IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received December 17, 2021, accepted January 15, 2022, date of publication February 16, 2022, date of current version February 24, 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3151856

Variational Multi-Prototype Encoder for Object Recognition Using Multiple Prototype Images

JUN SEOK KANG AND SANG CHUL AHN, (Member, IEEE)
Division of Nano and Information Technology, Korea Institute of Science and Technology (KIST) School, University of Science and Technology, Seoul 02792, South Korea
Center for Artificial Intelligence, Korea Institute of Science and Technology, Seoul 02792, South Korea

Corresponding author: Sang Chul Ahn (asc@kist.re.kr)

This work was supported by the Korea Institute of Science and Technology (KIST) Institutional Program under Project 2E31061 and Project 2E31591.

ABSTRACT In the recent research of Variational Prototyping-Encoder (VPE), the problem of classifying 2D flat objects of the unseen class has been addressed. VPE solves this problem by pre-learning the image translation task from real-world object images to their corresponding prototype images as a meta-task. VPE uses a single prototype for each object class. However, in general, a single prototype is not sufficient to represent a generic object class because the appearance can change significantly according to viewpoints and other factors. In this case, using VPE and a single prototype for each class in training can result in overfitting or performance degradation. One solution may be the use of multiple prototypes. However, this also requires costly sub-labeling for dividing the input class into smaller classes and assigning a prototype to each. Therefore, we propose a new learning method, the variational multi-prototype encoder (VaMPE), which can overcome the limitations of VPE and use multiple prototypes for each object class. The proposed method does not require additional sub-labeling other than simply adding multiple prototypes to each class. Through various experiments, we demonstrate that the proposed method outperforms VPE.

INDEX TERMS Deep learning, variational encoder, prototype learning, embedding space, image classification.

I. INTRODUCTION

An object has various appearances depending on the viewing direction, lighting, color variation, small deformations, etc. To conveniently represent an object, we commonly use a representative image called the prototype of the object. Prototype images are more refined (under uniform lighting and free from clutter or noise) than typical images, and show the features of the objects well. Therefore, a prototype is suitable as a reference to classify or distinguish objects (ex. support sets in few-shot learning [1], [2] and a neighbor in the k-nearest neighbor [3]).

In general, the prototype looks similar to the object, but sometimes it may have different characteristics. For example, the image of a traffic sign on the street and a graphic image of the traffic sign in a book may appear similar, but the domain of each image is completely different. One is a real-world image, and the other is a synthetic image. This difference is called the domain discrepancy [4], which makes it difficult to directly use the prototype as a reference for object classification. In a previous work, Variational Prototyping-Encoder(VPE) [5], the authors attempted to train a deep learning model that could effectively reduce the domain discrepancy by using prototypes as references for classification. To this end, they made the model learn the meta-task of converting the input images into their corresponding prototype images. Through this meta-task, it was possible to reduce the domain discrepancy between the input image and the prototype in the embedding space of the model. Consequently, the prototype can be used for object classification through the embedding space of the trained model.

However, VPE only deals with cases in which an object class is well represented by a single prototype. The actual application targets were limited to 2D flat objects (traffic signs and logos). This is a limitation of VPE, and VPE is not appropriate if an object requires multiple prototypes for representation. However, a single prototype is usually not sufficient to represent an ordinary object. Many objects drastically change their appearance depending on the situation. For example, in the case of a book, the front and

The associate editor coordinating the review of this manuscript and approving it for publication was Wai-Keung Fung.

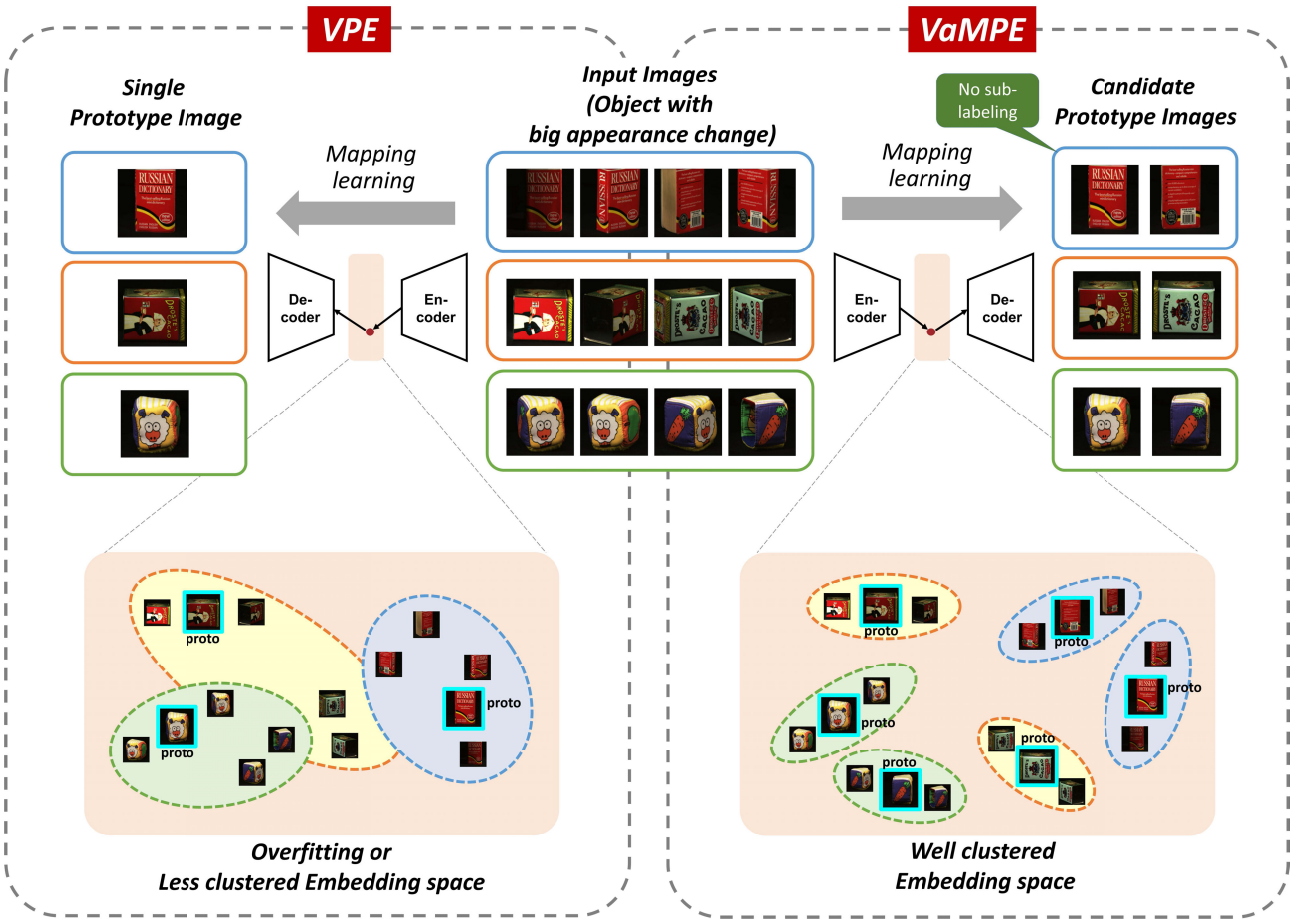

FIGURE 1. Learning the embedding space using a single or multi-prototype for image recognition. In the left part, we present the architecture of the baseline algorithm (VPE). The encoder and decoder networks learn to translate the input into a single prototype for each class. The feature representation in the embedding space can be used for classification after training. In many cases, there are significant changes in appearance even within a class, which makes it difficult for a single prototype to represent the class. Three example classes of inputs are shown at the center of the figure. The different colored boxes represent the different classes. Using single prototypes for such classes can lead to overfitting or a less clustered embedding space, as depicted in the VPE embedding space. Our new method, VaMPE, tackles this problem by using multiple prototypes. The right part of the figure illustrates VaMPE. VaMPE obtains a better-clustered embedding space than VPE by simply adding more prototypes for each class, without additional labeling.

