examples of bird species are *Long tailed Jaeger*, *Blue-winged Warbler*, *American Crow*, *Louisiana Waterthrush*, and *Herring Gull*. Figure 2 shows a few images from the CUB data set where the images were collected using Flickr image search and then filtered by showing each image to multiple users. Further information about the data set can be found here [2].

3.3 SCENE UNDERSTANDING WITH ATTRIBUTES DATABASE (SUN)

The Scene Understanding with Attributes database (SUN) [23] is the first large-scale scene attributes database. A crowd-sourced human study was used to establish a taxonomy of 102 discriminating attributes and the SUN attributes database was built on top of the fine-grained SUN categorical database. The SUN attributes database covers 717 categories and has 14,340 images. SUN is considered a fine-grained data set that is of medium scale in terms of number of images but large scale in terms of number of classes. This data set allows evaluation of the proposed approach in a fine-grained domain with a large number of classes. Few examples of scene classes in SUN are *street*, *indoor brewery*, *valley*, *classroom*, and *supermarket*. The SUN data set also provides a two-

[2] http://www.vision.caltech.edu/visipedia/CUB-200-2011.html

Figure 2: A few labelled images from the Caltech-UCSD Birds (CUB) data set.

level hierarchy for each of the 717 categories. The first level classifies scenes into broad categories such as *indoor*, *outdoor-natural*, and *outdoor-man-made*. The second level further classifies each of the first-level scenes into finer categories such as *workplace*, *shopping*, *forest*, *sports*, *cultural* etc. Figure 3 shows a few images from the SUN data set. The SUN data set is publicly available and further information about the data set can be found here [3].

Table 1: Summary Statistics for all three Data Sets

Data set	**Detail**	**Classes**	**Images**	**Attributes**
AWA2	coarse	50	37,322	85
CUB	fine	200	11,788	312
SUN	fine	717	14,340	102

[3]http://cs.brown.edu/ gmpatter/sunattributes.html

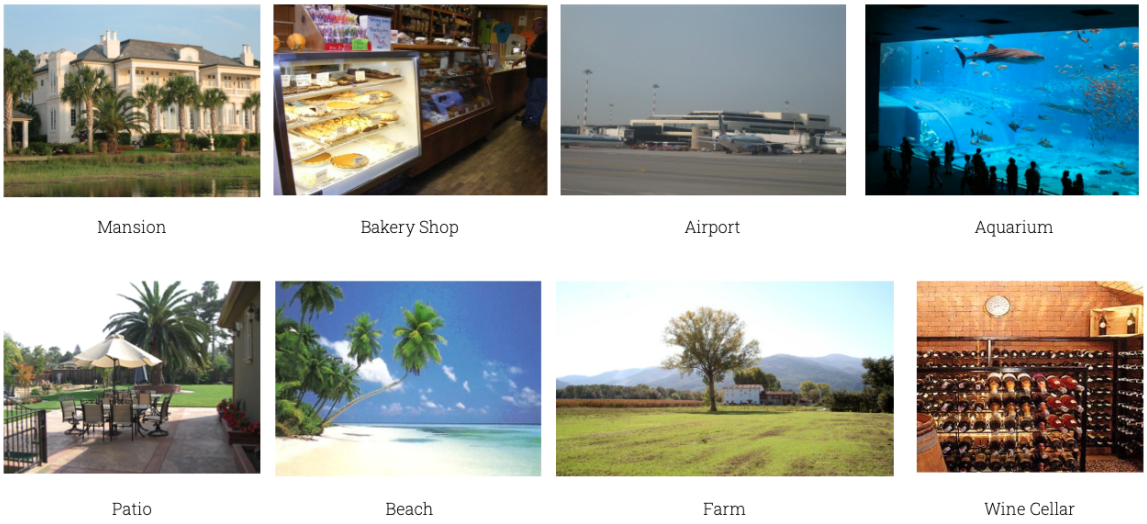

Figure 3: A few labelled images from the Scene Understanding with Attributes (SUN) data set.

CHAPTER 4

METHODOLOGY

Broadly, the proposed framework comprises of the following components:

1. Deep feature extraction
2. Incorporation of auxiliary information
3. Clustering of auxiliary Information
4. Multi-label classification of deep features
5. Predictions of categories/classes

Generally speaking, raw input images carry excess information that is not necessary for the purpose of discrimination and/or fine-grained analysis. Feature extraction enables us to collect important relevant features from an image while discarding the unnecessary information. The deep feature extraction component uses deep neural networks to extract essential information from each image. The first section in this chapter details the feature extraction procedure. Auxiliary information is used by ZSL techniques to learn the relationships between the seen and unseen classes. The second section in this chapter describes the various types of auxiliary information used within the proposed framework. In order to identify representative classes that enable us to predict the large number of unseen classes, we cluster the auxiliary information to find representative cluster centers. The third section in this chapter details the various clustering techniques used. A multi-label classifier is then trained which can discriminate between the representative classes identified in the clustering phase. The fourth section in this chapter describes the classifiers used and their parameters. For a test sample, once the appropriate representative cluster is identified, predictions about unseen classes are made by using a hypothesis generation procedure based on distance measures in the auxiliary space. The fifth section in this chapter describes the distance metrics used to generate the hypotheses from the auxiliary space. Figure 4 depicts a high-level schematic of all the components in the proposed framework.

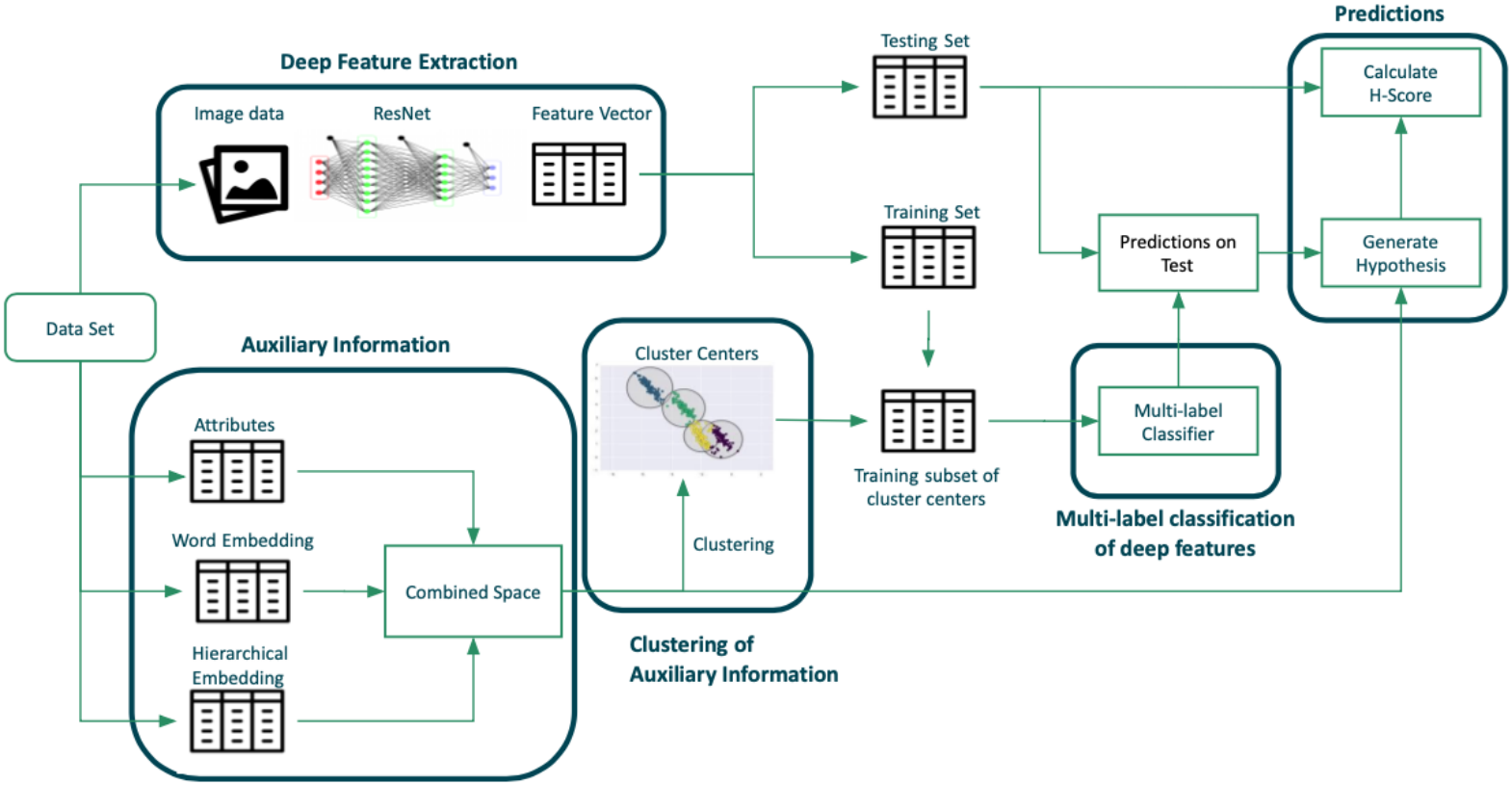

Figure 4: High-level schematic of the proposed framework.

