PMAL: Open Set Recognition via Robust Prototype Mining

Jing Lu[1*], **Yunlu Xu**[1*], **Hao Li**[1], **Zhanzhan Cheng**[1,2†], **Yi Niu**[1]

[1] Hikvision Research Institution, Hangzhou, China
[2] Zhejiang University, Hangzhou, China
{lujing6, xuyunlu, lihao50, chengzhanzhan, niuyi}@hikvision.com

Abstract

Open Set Recognition (OSR) has been an emerging topic. Besides recognizing predefined classes, the system needs to reject the unknowns. Prototype learning is a potential manner to handle the problem, as its ability to improve intra-class compactness of representations is much needed in discrimination between the known and the unknowns. In this work, we propose a novel Prototype Mining And Learning (PMAL) framework. It has a prototype mining mechanism before the phase of optimizing embedding space, explicitly considering two crucial properties, namely *high-quality* and *diversity* of the prototype set. Concretely, a set of high-quality candidates are firstly extracted from training samples based on data uncertainty learning, avoiding the interference from unexpected noise. Considering the multifarious appearance of objects even in a single category, a diversity-based strategy for prototype set filtering is proposed. Accordingly, the embedding space can be better optimized to discriminate therein the predefined classes and between known and unknowns. Extensive experiments verify the two good characteristics (*i.e.*, *high-quality* and *diversity*) embraced in prototype mining, and show the remarkable performance of the proposed framework compared to state-of-the-arts.

1 Introduction

Classic image classification problem is commonly based on the assumption of *close set*, *i.e.*, categories appeared in testing set should all be covered by training set. However, in real-world applications, samples of unseen classes may appear in testing phase, which will inevitably be misclassified into the specific known classes. To break the limitations of *close set*, Open Set Recognition (OSR) (Scheirer et al. 2013) was proposed, which has two sub-goals: known class classification and unknown class detection.

Methods based on *Prototype Learning* (PL) obtained promising performance (Yang et al. 2018; Chen et al. 2020) recently. This group generates clearer boundaries between the known and unknowns through learning more compact intra-class feature representations using *prototypes* (on behalf of the discriminative features of each class). In detail, (Yang et al. 2018) learns the CNN feature extractor and prototypes jointly from the raw data and predicts the categories by finding the nearest prototypes instead of the traditional SoftMax layer. (Chen et al. 2020) advanced the framework (Yang et al. 2020) by adversely using the prototypes named reciprocal points to represent the outer embedding space of each known class, and then limiting the embedding space of unknown class. The existing methods all conduct prototype learning and embedding optimization jointly, regarding the prototypes as parameterized vectors, without direct constraint on the procedure of obtaining prototypes. Here we call them *implicitly* learned prototypes, and oppositely, if imposing direct guidance on the prototypes themselves, we denote the prototypes as *explicitly* learned ones. All the above-mentioned methods belong to the former category, *i.e.*, the *implicit* prototype-based methods. While they inevitable encounter some problems, especially in complicated situations. Two typical problems are shown in Figure 1: (1) **Undesired learned prototypes close to feature space of low-quality**[1] **samples**. As in Figure 1(a), implicitly learned prototypes are mistakenly guided by the low-

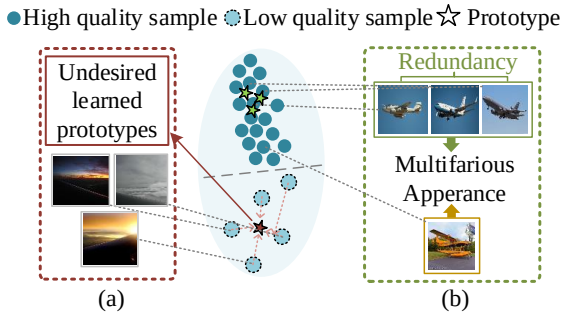

Figure 1: Two typical problems on implicitly learned prototypes. (a) Undesired learned prototypes arisen from low-quality samples. (b) Redundancy in similar prototypes and lacks of diversity. The in-between ellipse shows the feature distribution of a exemplary class 'airplane'.

