SIGN: Spatial-information Incorporated Generative Network for Generalized Zero-shot Semantic Segmentation

Jiaxin Cheng, Soumyaroop Nandi, Prem Natarajan, Wael Abd-Almageed
USC Information Sciences Institute, Marina del Rey, CA, USA
{chengjia,soumyrn,pnataraj,wamageed}@isi.edu

Abstract

Unlike conventional zero-shot classification, zero-shot semantic segmentation predicts a class label at the pixel level instead of the image level. When solving zero-shot semantic segmentation problems, the need for pixel-level prediction with surrounding context motivates us to incorporate spatial information using positional encoding. We improve standard positional encoding by introducing the concept of Relative Positional Encoding, which integrates spatial information at the feature level and can handle arbitrary image sizes. Furthermore, while self-training is widely used in zero-shot semantic segmentation to generate pseudo-labels, we propose a new knowledge-distillation-inspired self-training strategy, namely Annealed Self-Training, which can automatically assign different importance to pseudo-labels to improve performance. We systematically study the proposed Relative Positional Encoding and Annealed Self-Training in a comprehensive experimental evaluation, and our empirical results confirm the effectiveness of our method on three benchmark datasets.

1. Introduction

Zero-shot learning (ZSL) solves the task of learning in the absence of training data of that task (e.g., recognizing unseen classes). It has been widely adopted in classic computer vision problems, such as classification [1, 15, 2, 46, 48, 52, 58, 41, 54, 7, 38, 14, 34, 39, 19, 25, 51] and object detection [42, 3, 16, 11, 43, 4]. The key challenge in ZSL based tasks is to make the underlying model capable of recognizing classes that had not been seen during training. Earlier work focused on learning a joint embedding between seen and unseen classes [1, 15, 2, 46, 48, 52, 58, 41]. In recent works, knowledge and generative-based methods have gained more prominence. Knowledge-based methods [39, 19, 25, 51] use structured knowledge learned in another domain (e.g., natural language [36], knowledge graph[32], etc.) as constraints [19, 25] to transfer the learned information to unseen categories. Generative methods use attributes [14, 7], natural language [44] or word embeddings [14, 9] as priors to generate synthetic features for unseen categories.

In this work, we investigate the zero-shot semantic segmentation problem, which is less studied than other zero-shot computer vision problems [53, 5, 26, 18]. Generative methods have been widely adopted in zero-shot semantic segmentation problem. Bucher et al. [5] proposed ZS3Net, in which they leverage Generative Moment Matching Networks [31] to generate synthetic features for unseen categories by using word2vec [35] embeddings and random noise as prior (Fig. 1(a)). The use of random noise prevents the model from collapsing, which occurs due to lack of feature variety [59, 17]. Li et al. [30] extended ZS3Net [5] by adding a structural relation loss which constrains the generated synthetic features to have a similar structure as the semantic word embedding space. Gu et al. [18] suggested using a context-aware normal distributed prior when generating synthetic features instead of random noise (Fig. 1(b)).

We propose to incorporate spatial information to improve the performance of the zero-shot semantic segmentation problem. Our motivation arises from the assumption that knowing a pixel’s location may help semantic segmentation because it is a 2D prediction task. Incorporating spatial information in computer vision problems has recently attracted the attention of the community. In image classification, [12] slices the input image into nine patches and adds a positional vector for each patch to indicate the patches’ location. Other methods leverage [56] the relative position of objects through a space-aware knowledge graph for object detection. Zhang et al. [57] counted the co-occurrence of features to learn spatial invariant representation for semantic segmentation. However, to the best of our knowledge, spatial information has not been widely studied in previous zero-shot learning research. In this work, we propose to exploit spatial information by using Positional Encoding [50], as shown in Fig. 1(c). Positional Encoding generates a positional vector that indicates the position of a pixel in the image. Previous work [12] divided the input images into fixed number of patches and appended positional
embeddings on the images. However, in our case, dividing the input image into small patches is incompatible because the semantic segmentation problem requires the entire image as input. We mitigate this shortcoming by incorporating spatial information into the image features and propose Relative Positional Encoding to handle varying input image sizes.

In the zero-shot learning problem, unlabeled samples, including unseen classes, are sometimes available. Trained models can annotate unlabeled samples to obtain additional training data [60] and fine-tune the models with pseudo-annotated data. Such a training strategy is called self-training. In zero-shot semantic segmentation, self-training annotates pixel-level pseudo labels [5, 18]. To reduce the number of unreliable pseudo labels, [5] ranks the confidence score (i.e., the probability of classes after softmax function) of pseudo labels and uses only the most confident 75% pseudo labels. [18] eliminates pseudo labels if the confidence score is below a certain threshold.