4.1 DEEP FEATURE EXTRACTION

Deep neural networks (DNNs) have proven to perform well for feature extraction from images. We use a ResNet DNN [11] with a depth of 101 layers which is pre-trained on the ImageNet data set [6]. This reduces training time significantly and yields useful features since ImageNet consists of over 1 million images spanning 1000 categories. We extract features from the last ResNet layer which translates to 2048 features for each image. ResNet features for all three data sets are also publicly available [33]. Figure 5 shows the architecture and feature extraction map for an image using ResNet-101.

Extracted visual features are split into training and testing sets using a stratified 80:20 split. These splits are saved separately for further use in model training and in the prediction phases. Stratified sampling is used so that instances of seen and unseen classes are present in the test set, thus making this a generalized zero-shot learning setting. A seed is used during the splitting phase so that the results can be reproduced. Table 2 shows a summary of training and testing splits for each data set.

Table 2: Training and testing splits for each data set

Data Set	Classes	Total Images	Training Set Images	Testing Set Images
AWA2	50	37,322	29,857	7,465
CUB	200	11,788	9,430	2,358
SUN	717	14,340	11,472	2,868

4.2 AUXILIARY INFORMATION

Since instances of unseen classes are unavailable during the training process, auxiliary information is needed to establish semantic relationships between seen and unseen classes, which in turn helps address the ZSL problem. In this framework, we use three sources of auxiliary information for each of the three data sets.

Attributes. Humans can naturally perform zero-shot learning with the help of semantic back-

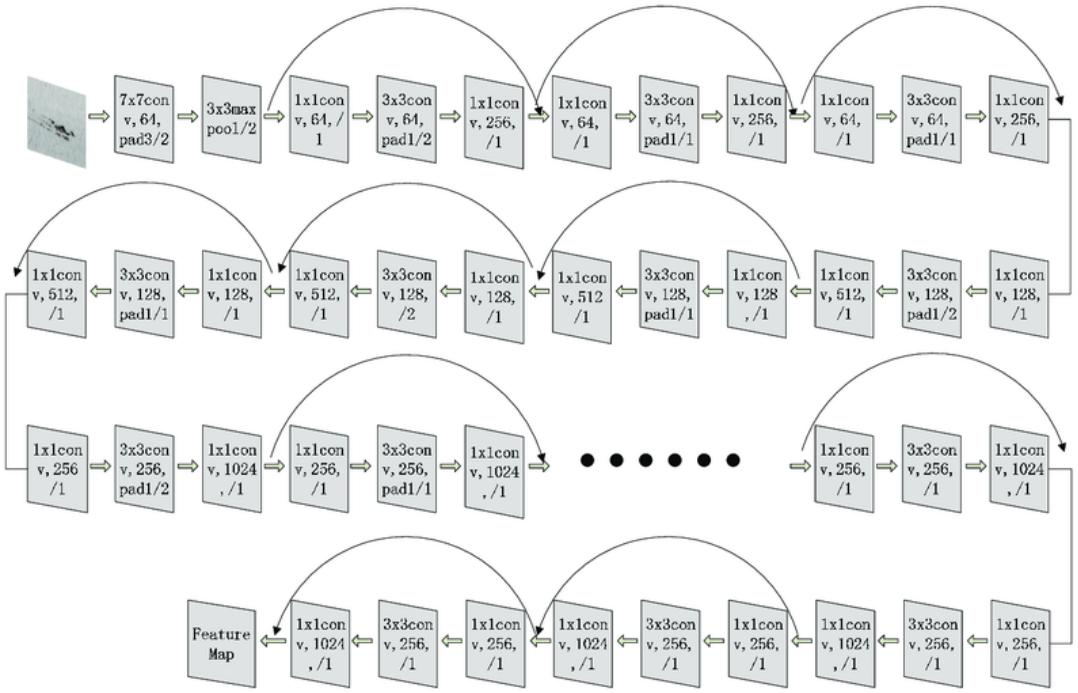

Figure 5: ResNet-101 feature extraction map [12]. Each image passes through the network and a feature vector is extracted from the last layer.

ground information. For instance, knowing that "a zebra looks like a horse with stripes" allows us to recognize a zebra without having seen one before, as long as we know what a horse looks like and what the pattern "stripe" looks like [10]. All three data sets that are used in this experiment have labelled attribute information for each of their classes. The AWA2 data set [14] includes attributes such as *black*, *small*, *walks*, *smart* etc. with 85 such attributes for each class. The CUB data set [32] includes attributes such as *primary color*, *wing color*, *wing shape*, *size* etc. with 312 such attributes for each class. The SUN data set [23] includes attributes such as *man-made*, *natural light*, *medical activity*, *diving* etc. with 102 such attributes for each class.

Text Embeddings. A popular idea in modern machine learning is to represent words by vectors that capture hidden information about a language. The learned vector representations for words (i.e., word/text embeddings) from a large general text corpus can help to construct semantic relationships between the seen and unseen class labels. There are three widely used word embeddings in the research literature: Word2Vec [18], GloVe [24], and FastText [4]. FastText has been

shown to perform better than the others since it treats each word as composed of character n-grams. In FastText the vector for each word is composed as the sum of its character n-grams whereas, in Word2Vec and GloVe each word in the corpus is treated as an atomic entity and a unique vector is generated for each word. As a result, FastText can also generate vectors for a combination of words. For instance, "polar bear" has a unique FastText vector representation. FastText has been trained on a very large Wikipedia corpus and is publicly available. In this work, we extract FastText word representations for each class label resulting in a 300-dimensional vector for each class label.

Hierarchy Embeddings. Creating a hierarchy of categories present in a data set allows us to derive taxonomy-based relationships between the classes and improve ZSL performance. For the AWA2 and CUB data sets, Lee et al. [15] propose a two-stage approach for generating hierarchy embeddings where they first derive a top-down hierarchy using WordNet [19] and then create a flattened hierarchy by representing the probabilities of all the leaf nodes as a single probability vector. Figure 6 illustrates how hierarchy embeddings are generated for each leaf class in the classification tree. In the case of the AWA2 data set, this results in a 61-dimensional vector and in the case of the CUB data set a 193-dimensional vector. The SUN data set provides its own two-level hierarchy information for each of its 717 categories resulting in a 19-dimensional vector.

Combined Semantic Space. The vector spaces of attributes, text embeddings, and hierarchy embeddings are combined into a unified space with reduced dimensions, while retaining the most important information. The dimensionality reduction of the semantic space reduces the computational complexity of the clustering phase and creates robust clusters. We use Principal Component Analysis (PCA) for dimensionality reduction since it retains the variance in the input data while reducing the data dimensionality resulting in a compact combined semantic space. PCA achieves dimensionality reduction by re-projecting the original data along the principal component axes where the principal components are determined via eigenvalue decomposition of the data covariance matrix. The combined, reduced-dimensional semantic space for each data set is computed and retained for use in the clustering and prediction phases. Figures 7, 8, and 9 show the t-distributed stochastic neighbour embedding (t-SNE) plots [17] of the combined semantic space for the AWA2, CUB, and SUN data sets respectively. The t-SNE [17] is a non-linear dimensionality reduction

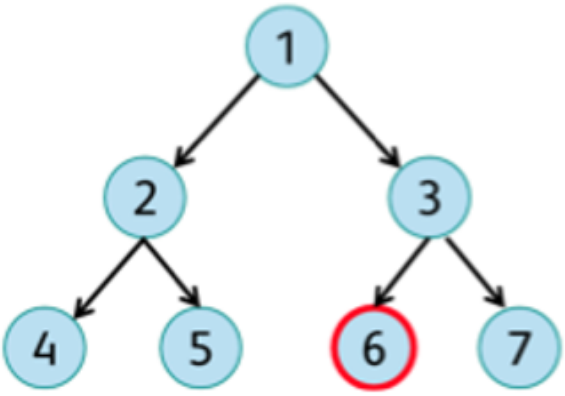

Illustration of Hierarchy embedding. Given 7 classes, class 6 is encoded as H(6) = [1,0,1,0,0,1,0]

Figure 6: An illustration of hierarchy embedding [1].

technique that embeds high-dimensional data into a low-dimensional space to aid in data visualization. Specifically, it models each high-dimensional vector as a two- or three-dimensional vector such that similar objects are closer and dissimilar objects are farther apart in the resulting two- or three-dimensional space. In the t-SNE plots for the CUB and SUN data sets, in Figures 8 and 9 respectively, 50 random classes are depicted in the interest of readability. In the t-SNE plots, we observe that classes in the CUB and SUN data set group closer to each other compared to those in the AWA2 data set. This is because AWA2 is a coarse-grained data set whereas CUB and SUN are fine-grained data sets.

