Zero-Shot Learning (ZSL) has shown great promise at the intersection of vision and language, and generative methods for ZSL are predominant owing to their efficiency. Moreover, textual description or attribute plays a critical role in transferring knowledge from the seen to unseen classes in ZSL. Such generative approaches for ZSL are very costly to train and require the class description of the unseen classes during training. In this work, we propose a non-generative gating-based attribute refinement network for ZSL, which achieves similar accuracies to generative methods of ZSL, at a much lower computational cost. The refined attributes are mapped into the visual domain through an attribute embedder, and the whole network is guided by the circle loss and the well-known softmax cross-entropy loss to obtain a robust class embedding. We refer to our approach as Circle loss guided gating-based Attribute Refinement Network (CARNet). We perform extensive experiments on the five benchmark datasets over the various challenging scenarios viz., Generalized ZSL (GZSL), Continual GZSL (CGZSL), and conventional ZSL. We observe that the CARNet significantly outperforms recent non-generative ZSL methods and most generative ZSL methods in all three settings by a significant margin. Our extensive ablation study disentangles the performance of various components and justifies their importance.

1 Introduction

Humans can recognize samples from unseen classes by leveraging the visual information of seen categories and textual descriptions of seen and unseen classes (Larochelle et al., 2008; Palatucci et al., 2009; Lampert et al., 2009). Zero-Shot Learning, inspired by this recognition ability of humans,

Figure 1: It can be seen that the attributes "small bird" and "long black beak" (words in bold) are common in all three species of the Hummingbird. However, attributes like "ruby red throat" or "orange bird" (words in red) distinguish one species from another. Hence the distinguishing attributes must be given more weight than the common attributes in the class attribute vector. We achieve this through the gating unit in the attribute refinement network.

learns unseen classes through the textual description (also referred to as side-information or class attribute vector or semantic information) (Xian et al., 2017). A typical ZSL algorithm does not need training samples from unseen classes. However, it requires the class description for both seen and unseen classes (Zhang and Saligrama, 2015; Reed et al., 2016).

The generative model has recently been the most popular approach for ZSL. It uses generators like VAE (Mishra et al., 2018; Schonfeld et al., 2019) or GAN (Narayan et al., 2020; Vyas et al., 2020) to generate synthetic samples for unseen classes using the class attribute vector. However, despite their promising results, such methods are not very efficient due to the following reasons: (i) The method requires the knowledge of the number of unseen classes and respective attribute vectors during training which is not always feasible, and (ii) retraining of the classifier with seen and unseen samples, with each new unseen classes. On the other hand, non-generative approaches for ZSL alleviates the
above problems but shows inferior accuracies. Typically, the non-generative models learn mapping in three ways: (i) visual to attribute space (Xian et al., 2016) or (ii) attribute to visual space (Zhang et al., 2017; Li et al., 2019), or (iii) joint embedding of attribute and visual space (Cacheux et al., 2019). It is to be noted that most of the existing non-generative or embedding-based ZSL approaches are formulated to learn embedding from visual to attribute space. They assume that the seen and unseen classes share the same representational characteristics and are linked in the attribute space (Frome et al., 2013; Wang et al., 2019; Chen et al., 2021a). However, this approach leads to the well-known hubness problem (Dinu et al., 2014), where the representations are skewed towards seen classes (Zhang et al., 2017; Li et al., 2019). Another problem with this approach is that it implicitly loses the discriminative power of visual features that are generally extracted from a powerful pre-trained deep learning model (like ResNet (Xian et al., 2018a, 2016) and GoogleNet (Song et al., 2018)) but are then mapped to a different smaller attribute space (Li et al., 2019, 2018).

Although the aforementioned issues in the non-generative model are mitigated by mapping the attribute to visual features (Zhang et al., 2017; Li et al., 2019; Skorokhodov and Elhoseiny, 2021), they have lower classification accuracies. In this work, we propose a non-generative method with an Attribute-Refinement Network (ARN) that leverages the gating mechanism. The ARN enables highly robust representation of the description/attribute vector for the seen and unseen classes. In recent years, the gating mechanism has shown good performance without any complex architecture in supervised learning tasks (Srivastava et al., 2015; Dauphin et al., 2017; Sandler et al., 2018; Wu et al., 2018; Liu et al., 2021). In this paper, we propose a gating mechanism for refining the textual description in the ZSL task. The ARN learns to refine the attribute in a self-weighing manner from the seen class attribute (Fig.1). These refined attributes are mapped onto the visual space using an attribute embedder (AE) to obtain the class prototype vector of each class. The class prototype vector is then combined with the visual features in the feature-prototype combiner (FPC) to obtain classifications. The ARN, AE, and FPC are trained jointly using the circle loss and standard softmax cross-entropy in such a way that it minimizes inter-class and maximizes intra-class similarity. The circle loss achieves better within-class compactness and between-class discrepancy compared to triplet loss and adaptive margin softmax loss, as it unifies both classification and pair-wise similarity representation objectives (Sun et al., 2020). We refer to our approach as Circle loss guided gating-based Attribute-Refinement Network (CARNet).

