Exploiting a Joint Embedding Space for Generalized Zero-Shot Semantic Segmentation

Donghyeon Baek* Youngmin Oh* Bumsub Ham†
School of Electrical and Electronic Engineering, Yonsei University
https://cvlab.yonsei.ac.kr/projects/JoEm

Abstract

We address the problem of generalized zero-shot semantic segmentation (GZS3) predicting pixel-wise semantic labels for seen and unseen classes. Most GZS3 methods adopt a generative approach that synthesizes visual features of unseen classes from corresponding semantic ones (e.g., word2vec) to train novel classifiers for both seen and unseen classes. Although generative methods show decent performance, they have two limitations: (1) the visual features are biased towards seen classes; (2) the classifier should be retrained whenever novel unseen classes appear. We propose a discriminative approach to address these limitations in a unified framework. To this end, we leverage visual and semantic encoders to learn a joint embedding space, where the semantic encoder transforms semantic features to semantic prototypes that act as centers for visual features of corresponding classes. Specifically, we introduce boundary-aware regression (BAR) and semantic consistency (SC) losses to learn discriminative features. Our approach to exploiting the joint embedding space, together with BAR and SC terms, alleviates the seen bias problem. At test time, we avoid the retraining process by exploiting semantic prototypes as a nearest-neighbor (NN) classifier. To further alleviate the bias problem, we also propose an inference technique, dubbed Apollonius calibration (AC), that modulates the decision boundary of the NN classifier to the Apollonius circle adaptively. Experimental results demonstrate the effectiveness of our framework, achieving a new state of the art on standard benchmarks.

1. Introduction

Recent works using convolutional neural networks (CNNs) [6, 36, 45, 57] have achieved significant success in semantic segmentation. They have proven effective in various applications such as image editing [34] and autonomous driving [54], but semantic segmentation in the wild still has two limitations. First, existing methods fail to generalize to new domains/classes, assuming that training and test samples share the same distribution. Second, they require lots of training samples with pixel-level ground-truth labels prohibitively expensive to annotate. As a result, current methods could handle a small set of pre-defined classes only [23].

As alternatives to pixel-level annotations, weakly-supervised semantic segmentation methods propose to exploit image-level labels [19], scribbles [35], and bounding boxes [8], all of which are less labor-intensive to annotate. These methods, however, also require a large number of weak supervisory signals to train networks for novel classes. On the contrary, humans can easily learn to recognize new concepts in a scene with a few visual examples, or even with descriptions of them. Motivated by this, few-and zero-shot learning methods [29, 42, 48] have been proposed to recognize objects of previously unseen classes with

*Equal contribution. †Corresponding author.
a few annotated examples and even without them, respectively. For example, few-shot semantic segmentation (FS3) methods [47, 49] typically exploit an episode training strategy, where each episode consists of randomly sampled support and query sets, to estimate query masks with a few annotated support examples. Although these FS3 methods show decent performance for unseen classes, they are capable of handling a single unseen class only. Recently, the work of [56] first explores the problem of zero-shot semantic segmentation (ZS3), where it instead exploits pre-trained semantic features using class names (i.e., word2vec [38]). This work, however, focuses on predicting unseen classes, even if a given image contains both seen and unseen ones. To overcome this, generalized ZS3 (GZS3) has recently been introduced to consider both seen and unseen classes in a scene during inference. Motivated by generative approaches [2, 50, 52] in zero-shot image classification, many GZS3 methods [3, 15, 32] first train a segmentation network that consists of a feature extractor and a classifier with seen classes. They then freeze the feature extractor to extract visual features, and discard the classifier. With the fixed feature extractor, a generator [14, 25] is trained to produce visual features from semantic ones (e.g., word2vec) of corresponding classes. This enables training novel classifiers with real visual features of seen classes and generated ones of unseen classes (Fig. 1 top). Although generative methods achieve state-of-the-art performance in GZS3, they have the following limitations: (1) the feature extractor is trained without considering semantic features, causing a bias towards seen classes. The seen bias problem becomes even worse through a multi-stage training strategy, where the generator and novel classifiers are trained using the feature extractor; (2) the classifier needs to be re-trained whenever a particular unseen class is newly included/excluded, hindering deployment in a practical setting, where unseen classes are consistently emerging.