4.3 CLUSTERING OF AUXILIARY INFORMATION

The goal of the clustering stage is to identify object categories that are good representatives for a large number of similar object categories. The underlying idea is that these cluster centers would have a strong relationship with its cluster members, thus aiding us to infer cluster members using the cluster centers alone. We use two clustering techniques, i.e., the Gaussian Mixture Model

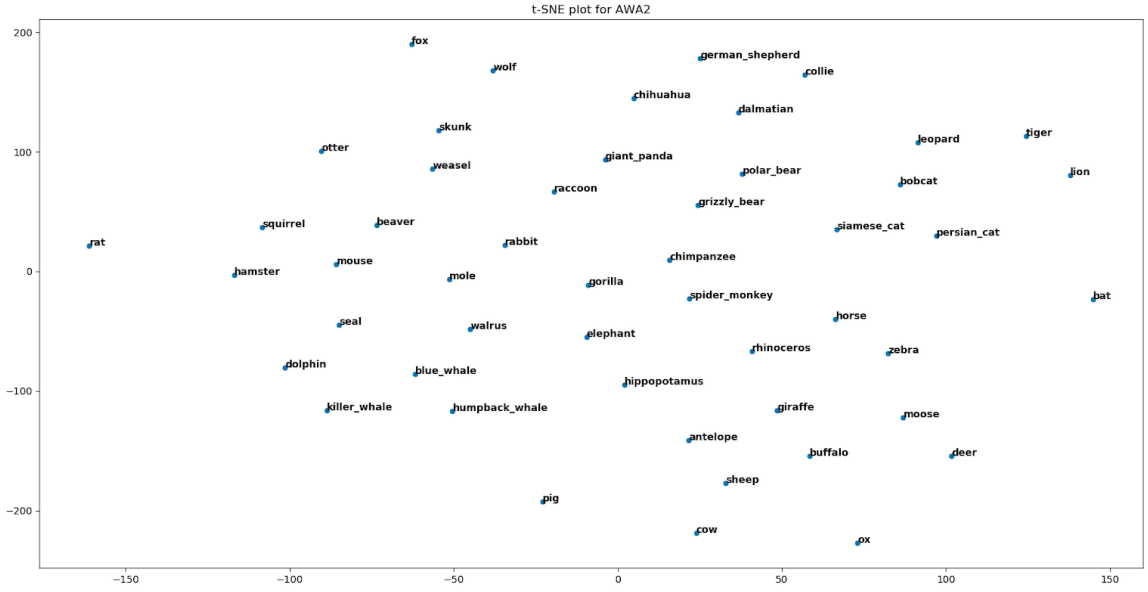

Figure 7: t-SNE plot of the combined semantic space in the AWA2 data set with all classes.

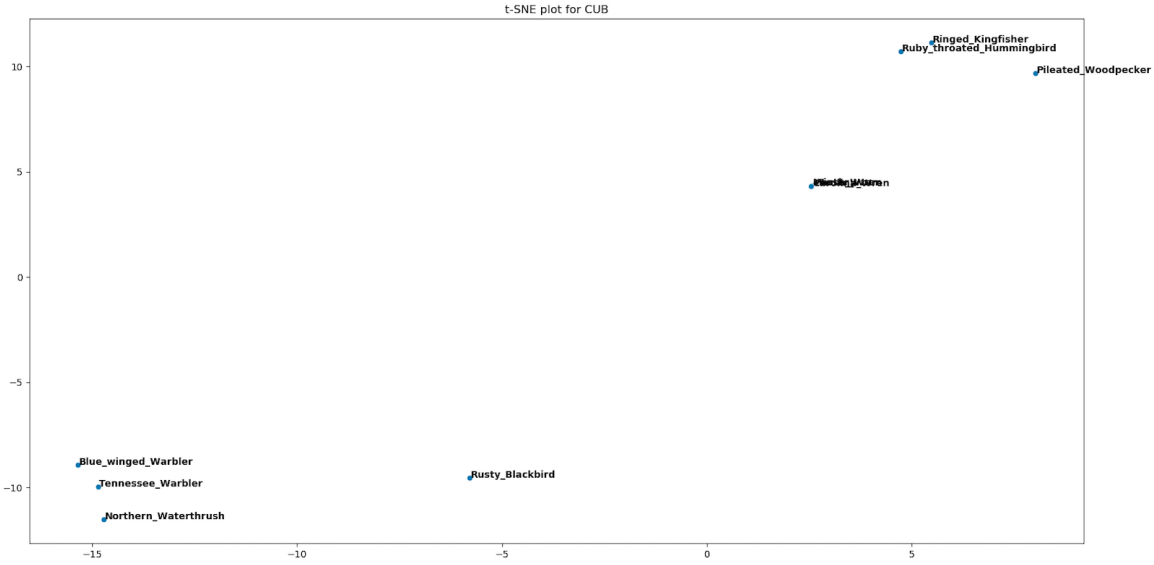

Figure 8: t-SNE plot of the combined semantic space in the CUB data set with 10 classes.

(GMM) [20] and Affinity Propagation (AP) [7], to identify the clusters and representative classes for the clusters. Clustering is performed in the combined reduced-dimensional semantic space for each data set mentioned in the previous section. The clusters centers are noted for further use in the classifier stage.

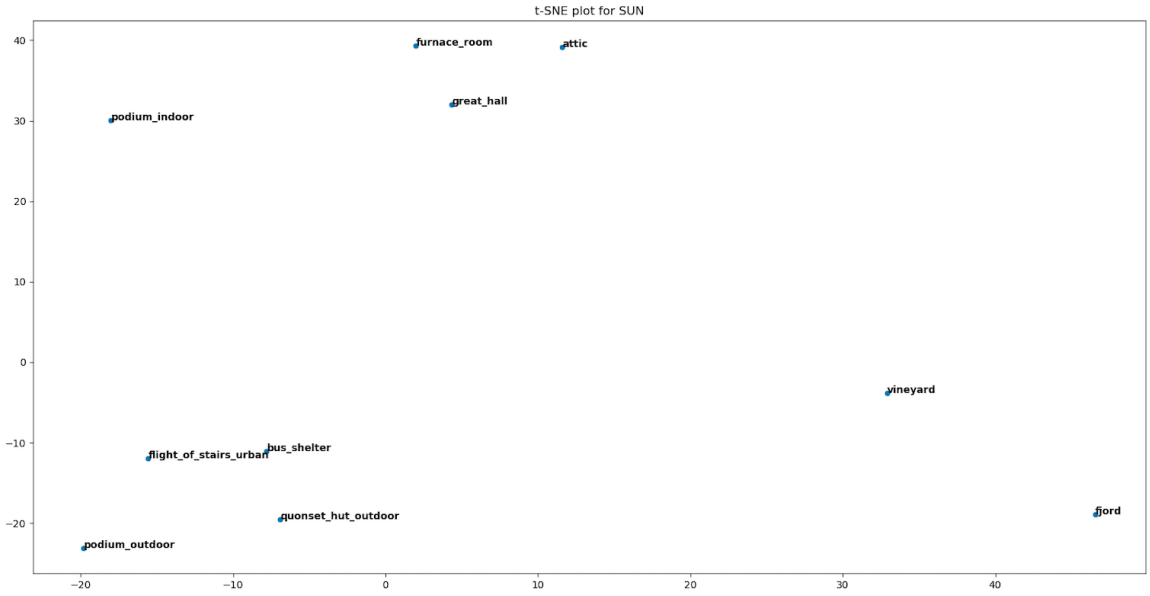

Figure 9: t-SNE plot of the combined semantic space in the SUN data set with 10 classes.