We evaluate the performance of CARNet in three scenarios: (a) Conventional Zero-Shot Learning (ZSL), where only the unseen classes are available during inference (b) Generalized Zero-Shot Learning (GZSL), where both the seen and unseen classes are available during inference and (c) Continual Generalized Zero-Shot Learning (CGZSL), where data arrives as a sequence of tasks and only current task data is available during training, with the challenge of handling catastrophic forgetting of the past tasks. The performance of CARNet for conventional ZSL and GZSL is evaluated on five standard datasets. The CGZSL method is evaluated for the challenging CUB and SUN datasets. The extensive experiment shows that CARNet outperforms the recent generative (unlike the generative model, we do not require the unseen class description during training) and the non-generative model by a significant margin. Our ablation study emphasizes the significance of each component of the proposed learning algorithm. The main contributions of our work are summarised as follows: (i) We propose a gating-based attribute-refinement network (ARN) to enhance the class description/attribute for zero-shot learning. (ii) The ARN and AE are guided by circle loss to achieve better within-class compactness and between-class discrepancy. (iii) We propose a highly competitive, simple, and fast non-generative method. Our model achieves ∼70× speedup compared to generative ZSL methods.

2 Related Work

The proposed CARNet is evaluated for the three kinds of ZSL settings: conventional ZSL, GZSL, and CGZSL. We provide a brief survey on all these three settings. ZSL aims to construct the recognition model for the samples from unseen classes using the textual description (i.e., attribute information) of unseen classes. These attribute information can be obtained through various ways, like human-annotated attributes (Farhadi et al., 2009), textual descriptions (Reed et al., 2016), and word vectors (Socher et al., 2013; Frome et al., 2013). In recent
years, there has been a surge of interest in this area. The whole literature of ZSL can be broadly categorized into two parts: generative and non-generative (i.e., embedding-based) approaches.

The first popular category is the generative approach, which solves the ZSL problem by synthesizing the unseen class samples. To synthesize the samples from the unseen classes, models leverage on a powerful generative models like conditional variational autoencoder (VAE) (Mishra et al., 2018; Kumar Verma et al., 2018) or generative adversarial network (GAN) (Vyas et al., 2020; Xian et al., 2018b; Felix et al., 2018; Keshari et al., 2020; Verma et al., 2020) or a combination of VAE and GAN (Xian et al., 2019; Narayan et al., 2020).

Another popular category is the non-generative approach, and it does not need class attribute information of unseen classes during training. In the early ZSL work (Lampert et al., 2009; Farhadi et al., 2009; Lampert et al., 2013), models directly predict the attribute confidence from images. Methods based on this approach can be further divided into three groups. In the first group, we project visual feature into attribute (i.e., semantic) space (Lampert et al., 2013; Socher et al., 2013; Frome et al., 2013; Akata et al., 2016; Fu and Sigal, 2016). In the second group, both visual and attribute data are projected into intermediate space (Akata et al., 2015; Fu et al., 2014; Lei Ba et al., 2015; Romera-Paredes and Torr, 2015; Cacheux et al., 2019). In the third group, visual space is spanned by attribute to visual mapping (Zhang et al., 2017; Li et al., 2019; Skorokhodov and Elhoseiny, 2021). ZSL methods developed based on the projection from attribute space to visual space approach are more suitable for mitigating the hubness problem, and recent works (Zhang et al., 2017; Li et al., 2019; Skorokhodov and Elhoseiny, 2021) show promising results for the ZSL and GZSL setting. Surprisingly, despite the fast, accurate, and realistic setting, this approach has not been explored much in the past. In this work, we consider the non-generative model for further exploration and learn the mapping from the attribute space to the visual space similar to (Li et al., 2019; Skorokhodov and Elhoseiny, 2021).

The above-discussed ZSL methods can handle data only in an offline setting, and cannot be used in a setting with a streaming sequence of tasks (Delange et al., 2021), known as Continual GZSL (CGZSL). Only a handful of research is available for CGZSL (Chaudhry et al., 2019a; Wei et al., 2020; Skorokhodov and Elhoseiny, 2021; Gautam et al., 2021a, 2020, 2021b). For the extensive evaluation, apart from the conventional ZSL and GZSL, CARNet is also evaluated on the CGZSL setting as proposed in Skorokhodov and Elhoseiny (2021).