[*] These authors contributed equally.
[†] Corresponding author.

[1] Low quality can be caused by various noise, *e.g.*, occlusion, blur or background interference.

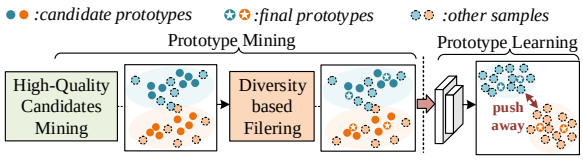

Figure 2: The proposed prototype mining and learning framework. Different colors denote different classes.

quality samples. As claimed in (Shi and Jain 2019), embedding of *high-quality* samples is discriminative while low-quality samples correspond to ambiguous features. Prototypes should represent the discriminative features of each class, so only *high-quality* samples are suitable. (2) **Redundancy in similar prototypes and lack of diversity.** Without explicit guidance, prototypes in one category show much redundant and cannot sufficiently represent the multifarious appearance. As in Figure 1(b), prototypes marked in green are adjacent in feature space and samples nearby show similar appearance, which implies the redundancy in learned prototypes. Besides, the airplanes in green and yellow rectangles show great distinctions, and their embeddings are located at separated positions. Obviously only using prototypes in green can not fully captures the multifarious appearance, which we require the *diversity* of prototype set.

Upon the above, we take *high-quality* prototypes and their *diversity* into consideration and propose to explicitly design the prototype[2] mining criteria, and then conduct PL with the chosen desirable prototypes. Note that different from the existing *implicit* prototypes, our proposed can be regarded as *explicit* ones. We name the novel framework as Prototype Mining And Learning (PMAL). The framework is illustrated in Figure 2, which can be divided into two phases, the prototype mining and embedding learning orderly. (1) **The prototype mining phase**. *High-quality* candidates are first extracted from training set according to the novelly proposed metric *embedding topology robustness*, which captures the data uncertainty contained in samples arisen from inherent low-quality factors. Then the prototype set filtering is designed to incorporate *diversity* for prototypes in each class. The step not only prevents the redundancy of similar prototypes, but also preserves the multifarious appearance of each category. (2) **The embedding optimization phase.** In this phase, given high-reliable prototypes, the embedding space is optimized via a well-designed point-to-set distance metric. The training burden is also reduced via mining prototypes in advance and feature optimization orderly, as the latter phase only work on embedding space.

Our main contributions are as follows. (1) Different from the common usage of implicitly learnable prototypes, we pay more attention on choosing prototypes with explicit criteria for OSR tasks. We point out the two important attributes of prototypes, namely the *high-quality* and *diversity*. (2) We design a OSR framework by prototype mining and learning. In the *prototype mining* phase, the above two key attributes are taken into consideration. In the *embedding learning* phase, with the chosen prototypes as fixed anchors for each class, a better embedding space is learned, without any sophisticated skills for convergence. (3) Extensive experiments on multiple OSR benchmarks show that our method is powerful to discriminate the known and unknowns, surpassing the state-of-the-art performance by a large margin, especially in complicated large-scale tasks.

[2]In our method, prototypes refer to samples, not features.

2 Related work

OSR is theoretically defined by Scheirer *et al.*(Scheirer et al. 2013), where they added an hyperplane to distinguish unknown samples from knowns in an SVM-based model. With rapid development of deep neural networks, Bendale *et al.*(Bendale and Boult 2016) incorporated deep neural networks into OSR by introducing the OpenMax function. Then both Ge (Ge, Demyanov, and Garnavi 2017) and Neal (Neal et al. 2018) tried to synthesize training samples of unseen classes via the popular Generative Adversarial Network.

Recently, reconstruction-based (Yoshihashi et al. 2019; Oza and Patel 2019; Sun et al. 2020) approaches are widely studied, among which Sun *et al.*(Sun et al. 2020) achieved promising results by learning conditional gaussian distributions for known classes then detecting unknowns. Zhang *et al.*(Zhang et al. 2020) added a flow density estimator on top of existing classifier to reject unseen samples. These methods all incorporate auxiliary models (*e.g.*, auto-encoders) for OSR, thus inevitably bring extra computational cost.