We introduce a discriminative approach for GZS3, dubbed JoEm, that addresses the limitations of generative methods in a unified framework (Fig. 1 bottom). Specifically, we exploit visual and semantic encoders to learn a joint embedding space. The semantic encoder transforms semantic features into semantic prototypes acting as centers for visual features of corresponding classes. Our approach to using the joint embedding space avoids the multi-stage training, and thus alleviates the seen bias problem. To this end, we propose to minimize the distances between visual features and corresponding semantic prototypes in the joint embedding space. We have found that visual features at object boundaries could contain a mixture of different semantic information due to the large receptive field of deep CNNs. Directly minimizing the distances between the visual features and semantic prototypes might distract discriminative feature learning. To address this, we propose a boundary-aware regression (BAR) loss that exploits semantic prototypes linearly interpolated to gather the visual features at object boundaries along with its efficient implementation. We also propose to use a semantic consistency (SC) loss that transfers relations between seen classes from a semantic embedding space to the joint one, regularizing the distances between semantic prototypes of seen classes explicitly. At test time, instead of re-training the classifier as in the generative methods [3, 15, 32], our approach to learning discriminative semantic prototypes enables using a nearest neighbor (NN) classifier [7] in the joint embedding space. In particular, we modulate the decision boundary of the NN classifier using the Apollonius circle. This Apollonius calibration (AC) method also makes the NN classifier less susceptible to the seen bias problem. We empirically demonstrate the effectiveness of our framework on standard GZS3 benchmarks [10, 40], and show that AC boosts the performance significantly. The main contributions of our work can be summarized as follows:

- We introduce a simple yet effective discriminative approach for GZS3. We propose BAR and SC losses, which are complementary to each other, to better learn discriminative representations in the joint embedding space.
- We present an effective inference technique that modulates the decision boundary of the NN classifier adaptively using the Apollonius circle. This alleviates the seen bias problem significantly, even without re-training the classifier.
- We demonstrate the effectiveness of our approach exploiting the joint embedding space on standard benchmarks for GZS3 [10, 40], and show an extensive analysis with ablation studies.

2. Related work

Zero-shot image classification. Many zero-shot learning (ZSL) [11, 29, 42] methods have been proposed for image classification. They typically rely on side information, such as attributes [11, 27], semantic features from class names [37, 55], or text descriptions [30, 44], for relating unseen and seen object classes. Early ZSL methods [1, 12, 44, 55] focus on improving performance for unseen object classes, and typically adopt a discriminative approach to learn a compatibility function between visual and semantic embedding spaces. Among them, the works of [13, 30, 37, 53] exploit a joint embedding space to better align visual and semantic features. Similarly, our approach leverages the joint embedding space, but differs in that (1) we tackle the task of GZS3, which is much more challenging than image classification, and (2) we propose two complementary losses together with an effective inference technique, enabling learning better representations and alleviating a bias towards seen classes. Note that
a straightforward adaptation of discriminative ZSL methods [4, 28, 41] to generalized ZSL (GZSL) suffers from the seen bias problem severely. To address this, a calibrated stacking method [5] proposes to penalize scores of seen object classes at test time. This is similar to our AC in that both aim at reducing the seen bias problem at test time. The calibrated stacking method, however, shifts the decision boundary with a constant value, while we modulate the decision boundary adaptively. Recently, instead of learning the compatibility function between visual and semantic embedding spaces, generative methods [2, 22, 31, 46, 50, 52] attempt to address the task of GZSL by using generative adversarial networks [14] or variational auto-encoders [25]. They first train a generator to synthesize visual features from corresponding semantic ones or attributes. The generator then produces visual features of given unseen classes, and uses them to train a new classifier for both seen and unseen classes. In this way, generative methods reformulate the task of GZSL as a standard classification problem, outperforming the discriminative ones, especially on the generalized setting.