3 Problem Definition

In this section, we define the problem formally and introduce the notations. The objective of ZSL is to learn a model that can generalize the novel classes (i.e., unseen classes) with the help of side information (attribute/descriptions) without training data for the novel classes. The attribute vector of each class is constructed by either using a word embedding vector generated from a language model or manually defining the key features like color, size, shape, pattern, etc. Primarily, the ZSL setting consists of two sets of classes known as seen and unseen classes. Let $D^s_{tr}$ and $D^s_{ts}$ be the training and testing data, respectively, for the $C^s$ seen and $C^u$ unseen classes. We also have set of seen ($\{C^s\}$) and unseen ($\{C^u\}$) classes where $\{C^s\} \cap \{C^u\} = \emptyset$ i.e. seen and unseen classes set are disjoint. It is to be noted that $\{C^s\}$ and $\{C^u\}$ denote the set of seen and unseen classes, while $C^s$ and $C^u$ denote the number of seen classes and unseen classes, respectively. Corresponding to each seen class $i$ ($i \in \{C^s\}$) and unseen class $j$ ($j \in \{C^u\}$), there is a $d$-dimensional class attribute vector, i.e., $A^s_i \in \mathbb{R}^d$ and $A^u_j \in \mathbb{R}^d$, respectively. In the ZSL training, data is represented as: $D^s_{tr} = \{x_i, y_i, A^s_{y_i}\}_{i=1}^N$ where $N$ is the number of seen class images and $\{x_i, y_i\}$ is the image and label pair. During inference for conventional ZSL, we have $D^s_{ts} = \{x_j\}_{j=1}^M$ with attribute set $A = A^u$ where $\forall j, x_j$ belongs to the unseen class. However, in GZSL, we have $D^s_{ts} = \{x_j\}_{j=1}^M$ with attribute set $A = A^s \cup A^u$ where $\forall j, x_j$ belongs to either seen or unseen class. Here $A^s$ and $A^u$ are the seen and unseen class attribute information, respectively. Overall, our objective is to develop a model based on the training dataset $D^s_{tr}$ (i.e., seen data), and it needs to be generalized over all class labels $\{C\}$ where $\{C\} = \{C^s\} \cup \{C^u\}$ and the total number of classes in $\{C\}$ is $C$.

4 Proposed Method

Zero-Shot Learning (ZSL) aims at classification in the absence of input images for unseen classes, using textual description, namely attribute vectors. In this section, we propose the CARNet for zero-shot
Figure 2: CARNet: circle loss guided gating-based attribute refinement network for ZSL. It primarily consists of three blocks: (i) ARN, (ii) AE, and (iii) FPC.

4.1 Gating-based Attribute Refinement Network (ARN)

The class attribute vector plays a very crucial role in ZSL, as there are no visual samples for unseen classes during training. Moreover, the attribute vector is the only information that is available for both the seen and unseen classes. Therefore, it is highly important that the attribute representation has minimal noise and highlights its prime dimensions. The objective of the ARN is to obtain an accurately representative class attribute vector with high weight on its key dimensions, as shown in Fig. 2. Let $A^s \in \mathbb{R}^{C_s \times d}$ be the class attribute matrix for $C^s$ seen classes where each row corresponds to the $d$ dimensional class attribute vector of the corresponding class. The ARN consists of the following stack of operations.

We first normalize the input $A^s$ across the dimension $d$ for each class independently using layer-norm (Ba et al., 2016), as $A^s_L = LayerNorm(A^s)$. Over the layer-norm, we perform the linear projection followed by the Gaussian error linear unit (GELU) (Hendrycks and Gimpel, 2016) activation function as $A^s_P = GELU(A^s_L W_1)$. Here, the linear projection helps in the expansion of the current dimension of the class attribute vector. Here, $W_1 \in \mathbb{R}^{d \times h}$ denotes weight for the linear projection and $A^s_P \in \mathbb{R}^{C^{\prime} \times h}$. Further, we apply the gating unit, which helps achieve a better representation of the attribute information. In the ARN, the gating unit performs cross-feature learning on the higher dimensional class attribute information vector ($A^s_P$). For this purpose, we split the $A^s_P$ into two parts, each with half the dimension, i.e. $A^s_{P1} \in \mathbb{R}^{C^{\prime} \times h/2}$ and $A^s_{P2} \in \mathbb{R}^{C^{\prime} \times h/2}$. Both halves are processed as