The **Gaussian Mixture Model (GMM)** can accommodate clusters that have different sizes and correlation structures within them, as opposed to k-means clustering. The GMM allows for flexible clustering or soft clustering of the input data. Soft clustering methods assign a score to a data point with respect to each cluster. The value of the score indicates the association strength of the data point to the cluster; in our case the relationship of the data point to the cluster of object categories. We denote the number of clusters by the variable k. GMM-based clustering requires us to specify the number of clusters k before fitting the model. The number of clusters k determines the number of components in the GMM. In our experiments, k starts with a value 5 and ends with a value that equals the total number of classes for a given data set in steps of 5. After identifying all the clusters, the silhouette score [26] is used to find the optimal value of k for the GMM.

Affinity Propagation (AP) is a clustering algorithm based on concept of "message passing" between data points. Unlike the GMM, AP does not require us to specify in advance the number of clusters k to be determined, the algorithm itself provides the optimal number of clusters, i.e., the value of k. AP discovers "exemplars" that are members of the input set that are representative of the clusters.

The optimal number of clusters for each data set using both clustering techniques is shown in Table 3. It should be noted that since GMM-based clustering was performed using a step size of 5 for k, these results may not be the true optimal values for k. In this work, we focus primarily on the GMM-based clustering technique since we would like to study how changing number of seen classes i.e. the value of k, impacts the classification accuracy.

Table 3: Optimal number of clusters (k value) for each clustering technique.

Data Set	***k*-GMM**	***k*-AP**	**Total Classes**
AWA2	15	15	50
CUB	25	24	200
SUN	35	31	717

4.4 MULTI-LABEL CLASSIFICATION OF DEEP FEATURES

After determining the clusters and the representative object for each cluster for different values of k, the next step is to train visual classifiers for each cluster center or representative object. A trained visual classifier is used to classify each new test image into one of the representative objects corresponding to the clusters.

For each value of k, the training set is filtered for the class labels associated with the cluster centers. For example, in the AWA2 data set, for $k = 5$, we have class labels *chimpanzee*, *hamster*, *humpback whale*, *bobcat*, and *ox* as the cluster centers. The image features associated with the class labels of cluster centers alone are considered as the training set and a multi-class visual classifier is trained using this training set. Thus, for each value of k, a multi-class classifier is trained, with the training data increasing with increasing number of clusters k. Algorithm 1 shows the steps involved in clustering and training process.

Visual Classifier. There are a number of machine learning techniques one can employ to design a multi-class visual classifier such as logistic regression, decision tree, support vector machine (SVM), and neural network. In this work we use a Random Forest (RF) classifier. The RF classifier builds an ensemble of decision trees using a bagging ensemble technique. Simply put, RF builds

multiple decision trees and then merges them together to get a more accurate and stable prediction. We use an RF classifier since it is highly scalable for a large number of classes and yields good results. Since we have to train multiple classifiers for varying values of k, when using GMM-based clustering, it is difficult to train a neural network or an SVM because of the time complexity and parameter tuning involved. The parameters used in the training of the RF are mentioned in Appendix A.

Algorithm 1: Clustering of semantic space and training of visual classifiers. Here, clusteringTechnique denotes the type of clustering technique used. The visual classifier used is a random forest.

```
Data: Combined semantic space, Training set of image features
Result: Cluster centers and a Trained visual classifier
initialization;
N ← Total number of classes in data set;
if clusteringTechnique = GMM then
    for k in 1 to N with increments of 5 do
        Perform GMM clustering on combined semantic space using k = k;
        Seen classes ← List of cluster centers;
        Training subset ← Subset entire training set for Seen classes alone;
        Train visual classifier using Training subset;
        Save classifier
    end
end
if clusteringTechnique = AF then
    Perform AF clustering on combined semantic space;
    Seen classes ← List of cluster centers;
    Training subset ← Subset entire training set for Seen classes alone;
    Train visual classifier using Training subset;
    Save classifier
end
```

4.5 GENERATION OF PREDICTIONS OR ALTERNATIVE HYPOTHESES

Once visual classifiers are trained, each test instance is classified into one of the representative clusters for a given value of k. Alternative hypotheses or predictions are then generated using a similarity measure within the combined semantic space.

Similarity measures. In machine learning, a similarity measure is an inverse distance metric with dimensionality determined the class feature space. If the distance between two data points

is small then there is a high degree of similarity between the classes and vice versa. There are a number of similarity measures using in machine learning such as ones based on Euclidean distance, Manhattan distance, Jaccard distance, and cosine similarity. In this work, we use the cosine similarity measure which computes the cosine of the angle between two vectors as shown in Equation (1). Cosine similarity is advantageous because even if the two classes are far apart based on a standard distance metric, it may be possible for their corresponding vectors to be oriented closer together in terms of their angular separation. The smaller the angular separation between the two vectors, the higher the cosine similarity measure.

$$\cos(\mathbf{a}, \mathbf{b}) = \frac{\mathbf{ab}}{\|\mathbf{a}\|\|\mathbf{b}\|} = \frac{\sum_{i=1}^{n} \mathbf{a}_i \mathbf{b}_i}{\sqrt{\sum_{i=1}^{n} (\mathbf{a}_i)^2} \sqrt{\sum_{i=1}^{n} (\mathbf{b}_i)^2}} \tag{1}$$

Testing. Algorithm 2 shows the steps involved in the prediction phase. The test set is split into two subsets. The first subset denotes the seen classes, comprising of data pertaining to class labels that are present in the training set. The second subset denotes the unseen classes, comprising of data pertaining to class labels that are absent from the training set. For each of these subsets, we determine top prediction using the trained visual classifier and then find the closest class label in the combined semantic space using the cosine similarity measure. Classification accuracy is computed for each subset separately followed by the computation of the *harmonic score* (H-score) using Equation (2).

$$H - score = \frac{2 * (SeenClassAccuracy) * (UnseenClassAccuracy)}{(SeenClassAccuracy) + (UnseenClassAccuracy)} \tag{2}$$

We choose the harmonic mean as the performance metric instead of the arithmetic mean because in the case of the latter, if the seen class accuracy is much higher than the unseen class accuracy, the overall arithmetic mean is significantly skewed towards the seen class accuracy [33]. However, since our aim is to attain high classification accuracy on both the seen and unseen classes, the harmonic mean is a better quantifier of overall classification accuracy.

Algorithm 2: Prediction phase algorithm. Here, clusteringTechnique denotes the type of clustering technique used.

Data: Combined semantic space, Trained Classifiers, Trained Clusters, Testing set of image features
Result: Overall H-Score on the test set

initialization;
N ← Total number of classes in data set;
C ← List of all class labels in data set;
if *clusteringTechnique* = *GMM* **then**
    for *k in 1 to N with increments of 5* **do**
        Cluster centers ← List of cluster centers from clusters trained using k = *k*;
        Seen classes ← Subset test set for Cluster centers alone;
        Unseen classes ← Subset test set for [C - Cluster centers];
        Use classifier trained for k=*k* to predict on seen and unseen classes;
        Use cosine similarity to find closest neighbour in Combined semantic space;
        Calculate accuracy;
        Calculate H-Score;
    end
end
if *clusteringTechnique* = *AF* **then**
    Cluster centers ← List of cluster centers from clusters trained using k = *k*;
    Seen classes ← Subset test set for Cluster centers alone;
    Unseen classes ← Subset test set for [C - Cluster centers];
    Use classifier trained for k=*k* to predict on seen and unseen classes;
    Use cosine similarity to find closest neighbour in Combined semantic space;
    Calculate accuracy;
    Calculate H-Score;
end

CHAPTER 5

EXPERIMENTAL RESULTS AND DISCUSSION

As stated in Chapters 1 and 2, the experimental setup in the proposed approach differs from the standard ZSL setup. The proposed approach aims to minimize the number of seen classes within a data set while delivering reasonable performance on the unseen classes. This makes it hard to compare our results with those of other ZSL frameworks mentioned in the research literature. In order to achieve a fair comparison of the proposed framework with other ZSL approaches, we modify a well known ZSL framework to work with our experiment setup.