**Zero-shot semantic segmentation.** Recently, there are many attempts to extend ZSL methods for image classification to the task of semantic segmentation. They can be categorized into discriminative and generative methods. The work of [56] adopts the discriminative approach for ZS3, focusing on predicting unseen classes in a hierarchical way using WordNet [39]. The work of [21] argues that adverse effects from noisy samples are significant especially in the problem of ZS3, and proposes uncertainty-aware losses [24] to prevent a segmentation network from overfitting to them. This work, however, requires additional parameters to estimate the uncertainty, and outputs a binary mask for a given class only. SPNet [51] exploits a semantic embedding space to tackle the task of GZS3, mapping visual features to fixed semantic ones. Differently, we propose to use a joint embedding space, better aligning visual and semantic spaces, together with two complementary losses. In contrast to discriminative methods, ZS3Net [3] leverages a generative moment matching network (GMMN) [33] to synthesize visual features from corresponding semantic ones. Training ZS3Net requires three stages for a segmentation network, the GMMN, and a new classifier, respectively. While ZS3Net exploits semantic features of unseen classes at the last stage only, CSRL [32] incorporates them in the second stage, encouraging synthesized visual features to preserve relations between seen and unseen classes in the semantic embedding space. CaGNet [15] proposes a contextual module using dilated convolutional layers [43] along with a channel-wise attention mechanism [20]. This encourages the generator to better capture the diversity of visual features. The generative methods [3, 15, 32] share the common limitations as follows: First, they require re-training the classifier whenever novel unseen classes are incoming. Second, they rely on the multi-stage training framework, which might deteriorate the seen bias problem, with several hyperparameters (e.g., the number of synthesized visual features and the number of iterations for training a new classifier). To address these limitations, we advocate using a discriminative approach that avoids the multi-stage training scheme and re-training the classifier.

3. **Method**

In this section, we concisely describe our approach to exploiting a joint embedding space for GZS3 (Sec. 3.1), and introduce three training losses (Sec. 3.2). We then describe our inference technique (Sec. 3.3).

3.1. **Overview**

Following the common practice in [3, 15, 32, 51], we divide classes into two disjoint sets, where we denote by $S$ and $U$ sets of seen and unseen classes, respectively. We train our model including visual and semantic encoders with the seen classes $S$ only, and use the model to predict pixel-wise semantic labels of a scene for both seen and unseen classes, $S$ and $U$, at test time. To this end, we jointly update both encoders to learn a joint embedding space. Specifically, we first extract visual features using the visual encoder. We then input semantic features (e.g., word2vec [38]) to the semantic encoder, and obtain semantic prototypes that represent centers for visual features of corresponding classes. We have empirically found that visual features at object boundaries could contain a mixture of different semantics (Fig. 2(a) middle), which causes discrepancies between visual features and semantic prototypes. To address this, we propose to use linearly interpolated semantic prototypes (Fig. 2(a) bottom), and minimize the distances between the visual features and semantic prototypes (Fig. 2(b)). We also encourage the relationships between semantic prototypes to be similar to those between semantic features explicitly (Fig. 3). At test time, we use the semantic prototypes of both seen and unseen classes as a NN classifier without re-training. To further reduce the seen bias problem, we modulate the decision boundary of the NN classifier adaptively (Fig. 4(e)). In the following, we describe our framework in detail.

3.2. **Training**

We define an overall objective for training our model end-to-end as follows:

$$\mathcal{L} = \mathcal{L}_{ce} + \mathcal{L}_{bar} + \lambda \mathcal{L}_{ac},$$  \hspace{1cm} (1)$$

where we denote by $\mathcal{L}_{ce}$, $\mathcal{L}_{bar}$, and $\mathcal{L}_{ac}$ cross-entropy (CE), BAR, and SC terms, respectively, balanced by the parameter $\lambda$. In the following, we describe each loss in detail.
Comparison of visual feature and semantic prototype maps.

(a) Discrepancy between visual feature and semantic prototype maps.

(b) Comparison of $L_{\text{center}}$ and $L_{\text{bar}}$.

Figure 2: (a) While the semantic feature map abruptly changes at object boundaries due to the stacking operation using a ground-truth mask (top), the visual one smoothly varies due to the large receptive field of the visual encoder (middle). We leverage a series of nearest-neighbor and bilinear interpolations to smooth a sharp transition at object boundaries in an efficient way (bottom). (b) Visual features at object boundaries might contain a mixture of different semantics, suggesting that minimizing the distances to the exact semantic prototypes is not straightforward (dashed lines). Our BAR loss exploits a virtual prototype to pull the visual features at object boundaries (solid lines). Best viewed in color.