learning. CARNet refines the class attribute vectors for better representation with the help of end-to-end joint loss. Here, a gating-based attribute refinement network (ARN) is used to refine the class attribute vectors. These refined class attribute vectors are mapped through an attribute embedder (AE) to extract an efficient class prototype vector corresponding to each class. The visual features are extracted using a pre-trained ResNet-101 model. These visual features are then combined with the class prototype vectors in the feature-prototype combiner (FPC) for classification, as shown in Fig. 2. The ARN, the AE, and the FPC are trained end-to-end based on the sum of two losses, namely, the circle loss and the softmax cross-entropy loss. These two losses guide the gating unit to yield refined attributes, which lead to better class prototype vectors through AE. In this section, we present a detailed description of our model CARNet.
follows:
\[
A^s_G = A^s_{P1} \odot \text{Conv1D}(\text{LayerNorm}(A^s_{P2})), \tag{1}
\]
where Conv1D represents 1-D Convolution, which enables the spatial projection, \(A^s_G \in \mathbb{R}^{C \times h/2}\), and \(\odot\) represents Hadamard product (i.e., element-wise multiplication) which is a linear gating. Also, we can interpret this operation as self-weighting on each dimension of the attribute vector. The important dimensions will get high weight while the other has lower weight. The Hadamard product enables the refinement network to keep both information (i.e., raw \(A^s_{P1}\) and spatial projection of \(A^s_{P2}\)) in the output of the gating unit, i.e., \(A^s_G\). During training, the Conv1D(LayerNorm(A^s_{P2})) is initialized as an identity matrix. Finally, the output of the ARN is obtained through residual learning, as shown below:
\[
A^s_R = A^s_G W_2 + A^s \tag{2}
\]
where \(W_2 \in \mathbb{R}^{h/2 \times d}\) denotes weight for the linear projection, \(\odot\) denotes element-wise-addition, and \(A^s_R \in \mathbb{R}^{C \times d}\) is the final refined class attribute information. The linear projection helps \(A^s_G\) to have the same dimension as \(A^s\).

Overall, the refinement network stacks the above-mentioned operations one after the other, as shown in Fig. 2, and this set of operations can be repeated multiple times. As the set of operations can be repeated multiple times for better attribute refinement, it can cause the vanishing gradient problem that is very common in typical gating units. However, the residual learning in Eq. (2) helps to alleviate this issue.

4.2 Attribute Embedder (AE)
After getting the refined attribute for seen classes, we perform attribute to visual mapping using the AE (as shown in Fig. 2) to obtain the class prototype matrix \(P^s\) for the \(C^s\) seen classes, where each row corresponds to the class prototype vector \(p^s_c\) of the respective seen class \(c \in \{C^s\}\).
\[
P^s = AE(A^s_R), \quad P^s \in \mathbb{R}^{C^s \times f}, \tag{3}
\]
Overall, the AE is a simple 3-layered multi-layer perceptron (MLP) architecture, which is used to perform the attribute-to-visual mapping. Here, \(f\) denotes the dimension of the visual feature vector.

4.3 Feature-Prototype Combiner (FPC)
The visual features \(V^s_r \in \mathbb{R}^{N \times f}\) are extracted by passing the images of the seen classes \(D^s_r\) through a pretrained ResNet-101 model (no fine-tuning). These visual features are combined with the class prototype vectors in the FPC through scaled cosine similarity between the \(P^s\) and the \(V^s_r\) (Skorokhodov and Elhoseiny, 2021). The scaled cosine similarity \(\text{scos}\) scales and normalizes the class prototype vectors and the extracted visual features before computing the dot product between them as follows:
\[
\text{scos}(v^s_{tr}, p^s_c) = \left( \beta \cdot \frac{v^s_{tr}}{\|v^s_{tr}\|} \right)^\top \left( \beta \cdot \frac{p^s_c}{\|p^s_c\|} \right), \tag{4}
\]
where \(v^s_{tr} \in V^s_r\) is the \(f\)-dimensional extracted visual feature for a sample, \(p^s_c\) is the class prototype vector of class \(c \in \{C^s\}\). \(\beta\) is the scaling hyperparameter, which has the same impact as setting a high temperature of \(\beta^2\) in softmax (Liu et al., 2018). Here, normalization reduces the variance of the class prototype vectors and the visual features, which helps in achieving better performance.

4.4 Training of the CARNet using only Seen Classes
The CARNet is trained by minimizing the circle loss and the softmax cross-entropy loss over the end-to-end network comprising the ARN and the AE. We present these loss functions and the learning algorithm of CARNet in this subsection. Without loss of generality, let us assume that \(v^s_{tr} \in V^s_r\) be the extracted visual feature of a sample, which belongs to the seen class \(k \in \{C^s\}\).