Attribute Label Embedding (ALE) [1] uses a bi-linear compatibility function to associate visual and auxiliary information. It embeds each class in the space of attribute vectors. The approach also generalizes to any type of auxiliary information that can be encoded in matrix form such as word embeddings. A comparison study performed in [33] shows that ALE outperforms other ZSL frameworks in the generalized ZSL setting. Recent generative methods described in the literature could potentially perform better than ALE but such a comparison study using same data sets, experimental conditions, and evaluation metrics has not been performed yet. Hence, we chose to compare the proposed approach with the ALE framework. A working implementation of the ALE framework was obtained from a recent GitHub repository [4] and modified to work within our experimental setting. Once the clusters centers are determined for each value of k, the ALE procedure is performed on the appropriate training and testing sets. Table 4 and Table 5 show a summary of the comparison between the proposed approach and ALE using both clustering techniques, i.e., GMM-based clustering and AP-based clustering. Table 6 in Appendix B provides comprehensive results on all three data sets when using GMM-based clustering.

Results on the AWA2 data set. The AWA2 data set has 50 classes and the training set consists of 560 images per class on average with the category *horse* having the highest number of images (1316), and the category *mole* having the least number of images (80).

[4] https://github.com/cetinsamet/attribute-label-embedding

We study performance of the proposed approach for $k = 25$ which renders half the classes as seen and half the classes as unseen for the model. The seen classes exhibit an average classification accuracy of 85% whereas the unseen classes exhibit an average classification accuracy of 27% on the test set.

Among the seen classes, we find there are three cases:

Seen Case: 1 Seen classes exhibiting $\geq$ 90% classification accuracy on the test set. In the AWA2 data set, 16 of 25 seen classes fall in this category and are expected to aid very well in the inference of unseen classes related to these seen classes. These seen classes have a good number of images to train on and the classifier is able to clearly identify distinguishing features for each class. For example, *humpback whale* is a seen class that exhibits 100% classification accuracy on the test set. Ideally, we would want all seen classes to fall into this category.

Seen Case: 2 Seen classes exhibiting classification accuracy $\geq$ 60% but $<$ 90% on the test set. In the AWA2 data set, 7 of 25 seen classes fall in this category and are expected to aid reasonably in the inference of unseen classes. Although these classes have a reasonable number of images to train on, these classes fall into a case where the classes are very close to each other but are still seen classes. For example, *ox*, *moose*, and *cow* fall in this category. It is understandable why the visual classifier would not be able to clearly distinguish between these categories, they are quite similar compared to other categories in this data set.

Seen Case: 3 Seen classes exhibiting classification accuracy $<$ 60% on the test set. In the AWA2 data set, 2 of 25 seen classes fall in this category and are expected to negatively impact the unseen class inference process. These 2 classes have only an average of 160 images in the training set, hence the visual classifiers were not be trained enough on these classes to learn the discriminating features.

Among the unseen classes also we find three cases:

Unseen Case: 1 Unseen classes exhibiting $>$ 60% classification accuracy on the test set. In the AWA2 data set, 6 of 25 unseen classes fall in this category. These are cases when a particular unseen class is

inferred from a seen class falling in Case 1 of seen classes. For example, *blue whale* is an unseen class that exhibits 100% accuracy on the test set, and it is inferred from *humpback whale* which falls under Case 1 of the seen classes.

Unseen Case: 2 Unseen classes exhibiting classification accuracy $\geq 1\%$ but $\leq 60\%$ on the test set. In the AWA2 data set, 6 of 25 unseen classes fall in this category. These are cases where the unseen classes are inferred from the seen classes falling in Case 2 and Case 3 of seen classes. These unseen classes exhibit poor performance since the seen classes they are inferred from are not clearly distinguishable by the visual classifier. For example, *deer* is an unseen class that is inferred from the seen class *moose* which falls in Case 2 of the seen classes.

Unseen Case: 3 Unseen classes that exhibit 0% accuracy on the test set. In the AWA2 data set, 12 of 25 unseen classes fall in this category. Since our inference process only allows for guessing 1 unseen class per seen class, unseen classes that are farther away from seen classes cannot be inferred and fall in this category. For example, *antelope* is an unseen class that falls in the cluster of *moose*, but only *deer* can be inferred from *moose* since it is the closest neighbor to *moose* based on the cosine similarity measure.

The proposed model performs well when a seen class has a sufficient number of images to train on and the unseen class being inferred from it is proximal to the seen class and it performs poorly when the unseen class being inferred is distant from any of the representative classes. Ideally, the proposed model would perform best when there are clusters of size 2 and the representative object for the cluster has clear discriminating features, while the unseen class in the cluster is also closer to the representative class than any other class. Figure 10 shows a comparison of the H-scores obtained on the AWA2 data set by the proposed model and ALE for all values of k. As seen from the graph, our model performs significantly better than ALE for all values of k on this data set. The H-score keeps increasing as the value of k increases which is expected since the number of seen classes increase which enables more unseen classes to be inferred more accurately.

In order to determine the optimal numbers of classes required to achieve reasonable performance, we choose a k value for which we achieve a greater than average H-score performance with

the proposed model. For $k = 20$, we achieve a H-score of 46% with the proposed model whereas the average H-score on the AWA2 data set is 45% across all categories. Thus, on the AWA2 data set, we need to have at least 20 categories as seen classes to reasonably infer the unseen classes with greater than average accuracy with the proposed model.

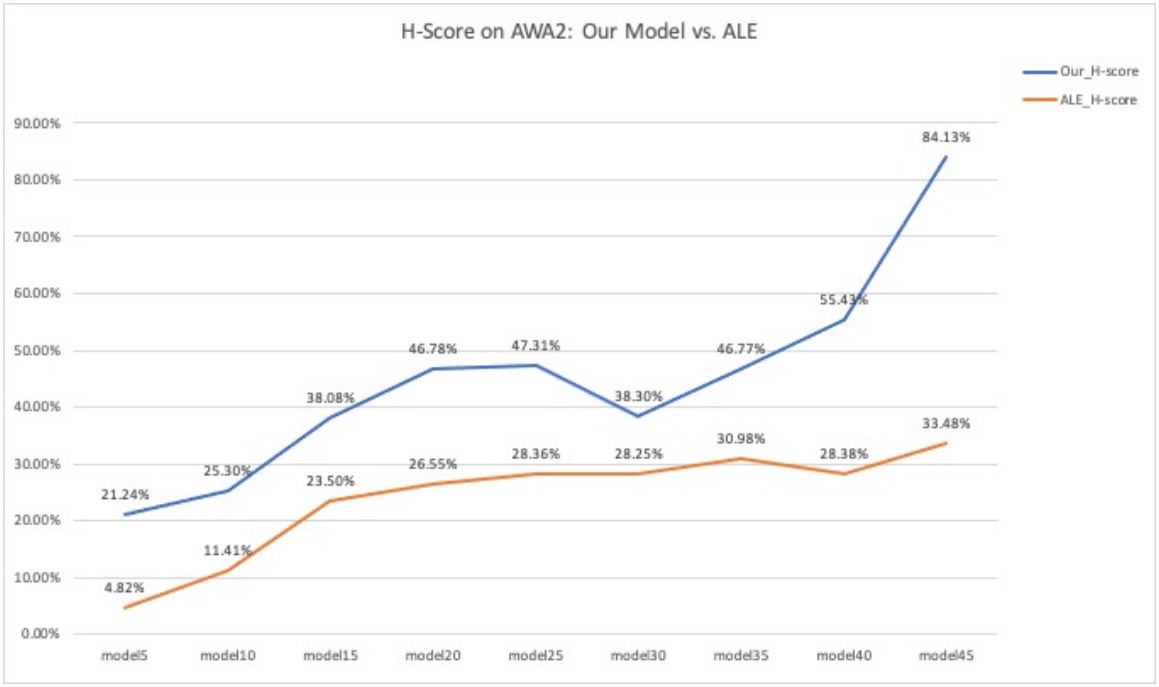

Figure 10: H-score comparison between the proposed model and ALE on AWA2 data set when using GMM-based clustering.

Results on the CUB data set. The CUB data set has 200 classes with 47 images per class on average in the training set. This is a small number of images to train on for each seen class. However, the proposed model still performs well on this data set.

Figure 11 shows a comparison of H-scores obtained on the CUB data set using the proposed model and ALE for all values of k. As seen from the graph, the proposed model performs better than ALE for values of $k \leq 65$. For $65 < k \leq 115$, the proposed model and ALE exhibit comparable performance and for $k > 115$, the proposed model significantly outperforms ALE on the CUB data set. Thus, across a large range of k values, the proposed model performs better than ALE on the CUB data set.

For $k = 80$, we achieve a H-score of 33.5% with the proposed model whereas the average H-score on the CUB data set is 33% across all categories. Thus, on the CUB data set, we need to use at least 80 categories as seen classes to reasonably infer the unseen classes with higher than average accuracy with the proposed model.