This work proposes a knowledge distillation-inspired self-training strategy, namely Annealed Self-Training (AST), to generate better pseudo-annotations for self-training. Knowledge distillation is widely used in the form of teacher-student learning [22, 37], where the student network is trained on the soft labels from the teacher network in addition to (one-hot) hard labels since soft labels have more information due to high entropy [22]. In the zero-shot semantic segmentation problem, previous methods [5, 18] set a threshold to eliminate pseudo labels from the low confidence scores, while assigning the same loss weights to the remaining pseudo labels. This is similar to using hard labels in teacher-student learning. However, it is hard to ensure that the threshold can generalize for each sample, and again low confidence pseudo labels may also contain some useful information. To avoid the shortcomings of setting a threshold and assigning the same loss weights to pseudo labels, in AST, we use pseudo-annotations of all unlabeled pixels while re-weighing their importance according to their confidence score. We leverage the annealed softmax [22] function to normalize the pseudo labels’ weights and control their relative importance by adjusting the annealed temperature in the softmax function.

We make the following contributions in this paper. Firstly, we introduce the Spatial Information Module to incorporate spatial information in semantic segmentation using a novel Relative Positional Encoding (RPE) scheme. Compared to previous work [12], RPE does not need patch-sliced input and can handle varying image sizes. Secondly, we propose a knowledge distillation-inspired self-training strategy, namely Annealed Self-Training (AST). AST generates pseudo-annotations for unlabeled samples and adjusts their importance during self-training with a tunable annealed temperature. Finally, we evaluate the performance on three benchmark datasets and conduct extensive ablation experiments to demonstrate the effectiveness of our method.

2. Related Work

Zero-shot Learning Without loss of generality, approaches to zero-shot classification can be categorized into three families — joint embedding, generative and knowledge-based methods. Earlier works focused on linear embedding [1, 15, 2, 46], non-linear embedding [48, 52], and hybrid embedding [58, 41] methods. In the embedding-based methods, the basic idea is to learn encodings for images and attributes (e.g., description) and maximize a linear/non-linear score between matched pairs. Generative methods use attributes to create synthetic images with a generative model (e.g., generative adversarial network [17] or a conditional variational autoencoder [49]) and then train a classifier based on seen and synthetic unseen categories. Knowledge-based method often use structured knowledge as constraints [19, 25, 51] of relationships between classes and employ graph networks, e.g., Graph Convolutional Network [29], to generalize learned information from seen categories to unseen ones.
Meanwhile, in zero-shot semantic segmentation problems, two parts: seen categories and unseen categories.

3.1. Problem Formulation

In zero-shot learning problems, class labels consist of two parts: seen categories $C^S$ and unseen categories $C^U$. Meanwhile, in zero-shot semantic segmentation problems, the training set $D^S$ is composed of images and labels of seen categories to which image pixels belong. In other words, $D^S = \{(x, y) | \forall i, y_i \in C^S\}$, where $x$ is an image and $y$ is its corresponding ground-truth label, $y_i$ is the ground-truth label for pixel $i$. Other images that include pixels of unseen categories are denoted by $D^U = \{(x, y) | \exists i, y_i \in C^U\}$ and are only encountered at the inference time.

3.2. Spatial-information Incorporated Generative Network

Fig. 2 illustrates the proposed Spatial-information Incorporated Generative Network (SIGN). SIGN is composed of one unlearnable mapping network $M$ and five learnable networks — feature encoder ($E$), generator ($G$), classifier ($C$), discriminator ($D$), and Spatial Information Module ($SIM$). During training, an input image first goes through the feature encoder $E$. Then, $SIM$ conducts RPE on image features and produces $F_x$ for classifier $C$, and a stochastic vector $z$ for generator $G$. $G$ synthesizes features from semantic word embeddings and $z$ with a target of image features. To generalize the model on unseen categories, a random label including unseen classes is passed to $M$ and $G$, and the synthesized features is used to train $C$. At inference time, the test image only passes $E$.