CE loss. Given an image of size $H_o \times W_o$, the visual encoder outputs a visual feature map $v \in \mathbb{R}^{H \times W \times C}$, where $H, W, C$ are height, width, and the number of channels, respectively. We denote by $y$ a corresponding ground-truth mask, which is resized to the size of $H \times W$ using nearest-neighbor interpolation, and $v(p)$ a $C$-dimensional local visual feature at position $p$. To encourage these visual features to better capture rich semantics specific to the task of semantic segmentation, we use a CE loss widely adopted in supervised semantic segmentation. Differently, we apply this for a set of seen classes (i.e., $S$) only as follows:

$$L_{\text{ce}} = -\frac{1}{|S|} \sum_{c \in S} \sum_{p \in R_c} \log \frac{e^{w_c \cdot v(p)}}{\sum_{j \in S} e^{w_j \cdot v(p)}}$$,

where $w_c$ is a $C$-dimensional classifier weight for a class $c$ and $R_c$ indicates a set of locations labeled as the class $c$ in $y$. We denote by $| \cdot |$ the cardinality of a set.

BAR loss. Although the CE loss trains the classifier to discriminate seen classes, the learned classifier weights $w$ are not adaptable to recognize unseen ones. To address this, we instead use the semantic encoder as a hypernetwork [16] that generates classifier weights. Specifically, the semantic encoder transforms a semantic feature (e.g., $\text{word2vec}$ [38]) into a semantic prototype that acts as a center for visual features of a corresponding class. We then use semantic prototypes of both seen and unseen classes as a NN classifier at test time.

A straightforward way to implement this is to minimize the distances between visual features and corresponding semantic prototypes during training. To this end, we first obtain a semantic feature map $s$ of size $H \times W \times C$ as follows:

$$s(p) = s_c \text{ for } p \in R_c,$$

where we denote by $s_c \in \mathbb{R}^C$ a semantic feature for a class $c$. That is, we stack a semantic feature for a class $c$ into corresponding regions $R_c$ labeled as the same class in the ground truth $y$. Given the semantic feature map, the semantic encoder then outputs a semantic prototype map $\mu$ of size $H \times W \times C$, where

$$\mu(p) = \mu_c \text{ for } p \in R_c.$$

We denote by $\mu_c \in \mathbb{R}^C$ a semantic prototype for a class $c$. Accordingly, we define a pixel-wise regression loss as follows:

$$L_{\text{center}} = \frac{1}{|S|} \sum_{c \in S} \sum_{p \in R_c} d(v(p), \mu(p)),$$

where $d(\cdot, \cdot)$ is a distance metric (e.g., Euclidean distance). This term enables learning a joint embedding space by updating both encoders with a gradient of Eq. (5). We have observed that the semantic feature map $s$ shows a sharp transition at object boundaries due to the stacking operation, making the semantic prototype map $\mu$ discrete accordingly\(^1\), as shown in Fig. 2(a) (top). By contrast, the visual feature map $v$ smoothly varies at object boundaries due to

\(^1\)This is because we use a $1 \times 1$ convolutional layer for the semantic encoder. Note that we could not use a CNN as the semantic encoder since it requires a ground-truth mask to obtain the semantic feature map at test time.
to the large receptive field of the visual encoder as shown in Fig. 2(a) (middle). That is, the visual features at object boundaries could contain a mixture of different semantics. Thus, directly minimizing Eq. (5) might degrade performance, since this could also close the distances between semantic prototypes as shown in Fig. 2(b) (dashed lines). To address this, we exploit linearly interpolated semantic prototypes, which we refer to as virtual prototypes. The virtual prototype acts as a dustbin that gathers the visual features at object boundaries as shown in Fig. 2(b) (solid lines). However, manually interpolating semantic prototypes at all boundaries could be demanding.