Circle Loss: Generally, two kinds of losses are involved in the literature: one kind of losses, like L2-softmax, AM-softmax, and angular softmax are good candidates for classification, while the other kind of losses, like triplet loss, N-pair loss, contrastive loss, and the margin loss are good candidates for pair-wise similarity. The circle loss (Sun et al., 2020) aims to unify both classification and pair-wise similarity representation. Hence, it is a good candidate for optimizing ARN, AE, and FPC. Moreover, it enhances the feature learning and better separability by using flexible optimization and definite convergence target (Sun et al., 2020). The main objective of feature learning is to increase within-class similarity \(s_p\) while reducing between-class similarity \(s_n\). The circle loss unifies both
class-level labels and pair-wise similarity with $K$ within-class similarity scores ($s_p$) and $L$ between-class similarity scores ($s_n$) and is defined as:

$$\mathcal{L}_{circle} = \log \left[ 1 + \frac{1}{K} \sum_{j=1}^{K} \exp(\gamma \alpha_{n}^j (s_n^j - \Delta_n)) \right]$$

(5)

where $\alpha_{p}^j = [1 + m - s_p^j]_+$, and $\alpha_{n}^j = [s_n^j + m]_+$ are weighting factors, $\alpha_{p}^j > 0$, $\alpha_{n}^j > 0$, and $\gamma$ is a scaling factor. Here $[.]_+$ denotes cut-off at Zero. $\Delta_n = m$ and $\Delta_p = 1 - m$ are the between-class and within-class margins, respectively.

In CARNet, we use the sample $v_{ts}^k$, as the anchor and the corresponding class prototype vector $p_k^C$ of class $k$ as the positive sample and the remaining class prototype vectors $p_j^C$ of the seen classes as negative samples. The cosine similarity is used to determine the positive similarity $s_p$ and negative similarity $s_n$ as follows:

$$s_p^k = \frac{v_{ts}^k \cdot p_k^C}{\|v_{ts}^k\| \|p_k^C\|}$$

(6)

$$s_n^j = \frac{v_{ts}^k \cdot p_j^C}{\|v_{ts}^k\| \|p_j^C\|}$$

(7)

Hence, the circle loss in Eq. (5) is modified in CARNet as:

$$\mathcal{L}_{circle} = \log \left[ 1 + \frac{1}{L} \sum_{j \in \{C\} \setminus \{k\}} \exp(\gamma \alpha_{n}^j (s_n^j - \Delta_n)) \right]$$

(8)

**Softmax Cross-Entropy Loss:** To improve the classification, the softmax cross-entropy loss ($\mathcal{L}_{soft-ce}$) is applied over the computed scaled cosine similarity in Eq. (4), as shown below:

$$\mathcal{L}_{soft-ce} = - \log \frac{e^{scos(v_{ts}, p_i)}}{\sum_{i \in \{C\}} e^{scos(v_{ts}, p_i)}}$$

(9)

Thus, the training of the CARNet is achieved by learning the weights of the ARN and the AE using the losses in Eq. (8) and Eq. (9), as shown below:

$$\mathcal{L}_{CARNet} = \mathcal{L}_{soft-ce} + \lambda \mathcal{L}_{circle}$$

(10)

Thus, the loss in Eq. (10) is optimized during training. It is to be noted that only the seen class information ($D_{tr}, \{C^s\}, A^s$) is used during training.

### 4.5 Inference: Seen and Unseen Classes

The proposed CARNet method is based on the fixed body and dynamic head (classification layer) architecture. As the model is trained with only seen classes, the classification layer has neurons corresponding to the seen classes only, i.e., $C^s$ neurons. Further, we simply modify the output head and enable it for unseen classes ($\{C^u\}$) using its class attribute information ($A^u$) as per the following procedure:

1. Pass the unseen class attribute information $A^u$ to the trained ARN and get the output $A^u_{R}$.
2. Pass the $A^s_{R}$ to the trained AE and get the unseen class prototype vectors $P^u \in \mathbb{R}^{C_u \times f}$ for $C_u$ unseen classes.
3. The unseen class prototype vectors ($P^u$) are stacked with seen class prototype vectors ($P^s$) as follows:

$$P = \left[ P^s \quad P^u \right]$$

(11)

After computation of $P$, we compute the scaled cosine similarity score $scos(v_{ts}, p_i)$ as follows:

$$scos(v_{ts}, p_i) = \beta \cdot \frac{v_{ts} \cdot p_i}{\|v_{ts}\| \|p_i\|}$$

(12)

where $p_i \in P$ is the class prototype vector of class $i \in \{C^s\} \cup \{C^u\}$, and $v_{ts} \in V_{ts}^s \cup V_{ts}^u$. Here, $V_{ts}^s$ and $V_{ts}^u$ are the extracted visual features using pretrained Resnet-101 model for the test images of seen and unseen classes, respectively. Finally, we perform a traditional way of classification and choose the class based on the highest cosine similarity score.

### 5 Experiments

We conduct extensive experiments on five benchmark ZSL datasets (description given in Table 5 in appendix) to evaluate the performance of ZSL in two settings, i.e., conventional ZSL and GZSL. In conventional ZSL, test samples only consist of unseen classes, and we compute Top-1 accuracy for unseen classes (Acc) during inference. In GZSL, test samples consist of both seen and unseen classes, and we compute Top-1 accuracy for both seen ($SA$) and unseen ($UA$) classes. Further, we compute its corresponding harmonic mean (HM) using $SA$ and $UA$, which is defined as $HM$. We also evaluate the model on the continual GZSL (CGZSL) setting.
### 5.1 Comparison with Baseline Methods

In this section, the performance of CARNet is evaluated against strong baseline models for three ZSL settings. Results for both the ZSL settings are provided in Table 1.