The model is optimized in three stages — (1) Training Step ($SIM$) updates the main feature encoder $E$, the Spatial Information Module $SIM$, and the classifier $C$ in a standard...
semantic segmentation fashion. (2) Training Step (G) trains a generator $G$ that produces synthetic features with a target of real image features. (3) Transfer Learning Step uses synthesized features to fine-tune classifier $C$, which enables $C$ to recognize unseen categories.

The Training Step (SIM) objective is to fine-tune the upstream feature encoder $E$ and train $SIM$. $E$ is a semantic segmentation backbone network, which extracts image features. The choice of the backbone is architecture-agnostic, and any CNN-based network can be used (e.g., DeepLab [8], UNet [47], FCN [33]). The Spatial Information Module (SIM) takes image feature inputs $E(x)$, and (1) incorporates spatial information into image features through positional encoding and (2) produces a space-aware stochastic latent representation $z$, as shown in Eq. (1), where $\oplus$ denotes concatenation operation and $PE$ stands for positional encoding vector.

$$F_x, z = SIM(E(x) \oplus PE) \quad (1)$$

We use KL divergence to force $z$ to converge to a normal distribution [18] to ensure stochasticity, as shown in Eq. (2).

$$L_{KLD} = KL(z||\mathcal{N}(0, 1)) \quad (2)$$

We use the standard categorical cross-entropy loss to train classifier $C$, as shown in Eq. (3)

$$L_{pred}^{train}(p, y) = -\sum_{c} y_{c}\log(p_{c}) \quad (3)$$

where $p = C(F_x)$ is the categorical probabilities of features, $p_{c}$ is the probability of class $c$, and $y$ is the ground truth label. In Training Step (SIM), the classifier $C$ is trained only on real features (i.e., $c \in C^S$), and the total optimization target is the weighted sum of prediction loss and KL loss for $z$, where $\alpha$ is a hyperparameter to balance losses, as shown in Eq. (4).

$$E^*, SIM^*, C^* = \min_{E, SIM, C} L_{pred}^{train} + \alpha L_{KLD} \quad (4)$$

Training Step (G) attempts to train a generator $G$, with a fixed encoder $E$ and $SIM$. The generator is needed to synthesize image features of unseen categories so that the classifier $C$ can recognize unseen categories after being trained on synthetic features. $G$ generates synthetic features from a latent code. The latent code consists of two parts: (1) semantic word embedding $e$ and (2) normally distributed prior $z$. The stochasticity of $z$ prevents the generative model from collapsing, as discussed in [59, 17]. The mapping network $M$ maps ground truth annotations to semantic word embeddings, $e = M(y)$. Its weights are initialized with word2vec [35] and fasttext [24]. The generator $G$ produces a synthetic feature $\hat{F}_x = G(e \oplus z)$.

The synthesized features $\hat{F}_x$ have to be close to real features of seen categories. We follow previous work [5, 18] and use Maximum Mean Discrepancy (MMD) loss [31] to reduce the distribution distance between real and synthetic features. Total loss $L_{MMD}$ is the summation of MMD loss on seen classes $L_{MMD}(c)$, as shown in Eq. (5).

$$L_{MMD} = \sum_{c} L_{MMD}(c) \quad ; \quad c \in C^S \quad (5)$$

where,

$$L_{MMD}(c) = \sum_{f, f' \in F_{x,c}} k(f, f') + \sum_{\hat{f}, \hat{f}' \in \hat{F}_{x,c}} k(\hat{f}, \hat{f}') - 2 \sum_{f \in F_{x,c}} \sum_{\hat{f} \in \hat{F}_{x,c}} k(f, \hat{f}) \quad (6)$$

where $F_{x,c}$ and $\hat{F}_{x,c}$ are real and synthetic features for class $c$ in sample $x$’s feature, respectively. We choose Gaussian kernel function $k(f, f') = \exp(-\frac{1}{2}\|f - f'\|^2)$ as suggested in [5]. In order to make the synthesized image features realistic, we add a discriminator $D$ and training $G$ to deceive $D$ by optimizing an adversarial loss, as shown in Eq. (7) [17].

$$L_{adv} = \mathbb{E}_{f \in F_{x}}[\log(D(f))] + \mathbb{E}_{\hat{f} \in \hat{F}_{x}}[\log(1 - D(\hat{f}))] \quad (7)$$

The total loss for Training Step (G) is composed of MMD loss and adversarial loss, and hyperparameter $\beta$ controls the trade-off between two losses, as shown in Eq. (8).