We introduce a simple yet effective implementation that gives a good compromise. Specifically, we first downsample the ground-truth mask \( y \) by a factor of \( r \) using nearest-neighbor interpolation. Similar to the previous case, we stack semantic features but with the downsampled ground-truth mask, and obtain a semantic feature map. We upsample this feature map by a factor of \( r \) again using bilinear interpolation, resulting in an interpolated one \( \tilde{s} \) of size \( H \times W \times D \). Given the semantic feature map \( \tilde{s} \), the semantic encoder outputs an interpolated semantic prototype map \( \tilde{\mu} \) accordingly, as shown in Fig. 2(a) (bottom). Using the interpolated semantic prototype map \( \tilde{\mu} \), we define a BAR loss as follows:

\[
\mathcal{L}_{\text{bar}} = \frac{1}{\sum_{c \in S} |R_c|} \sum_{c \in S} \sum_{p \in R_c} d(v(p), \tilde{\mu}(p)). \tag{6}
\]

This term enables learning discriminative semantic prototypes. Note that it has been shown that uncertainty estimates of [21] are highly activated at object boundaries. We can thus interpret the BAR loss as alleviating the influence of visual features at object boundaries in that this term encourages the visual features at object boundaries to be closer to virtual prototypes than the exact ones. Note also that Eq. (5) is a special case of our BAR loss, that is, \( \mu = \tilde{\mu} \) when \( r = 1 \).

**SC loss.** Although CE and BAR terms help to learn discriminative representations in the joint embedding space, they do not impose explicit constraints on the distances between semantic prototypes during training. To complement this, we propose to transfer the relations of semantic features in the semantic embedding space to the semantic prototypes in the joint one. For example, we reduce the distances between semantic prototypes in the joint embedding space if corresponding semantic features are close in the semantic one (Fig. 3). Concretely, we define the relation between two different classes \( i \) and \( j \) in the semantic embedding space as follows:

\[
r_{ij} = \frac{e^{-\tau_s d(s_i, s_j)}}{\sum_{j \in S} e^{-\tau_s d(s_i, s_j)}}, \tag{7}
\]

where \( \tau_s \) is a temperature parameter that controls the smoothness of relations. Similarly, we define the relation in the joint embedding space as follows:

\[
\hat{r}_{ij} = \frac{e^{-\tau_{\mu} d(\mu_i, \mu_j)}}{\sum_{j \in S} e^{-\tau_{\mu} d(\mu_i, \mu_j)}}, \tag{8}
\]

where \( \tau_{\mu} \) is a temperature parameter. To encourage the consistency between two embedding spaces, we define a SC loss as follows:

\[
\mathcal{L}_{\text{sc}} = -\sum_{i \in S} \sum_{j \in S} r_{ij} \log \frac{\hat{r}_{ij}}{r_{ij}}. \tag{9}
\]

This term regularizes the distances between semantic prototypes of seen classes. Similarly, CSRL [32] distills the relations of real visual features to the synthesized ones. It however exploits semantic features of unseen classes during training, suggesting that both generator and classifier should be trained again to handle novel unseen classes.

### 3.3. Inference

Our discriminative approach enables handling semantic features of arbitrary classes at test time without re-training, which is suitable for real-world scenarios. Specifically, the semantic encoder takes semantic features of both seen and unseen classes, and outputs corresponding semantic prototypes. We then compute the distances from individual visual features to each semantic prototype. That is, we formulate the inference process as a retrieval task using the semantic prototypes as a NN classifier in the joint embedding space. A straightforward way to classify each visual feature is to assign the class of its nearest semantic prototype as follows:

\[
\hat{y}_{\text{NN}}(p) = \arg\min_{c \in S \cup U} d(v(p), \mu_c). \tag{10}
\]

Although our approach learns discriminative visual features and semantic prototypes, visual features of unseen classes might still be biased towards those of seen classes (Fig. 4(a)), especially when both have similar appearance. For example, a cat (a unseen object class) is more

Figure 3: We visualize the relations between seen classes in semantic and joint embedding spaces. Our SC loss transfers the relations from the semantic embedding space to the joint one. This adjusts the distances between semantic prototypes explicitly, complementing the BAR loss. Best viewed in color.

\[\text{Table 1}\]