#### Conventional ZSL:
From Table 1, it is observed that the proposed CARNet outperforms all non-generative ZSL methods in conventional ZSL setting, by 10.3%, 4.1%, 2.6%, 1.7% and 1.3% absolute gain for CUB (Welinder et al., 2010), AWA1 (Lampert et al., 2009), AWA2 (Lampert et al., 2013), and aPY (Farhadi et al., 2009) and SUN (Patterson and Hays, 2012) datasets, respectively. Also, in comparison to generative methods of ZSL, the proposed approach shows 6% and 0.4% absolute gain over CUB and AWA1 datasets, respectively.

On the remaining datasets, the model outperforms all the generative models but shows competitive performance to IZF (Shen et al., 2020).

### Generalized zero-shot learning:
In this setting, the proposed non-generative CARNet model outperforms all non-generative ZSL methods with the absolute gain of 1.2%, 4.1%, 2.9%, 4.4%, and 4.2% for SUN, CUB, AWA1, AWA2, and aPY datasets, respectively. Moreover, CARNet yields the best $HM$ compared to all generative/non-generative methods for CUB, AWA1, and AWA2 datasets and better/similar $HM$ on aPY dataset. The performance of CARNet is outperformed by IZF (Shen et al., 2020) on the SUN dataset. It should be noted that IZF is an invertible flow-based generative model which learns from the bi-directional mapping between the visual and the
attribute space, enabling it to have better performances than other generative ZSL methods. However, CARNet outperforms IZF for CUB, AWA1, and AWA2 by a significant margin of 2.8%, 2.4%, and 3.7%, respectively, and yields similar result for aPY dataset (only a difference of 0.1%). In addition to performance gains, the proposed approached CARNet is characteristically advantageous over generative ZSL approaches in that, while the CARNet uses only the attribute vectors of seen classes during training, the generative ZSL methods use the attribute vectors of both seen and unseen classes during training which is not a realistic scenario in a dynamic environment.

Table 3 presents the computational time required to train the various ZSL methods. It can be observed from this table that the CARNet is at least $68 \times$ slower than generative methods for SUN, CUB, AWA1, and AWA2, as observed in Table 3. This can be attributed to the fact that CARNet only needs to process class attribute vectors through ARN and AE.

Thus, the proposed CARNet is a desirable candidate for conventional and generalized ZSL, owing to its performance, data requirements and computational speed.

Continual generalized zero-shot learning (CGZSL): While ZSL assumes data for all tasks to be available apriori, data may arrive in a sequential manner in real-world, and collecting all the data in memory is cumbersome. Hence, we further evaluate the performance of CARNet for the highly challenging CGZSL setting proposed in Skorokhodov and Elhoseiny (2021). This setting assumes that the data arrives in a sequence of tasks and only the current task data is available for training. Thus, after training for a sequence of $[1, \ldots, t]$ tasks, all classes in the $[1, \ldots, t]$ tasks are considered as seen classes and classes from $(t + 1)$ onward are considered as unseen classes. As experience replay-based methods generally outperform regularization-based methods in the literature (Delange et al., 2021), CARNet is equipped for CGZSL using experience replay (ER) (Chaudhry et al., 2019b) strategy with class-balanced reservoir (CBR) sampling (Chrysakis and Moens, 2020). We measure the performance of CGZSL method using $SA$, $UA$, and $HM$ at each task. Further, we compute the mean of $SA$, $UA$, and $HM$ of overall tasks and denote it as $mSA$, $mUA$, and $mHM$ (Skorokhodov and Elhoseiny, 2021). We present the CGZSL results in Table 2, along with the state-of-the-art CGZSL methods. Our method outperforms all existing methods by an absolute gain of 7.1% and 0.4% for CUB and SUN datasets, respectively. We also provide the performance of CARNet with CBR sampling (CARNet-ER+CBR) and without CBR sampling (CARNet-ER) in Table 2.

5.2 Ablation Study: Significance of Individual Components in CARNet

In this section, to emphasize the significance of individual components of CARNet, we perform an extensive ablation study over all the components and hyperparameters.