Since (Yang et al. 2020; Chen et al. 2020) attempted to combine prototype learning with deep neural networks for OSR, they achieved the new state-of-the art. Prototypes refer to representative samples or latent features for each class. It is inspired by *Prototype Formation* theory in psychology cognition field (Rosch 1973), and is later incorporated in some deep networks, *e.g.*, face recognition (Ma et al. 2013; Wang et al. 2016), few-shot learning (Snell, Swersky, and Zemel 2017). Yang *et al.*(Yang et al. 2020, 2018) introduced Convolutional Prototype Network (CPN), in which prototypes per class were jointly learned during training. Chen *et al.*(Chen et al. 2020) learned discriminative reciprocal points for OSR, which can be regarded as the inverse concept of prototypes. However, these methods suffer from unreliable prototypes caused by low-quality samples and lack of diversity, leading to the limited representativeness of prototypes.

3 Notation and Preliminaries

3.1 Notations

Let $X \to Z$ denote the mapping from input dataset $X=\{x_i\}_{i=1}^N$ into its embedding space $Z=\{z(x_i)\}_{i=1}^N$ by a trained deep classification model, where $Z \in \mathbb{R}^{N \times D}$, N is the number of samples and D is the embedding channel size. The feature region occupied by samples of the known class k in Z is referred as *embedding region* Z_k where $k \in \{1, ..., K\}$, K is the number of known classes.

Given the input $x_i \in$X, the extracted feature $z(x_i)$ (simply denoted as z_i) is fed into the ultimate linear layer, then SoftMax operation is conducted to obtain the probability $p(\cdot)$ of

x_i belonging to the k-th class, which is:

$$p(\hat{y}_i = k|z_i) = exp(z_i w_k + b_k)/\sum_{n=1}^{K} exp(z_i w_n + b_n), \quad (1)$$

where $\hat{y}_i$ is predicted class and W=$(w_1, ..., w_K) \in \mathbb{R}^{D\times K}$ and $b \in \mathbb{R}^K$ are the weight and bias term of the linear layer.

3.2 Preliminaries of Uncertainty

In deep uncertainty learning, *uncertainties* (Chang et al. 2020) can be categorised into *model uncertainty* and *data uncertainty*. *Model uncertainty* captures the noise of parameters in deep neural networks. What we mention in this work is *data uncertainty*, which captures the inherent noise in input data. It has been widely explored in deep learning to tackle various computer vision tasks, *e.g.*, face recognition (Shi and Jain 2019), semantic segmentation (Kendall, Badrinarayanan, and Cipolla 2016) *etc.*. Generally, inherent noise is attributed to two factors: the low quality of image and the label noise. In the scope of this work for assessing qualified samples in PL, we only regard the former. Following (Chang et al. 2020), when mapping an input sample x_i into Z, its inherent noise, *i.e.*, *data uncertainty*, contained in input will also be projected into embedding space, the embedded feature $z(x_i)$ can be formulated as:

$$z(x_i) = \phi(x_i) + n(x_i),\ n(x_i) \sim \mathcal{N}(0, \sigma(x_i)) \quad (2)$$

where $\phi(x_i)$ represents the discriminative class-relevant feature of x_i, which can be seen as the ideal embedding for representing its identity. ϕ denotes the embedding model. $n(x_i)$ is drawn from a Gaussian distribution with mean of zero and x_i-dependent variance $\sigma(x_i)$, $\sigma(x_i)$ represents the *data uncertainty* (*i.e.*, class-irrelevant noisy information caused by low quality) of x_i in Z. *The more noise contained in x_i, the larger uncertainty $\sigma(x_i)$ exists in embedding space.* We denote $z(x_i)$, $\phi(x_i)$, $\sigma(x_i)$ as z_i, ϕ_i, σ_i for simplicity hereinafter.