<table>
<thead>
<tr>
<th>Category</th>
<th>Score</th>
</tr>
</thead>
<tbody>
<tr>
<td>Animal</td>
<td>85.2</td>
</tr>
<tr>
<td>Plant</td>
<td>78.9</td>
</tr>
<tr>
<td>Vehicle</td>
<td>89.4</td>
</tr>
<tr>
<td>Furniture</td>
<td>72.5</td>
</tr>
<tr>
<td>Food</td>
<td>87.8</td>
</tr>
</tbody>
</table>

\[\text{Figure 2}\]

**Visual Features Comparison**

- Animal: High scores for animals.
- Plant: Moderate scores for plants.
- Vehicle: High scores for vehicles.
- Furniture: Low scores for furniture.
- Food: High scores for food items.

\[\text{Equation 1}\]

\[
L = \frac{1}{|R|} \sum_{c \in S} \sum_{p \in R_c} d(v(p), \mu_c).
\]

\[\text{Equation 7}\]

\[
r_{ij} = \frac{e^{-\tau_s d(s_i, s_j)}}{\sum_{j \in S} e^{-\tau_s d(s_i, s_j)}}.
\]
Figure 4: Comparison of CS [5] and AC. We visualize semantic prototypes and visual features by stars and circles, respectively. The decision boundary of the NN classifier is shown as dashed lines. (a) We show the seen bias problem with the distribution of visual features in two cases. One is when two different semantic prototypes are distant (left), and the other is the opposite situation (right). Note that visual features of seen classes are tightly clustered, while those of unseen classes are skewed. (b) CS shifts the decision boundary to semantic prototypes of seen classes. Although CS alleviates the seen bias problem (left), it might degrade performance for seen classes (right). Thus, the value of $\gamma$ should be chosen carefully. (c) We modulate the decision boundary with the Apollonius circle. This gives a good compromise between improving performance for unseen classes (left) and preserving that for seen ones (right). Best viewed in color.

![Diagram of seen bias problem](image)

![Diagram of CS](image)

![Diagram of AC](image)

4. Experiments

4.1. Implementation details

**Dataset and evaluation.** We perform experiments on standard GZS3 benchmarks: PASCAL VOC [10] and PASCAL Context [40]. The PASCAL VOC dataset provides 1,464
Table 1: Quantitative results on the PASCAL VOC [10] and Context [40] validation sets in terms of mIoU. Numbers in bold are the best performance and underlined ones are the second best. We report our average scores over five runs with standard deviations in parentheses.

<table>
<thead>
<tr>
<th>Datasets</th>
<th>Methods</th>
<th>unseen-2</th>
<th>unseen-4</th>
<th>unseen-6</th>
<th>unseen-8</th>
<th>unseen-10</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>mIoU$_S$</td>
<td>mIoU$_U$</td>
<td>hIoU</td>
<td>mIoU$_S$</td>
<td>mIoU$_U$</td>
</tr>
<tr>
<td>VOC</td>
<td>DeVISE [12]</td>
<td>68.1</td>
<td>3.2</td>
<td>6.1</td>
<td>64.3</td>
<td>2.9</td>
</tr>
<tr>
<td></td>
<td>SPNet [51]</td>
<td>71.8</td>
<td>34.7</td>
<td>46.8</td>
<td>67.3</td>
<td>21.8</td>
</tr>
<tr>
<td></td>
<td>ZS3Net [3]</td>
<td>72.0</td>
<td>35.4</td>
<td>47.5</td>
<td>66.4</td>
<td>23.2</td>
</tr>
<tr>
<td></td>
<td>CSRL [32]</td>
<td>73.4</td>
<td>45.7</td>
<td><strong>56.3</strong></td>
<td>69.8</td>
<td>31.7</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>68.9</td>
<td><strong>43.2</strong></td>
<td><strong>53.1</strong></td>
<td><strong>67.0</strong></td>
<td><strong>33.4</strong></td>
</tr>
<tr>
<td></td>
<td></td>
<td>(0.4)</td>
<td>(0.9)</td>
<td>(0.4)</td>
<td>(0.4)</td>
<td>(0.8)</td>
</tr>
<tr>
<td>Context</td>
<td>DeVISE [12]</td>
<td>38.5</td>
<td>2.7</td>
<td>5.0</td>
<td>33.4</td>
<td>2.5</td>
</tr>
<tr>
<td></td>
<td>SPNet [51]</td>
<td>38.2</td>
<td>16.7</td>
<td>23.2</td>
<td>36.3</td>
<td>18.1</td>
</tr>
<tr>
<td></td>
<td>ZS3Net [3]</td>
<td>41.6</td>
<td>21.6</td>
<td>28.4</td>
<td>37.2</td>
<td>24.9</td>
</tr>
<tr>
<td></td>
<td>CSRL [32]</td>
<td>41.9</td>
<td>27.8</td>
<td>33.4</td>
<td>39.8</td>
<td>23.9</td>
</tr>
<tr>
<td></td>
<td>Ours</td>
<td>38.2</td>
<td>32.9</td>
<td><strong>35.3</strong></td>
<td><strong>36.9</strong></td>
<td><strong>30.7</strong></td>
</tr>
</tbody>
</table>