back covers are significantly different. It is inevitable to use multiple prototypes to represent this object. Some examples are shown in the center of Fig.1. In this case, if we want to use VPE, we have to divide the images of the object into sub-classes by matching each image with a suitable prototype among the multiple prototypes. We refer to this task as a sub-labeling task. Suppose that learning is performed using an arbitrary prototype, without such a process. In this case, there may be the problem of overfitting the training data and learning a less clustered embedding space by learning incorrect matching. This is depicted on the left-hand side of Fig.1. The sub-labeling task has two issues. First, it is costly. It is usually challenging to find a suitable prototype for each image. Second, determining which prototype best fits the images of an object can sometimes be very ambiguous. For example, when using an object's front and back images as prototypes, it is difficult to choose which prototype is more suitable for the side image.

Therefore, we propose a new method called the variational multi-prototype encoder (VaMPE) that can overcome the limitations of VPE and use multiple prototypes for each object. VaMPE introduces a newly proposed mechanism called epsilon-greedy loss selection, which automatically selects one prototype from several prototypes for each input image. This mechanism was inspired by the epsilon-greedy search method for reinforcement learning [6]. VaMPE then trains a deep learning model through a meta-task that translates the input image into the selected prototype. VaMPE can reduce overfitting by utilizing multiple prototypes. As a result, VaMPE can learn a more generalized embedding space than VPE. This concept is depicted on the right side of Fig.1. In addition, the proposed method is free from sub-labeling tasks as it automatically selects a suitable prototype for each input image during the training process. The VaMPE pipeline is shown in Fig.2.

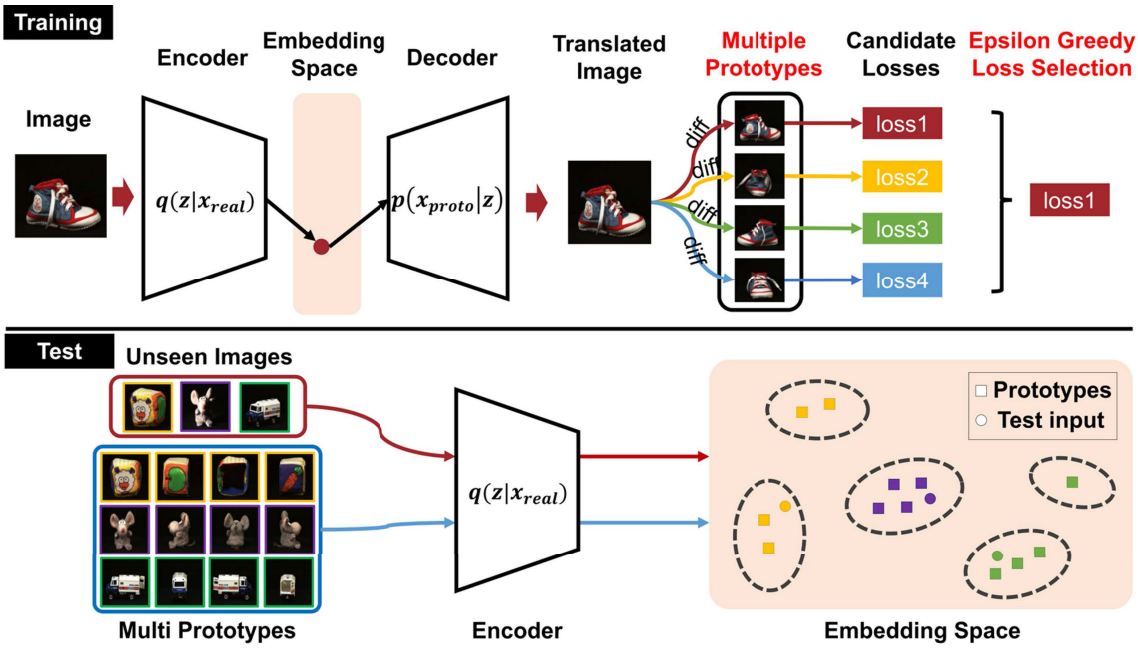

FIGURE 2. The entire VaMPE pipeline. VaMPE aims to translate a real image into one of multiple prototypes for learning a generalized embedding space in the training phase. The encoder embeds the input into a single latent vector. The decoder converts this into a single output image. The losses are computed between the output image and multiple prototype images. We select one of the losses according to the epsilon-greedy loss selection mechanism. We explain the details of the epsilon-greedy loss selection in Section III.B. The trained encoder and embedding space are used in the test phase. We use unseen object images as inputs that we want to recognize. Once the unseen object images are given, the frozen encoder converts them to latent vectors one by one. We apply the same operation to multiple prototype images. Then we compute the distances between the latent vector of each unseen object image and those of the prototype images. The label of the nearest prototype from each input is used as the label of the image. The image samples in this figure are obtained from the ALOI dataset.

We demonstrate the advantages of VaMPE through various experiments. VaMPE showed higher accuracy than the previous methods in all experiments. Particularly for datasets in which it is difficult to represent the class with a single prototype image, the classification accuracy improved significantly(83.92% $\rightarrow$ 96.6%). We also compare the embedding spaces induced by VPE and VaMPE through t-stochastic neighbor embedding(t-SNE) visualization, and show the reason for the effectiveness of VaMPE. The code for the model used in the experiments will be released.

II. RELATED WORKS

For object recognition with a limited number of prototype samples, few-shot learning [1], [2] is generally used. It uses the knowledge learned from previous tasks for a new task, which is human-like learning [7]. Recently, few-shot learning has become a hot research topic, and various studies have been conducted [8]. In recent years, deep neural networks have shown strong performance, but implementing few-shot learning with a deep neural network(DNN) is not easy because of overfitting problems owing to limited data [9]. Recent deep learning research has aimed to create a more general embedding space to solve this problem.

In [10]–[13], the authors attempted to create a generalized embedding space for novel classes by using user-defined metrics. However, it has been shown that embedding space made using a discrete user-defined metric is difficult to embed enough features to represent novel classes when we use it to recognize prototypes [5]. Kim *et al.* [5] showed that a more general embedding space can be learned through training using a meta-task of translating a real-world image into a single prototype image. They also demonstrated that their method(VPE) can solve domain discrepancy and data imbalance issues. However, VPE considers only datasets in which a single prototype is suitable for representing a class. Learning an excessive converging transformation in VPE may incur overfitting training datasets if a single prototype is insufficient. VPE is the baseline of our research, and we want to generalize the method for application to multimodal datasets.

Zhang *et al.* [14] proposed a variational Bayesian framework where a class is represented by a class-specific distribution, rather than a single point latent vector. Edwards and Storkey [15] used context as an additional latent variable, making it possible to learn distributions with different means and variances for each class (not samples). Both the studies used a single modal distribution in the latent space for each class. Therefore, this is different from the approach of our study.

Previous attempts have been made to use prototypes for classification [11], [13], [16]. Jetley *et al.* [16] suggested a feature transform approach that makes the features of

real images similar to the predefined features of prototypes. In [11], the authors used deep quadruplet networks to map prototype and real images' embeddings into a common space. In [13], the authors suggested prototypical networks for few-shot learning, as an extension of Vinyals *et al.* [12]. However, they used the mean of the support examples as the prototype of a class. In contrast, our method is different because it exploits an image translation task and uses multiple prototypes without integrating them.