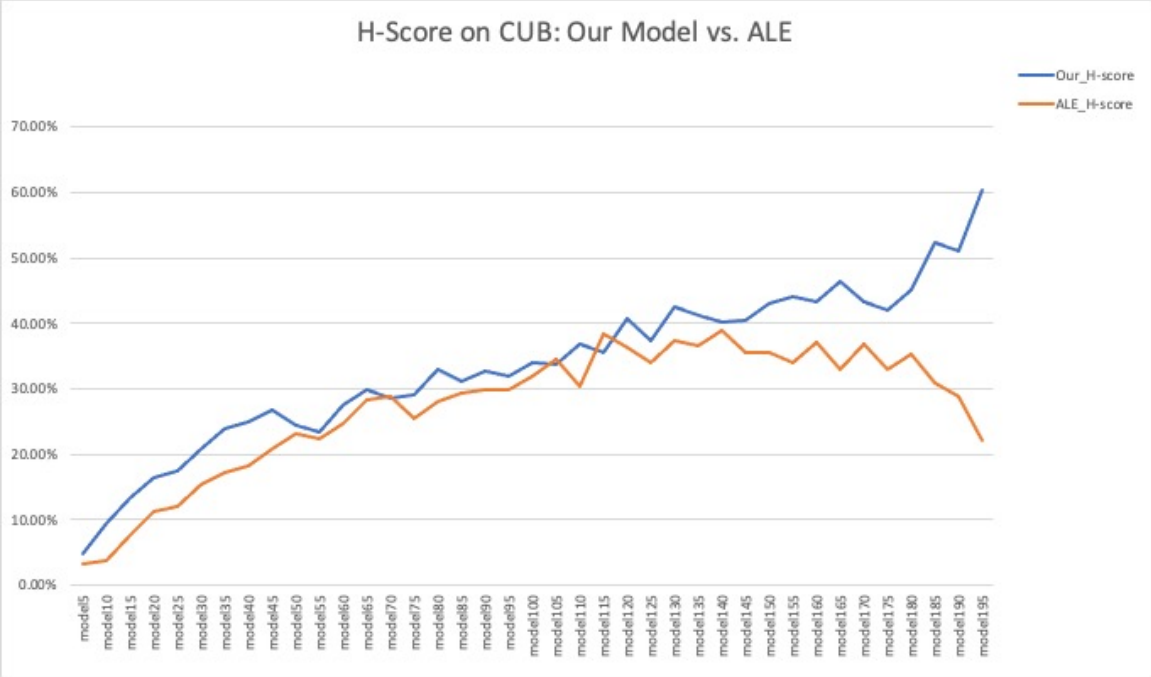

Figure 11: H-score comparison between the proposed model and ALE on CUB data set when using GMM-based clustering.

Results on the SUN data set. The SUN data set has 717 classes with 16 images per class on average in the training set. This makes it hard for the visual classifier to learn distinguishing features for each class because of the large number of classes and small number of images for each class. Nevertheless, the proposed model exhibits performance that is comparable to ALE on this data set. Figure 12 shows a comparison of H-scores obtained on the SUN data set using the proposed model and ALE for all values of k. As seen from the graph, ALE performs better than the proposed model for values of $k \leq 560$ and the proposed model performs better than ALE for values of k beyond 560.

For $k = 360$, we achieve a H-score of 22% with the proposed model and the average H-score on the CUB data set is 21.6% across all categories. Thus, on the SUN data set, we need to use at

least 360 categories as seen classes to reasonably infer the unseen classes with greater than average accuracy with the proposed model.

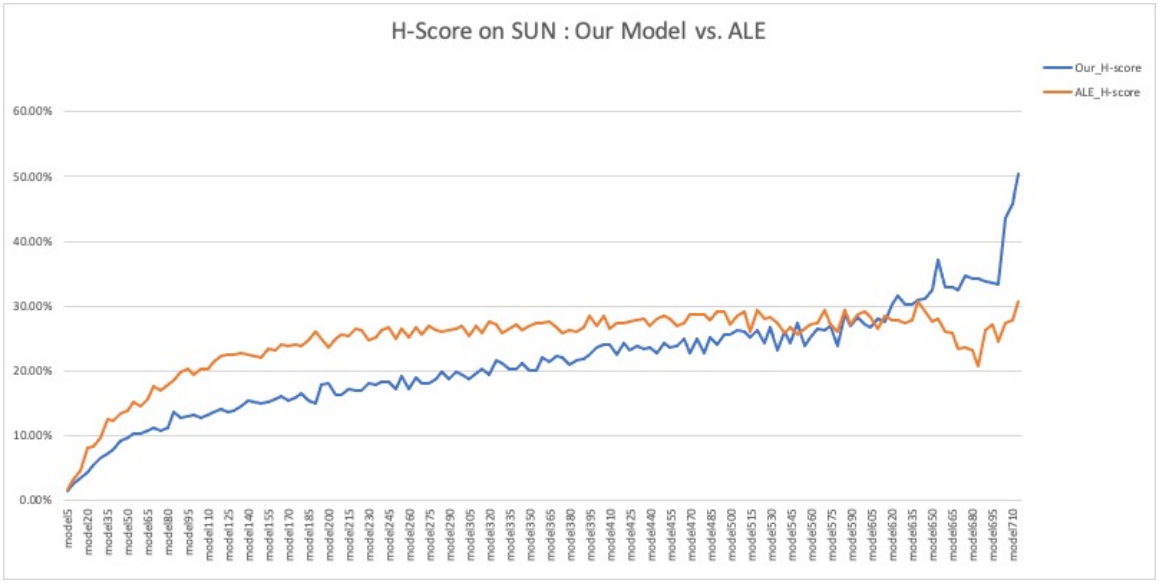

Figure 12: H-score comparison between the proposed model and ALE on SUN data set when using GMM-based clustering.

Table 4: Comparison of average classification accuracy across k values between the proposed model and ALE when using GMM-based clustering.

	Proposed Model			**ALE**		
Data Set	**Avg. Seen**	**Avg. Unseen**	**Avg. H-Score**	**Avg. Seen**	**Avg. Unseen**	**Avg. H-Score**
AWA2	94%	32%	**45%**	90%	14%	24%
CUB	78%	22.86%	**33%**	70%	17.50%	27.19%
SUN	58.20%	15%	21.60%	41.50%	17.80%	**25%**

Hardware Specifications. All the experiments were performed on an Intel I7 6850K CPU with 128GB RAM. For the AWA2 data set, it takes $\approx$ 20 minutes to train visual classifiers for all values of k. Likewise, for the CUB and SUN data sets, it takes $\approx$ 2.5 hours and $\approx$ 40 hours, respectively, to train visual classifiers for all values of k.

Table 5: Comparison of average classification accuracy between the proposed model and ALE when using AP-based clustering.

	Proposed Model			**ALE**		
Data Set	**Seen**	**Unseen**	**H-Score**	**Seen**	**Unseen**	**H-Score**
AWA2	96.44%	23.43%	**37.7%**	83.40%	10%	17.50%
CUB	91%	9.70%	**17.50%**	55%	8.33%	14.40%
SUN	83.10%	4.30%	8.20%	24.40%	8.20%	**12.30%**

CHAPTER 6

CONCLUSIONS AND FUTURE WORK

In this thesis, we propose a framework for generalized zero-shot learning (ZSL) that is simple yet very effective. The proposed framework offers an intuitive approach to aid in the training data collection process for image recognition tasks by identifying representative classes using various clustering techniques. It also provides a method to infer unseen classes using cosine similarity measure. The proposed framework achieves accuracy figures that are 21% greater on the AWA2 data set and 6% greater on the CUB data set when compared to the well known Attribute Label Embedding (ALE) scheme for GZSL. On the SUN data set, the proposed model exhibits performance that is comparable to that of ALE. We also determine the minimum number of categories needed to considered as seen classes to achieve reasonable classification accuracy results on all the three data sets using the proposed model.

One of the drawbacks of the proposed framework is the inability to infer unseen classes that are distant from the representative classes in the semantic space. There is significant scope for future improvement of the proposed framework in this aspect. A potential solution could be a scheme to map the distance between each unseen class and representative class in a cluster to the classification probabilities obtained from the visual classifier. In this way, the framework would be able to infer all unseen classes, regardless of the distance, with some non-zero probability.

Another important future task is the evaluation of the proposed framework on a very large data set such as ImageNet. ImageNet spans more than 1000 classes and has several images in each class unlike the SUN data set which while having over 700 classes, has very few images per class.