$$G^*, D^* = \min_{G} \max_{D} L_{adv} + \beta L_{MMD} \quad (8)$$

Since the trainable networks and the losses do not overlap in Eqs. (4) and (8), we jointly optimize them for efficiency. Finally, to synthesize features for unseen categories, during the Transfer Learning Step, a pseudo-ground-truth $\hat{y}$ (i.e., a pseudo label including unseen categories) is fed into $M$ and $G$. The synthetic features are used to train the classifier $C$ so that $C$ can recognize the unseen categories. The prediction loss of the Transfer Learning Step is shown in Eq. (9).

$$L_{pred}^{trans}(p, \hat{y}) = -\sum_{c} \hat{y}_{c}\log(p_{c}) \quad (9)$$

The classifier $C$ is optimized on synthetic features as well as real features to avoid performance drop on seen categories, as shown in Eq. (10).

$$C^* = \min_{C} L_{pred}^{train} + L_{pred}^{trans} \quad (10)$$

### 3.3. Relative Positional Encoding

As illustrated in Fig. 2(d), the Spatial Information Module consists of the encoding module and the reparameterization module. The encoding module uses a residual structure [21] and incorporates spatial information into image features using positional encoding [50]. The reparameterization module [28] takes the output of the encoding module and generates a stochastic latent code. In Section 4.4, we discuss and compare different architectures for SIM.

RPE uses sine and cosine functions to incorporate pixel positions into a feature map [50]. To handle 2D positional encoding, we use a 600-dimensional vector, in which the first 300 dimensions are used for horizontal location encoding and the last 300 dimensions are used for vertical loca-
tion encoding. Eqs. (11) and (12) show horizontal positional encoding and Eqs. (13) and (14) show vertical positional encoding. \(i^{pos_x}\) and \(i^{pos_y}\) represent the relative horizontal and vertical positions of pixel \(i\), respectively, and \(d\) denotes the dimension. The overall dimensionality of positional encoding in each direction is \(d_{model} = 300\).

\[
PE(i^{pos_x}, 2d) = \sin(i^{pos_x}/10000^{2d/d_{model}}) \tag{11}
\]

\[
PE(i^{pos_x}, 2d + 1) = \cos(i^{pos_x}/10000^{2d/d_{model}}) \tag{12}
\]

\[
P\varepsilon(i^{pos_y}, 2d) = \sin(i^{pos_y}/10000^{2d/d_{model}}) \tag{13}
\]

\[
P\varepsilon(i^{pos_y}, 2d + 1) = \cos(i^{pos_y}/10000^{2d/d_{model}}) \tag{14}
\]

In order to handle arbitrary image sizes, we do not use the absolute position \(pos\) of a given pixel in the feature map. Rather, we use the relative position \(pos^*\), as shown in Eq. (15) and Eq. (16).

\[
i^{pos_x} = c \cdot i^{pos_x}/W \tag{15}
\]

\[
i^{pos_y} = c \cdot i^{pos_y}/H \tag{16}
\]

where \(c\) is a constant which we set to 512, and \(H\) and \(W\) are the height and width of the image feature, respectively. Please note that despite using the same name, our RPE is different from the ones [10, 23] used in natural language problems, which use pair-wise token relation for positional encoding and is fundamentally different from positional encoding for image features.

3.4. Annealed Self-Training

In prior literature [60], self-training was used to leverage the model’s prediction on unlabeled samples to obtain additional pseudo-annotations for fine-tuning the model. In zero-shot segmentation, the model produces class labels and confidence values (i.e., output of softmax layer) upon encountering pixels of unseen classes. The produced class labels are used as pseudo labels for self-training to learn unseen classes, based on the output confidence values. We anticipate that the generated pseudo-labeled pixels may be incorrect, and therefore, noisy labels may degrade the performance of the segmentation model. Previous methods [5, 18] threshold the prediction confidence and use only high confidence pseudo labels (e.g., highest 75% in [5]) during training to reduce the influence of incorrect pseudo-annotations.

However, finding a suitable threshold is not trivial since the model’s confidence in each sample is different. Inspired by knowledge distillation in transfer learning [22], we propose Annealed Self-Training (AST), which uses all pseudo-annotations but assigns different loss weights according to the confidence score, as shown in Eq. (17).

\[
w_i = \frac{1}{Z} \exp(p_i/T) \tag{17}
\]

where \(Z\) is a normalization term so that the maximum value of loss weights (\(\max(w_i)\)) is 1. The loss re-weighting is achieved by applying annealing softmax function on confidence score \(p\), and the annealed temperature \(T\) is used to adjust re-weighting intensity. Note that we only do loss re-weighting on pseudo-annotations and the loss weights of seen classes are always 1.