III. PROPOSED METHOD

A. VARIATIONAL PROTOTYPING-ENCODER (VPE)

Before introducing our approach, we briefly describe VPE (Variational Prototyping-Encoder) [5] that is the basis of this study. VPE is a deep neural network for one-shot classification of graphic symbols. Given a single prototype image for each symbol class, VPE classifies a query into its corresponding category without requiring a large, fully supervised dataset. The network structure of VPE is the same as that of variational autoencoder(VAE) [17], consisting of an encoder and a decoder. The difference is that the reconstruction target is the prototype image in VPE and the input image in VAE. The output of the encoder induces a latent feature embedding space. VPE learns to transform an input image into a target prototype image while regularizing the latent space distribution to an assumed prior distribution. Only the learned encoder is used as a feature extractor in the test phase. Given the prototypes of the novel classes, VPE extracts their features from the encoder and stores them in the support set. When a query is given, VPE extracts its features from the encoder and performs classification by nearest neighbor search using the support set.

VPE learns an embedding space using the meta-task of image translation from a real image to a corresponding prototype image. VPE has a few advantages compared with previous methods(VAE, metric learning) as follows [5]: 1) Due to the prototype image reconstruction loss, VPE indirectly learns the relative similarities between a real image and a corresponding prototype image according to the apparent similarity. It reduces the domain discrepancy between the two images and allows a better generalization of the embedding space. 2) The image translation to a prototype diminishes the influence of extraneous perturbations such as background clutter and geometric and photometric perturbations. This also makes the induced embedding space robust to perturbations.

B. VARIATIONAL MULTI-PROTOTYPE ENCODER (VaMPE)

In [5], VPE was applied to an open-set graphic symbol recognition problem. We found three problems when applying this method to a more general object recognition problem. First, when a class of real object images is hard to represent by using a single prototype(ex. large appearance difference between the front and back images of an object), image translation to a single prototype may cause the model to overfit the training data. This is similar to enforcing mapping from a multimodal data distribution to a single modal data distribution. This enforcement could hurt the generalization capability of the embedding space. Second, it would be better to use multiple prototypes instead of a single prototype for each class to handle multimodal data distribution. However, suppose we want to use multiple prototypes for a class with VPE. In this case, we need additional preprocessing to find and separate the images that correspond to each of the multiple prototypes (we call this process sub-labeling or fine-grained labeling). It is a costly process. Third, even when it seems possible to cover a class using a single prototype image, selecting a prototype image for the class may not be trivial if there are multiple prototype candidates or variants of a prototype. We could not be sure which is more suitable as a prototype image among the possible candidates to enable the model to learn a better embedding space.

To solve these problems, we propose a new learning method called the variational multi-prototype encoder (VaMPE). VaMPE uses the same network structure as VPE. However, it learns a more general embedding space than VPE by training the network model with multiple prototypes for each class. We can say that VPE learns a sort of convergent transformation into a prototype for the input samples of each class. Convergent transformation should absorb possible variations in each class. However, if the variation is too significant, it is difficult to apply VPE. Allowing for multiple prototypes, VaMPE can handle a more extensive range of datasets, including datasets with multimodal distributions. As mentioned above, simply increasing the number of prototypes incurs a fine-grained labeling cost. Therefore, we also designed and adopted a new automatic prototype selection mechanism called epsilon-greedy loss selection in VaMPE. This new mechanism was inspired by the epsilon-greedy search mechanism used in reinforcement learning [18]. The epsilon-greedy search mechanism performs global exploration by selecting the best choice most of the time but sometimes selecting a random choice with an epsilon probability. With the epsilon-greedy loss selection mechanism, VaMPE computes the reconstruction losses for multiple prototypes and usually picks the smallest loss to train the model, but sometimes randomly selects one of the losses with an epsilon probability.

To formulate a modified loss function, let us consider $x^{(i)}$ as the i^{th} image in the image set X, and denote $t_k^{(i)}$ as the k^{th} prototype of $x^{(i)}$'s class, where k ranges from 1 to K (assuming K prototypes in each class). We denote q, p, ϕ, θ as the encoder, decoder, and their related parameters, respectively. As in VPE, if a prototype is given, each log marginal likelihood of the individual prototype $\log p_\theta(t)$ can be lower bounded by Jensen's inequality:

$$\log p_\theta(t) \geq E_{q_\phi(z|x)}[\log p_\theta(t|z)] - D_{KL}[q_\phi(z|x)||p_\theta(z)], \quad (1)$$

where t is the prototype image of input x's class and latent code z is generated from a prior distribution $p_\theta(z)$. $D_{KL}[\cdot]$ is the Kullback-Leibler(KL) divergence, and the proposal

distribution $q_\phi(z|x)$ approximates the intractable true posterior. The log marginal likelihood can be maximized by maximizing the variational lower bound. Therefore, the loss function is defined as follows:

$$L(x,t;\phi,\theta) = \frac{1}{S}\sum_{s=1}^{S} -\log p_\theta(t^{(s)}|z^{(s)}) + D_{KL}[q_\phi(z|x)||p_\theta(z)], \quad (2)$$

where s denotes each sample and S is the total number of samples. When we introduce multiple prototypes and the epsilon-greedy loss selection mechanism, the loss function is changed to:

$$L(x,t;\phi,\theta) = \frac{1}{S}\sum_{s=1}^{S} L_{\text{recon}} + D_{KL}[q_\phi(z|x)||p_\theta(z)], \quad (3)$$

$$L_{\text{recon}} = \begin{cases} \min(-\log p_\theta(t_1^{(s)}|z^{(s)}), \ldots, -\log p_\theta(t_K^{(s)}|z^{(s)})), \\ \qquad \text{if } \varepsilon \geq \varepsilon_{th} \\ \text{rand}(-\log p_\theta(t_1^{(s)}|z^{(s)}), \ldots, -\log p_\theta(t_K^{(s)}|z^{(s)})), \\ \qquad \text{if } \varepsilon < \varepsilon_{th} \end{cases} \quad (4)$$

where L_{recon} represents the reconstruction loss, and epsilon ε is a randomly sampled number from the continuous uniform distribution U(0,1). The loss function is determined according to the ε selected in each iteration. We set ε_{th} as $\frac{1}{current\ epoch}$, decreasing as the number of iterations increases. In (4), the smallest reconstruction loss is generally chosen, which is equivalent to selecting the prototype that produces the smallest loss among the multiple prototypes. If the sampled ε is less than ε_{th}, one of the multiple prototypes is chosen randomly and its reconstruction loss is used regardless of its magnitude. In the early stage of learning, ε_{th} is large; therefore, random selection is primarily used. As learning progresses, ε_{th} gradually decreases, and the probability of choosing a greedy selection increases. This induces the model to explore several cases to find the optimal matching between the reconstructed image and the prototypes during the early stage of learning. After optimal matching is found, the model continues learning with the corresponding prototype.

Multi-Prototype in Training In [5], it is shown that learning appearance similarity in the image domain allows better generalization than metric-based approaches that adopt metric losses induced by labels. This means that an image translation task can help learn the rich features used for unseen classes. In the same context, training using multiple prototypes can help learn more meaningful features than just using a single prototype. It can teach various matching cases between input images and multiple prototypes. This makes the induced embedding space more general. We demonstrate this in an experiment in Section IV.D.