Training. For fair comparison, we use DeepLabV3+ [6] with ResNet-101 [18] as our visual encoder. Following ZS3Net [3], ResNet-101 is initialized with the pre-trained weights for ImageNet classification [9], where training samples of seen classes are used only. We train the visual encoder using the SGD optimizer with learning rate, weight decay, and momentum of 2.5e-4, 1e-4, and 0.9, respectively. We adopt a linear layer as the semantic encoder, and train it using the Adam optimizer with learning rate of 2e-4. The entire model is trained for 50 and 200 epochs with a batch size of 32 on PASCAL VOC [10] and Context [40], respectively. We use the poly schedule to adjust the learning rate. In all experiments, we adopt a Euclidean distance for $d(\cdot, \cdot)$. Hyperparameters. We empirically set $(\tau, \sigma, \tau_m)$ to (4, 5, 1) and (4, 5, 1) for PASCAL VOC [10] and Context [40], respectively. Other parameters $(\lambda, \sigma)$ are chosen by cross-validation for each split as in [2]. We provide a detailed analysis on these parameters in the supplementary material.

4.2. Results

We compare in Table 1 our approach with state-of-the-art GZS3 methods on PASCAL VOC [10] and Context [40]. We report average scores over five runs with standard deviations. All numbers for other methods are taken from CSRL [32]. From this table, we have three findings as follows: (1) Our approach outperforms SPNet [51] on both datasets by a considerable margin in terms of mIoU$_U$ and hIoU. This confirms that exploiting a joint embedding space enables learning better representations. (2) We achieve a new state of the art on four out of five PASCAL VOC splits. Although CSRL shows better results on the unseen-2 split, they require semantic features of unseen classes during training. This suggests that both generator and classifier of CSRL should be retrained whenever novel unseen classes appear, which is time consuming. Our discriminative approach is more practical in that the semantic encoder takes semantic features of arbitrary classes without the retraining process. (3) We can clearly see that our approach outperforms all other methods including the generative methods [3, 32] on all splits of PASCAL VOC. A plausible reason is that PASCAL Context contains four times more seen classes including stuff ones than VOC. This makes the generative methods suffer from a severe bias problem towards seen classes.

4.3. Discussion

Ablation study. In the first four rows of Table 2, we present an ablation analysis on different losses in our framework. We adopt a simple NN classifier to focus on the effect of each term. Since the CE loss is crucial to learn discriminative visual features, we incorporate it to all variants. Ablation study on these parameters in the supplementary material.
between the first two rows is whether a series of two inter-
polations is applied to a semantic feature map or not, be-
fore inputting it to a semantic encoder (see Sec. 3.2). We
can also see that explicitly regularizing the distances be-
tween semantic prototypes improves performance for un-
seen classes in the third row. The fourth row demonstrates
that BAR and SC terms are complementary to each other,
achieving the best performance.