4. Experimental Evaluation

4.1. Benchmark Datasets

Following [53, 18], we used (1) Pascal Visual Object Classes (VOC) [13], (2) Pascal Context [40] and (3) COCO Stuff [6] for evaluation. Pascal VOC contains 20 categories with 1,464 and 1,449 images, for training and testing, respectively. Since Pascal VOC is relatively small, external Semantic Boundaries Dataset (SBD) dataset [20] is also used during training as suggested in previous works [53, 5, 18]. After introducing SBD and excluding duplicate images in Pascal VOC test set, there are 8,284 and 2,299 images for training and validation, respectively. Pascal Context contains 33 categories, including 4998, 500 and 5105 images, for training, validation and testing. COCO Stuff is a large semantic segmentation dataset with 171 categories. There are 118,287 images for training and 5,000 for testing. We split the last 10,000 images in the training set for validation.

We follow the evaluation protocol from [53, 18] for splitting seen and unseen categories. For Pascal VOC, the last five classes (potted plant, sheep, sofa, train, tv-monitor) are used as unseen categories [53, 18]. Class “background” is ignored in Pascal VOC during both training and testing, as suggested by [18], since it is unreasonable to use single semantic word representation for all kinds of background objects (e.g., sky, road). Four categories (cow, motorbike, sofa, cat) are classified as unseen classes in Pascal Context [18]. For COCO Stuff, 15 classes (frisbee, skateboard, card-board, carrot, scissors, suitcase, giraffe, cow, road, wall-concrete, tree, grass, river, clouds, playingfield) are treated as unseen [53, 18].

4.2. Experiment Setup And Evaluation Metrics

Implementation Details: We use word embeddings from both word2vec [35] and fasttext [24], and concatenate them to represent words (a total of 600 dimensional vector; 300 for each). We follow [18] to use average word embeddings when a category has multiple words. We use Deeplab-v2 [8] built upon ResNet-101 [21] as the semantic segmentation backbone. We apply SGD [45] optimizer for Deeplab-v2 backbone, SIM and classifier with initial learning rate \(2.5 \times 10^{-4}\), and Adam [27] optimizer for the generator with initial learning rate \(2 \times 10^{-4}\). A poly learning rate scheduler was applied for backbone as suggested by [53]. We empirically set loss weights to \(\alpha = 100\) and \(\beta = 50\).

Evaluation Metrics: We report performance based on mean intersection-over-union (mIoU) and conduct evalu-
The effectiveness of our method on recognizing unseen categories. In addition, our Annealed Self-Training further improves performance over conventional self-training. Compared to seen categories, AST works better for unseen ones due to higher utilization of pseudo-annotations. We notice that the performance impact of AST on Pascal VOC is higher than Context and COCO Stuff. Performance difference can be attributed to the smaller number of categories in Pascal VOC, which leads to a higher chance of correct pseudo labels.

Fig. 3 shows a qualitative comparison between SIGN and baselines. We can see that SIGN achieves high accuracy, even when there are multiple unseen categories (please see last row).

### 4.4. Ablation Studies

We conduct ablation studies on Pascal VOC dataset to show the effectiveness of Relative Positional Encoding and Annealed Self-Training.

**SIM Architecture:** We designed SIM as a residual module [21] and experimented with four architectures. (A) Convolution-based SIM uses simple residual blocks [21] with consecutive convolution and activation layers. (B) Attention-based [18] SIM learns three attention maps of different scales from deep features. Deep features are then element-wise multiplied with attention maps, and concatenated together. (C) Self-attention-based SIM uses the structure of Transformer Encoder [50]. The difference between self-attention- and attention-based SIM is: (1) attention map is computed according to the correlation of pixels on deep features instead of pixel-wise attention, and (2) feature aggregation computes the weighted sum of previous deep features, rather than concatenating features of different scales. (D) Multihead self-attention-based [50] runs several self-attention in parallel. The input feature is first linearly transformed into smaller dimension in each head, and self-attention is applied separately. Then, the attention results are concatenated together and linearly transformed back to the original dimension.

Table 2 summarizes the number of parameters of the four SIM architectures and their performance on Pascal VOC. The best performance is achieved by multihead self-attention-based SIM, followed by attention-based SIM. According to the performance in Table 2, performance numbers in all following experiments are reported based on multihead self-attention-based SIM. Please refer to Appendix A for the detailed model structure.