Role of Epsilon Selection If ε_{th} is always 1, the loss function is denoted as

$$L(x,t;\phi,\theta) = \frac{1}{S}\sum_{s=1}^{S} \text{rand}(-\log p_\theta(t_1^{(s)}|z^{(s)}), \ldots, -\log p_\theta(t_K^{(s)}|z^{(s)})) + D_{KL}[q_\phi(z|x)||p_\theta(z)]. \quad (5)$$

Assuming that the model parameter change is small between each iteration and that the random selection follows a uniform distribution, we can calculate the expectation of the reconstruction loss function(the first term of the loss) as

$$\begin{aligned} E(L_{\text{recon}}) &= \frac{1}{N}\sum_{n=1}^{N}(\frac{1}{S}\sum_{s=1}^{S}\text{rand} \\ &\qquad (-\log p_\theta(t_1^{(s)}|z^{(s)}), \ldots, -\log p_\theta(t_K^{(s)}|z^{(s)}))) \\ &= \frac{1}{S}\sum_{s=1}^{S}(\frac{1}{N}\sum_{n=1}^{N}\text{rand} \\ &\qquad (-\log p_\theta(t_1^{(s)}|z^{(s)}), \ldots, -\log p_\theta(t_K^{(s)}|z^{(s)}))) \\ &\approx \frac{1}{S}\sum_{s=1}^{S} E_{t_k}(-\log p_\theta(t_k^{(s)}|z^{(s)})). \end{aligned} \quad (6)$$

N is the number of iterations during training. For convenience of explanation, we assume that we use binary cross entropy(BCE) loss for the reconstruction loss function. Then, the equation is changed to

$$\begin{aligned} E(L_{\text{recon}}) &= \frac{1}{S}\sum_{s=1}^{S} -E_{t_k}(t_k^{(s)}\log y^{(s)} + (1-t_k^{(s)})\log(1-y^{(s)})) \\ &= \frac{1}{S}\sum_{s=1}^{S} -(E_{t_k}(t_k^{(s)})\log y^{(s)} + E_{t_k}(1-t_k^{(s)})\log(1-y^{(s)})) \\ &\approx \frac{1}{S}\sum_{s=1}^{S} -\log p_\theta(E_{t_k}(t_k^{(s)})|z^{(s)}), \end{aligned} \quad (7)$$

where $\log p_\theta(t^{(s)}|z^{(s)}) \approx t^{(s)}\log y^{(s)} + (1-t^{(s)})\log(1-y^{(s)})$, and $y^{(s)}$ is the output of the decoder when the decoder input is $z^{(s)}$.

In other words, the expected reconstruction loss is similar to that when using the average prototype as a single prototype. This is similar to moving all prototypes towards the average prototype. Therefore, in the early stage of learning, the epsilon-greedy selection mechanism has the effect of clustering samples in each class in the embedding space. As ε_{th} decreases, greedy selection cases occur more frequently. This makes each sample closer to its corresponding prototype. In summary, we can think that the epsilon-greedy selection mechanism brings all the samples and prototypes in each class closer in the early stage of learning (global learning)

and gradually leads input samples to create subgroups around corresponding prototypes (fine-tuning).

In this study, we used BCE loss as the reconstruction loss function. Depending on the problem, the mean square error(MSE) or other loss(from basic to advanced loss [19]–[21]) can also be used. The stochastic gradient descent(SGD) was used as the optimization algorithm. We used the reparameterization trick [17] to make the model trainable.

The proposed method uses multiple prototypes for each class. However, the proposed method can also be applied to datasets with a single prototype for each class through prototype augmentation, which transforms a single prototype into several ones. This gives the model a chance to choose a more suitable prototype among augmented prototypes instead of just a manually selected prototype.

Test Phase We use only the encoder part in the test phase. Once multiple prototypes are provided for each class as a support set, we obtain their latent vectors in the embedding space using the trained encoder. We store the latent vectors for use as anchors in the tests. When a real image is given, it is input into the encoder to obtain a latent vector. We compute the distances between the latent vectors of the image and the prototypes. The label of the prototype with the shortest distance is assigned to the image. Various measures can be used for this distance function. In this study, the Euclidean distance function is used.

C. NETWORK ARCHITECTURE

For a fair comparison, we used the same network structure as the VPE network [5]. The encoder consists of three convolution layers, and the stride and downsizing factor are set to two for all layers. The kernel size was set as $\{7 \times 7, 4 \times 4, 4 \times 4\}$ for each layer. Every layer contains a leaky rectified linear unit(ReLU) activation and batch normalization. We attached a spatial transformer network(STN) [22] to compensate for the image transform before the first and third layers. Each STN consisted of {maxpool(2×2) - conv2d(5×5) - conv2d(5×5) - fc}. After the final convolutional layer, a fully connected layer with an output size of 300 is attached. The decoder has the inverse structure of the encoder except for the fully connected layer. We used 2 as the up-convolution parameter, and kernel sizes were 3×3. All decoder layers include leaky ReLU activation and batch normalization. We did not use the STN layers in the decoder.

D. DATA AUGMENTATION FOR TRAINING

For a fair comparison, we used the same data augmentations as in [5] for the traffic sign and logo datasets. We applied random rotation and flipping to each image, and the same augmentation to the corresponding prototype. However, the augmentation to prototypes was different from that for generating augmented multi-prototypes in sections IV-B and IV-C. This increases the variety of input images and the generalization performance of the trained models. For the ALOI dataset, we applied augmentation only to the inputs, not to the prototypes. This is because the ALOI dataset contains more viewpoint changes. We wanted the model to be more robust to changes in the viewpoint. We trained the network to generate an unaugmented prototype regardless of the input viewpoint change.

IV. EXPERIMENTS

A. PREPARATION

Dataset In this study, we used six datasets for the experiments. Two were traffic sign datasets, the other three were corporate logo datasets, and the last was a general object dataset. We used the GTSRB [23] and TT100K [24] datasets for the traffic sign classification experiments. The BelgaLogos [25], [26], FlickrLogos-32 [27], and TopLogo-10 [28] datasets were used for the logo classification experiments. The ALOI dataset [29] was used as a general object dataset. The ALOI dataset contained multiview images of general 3D objects.

The size and number of classes for each dataset are presented in Table 1. We provided single prototypes for the traffic sign and corporate logo datasets, and multiple prototypes for the general object dataset. Examples of the provided prototypes are shown in Fig.2 and 3. Each dataset was divided into training and test parts. As in previous papers, we denote 'All' for evaluating the entire classes and 'Unseen' for evaluating the classes excluded from the training.

Implementation Details All previous methods (SiamNet, QuadNet, MatchNet, VPE) used IdsiaNet as the base model. They used an ADAM optimizer with a learning rate of 10^{-4}, $\beta = (0.9, 0.999)$, $\epsilon = 10^{-8}$, and a mini-batch size of 128 to train the networks. SiamNet adopts a Siamese network structure to minimize the distance in the embedding space between a real image and a prototype image. QuadNet uses a quadruplet network to train its model to recognize objects. QuadNet uses quadruplet loss, which uses two pairs of real images and prototype images picked from different classes. MatchNet used an attention kernel in its model to maximize the similarity of the embedding vector between the real and prototype images. For VPE, we used the same network architecture as VaMPE and a single prototype for training and testing. For VaMPE, we used the same settings (base model, optimizer, optimizer parameter, and number of mini-batches) as the previous for a fair comparison. The size of the input image was 64×64 pixels. During the training process, prototype and random images were sampled at a ratio of 1:200 and used as the input images of the model. We used $\frac{1}{epoch}$ as ε_{th} in the epsilon-greedy loss selection method. The epoch is the number of times the entire dataset is explored in training.

Setting for each experiment The detailed settings for each experiment are described in each subsection, but we summarize the experimental settings in Table 3 to prevent confusion and for easy understanding.