We study the effect of individual components of the CARNet, namely, (i) ARN (ii) AE (iii) circle loss (iv) softmax cross-entropy loss (v) scaled cosine similarity or Dot Product. We present the

Table 2: Continual Generalized Zero-shot Learning Results

<table>
<thead>
<tr>
<th>Methods</th>
<th>CUB</th>
<th>SUN</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>mUA</td>
<td>mSA</td>
</tr>
<tr>
<td>Seq-CARNet</td>
<td>4.5</td>
<td>11.3</td>
</tr>
<tr>
<td>Seq-CNZSL (Skorokhodov and Elhoseiny, 2021)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Seq-CVAE (Mishra et al., 2018)</td>
<td>8.6</td>
<td>24.7</td>
</tr>
<tr>
<td>Seq-CADA-V AE (Schonfeld et al., 2019)</td>
<td>14.4</td>
<td>40.8</td>
</tr>
<tr>
<td>CNZSL-AGEM (Skorokhodov and Elhoseiny, 2021)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CNZSL-EWC-online (Skorokhodov and Elhoseiny, 2021)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>CNZSL-MAS-online (Skorokhodov and Elhoseiny, 2021)</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>GRCZSL (Gautam et al., 2021a)</td>
<td>14.1</td>
<td>41.9</td>
</tr>
<tr>
<td>CZSL-CV+res (Gautam et al., 2020)</td>
<td>13.5</td>
<td>44.9</td>
</tr>
<tr>
<td>CZSL-CA+res (Gautam et al., 2020)</td>
<td>32.8</td>
<td>44.0</td>
</tr>
<tr>
<td>Tf-GCZSL (Gautam et al., 2021b)</td>
<td>32.4</td>
<td>46.6</td>
</tr>
<tr>
<td>CARNet-ER (Ours)</td>
<td>43.0</td>
<td>45.8</td>
</tr>
<tr>
<td>CARNet-ER+CBR (Ours)</td>
<td>43.4</td>
<td>47.4</td>
</tr>
</tbody>
</table>
results of this study in Table 4. The softmax cross-entropy loss is an imperative loss for the model, as the proposed CARNet has to perform classification. Therefore, we kept it in all cases in the component analysis of Table 4. It is very evident from the results that attribute refinement significantly boosts the performance of CARNet. Moreover, the scaled-cosine similarity is another important component, and helps to outperform the model with another potential candidate, namely, dot product by a large margin.

<table>
<thead>
<tr>
<th>Methods</th>
<th>SUN</th>
<th>CUB</th>
<th>AWA1</th>
<th>AWA2</th>
</tr>
</thead>
<tbody>
<tr>
<td>RelNet (Sung et al., 2018)</td>
<td>-</td>
<td>25 min</td>
<td>40 min</td>
<td>40 min</td>
</tr>
<tr>
<td>DCN (Liu et al., 2018)</td>
<td>40 min</td>
<td>50 min</td>
<td>-</td>
<td>55 min</td>
</tr>
<tr>
<td>CIZSL (Elhoseiny and Elfeki, 2019)</td>
<td>3 Hr</td>
<td>2 Hr</td>
<td>3 Hr</td>
<td>3 Hr</td>
</tr>
<tr>
<td>CVC-ZSL (Li et al., 2019)</td>
<td>3 Hr</td>
<td>3 Hr</td>
<td>1.5 Hr</td>
<td>1.5 Hr</td>
</tr>
<tr>
<td>LsrGAN (Vyas et al., 2020)</td>
<td>1.1 Hr</td>
<td>1.25 Hr</td>
<td>-</td>
<td>1.5 Hr</td>
</tr>
<tr>
<td>TF-VAEGAN (Narayan et al., 2020)</td>
<td>1.5 Hr</td>
<td>1.75 Hr</td>
<td>-</td>
<td>2 Hr</td>
</tr>
<tr>
<td>CNZSL (Skorokhodov and Elhoseiny, 2021)</td>
<td>20 sec</td>
<td>20 sec</td>
<td>30 sec</td>
<td>30 sec</td>
</tr>
<tr>
<td>CARNet (Ours)</td>
<td>35 sec</td>
<td>22 sec</td>
<td>110 sec</td>
<td>77 sec</td>
</tr>
</tbody>
</table>

Table 4: Component Analysis

6 Conclusion

In this work, we developed the circle loss guided gating-based attribute-refinement network for handling ZSL, GZSL, and continual-GZSL tasks. CARNet refines the attribute through a gating unit where it improves the attribute representation by learning a self-weight on each attribute dimension in a projected space. These refined attributes improve the embedding, which helps to overcome the model bias towards the seen classes. The whole model is guided by the circle loss along with the standard softmax cross-entropy loss, which maximizes the inter-class separability and intra-class similarity. Also, unlike the generative method, CARNet does not require the attribute vector of the unseen classes during training. The proposed method is quite fast, as the attribute refinement network and the attribute embedder need to process only the class attribute vectors during training. This work shows that a simple MLP-based architecture can outperform various highly computationally expensive ZSL methods. This approach needs to be explored with generative methods and other applications of ZSL, like zero-shot for sketch-based image retrieval, action recognition, and natural language processing.