**Relative versus Absolute Positional Encoding:** To evaluate the effectiveness of the proposed Relative Positional Encoding, we compare it with two other positional encoding strategies: (1) Absolute Positional Encoding (APE), which use the absolute index of pixel to compute positional vec-
Table 2. Number of parameters in four different SIM architecture and the corresponding performance on Pascal VOC.

<table>
<thead>
<tr>
<th>Archi.</th>
<th>Conv</th>
<th>Attention</th>
<th>Self-Attn.</th>
<th>Multi SA</th>
</tr>
</thead>
<tbody>
<tr>
<td># Params</td>
<td>5.25M</td>
<td>5.24M</td>
<td>4.60M</td>
<td>4.86M</td>
</tr>
<tr>
<td>H. mIoU(%)</td>
<td>39.13</td>
<td>40.86</td>
<td>40.51</td>
<td><strong>41.74</strong></td>
</tr>
</tbody>
</table>

Figure 3. Qualitative comparison with SPNet [53], ZS3 [5] and CaGNet [18]. The top three samples are from Pascal VOC and the bottom three samples are from COCO Stuff. Color bar below the samples indicate the correspondence between colors and categories (including false positive categories). The square(s) on the right indicates the unseen class(es) in the sample on the left.

The results are presented in Table 3. We see that on unseen categories mIoU and harmonic mIoU, RPE improves performance by 3% compared to APE. Adding bilinear interpolation to APE improves performance by roughly 2% but still cannot match the performance of RPE. Interestingly, we notice a performance degradation of APE compared to the model without PE. We speculate that this is due to the mismatch between training and test image size, and due to the larger test image size, APE fails to encode all spatial information. Please note that larger image sizes or even multi-scale input sizes are commonly used during testing, because it can provide better prediction performance [33, 8].

Effect Of Annealed Temperature: In annealed self-training, pseudo-annotations with higher confidence are
Table 3. Mean IoU of model without PE, Absolute PE, Absolute PE with interpolation and Relative PE on Pascal VOC. Numbers in parentheses show the improvement over model without PE.

<table>
<thead>
<tr>
<th>Methods</th>
<th>Seen(%)</th>
<th>Unseen(%)</th>
<th>Harmonic(%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>w/o PE</td>
<td>71.86</td>
<td>26.07</td>
<td>38.26</td>
</tr>
<tr>
<td>APE</td>
<td>70.44</td>
<td>25.68</td>
<td>37.64</td>
</tr>
<tr>
<td>APE w/ Inter.</td>
<td>71.17</td>
<td>27.36</td>
<td>39.53</td>
</tr>
<tr>
<td>RPE</td>
<td><strong>75.40</strong> (+3.54)</td>
<td><strong>28.86</strong> (+2.79)</td>
<td><strong>41.74</strong> (+3.21)</td>
</tr>
</tbody>
</table>

Table 4. mIoU(%) on seen categories before transfer learning on Pascal VOC.

<table>
<thead>
<tr>
<th></th>
<th>w/o SIM</th>
<th>Conv</th>
<th>Attention</th>
<th>Self-att.</th>
<th>Multi SA</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>76.59</td>
<td>76.83</td>
<td>76.97</td>
<td>77.75</td>
<td><strong>77.87</strong></td>
</tr>
</tbody>
</table>

5. Conclusion

We proposed a new zero-shot semantic segmentation framework that incorporates spatial information into prediction. Our method is flexible to handle varying image size using a novel Relative Positional Encoding scheme. We introduced a new self-training strategy - Annealed Self Training, which automatically adjusts the importance of pseudo-annotations from prediction confidence. We conducted an extensive experimental study and validated the effectiveness of the proposed RPE and AST, and also investigated network architectures for encoding spatial information. Finally, our SIGN model showed state-of-the-art performance for zero-shot semantic segmentation on benchmark datasets and has the potential to improve performance on conventional semantic segmentation problems.

Acknowledgement This material is based on research sponsored by Air Force Research Laboratory (AFRL) under agreement number FA8750-19-1-1000. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation therein. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of Air Force Laboratory, DARPA or the U.S. Government.
References


Danilo Jimenez Rezende, Shakir Mohamed, and Daan Wierstra. Stochastic backpropagation and approximate inference in deep generative models. 2014. 5