FIGURE 3. Examples of augmented prototypes. Original single prototypes (red box) and augmented prototypes with affine transformations and brightness changes(blue box).

TABLE 1. Datasets used in experiments. The largest, medium, and smallest classes have the highest number of instances, median number of instances, and smallest number of instances, respectively.

Dataset	GTSRB	TT100K	BelgaLogos	FlickrLogos-32	TopLogo-10	ALOI
Instances	51,839	11,988	9,585	3,404	848	108,000
Classes	43	36	37	32	11	1,000
# of instances in largest class	3,000	2,051	2,192	204	99	108
# of instances in medium class	780	144	147	99	80	108
# of instances in smallest class	270	240	2	73	19	108
Type of image	RGB	RGB	RGB	RGB	RGB	RGB
Resolution	25x25∼266x232	22x21∼438x351	6x8∼645x433	16x16∼881x875	37x16∼544x196	192x144

TABLE 2. Dataset settings for the experiments. 'Way' in the table indicates the number of prototypes provided. For the GTSRB experiment, we used a portion of the GTSRB dataset as the validation dataset. In the Belga → Flickr and Belga → Toplogos experiments, we used the Toplogos and Flickr datasets for validation, respectively.

Dataset	GTSRB	GTSRB → TT100K	Belga → Flickr	Belga → Toplogos	ALOI
Training class	22	43	37	37	500
Val class	22	-	11	32	-
Test class	21(43 way)	36(36 way)	32(32 way)	11(11 way)	500(500 way)
# of training images	32,189	51,839	9,585	9,585	54,000
# of validation images	7,020	-	848	3,404	-
# of test images	12,630	11,988	3,404	848	54,000

B. SINGLE PROTOTYPE VS. AUGMENTED MULTI-PROTOTYPE

For the case in which a single prototype was provided, we compared the accuracy of the existing methods with that of the proposed algorithm. We conducted experiments on logo and traffic sign classification.

In experiments using logo datasets, the BelgaLogos dataset was used for training, and the other two datasets were alternately used as the test and validation sets. We denote each experiment as Belga → Flickr and Belga → Toplogos, using the Flickr and Toplogos datasets as test datasets, respectively. In the Belga → Flickr and Belga → Toplogos experiments, the Toplogos and Flickr datasets were used for validation, respectively.

In the experiments using traffic sign datasets, we trained the model using the GTSRB dataset and then tested it using the TT100K dataset. In the GTSRB → TT100K experiment, we obtained test accuracy without validation. An experiment using only the GTSRB dataset was also conducted. In this experiment, 22 of the 43 classes in the GTSRB dataset were used as seen classes, and the remaining 21 were used as unseen classes. We created a training set to have both seen and unseen classes. After training with the seen classes, we validated the network with the unseen classes in the training set. We measured its accuracy with the test set. The dataset settings for the experiments are described in Table 2.

TABLE 3. The summary of settings for each experiment.

Experiments (Chapter IV)	Section B	Sec. C	Sec. D	Sec. E
Dataset	Traffic sign Logos	Traffic sign Logos	ALOI	ALOI
Training prototype(VPE)	Single	Single	Single	Single
Test prototype(VPE)	Single	Augmented Multi	Single	Multi
Training prototype(VaMPE)	Augmented Multi	-	Multi	-
Test prototype(VaMPE)	Augmented Multi	-	Multi	-

We must provide multiple prototypes for VaMPE; however, only a single prototype exists for each class in the datasets. Therefore, we created multiple prototypes for VaMPE by applying simple augmentation to the prototype. (Note: This is different from the augmentation applied during training. This augmentation is only applied to generate multiple prototypes from a single prototype.) We used brightness changes for the traffic sign datasets because the rotation or translation

TABLE 4. Classification accuracy (%) for traffic sign, logo, and object datasets. In the QuadNet experiments, augmentations were used for the logo dataset, but not for the traffic sign dataset. The value in parentheses for VPE-ALOI (green text) denotes the accuracy when trained with a single prototype and tested with multiple prototypes (see Section IV.D). The best accuracy is marked in blue. 'aug' denotes augmentation for input images, not for the prototypes in this table. We denote 'All' for evaluating the entire classes and 'Unseen' for evaluating the classes, excluding the classes used in training.

Dataset	GTSRB	GTSRB → TT100K		Belga → Flickr		Belga → TopLogo		ALOI
Split	Unseen	ALL	Unseen	ALL	Unseen	ALL	Unseen	Unseen
No. class	21	36	32	32	28	11	6	500
No. way	(22+21)-way	36-way		32-way		11-way		500-way
SiamNet +aug [10]	33.62	28.36	22.74	24.70	22.82	30.84	30.46	-
QuadNet +(aug) [11]	45.20	42.3	N/A	31.68	28.55	38.89	34.16	-
MatchNet +aug [12]	53.30	62.14	58.75	38.54	35.28	28.46	27.46	-
VAE [17] +aug	22.24	32.10	27.98	27.17	27.31	23.30	18.59	-
VPE [5] +aug+stn	83.79	73.98	71.80	56.60	53.53	58.65	57.75	83.92
(see sec.IV.C for # in red)	(83.41)	(69.53)	(66.46)	(55.83)	(52.49)	(59.53)	(60.15)	(94.61)
VaMPE_greedy +aug+stn	82.12	70.35	67.16	54.56	53.32	61.27	59.94	94.09
VaMPE +aug+stn	**87.05**	**74.47**	**71.81**	**60.28**	**58.26**	**62.79**	**67.37**	**96.65**

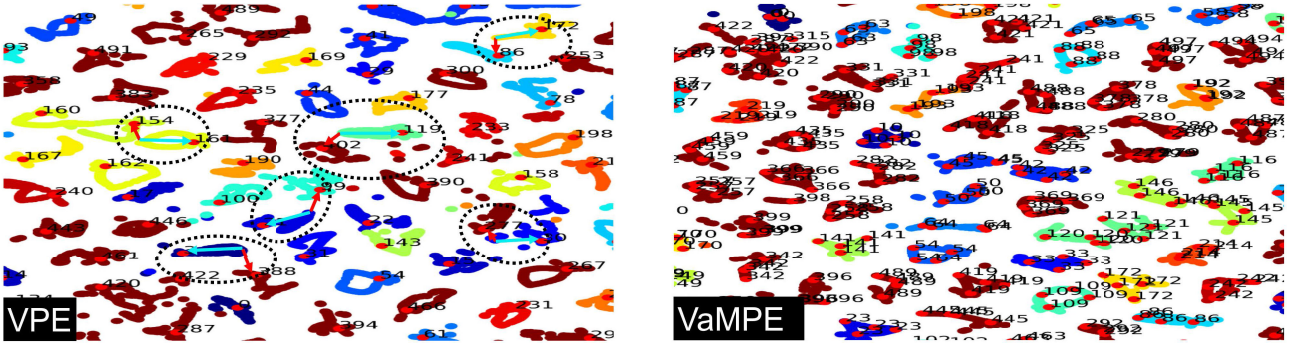

FIGURE 4. t-SNE visualization: single prototype vs. multi-prototype. This figure shows parts of the visualization of the object clusters in the embedding space learned using the ALOI dataset. The red dots denote each class's prototype, and the dots of different colors indicate the input images of different classes. In the left VPE plot, the sky-blue arrows and red arrows represent the distance from the input to the ground truth class prototype and the prototypes of a nearby class, respectively.

was insignificant, and a brightness-oriented difference was observed among the data samples. We created four prototypes by adding −120, −84, −48, and 0 to the image brightness. There were various variations in the logo datasets, so we made multiple prototypes by applying the brightness change in the Belga → TopLogo task and the affine transform in the Belga → Flickr task. For the brightness change, we used the same augmentation as before. We applied affine transforms to make three prototypes rotated by −36, 0, and 36 degrees along the yaw axis. Examples of augmented prototypes are shown in Fig.3.