REFERENCES

[1] Z. Akata et al. "Label-Embedding for Attribute-Based Classification". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 819–826.

[2] Z. Akata et al. "Evaluation of output embeddings for fine-grained image classification". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (June 2015). DOI: 10.1109/cvpr.2015.7298911. URL: http://dx.doi.org/10.1109/CVPR.2015.7298911.

[3] I. Biederman. "Recognition-by-components: A theory of human image understanding". In: *Psychological Review* 94 (1987), pp. 115–147.

[4] P. Bojanowski et al. "Enriching Word Vectors with Subword Information". In: *arXiv preprint arXiv:1607.04606* (2016).

[5] S. Changpinyo et al. *Synthesized Classifiers for Zero-Shot Learning*. 2016. arXiv: 1603.00550 [cs.CV].

[6] J. Deng et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[7] B. J. Frey and D. Dueck. "Clustering by Passing Messages Between Data Points". In: *Science* 315.5814 (2007), pp. 972–976. ISSN: 0036-8075. DOI: 10.1126/science.1136800. eprint: https://science.sciencemag.org/content/315/5814/972.full.pdf. URL: https://science.sciencemag.org/content/315/5814/972.

[8] A. Frome et al. "DeViSE: A Deep Visual-Semantic Embedding Model". In: *Advances in Neural Information Processing Systems*. Ed. by C. J. C. Burges et al. Vol. 26. Curran Associates, Inc., 2013, pp. 2121–2129. URL: https://proceedings.neurips.cc/paper/2013/file/7cce53cf90577442771720a370c3c723-Paper.pdf.

[9] Y. Fu et al. "Transductive Multi-View Zero-Shot Learning". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37.11 (Nov. 2015), pp. 2332–2345. ISSN: 2160-9292. DOI: 10.1109/tpami.2015.2408354. URL: http://dx.doi.org/10.1109/TPAMI.2015.2408354.

[10] Z. Fu et al. "Zero-shot object recognition by semantic manifold distance". In: *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015, pp. 2635–2644. DOI: 10.1109/CVPR.2015.7298879.

[11] K. He et al. "Deep Residual Learning for Image Recognition". In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385.

[12] Q. Jiang et al. "Object Detection and Classification of Metal Polishing Shaft Surface Defects Based on Convolutional Neural Network Deep Learning". In: *Applied Sciences* 10 (Dec. 2019), p. 87. DOI: 10.3390/app10010087.

[13] C. H. Lampert, H. Nickisch, and S. Harmeling. "Attribute-Based Classification for Zero-Shot Visual Object Categorization". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36.3 (2014), pp. 453–465. DOI: 10.1109/TPAMI.2013.140.

[14] C. H. Lampert, H. Nickisch, and S. Harmeling. "Attribute-Based Classification for Zero-Shot Visual Object Categorization". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36 (2014), pp. 453–465.

[15] K. Lee et al. "Hierarchical Novelty Detection for Visual Object Recognition". In: *CVPR*. 2018.

[16] J. Li et al. "Leveraging the Invariant Side of Generative Zero-Shot Learning". In: *IEEE Computer Vision and Pattern Recognition (CVPR)*. 2019.

[17] L. van der Maaten and G. Hinton. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9.86 (2008), pp. 2579–2605. URL: http://jmlr.org/papers/v9/vandermaaten08a.html.

[18] T. Mikolov et al. *Distributed Representations of Words and Phrases and their Compositionality*. 2013. arXiv: 1310.4546 [cs.CL].

[19] G. A. Miller. "WordNet: A Lexical Database for English". In: *Commun. ACM* 38.11 (Nov. 1995), pp. 39–41. ISSN: 0001-0782. DOI: 10.1145/219717.219748. URL: https://doi.org/10.1145/219717.219748.

[20] O. M. M. Mohamed and M. Jaïdane-Saïdane. "Generalized Gaussian mixture model". In: *2009 17th European Signal Processing Conference*. 2009, pp. 2273–2277.

[21] S. Narayan et al. "Latent Embedding Feedback and Discriminative Features for Zero-Shot Classification". In: *arXiv preprint arXiv:2003.07833* (2020).

[22] M. Norouzi et al. "Zero-shot learning by convex combination of semantic embeddings". English (US). In: 2nd International Conference on Learning Representations, ICLR 2014 ; Conference date: 14-04-2014 Through 16-04-2014. Jan. 2014.

[23] G. Patterson and J. Hays. "SUN attribute database: Discovering, annotating, and recognizing scene attributes". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*. 2012, pp. 2751–2758.

[24] J. Pennington, R. Socher, and C. D. Manning. "GloVe: Global Vectors for Word Representation". In: *Empirical Methods in Natural Language Processing (EMNLP)*. 2014, pp. 1532–1543. URL: http://www.aclweb.org/anthology/D14-1162.

[25] B. Romera-Paredes and P. H. S. Torr. "An Embarrassingly Simple Approach to Zero-Shot Learning". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, pp. 2152–2161.

[26] P. Rousseeuw. "Silhouettes: A Graphical Aid to the Interpretation and Validation of Cluster Analysis". In: *J. Comput. Appl. Math.* 20.1 (Nov. 1987), pp. 53–65. ISSN: 0377-0427. DOI: 10.1016/0377-0427(87)90125-7. URL: https://doi.org/10.1016/0377-0427(87)90125-7.

[27] W. J. Scheirer et al. "Toward Open Set Recognition". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.7 (2013), pp. 1757–1772. DOI: 10.1109/TPAMI.2012.256.

[28] R. Socher et al. "Zero-Shot Learning through Cross-Modal Transfer". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 935–943.

[29] V. K. Verma and P. Rai. *A Simple Exponential Family Framework for Zero-Shot Learning*. 2018. arXiv: 1707.08040 [cs.LG].

[30] M. R. Vyas, H. Venkateswara, and S. Panchanathan. *Leveraging Seen and Unseen Semantic Relationships for Generative Zero-Shot Learning*. 2020. arXiv: 2007.09549 [cs.CV].

[31] X. Wang, Y. Ye, and A. Gupta. *Zero-shot Recognition via Semantic Embeddings and Knowledge Graphs*. 2018. arXiv: 1803.08035 [cs.CV].

[32] P. Welinder et al. *Caltech-UCSD Birds 200*. Tech. rep. CNS-TR-2010-001. California Institute of Technology, 2010.

[33] Y. Xian, B. Schiele, and Z. Akata. "Zero-Shot Learning — The Good, the Bad and the Ugly". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (July 2017). DOI: 10.1109/cvpr.2017.328. URL: http://dx.doi.org/10.1109/CVPR.2017.328.

[34] Y. Xian et al. *Feature Generating Networks for Zero-Shot Learning*. 2018. arXiv: 1712.00981 [cs.CV].

[35] Y. Xian et al. *Latent Embeddings for Zero-shot Classification*. 2016. arXiv: 1603.08895 [cs.CV].

[36] Z. Zhang and V. Saligrama. "Zero-Shot Learning via Semantic Similarity Embedding". In: *CoRR* abs/1509.04767 (2015). arXiv: 1509.04767. URL: http://arxiv.org/abs/1509.04767.

APPENDIX A

MODEL PARAMETERS

A.1 Random Forest

- **AWA2** *n_estimators* : 1000, *max_depth* : 60, *n_jobs* : -1, *min_samples_split* : 2, *min_samples_leaf* : 1, *max_features* : 'auto', *bootstrap* : 'False'
- **CUB** *n_estimators* : 1000, *max_depth* : 60, *n_jobs* : -1, *min_samples_split* : 2, *min_samples_leaf* : 1, *max_features* : 'auto', *bootstrap* : 'False'
- **SUN** *n_estimators* : 500, *max_depth* : 60, *n_jobs* : -1, *min_samples_split* : 2, *min_samples_leaf* : 1, *max_features* : 'auto', *bootstrap* : 'False'

A.2 GMM Clustering

- **AWA2** k : 5 to 45 with steps of 5
- **CUB** k : 5 to 195 with steps of 5
- **SUN** k : 5 to 715 with steps of 5

A.3 Principal Component Analysis

- **AWA2** *explained_variance* : 0.70
- **CUB** *explained_variance* : 0.40
- **SUN** *explained_variance* : 0.60