7 Limitations

One major limitation is that the inference data must be from the same domain, as the proposed model cannot handle data from the other domains on which the model is not trained. Another limitation of the proposed method is that it requires task id during training in the CGZSL setting, without which CARNet cannot optimize the proposed model properly. However, in realistic scenarios, it is not necessary for the data to arrive with well-defined task-boundaries. Hence, the requirement of task id during training is a drawback of our proposed model.
Acknowledgements

We would like to thank the Wipro IISc Research and Innovation Network (WIRIN, Grant No-99325T) and National Research Foundation, Singapore under its AI Singapore Programme (AISG Award No: AISG2-RP-2021-027) for funding this research.

References


Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. 2009. 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.


A Appendix

A.1 ZSL Datasets

Five common benchmark datasets used in ZSL are Scene UNderstanding (SUN) dataset, Caltech-UCSD Birds (CUB) dataset, Animals With Attributes (AWA1, AWA2) dataset and Attribute Pascal and Yahoo (aPY) dataset. A brief description of these datasets is provided in Table 5.

A.2 Continual Learning Setup

In a continual learning setup, all the classes (both seen and unseen classes) of a given dataset are grouped together and split into T tasks. Further, all classes up to the current task t are taken as seen classes. All classes of the remaining tasks, namely t+1 to T, are taken as unseen classes. At task t, the input consists of training images for all classes in task t. During testing, the test data consists of images belonging to all classes of tasks 1 to T. The Task split for the CUB and SUN dataset are as follows:

1. The CUB dataset consists of 200 classes. The dataset is split into 20 tasks with 10 classes in each task.
2. The SUN dataset consists of 717 classes. The dataset is split into 15 tasks, with 47 classes in the first 3 tasks and 48 classes in the remaining 12 tasks.

A.3 Implementation Details

We use ResNet-101 as a pretrained model, which is pretrained on ImageNet as the backbone for visual feature extraction. CARNet is trained using the Adam optimizer with a learning rate of 0.001 for APY and 0.0005 for all remaining datasets. Further, weight decay of 0.001 for APY and SUN, and 0.0001 for other datasets are used. We choose $\beta = 5$ across all datasets for computing scaled cosine similarity and $m$, $\gamma$, $\lambda$ are taken as shown in Table 6. We performed all our experiments on RTX 2080 GPU with i7 processor and 32 GB RAM.

A.4 Impact of Hyper-parameters ($\gamma$ and $m$)

In Fig. 3, we provide three 3-D plots for $H_M$, $S_A$, and $U_A$ on AWA2 dataset to study the effects of $m$ and $\gamma$ on CARNet. From the figure, it is seen that $H_M$, $S_A$, and $U_A$ vary only 3%, 2%, and 4% with changes in $m$ and $\gamma$, and still outperforms most generative and all non-generative ZSL methods.
<table>
<thead>
<tr>
<th>Dataset</th>
<th>Attribute Dimension</th>
<th>#Seen Classes</th>
<th>#Unseen Classes</th>
<th>Total Classes</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SUN</td>
<td>102</td>
<td>645</td>
<td>72</td>
<td>717</td>
<td>Fine-grained</td>
</tr>
<tr>
<td>CUB</td>
<td>1024</td>
<td>150</td>
<td>50</td>
<td>200</td>
<td>Fine-grained</td>
</tr>
<tr>
<td>AWA1</td>
<td>85</td>
<td>40</td>
<td>10</td>
<td>50</td>
<td>Coarse-grained</td>
</tr>
<tr>
<td>AWA2</td>
<td>85</td>
<td>40</td>
<td>10</td>
<td>50</td>
<td>Coarse-grained</td>
</tr>
<tr>
<td>aPY</td>
<td>64</td>
<td>20</td>
<td>12</td>
<td>32</td>
<td>Coarse-grained</td>
</tr>
</tbody>
</table>

Table 5: Zero-shot learning benchmark datasets

Thus, it can be observed that the CARNet model is robust to large changes in the $\gamma$ and $m$.

<table>
<thead>
<tr>
<th>Hyperparameter</th>
<th>SUN</th>
<th>CUB</th>
<th>AWA1</th>
<th>AWA2</th>
<th>APY</th>
</tr>
</thead>
<tbody>
<tr>
<td>$m$</td>
<td>0.4</td>
<td>0.1</td>
<td>0.3</td>
<td>0.4</td>
<td>0.2</td>
</tr>
<tr>
<td>$\gamma$</td>
<td>0.5</td>
<td>0.9</td>
<td>1.0</td>
<td>0.5</td>
<td>1.0</td>
</tr>
<tr>
<td>$\lambda$</td>
<td>1.2</td>
<td>0.7</td>
<td>0.2</td>
<td>0.5</td>
<td>1.0</td>
</tr>
</tbody>
</table>

Table 6: Values taken by the Hyperparameters for different datasets