The accuracy is shown in Table 4. The aug and stn used in Table 4 represent the augmentation on the training input and a spatial transformer network attached to the encoder, respectively. We took the results related to metric learning from the previous papers(SiamNet, QuadNet, MatchNet) without retest [5]. In the case of VPE, we attempted to train the model for all datasets. However, because the optimal model parameter values for each dataset were not disclosed, the results of our training were below those of the original study. Therefore, we used the accuracy reported in a previous paper for comparison. In all cases, the VaMPE method showed higher accuracy than previous methods. Interestingly, we can obtain better results by simply creating multiple augmented prototypes from a single prototype instead of adding new prototypes. This means that with VaMPE, we can easily achieve better results. To guess why VaMPE always produces better results than conventional VPE: 1) it may be possible for the model to obtain invariants with the help of multiple augmented prototypes. 2) Some augmented prototypes may have been more appropriate for representing classes than the original prototype. This means that VaMPE may be a solution to the problem in which the user-chosen prototype may not be suitable for representing a class.

C. VPE: SINGLE PROTOTYPE TRAINING AND AUGMENTED MULTI-PROTOTYPE TEST

In this section, we demonstrate that training with multiple prototypes is essential for performance improvement. To this end, we trained the VPE model using a single prototype and tested it by providing multiple augmented prototypes as a support set. We then compared the classification results of

VPE with those of VaMPE. We used the GTSRB, TT100K, Belga, and Flickr datasets in this experiment.

We trained the VPE model for the test because there were no publicly available pretrained weights for the model. Unfortunately, it did not achieve the original accuracy (using a single prototype) as denoted in the VPE paper [5]. Therefore, we trained it several times and took the best result as the best accuracy of VPE using a single prototype. The experimental results are listed in Table 4(red text). The test results of VPE using multiple augmented prototypes as a support set were still less accurate than those of VaMPE. In some cases, it is less than that when using a single prototype. This indicates that training with a single prototype can generate a more biased embedding space than training with multiple prototypes. It implies that multiple prototypes can improve performance in the test phase, but more importantly, they can also help create a more general embedding space in the training phase.

D. SINGLE PROTOTYPE VS. MULTI-PROTOTYPE

We compared the results when it was challenging to represent the samples of a class using a single prototype. The ALOI dataset [29] was used in this experiment. Unlike previous datasets, the ALOI dataset contains multiview images of general 3D objects, making it difficult to represent all samples in a class (object) with a single prototype. We trained the model using 500 classes(seen) and measured the test accuracy using the remaining 500 classes(unseen). We used images captured in four directions around an object as its multiple prototypes. For training and testing in single prototype classification, only the frontal view image was used as a prototype. All four prototypes were used for training and testing in multi-prototype classification. Since ordinary objects are highly likely to be taken from different views in an actual situation, we applied additional augmentation of random 2D rotation to the input images when measuring the test accuracy.

We show the accuracy of VPE trained with a single prototype and that of VaMPE trained with multiple prototypes in the last column of Table 4. Unlike in the previous experiments (Sections B and C), we can see a substantial performance improvement in VaMPE. The ALOI dataset contains not only simple objects that can be covered with single prototypes, but also ordinary 3D objects that are difficult to represent with single prototypes. This result indicates that VaMPE can be used to learn more general objects. Despite this high accuracy improvement, the additional cost is only the addition of new prototypes to the existing class without difficult fine-grained labeling.

To verify the usefulness of embedding space learning using multiple prototypes, we compared the embedding spaces of VPE and VaMPE using the t-SNE algorithm [30]. We calculated the latent vectors in the embedding space for all unseen input images. We then used the t-SNE algorithm to visualize the distribution in two-dimensional space. In the analysis, we ran the t-SNE algorithm for 2000 iterations to generate the plots. In the visualization plot in Fig.4, a red dot denotes each class's prototype, and dots of other colors indicate the input images. In the left image of Fig.4, the sky-blue arrows and the red arrows represent the distance from the input to the ground truth class prototype and to the prototypes of other classes, respectively. We can see that some of the input images in one class are closer to those of another class on VPE, which uses a single prototype. This means that a single prototype cannot adequately cover the full input images. For this reason, we adopt multiple prototypes on VaMPE, and show that multiple prototypes can solve the above problem. Multiple prototypes make it possible to cover classes that are difficult to cover with a single prototype. This can lead to higher accuracy in the proposed VaMPE.

E. VPE: SINGLE PROTOTYPE TRAINING AND MULTI-PROTOTYPE TEST

For the same reason as in section IV.C, we conducted a similar experiment. We used multiple prototype images instead of multiple 'augmented' ones. The ALOI dataset was used in this experiment. The experimental result is presented in Table 4(green text). The test result of VPE using multiple prototypes is more accurate than that using a single prototype, but still less accurate than that of VaMPE. This means that the embedding space created using multiple prototypes is more generalized than that created using a single prototype. This implies that multiple prototypes can improve performance in the test phase, and they can also help create a more general embedding space in the training phase.

We compared the embedding spaces of VPE and VaMPE to verify the usefulness of multi-prototype training. We selected the 30 classes showing the highest accuracy difference between VPE and VaMPE from the test results and visualized them in the embedding space using t-SNE. In the t-SNE plot in Fig.5, we can see that the samples are dispersed and overlapped between classes in VPE, while they are well clustered by class in VaMPE. This shows that VaMPE can produce clearer clusters than VPE can. This means that VaMPE's embedding space captures rich features, making it easier to distinguish unseen samples.

F. ABLATION TEST: EPSILON-GREEDY LOSS SELECTION

We conducted an ablation test using the proposed epsilon-greedy loss selection method. We performed an experiment using only greedy loss selection(VaMPE_greedy) instead of epsilon-greedy loss selection, where the prototype with the least reconstruction loss was selected for training.

The experimental results are listed in the lower part of Table 4. With only greedy loss selection, the accuracy is reduced and even worse than that of VPE using a single prototype. We compared the t-SNE visualizations of VaMPE and VaMPE trained with only greedy loss selection (VaMPE_greedy) to observe the effect of epsilon on the embedding space. Fig.6 depicts the results. In Fig.6, we can find that the clusters of 'supreme logo' class are separated in VaMPE_greedy but not in VaMPE. This indicates that epsilon-greedy loss selection can better cluster the embedding space.

FIGURE 5. t-SNE visualization: effect of a multi-prototype in the test. These are the t-SNE plots for classifying 30 classes (out of 500 classes), showing the highest accuracy differences between VPE (with a multi-prototype support set) and VaMPE. We can discover that some classes have separate clusters in VPE with a multi-prototype test, while those classes are clustered well in VaMPE.