4 Prototype Mining

Prototype mining phase has two steps orderly, the *high-quality* candidate selection and *diversity*-based filtering.

4.1 High-Quality Candidate Selection

Since data uncertainty captures noise in samples caused by low-quality, we exploit it for selecting high-quality samples as candidate prototypes. To model data uncertainty, a simple yet efficient algorithm is proposed, which includes the following three steps: 1) embedding space initialization, 2) data uncertainty modeling and 3) candidate selection.

Embedding Space Initialization. Following Monte-Carlo simulation(Gal and Ghahramani 2016), we first acquire U SoftMax-based deep classifiers $\{M^u\}_{u=1}^U$ on the training set of known classes by repeating the training process U times. Then the input data is fed into the pre-trained classifiers, obtaining $\{Z^u\}_{u=1}^U$. Noticing that it is sufficient to formalize different embedding space by conducting repeated training processes with random parameter initialization and data shuffling, as proved in (Lakshminarayanan, Pritzel, and Blundell 2017). Here we set U to 2 for clearer illustration.

Data Uncertainty Modeling. Based on Sec. 3.2, the higher quality for a sample, the lower data uncertainty it has.

Property 1. *Given a high-quality sample x_i, its embedding z_i satisfies $z_i \approx \phi_i$.*

The high-quality sample x_i satisfy $\sigma_i \approx 0$, then combined with Equa. 2 we can easily obtain the above property. Suppose we select high-quality samples from training data to form the candidate prototype set C=$\{c_i\}_{i=1}^H \subseteq X$, where H is the total candidate number. Correspondingly, the set of their embedding in two different space Z^1 and Z^2 can be denoted as Φ^1=$\{z_i^1\}_{i=1}^H \approx \{\phi_i^1\}_{i=1}^H$ and Φ^2=$\{z_i^2\}_{i=1}^H \approx \{\phi_i^2\}_{i=1}^H$, where the superscript denotes the index of embedding space.

Property 2. *Given a sample pair (x_i, x_j), $\forall\ i, j \in \{1, ...H\}$, Mahalanobis distance in embedding space Z can be computed by $d_{\mathcal{M}}(z_i, z_j)$=$\sqrt{(z_i\text{-}z_j)\Sigma^{-1}(z_i\text{-}z_j)^{\mathrm{T}}}$ where Σ^{-1} is covariance matrix. If x_i, x_j are both of high quality, $d_{\mathcal{M}}(z_i, z_j)$ in different embedding space remains similar, i.e., $d_{\mathcal{M}}(z_i^1, z_j^1) \approx d_{\mathcal{M}}(z_i^2, z_j^2)$, $\forall x_i, x_j \in C$.*

Proofs. When only feeding the class-relevant feature ϕ_i^1 and ϕ_i^2 into the top linear layer of each classifier, the output probability for each category should remain consistent under the constraint of same class label y_i, *i.e.*, $p(\hat{y}_i^1$=$k|\phi_i^1) \approx p(\hat{y}_i^2$=$k|\phi_i^2)$, $\forall k \in \{1, ..., K\}$. Combining Equa. 1, we have the formulation:

$$\frac{exp(\phi_i^1 w_k^1 + b_k^1)}{\sum_{n=1}^K exp(\phi_i^1 w_n^1 + b_n^1)} \approx \frac{exp(\phi_i^2 w_k^1 + b_k^1)}{\sum_{n=1}^K exp(\phi_i^2 w_n^1 + b_n^2)}, \quad (3)$$

which can be deduced to

$$\phi_i^1(w_n^1\text{-}w_k^1)\text{+}b_n^1\text{-}b_k^1 \approx \phi_i^2(w_n^2\text{-}w_k^2)\text{+}b_n^2\text{-}b_k^2,\ n\text{=}1, ..., K. \quad (4)$$