Comparison with CS. The last two rows in Table 2 show a
quantitative comparison of CS [5] and AC in terms of mIoU
scores. We can see that both CS and AC improve perfor-
mance for unseen classes by large margins. A reason is that
visual features for unseen classes are skewed and biased to-
towards those of seen classes (Fig. 4(a)). It is worth noting
that AC further achieves a mIoU\textsubscript{U} gain of 2.7\% over CS
with a negligible overhead, demonstrating the effectiveness
of using the Apollonius circle. In Fig. 5, we plot perform-
ance variations according to the adjustable parameter for
each method, i.e., \( \gamma \) and \( \sigma \), in the range of \([0, 12]\) and \([0, 1]\)
with intervals of 0.5 and 0.05 for CS and AC, respectively.
We first compare the mIoU\textsubscript{U} curves in Fig. 5 (left).
For comparison, we visualize the mIoU\textsubscript{S} of the NN clas-
ifier by a dashed line. We can see that AC always gives
better mIoU\textsubscript{U} scores for all mIoU\textsubscript{S} values on the left-hand
side of the dashed line, suggesting that AC is more robust w.r.t.
the adjustable parameter. We also show that how false
natives of seen classes change according to true positives
of unseen classes (TP\textsubscript{U}) in Fig. 5 (right). In particular, we
compute false negatives of seen classes, when they are pre-
dicted as one of unseen classes, denoted by FN\textsubscript{S→U}. We
can clearly see that CS has more FN\textsubscript{S→U} than AC at the same
value of TP\textsubscript{U}, confirming once again that AC is more
robust to the parameter, while providing better results.

Analysis of embedding spaces. To verify that exploit-
ing a joint embedding space alleviates a seen bias prob-
lem, we compare in Table 3 variants of our approach with
ZS3Net [3]. First, we attempt to project visual features to
corresponding semantic ones without exploiting a semantic
encoder. This, however, provides a trivial solution that all
visual features are predicted as a background class. Second,
we adopt a two-stage discriminative approach, that is, train-
ing visual and semantic encoders sequentially. We first train
a segmentation network that consists of a feature extractor
and a classifier with seen classes. The learned feature ex-
tractor is then fixed and it is used as a visual encoder to
train a semantic encoder. This, however, provides a trivial
solution that all visual features are predicted as a background
class. Second, we adopt a two-stage discriminative approach, that is, training visual and semantic encoders sequentially. We first train a segmentation network that consists of a feature extractor and a classifier with seen classes. The learned feature extractor is then fixed and it is used as a visual encoder to train a semantic encoder. We can see from the first two rows that this simple variant with BAR and SC terms already outperforms ZS3Net, demonstrating the effectiveness of the discriminative approach. These variants are, however, outperformed by our approach that gives the best hIoU score of 44.6 (Table 1). To further verify our claim, we train the generator of ZS3Net using visual features extracted from our visual encoder (‘ZS3Net\textsuperscript{f}’). For compar-
ison, we also report the results obtained by our implementa-
tion of ZS3Net (‘ZS3Net\textsuperscript{f}’). From the last two rows, we
can clearly see that ‘ZS3Net\textsuperscript{f}’ outperforms ‘ZS3Net\textsuperscript{f}’. This
confirms that our approach alleviates the seen bias problem,
enhancing the generalization ability of visual features.

5. Conclusion

We have introduced a discriminative approach, dubbed
JoEm, that overcomes the limitations of generative ones in
a unified framework. We have proposed two complemen-
tary losses to better learn representations in a joint embed-
ding space. We have also presented a novel inference tech-
nique using the circle of Apollonius that alleviates a seen
bias problem significantly. Finally, we have shown that our
approach achieves a new state of the art on standard GZS3
benchmarks.

Acknowledgments. This work was supported in part
by the National Research Foundation of Korea (NRF)
grant funded by the Korea government (MSIP) (NRF-
2019R1A2C2084816) and the Yonsei University Research
Fund of 2021 (2021-22-0001).
References


[27] Christoph H Lampert, Hannes Nickisch, and Stefan Harmeling. Learning to detect unseen object classes by between-class attribute transfer. In CVPR, 2009. 2


[38] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed representations of words and
phrases and their compositionality. In NeurIPS, 2013. 2, 3, 4, 7


[40] Roozbeh Mottaghi, Xianjie Chen, Xiaobai Liu, Nam-Gyu Cho, Seong-Whan Lee, Sanja Fidler, Raquel Urtasun, and Alan Yuille. The role of context for object detection and semantic segmentation in the wild. In CVPR, 2014. 2, 6, 7, 8


[44] Scott Reed, Zeynep Akata, Honglak Lee, and Bernt Schiele. Learning deep representations of fine-grained visual descriptions. In CVPR, 2016. 2


[57] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In CVPR, 2017. 1