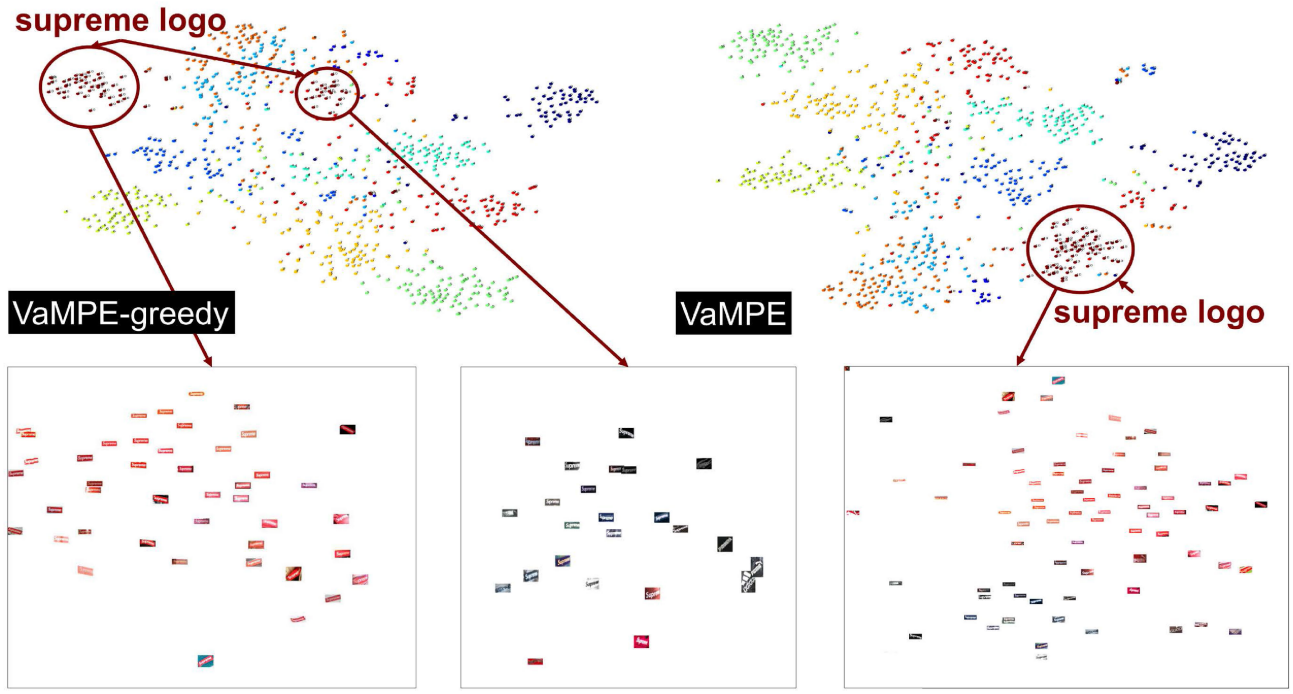

FIGURE 6. t-SNE visualization: effect of epsilon in training. This figure shows t-SNE plots for the TopLogos-10 dataset. We can discover separated clusters for a class (supreme logo) in the VaMPE_greedy plot (left) whereas the class is clustered well in the VaMPE plot (right). The bottom three plots are magnified views of the circles and show the samples as real supreme logos. We find that the samples are clustered by color in the VaMPE_greedy algorithm. However, in VaMPE, they are divided but still close because of their apparent similarity. This means that VaMPE learns a more general embedding space to represent unseen classes.

G. SUCCESSFUL AND FAILURE CASES OF VPE AND VaMPE

We show some examples of success and failure cases of VaMPE in Fig.7 and Fig.8, respectively. As shown in Fig.7, VaMPE works even when the background is not clean. We can find failure cases for both VPE and VaMPE. In particular, the logo experiment revealed some extreme failure cases. In the 'extreme failure case', the selected output prototype differed significantly from the input image. This means that the learned embedding space is not sufficiently generalized by training. We believe that it is due to the size of the training dataset. The logo datasets contain fewer training samples than the traffic sign and ALOI datasets do. It seems that the logo training datasets are not sufficient for learning a generalized embedding space, considering the various variations in logos. Therefore, we can confirm that a more diverse and sufficient dataset for training is recommended. As shown in Fig.8, in the ALOI dataset, where training data are sufficient, the failed classification results appear more similar to the inputs than in the logo case(3-6 rows). This means that the embedding space trained with sufficient data can learn more features, thereby reducing extreme failure cases. Moreover, the failure cases of VaMPE are more plausible than those of VPE. We presume

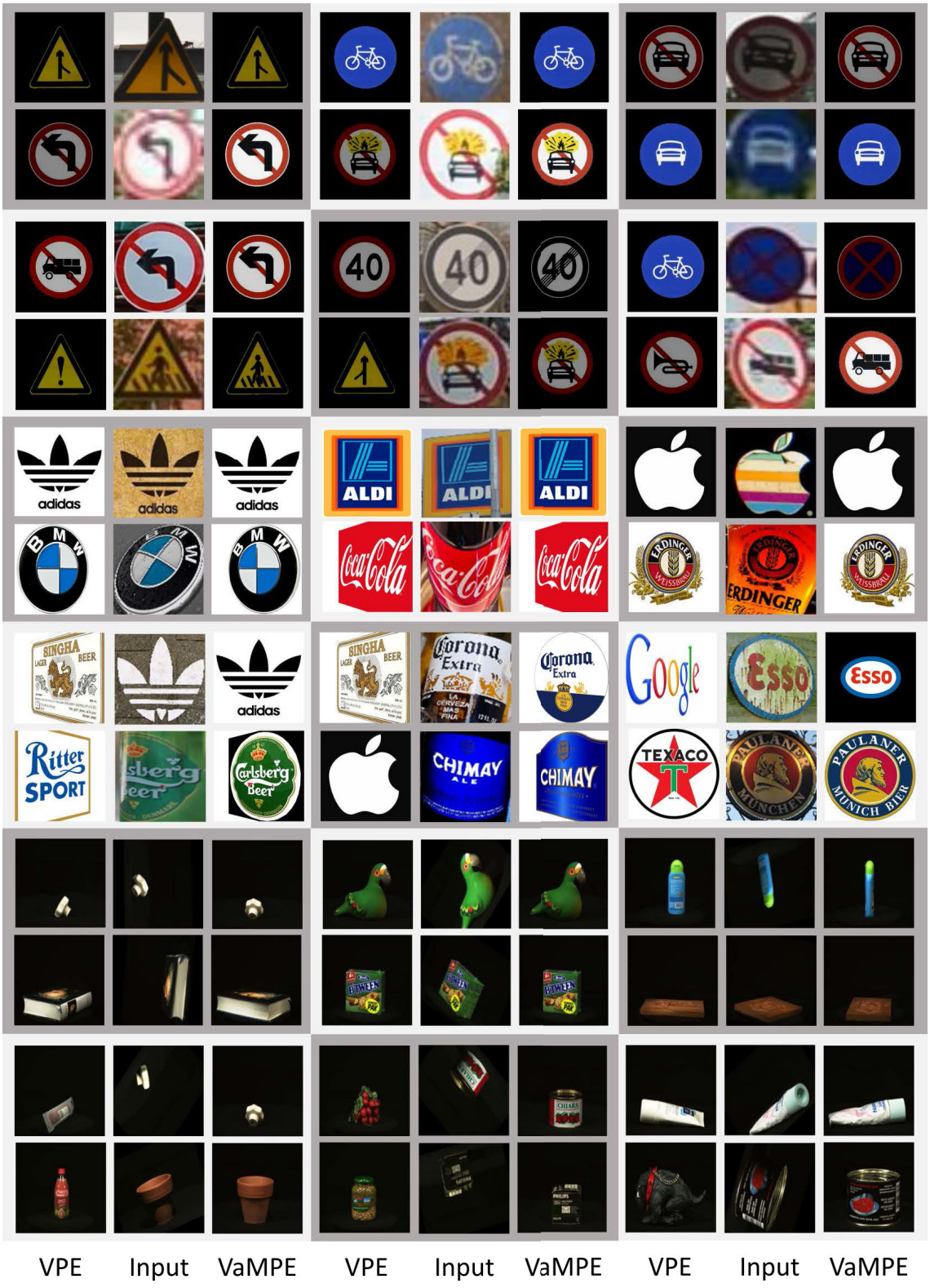

FIGURE 7. Successful cases of VaMPE on traffic sign, logo, and ALOI datasets. The second, fifth, and eighth columns show the inputs, and each left/right column represents the corresponding output of VPE/VaMPE. The 1, 2, 5, 6, 9th, and 10th rows show the case where both VPE and VaMPE are successful. The 3, 4, 7, 8, 11th, and 12th rows indicate when VPE fails, but VaMPE succeeds.