APPENDIX B

COMPLETE MODEL RESULTS

Table 6: Results on all data sets when using GMM-based clustering

	AWA2			CUB			SUN		
Model	**Seen**	**Unseen**	**H_score**	**Seen**	**Unseen**	**H_score**	**Seen**	**Unseen**	**H_score**
model5	96.72%	11.93%	21.24%	100.00%	2.52%	4.92%	90.00%	0.67%	1.33%
model10	96.82%	14.55%	25.30%	97.50%	4.96%	9.44%	100.00%	1.20%	2.37%
model15	95.97%	23.75%	38.08%	93.82%	7.11%	13.22%	93.33%	1.57%	3.09%
model20	94.10%	31.13%	46.78%	92.80%	9.00%	16.41%	87.50%	2.33%	4.54%
model25	93.57%	31.66%	47.31%	86.82%	9.65%	17.37%	93.00%	2.64%	5.13%
model30	92.82%	24.13%	38.30%	86.16%	11.83%	20.80%	90.83%	3.64%	7.00%
model35	92.36%	31.31%	46.77%	85.71%	13.93%	23.97%	87.86%	3.34%	6.44%
model40	92.07%	39.65%	55.43%	84.86%	14.61%	24.93%	83.75%	3.51%	6.74%
model45	90.76%	78.41%	84.13%	86.33%	15.90%	26.85%	80.00%	4.20%	7.98%
model50				81.49%	14.41%	24.49%	81.00%	3.97%	7.57%
model55				80.19%	13.67%	23.36%	79.55%	4.38%	8.30%
model60				81.95%	16.62%	27.64%	80.42%	4.34%	8.24%
model65				81.82%	18.26%	29.86%	78.46%	5.90%	10.97%
model70				81.71%	17.36%	28.64%	79.29%	5.02%	9.44%
model75				78.71%	17.76%	28.98%	77.00%	5.72%	10.65%
model80				81.66%	20.72%	33.05%	76.25%	5.30%	9.91%
model85				80.08%	19.28%	31.08%	73.24%	6.17%	11.38%
model90				78.79%	20.66%	32.74%	73.33%	5.58%	10.37%
model95				79.80%	20.02%	32.01%	73.68%	6.15%	11.35%
model100				77.65%	21.84%	34.09%	71.50%	6.00%	11.07%
model105				75.20%	21.79%	33.79%	69.52%	6.25%	11.47%
model110				76.23%	24.29%	36.84%	65.45%	6.05%	11.08%

	AWA2			CUB			SUN		
model115		76.27%	23.08%	35.44%	69.13%	5.86%	10.80%		
model120		75.09%	28.03%	40.82%	68.33%	7.41%	13.37%		
model125		74.76%	25.00%	37.47%	67.80%	6.88%	12.49%		
model130		75.36%	29.59%	42.49%	65.77%	6.94%	12.56%		
model135		74.64%	28.46%	41.21%	70.56%	7.17%	13.02%		
model140		73.50%	27.63%	40.16%	68.57%	6.89%	12.52%		
model145		72.26%	28.02%	40.38%	65.17%	6.64%	12.05%		
model150		71.66%	30.68%	42.97%	62.33%	7.10%	12.75%		
model155		70.08%	32.08%	44.01%	65.81%	7.34%	13.21%		
model160		68.40%	31.57%	43.20%	62.81%	7.85%	13.96%		
model165		69.84%	34.71%	46.37%	61.67%	6.88%	12.38%		
model170		69.36%	31.36%	43.19%	63.68%	7.68%	13.71%		
model175		68.66%	30.33%	42.07%	61.71%	6.96%	12.51%		
model180		68.11%	33.61%	45.01%	62.50%	8.33%	14.70%		
model185		67.45%	42.78%	52.35%	61.89%	8.60%	15.10%		
model190		66.26%	41.67%	51.16%	61.84%	8.06%	14.26%		
model195		64.58%	56.67%	60.37%	61.28%	8.19%	14.45%		
model200					62.38%	7.79%	13.85%		
model205					62.32%	8.45%	14.88%		
model210					59.88%	8.58%	15.01%		
model215					60.35%	8.12%	14.31%		
model220					59.43%	9.26%	16.02%		
model225					60.56%	9.15%	15.90%		
model230					59.89%	8.73%	15.24%		
model235					58.51%	9.13%	15.80%		
model240					58.96%	9.07%	15.72%		
model245					56.94%	9.32%	16.02%		
model250					57.80%	8.83%	15.32%		

	AWA2	CUB	SUN		
model255			54.80%	8.66%	14.96%
model260			57.40%	9.08%	15.68%
model265			55.94%	9.24%	15.86%
model270			55.56%	8.22%	14.32%
model275			58.45%	9.28%	16.02%
model280			56.52%	8.58%	14.90%
model285			57.54%	9.72%	16.63%
model290			55.26%	10.60%	17.79%
model295			54.41%	10.07%	16.99%
model300			55.08%	10.07%	17.03%
model305			55.41%	9.71%	16.52%
model310			54.68%	9.58%	16.30%
model315			53.49%	10.82%	18.00%
model320			53.20%	10.26%	17.20%
model325			54.31%	10.65%	17.81%
model330			51.97%	10.40%	17.33%
model335			52.99%	10.73%	17.85%
model340			52.06%	10.94%	18.08%
model345			51.88%	10.82%	17.91%
model350			53.07%	11.31%	18.65%
model355			51.27%	10.64%	17.62%
model360			51.04%	10.29%	17.13%
model365			50.62%	10.80%	17.80%
model370			52.16%	10.59%	17.61%
model375			51.20%	10.96%	18.06%
model380			51.51%	11.72%	19.10%
model385			49.81%	10.92%	17.91%
model390			48.85%	11.62%	18.77%

	AWA2	CUB	SUN		
model395			50.38%	11.72%	19.02%
model400			49.69%	13.56%	21.31%
model405			49.07%	12.82%	20.33%
model410			50.43%	11.89%	19.24%
model415			49.88%	12.75%	20.31%
model420			49.52%	11.36%	18.48%
model425			48.41%	12.59%	19.98%
model430			48.84%	13.07%	20.62%
model435			48.33%	14.36%	22.14%
model440			48.47%	12.00%	19.24%
model445			49.33%	13.69%	21.43%
model450			48.06%	13.67%	21.29%
model455			47.36%	13.45%	20.95%
model460			48.10%	14.69%	22.51%
model465			47.31%	13.10%	20.52%
model470			47.07%	12.25%	19.44%
model475			46.37%	14.67%	22.29%
model480			46.41%	15.19%	22.89%
model485			45.77%	14.12%	21.58%
model490			46.84%	14.32%	21.93%
model495			47.53%	14.19%	21.86%
model500			47.50%	15.78%	23.69%
model505			45.45%	14.50%	21.99%
model510			46.86%	15.82%	23.65%
model515			46.65%	14.85%	22.53%
model520			45.91%	14.97%	22.58%
model525			45.48%	16.28%	23.98%
model530			45.90%	14.71%	22.28%

	AWA2	CUB	SUN	
model535			44.53%	15.25%	22.72%
model540			44.72%	16.67%	24.29%
model545			44.63%	15.84%	23.38%
model550			44.73%	15.57%	23.10%
model555			44.55%	16.82%	24.42%
model560			44.15%	16.56%	24.09%
model565			43.36%	16.28%	23.67%
model570			43.86%	19.39%	26.89%
model575			43.57%	16.02%	23.43%
model580			43.23%	17.52%	24.93%
model585			43.03%	16.86%	24.23%
model590			43.98%	17.91%	25.45%
model595			42.73%	19.26%	26.55%
model600			42.63%	18.38%	25.69%
model605			42.77%	18.08%	25.42%
model610			43.36%	21.03%	28.32%
model615			42.80%	19.36%	26.66%
model620			42.82%	19.59%	26.88%
model625			41.76%	19.29%	26.39%
model630			41.31%	19.25%	26.26%
model635			40.83%	20.43%	27.23%
model640			41.41%	21.75%	28.52%
model645			41.40%	20.83%	27.72%
model650			40.08%	23.88%	29.93%
model655			40.76%	20.16%	26.98%
model660			40.00%	25.00%	30.77%
model665			40.15%	25.00%	30.81%
model670			40.63%	24.47%	30.54%

	AWA2	CUB	SUN	
model675			38.93%	30.36%	34.12%
model680			39.85%	25.00%	30.72%
model685			39.45%	26.56%	31.75%
model690			40.04%	34.26%	36.93%
model695			40.22%	31.82%	35.53%
model700			39.14%	30.88%	34.52%
model705			39.18%	39.58%	39.38%
model710			38.66%	35.71%	37.13%
model715			37.97%	75.00%	50.42%