FIGURE 8. Failure cases of VaMPE on traffic sign, logo, and ALOI datasets. The second, fifth, and eighth columns show the inputs, and each left/right column represents the corresponding output of VPE/VaMPE. The first, third, and fifth rows show the case where VPE succeeds, but VaMPE fails. The second, fourth, and sixth rows indicate when VPE and VaMPE are unsuccessful.

that this is because VaMPE can reduce the overfitting problem of VPE.

V. CONCLUSION

We have presented VaMPE, a new method for recognizing objects using multiple prototypes. The network was trained using generative losses and a new loss selection mechanism. VaMPE is a generalization of the variational prototyping-encoder(VPE) [5], which tackles open-set graphic symbol recognition. VaMPE increases the ability to learn a general embedding space using multiple prototypes. This allows us to apply this method to more general datasets. VaMPE retains the advantages of VPE, its strength against domain inconsistencies and data imbalances.

Quantitatively, we have shown that the proposed method can achieve a higher accuracy than previous methods in various experiments. It indirectly proves that VaMPE can learn a more generalized embedding space than the previous methods. In the qualitative analysis, we can verify that the images in the same class are more clustered than in VPE, and that multi-prototype is helpful for classification. Despite these performance improvements, the additional cost is simply the addition of multiple prototypes, without further fine-grained labeling. Note that our research deals with the case in which initial prototypes are provided. The determination of prototypes is another issue that can affect further performance improvement. This could be a research topic for future studies.

Meanwhile, the proposed method introduces a new problem of multi-target training, where there are only target candidates without any specific target for each input during training. We solved this problem by using a new loss selection mechanism called epsilon-greedy loss selection. The epsilon-greedy loss selection mechanism is meaningful because it introduces randomness into the loss function when the best target is unknown. We believe that this mechanism can inspire us to design loss functions in uncertain environments. This can be a link to reinforcement learning in the future.

REFERENCES

[1] M. Fink, "Object classification from a single example utilizing class relevance metrics," in *Proc. Adv. Neural Inf. Process. Syst.*, 2005, pp. 449–456.
[2] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.
[3] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *Amer. Statist.*, vol. 46, no. 3, pp. 175–185, 1992, doi: 10.1080/00031305.1992.10475879.
[4] W. Mei and D. Weihong, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, Jul. 2018.

[5] J. Kim, T.-H. Oh, S. Lee, F. Pan, and I. S. Kweon, "Variational prototyping-encoder: One-shot learning with prototypical images," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 9454–9462.
[6] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," 2013, *arXiv:1312.5602*.
[7] B. M. Lake, T. D. Ullman, J. B. Tenenbaum, and S. J. Gershman, "Building machines that learn and think like people," *Behav. Brain Sci.*, vol. 40, no. e253, pp. 1–72, 2017.
[8] Y. Wang, Q. Yao, J. Kwok, and L. M. Ni, "Generalizing from a few examples: A survey on few-shot learning," 2019, *arXiv:1904.05046*.
[9] B. M. Lake, R. Salakhutdinov, and J. B. Tenebaum, "Human-level concept learning through probabilistic program induction," *Science*, vol. 350, no. 6266, pp. 1332–1338, 2015.
[10] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *Proc. ICML Deep Learn. Workshop*, vol. 2, 2015.
[11] J. Kim, S. Lee, T.-H. Oh, and I. S. Kweon, "Co-domain embedding using deep quadruplet networks for unseen traffic sign recognition," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 6975–6982.
[12] O. Vinyals, C. Blundell, T. Lillicrap, K. kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3630–3638.
[13] J. Snell, K. Swersky, and R. Zemel, "Prototypical networks for few-shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 4077–4087.
[14] J. Zhang, C. Zhao, B. Ni, M. Xu, and X. Yang, "Variational few-shot learning," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 1685–1694.
[15] H. Edwards and A. J. Storkey, "Towards a neural statistician," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.
[16] S. Jetley, B. Romera-Paredes, S. Jayasumana, and P. Torr, "Prototypical priors: From improving classification to zero-shot learning," in *Proc. Brit. Mach. Vis. Conf.*, 2015, p. 120.
[17] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2014.
[18] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, S. Peterson, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[19] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.
[20] A. B. L. Larsen, S. K. Sønderby, H. Larochelle, and O. Winther, "Autoencoding beyond pixels using a learned similarity metric," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1558–1566.
[21] X. Hou, L. Shen, K. Sun, and G. Qiu, "Deep feature consistent variational autoencoder," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 1133–1141.
[22] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu, "Spatial transformer networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 2017–2025.
[23] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel, "Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition," *Neural Netw.*, vol. 32, pp. 323–332, Aug. 2012.
[24] Z. Zhu, D. Liang, S. Zhang, X. Huang, B. Li, and S. Hu, "Traffic-sign detection and classification in the wild," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 2110–2118.
[25] A. Joly and O. Buisson, "Logo retrieval with a contrario visual query expansion," in *Proc. seventeen ACM Int. Conf. Multimedia (MM)*, 2009, pp. 581–584.
[26] P. Letessier, O. Buisson, and A. Joly, "Scalable mining of small visual objects," in *Proc. 20th ACM Int. Conf. Multimedia (MM)*, 2012, pp. 599–608.
[27] S. Romberg, L. G. Pueyo, R. Lienhart, and R. van Zwol, "Scalable logo recognition in real-world images," in *Proc. 1st ACM Int. Conf. Multimedia Retr. (ICMR)*, 2011, pp. 1–8.
[28] H. Su, X. Zhu, and S. Gong, "Deep learning logo detection with data expansion by synthesising context," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Mar. 2017, pp. 530–539.
[29] J.-M. Geusebroek, G. J. Burghouts, and A. W. M. Smeulders, "The Amsterdam library of object images," *Int. J. Comput. Vis.*, vol. 61, no. 1, pp. 103–112, 2005.
[30] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

JUN SEOK KANG received the B.S. degree in electronics and information engineering from Korea University, Sejong, South Korea, in 2015, and the M.S. degree in HCI and robotics from the Division of Nano and Information Technology, Korea Institute of Science and Technology (KIST) School, University of Science and Technology (UST), Seoul, South Korea, in 2017, where he is currently pursuing the Ph.D. degree.

His main research interests include computer vision and image-based machine learning and deep learning.

SANG CHUL AHN (Member, IEEE) received the B.S., M.S., and Ph.D. degrees in control and instrumentation engineering from Seoul National University, Seoul, South Korea, in 1988, 1990, and 1996, respectively.

He was a Visiting Scholar with the Computer Science Department, University of Southern California (USC), USA, in 1996. He has been with the Korea Institute of Science and Technology (KIST), since 1997. Currently, he is a Principal Researcher at the Center for Artificial Intelligence in AI-Robotics Institute, KIST. He is also the Chief Professor of the Division of Nano and Information Technology, KIST School, University of Science and Technology (UST). His main research interests include computer vision, deep learning, mixed reality, and robotics. He is a member of ACM.