Averaging up all the equations for $\forall\ k \in \{1, ..., K\}$ leads to

$$\phi_i^1(w_n^1\text{-}\overline{w}^1)\text{+}b_n^1\text{-}\bar{b}^1 \approx \phi_i^2(w_n^2\text{-}\overline{w}^2)\text{+}b_n^2\text{-}\bar{b}^2,\ n\text{=}1, ..., K, \quad (5)$$

where $\overline{w}$=$(\sum_{l=1}^K w_l)/K$ and $\bar{b}$=$(\sum_{l=1}^K b_l)/K$. Taking A=$(w_1 - \overline{w}, ..., w_K - \overline{w})$ and B=$(b_1 - \bar{b}, ..., b_K - \bar{b})$, Equa. 5 can be rewritten as $\phi_i^1 A^1\text{+}B^1 \approx \phi_i^2 A^2\text{+}B^2$. Given another $c_j \in C$ where $j \neq i$, the same equation $\phi_j^1 A^1\text{+}B^1 \approx \phi_j^2 A^2\text{+}B^2$ can be obtained. Combining these two equations leads to $(\phi_i^1 - \phi_j^1)A^1 \approx (\phi_i^2 - \phi_j^2)A^2$, which is equivalent to:

$$\sqrt{(\phi_i^1\text{-}\phi_j^1)A^1 A^{1\mathrm{T}}(\phi_i^1\text{-}\phi_j^1)^{\mathrm{T}}} \approx \sqrt{(\phi_i^2\text{-}\phi_j^2)A^2 A^{2\mathrm{T}}(\phi_i^2\text{-}\phi_j^2)^{\mathrm{T}}} \quad (6)$$

Here, AA^{T}=$(w_1 - \overline{w}, ..., w_K - \overline{w})(w_1 - \overline{w}, ..., w_K - \overline{w})^{\mathrm{T}}$. As (Chang et al. 2020) pointed out, $w_n \in \{w_1, ...w_K\}$ in A can be seen as the center (or mean) of embedding region Z_n, *i.e.*, $E(z_i|y_i$=$n) \approx w_n$. Thus AA^{T} is a reasonable estimation for the covariance matrix Σ^{-1} of Z. Consequently, Equa. 6 educes $d_{\mathcal{M}}(z_i^1, z_j^1) \approx d_{\mathcal{M}}(\phi_i^1, \phi_j^1) \approx d_{\mathcal{M}}(\phi_i^2, \phi_j^2) \approx d_{\mathcal{M}}(z_i^2, z_j^2)$.

Definition 1. ***Embedding Topology Robustness.*** *Given a sample x_i, its relative position to other samples in embedding space Z is defined by 'embedding topology' as: $t(z_i) \triangleq (d_{\mathcal{M}}(z_i, z_1), ..., d_{\mathcal{M}}(z_i, z_N))$. Then the distance metric 'embedding topology robustness' is defined by:*

$$r(x_i) \triangleq exp(-||t(z_i^1) - t(z_i^2)||_2) \quad (7)$$

where $||\cdot||_2$ is Euclidean distance. Following *Property 2*, $r(\cdot)$ possesses the following characteristic.

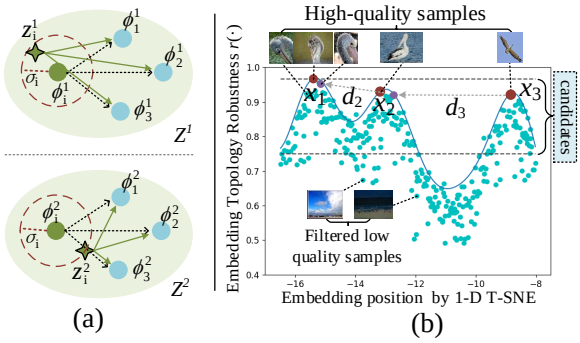

Figure 3: (a) Illustration for the effect of data uncertainty σ_{o_i} on *Embedding Topology Robustness*. (b)The Distribution of $r(\cdot)$ for class 'Pelican' on ImageNet. The blue curve fits the upper contour of the distribution.

Property 3. *High-quality samples have large embedding topology robustness $r(\cdot)$ near 1, while low-quality ones correspond to smaller $r(\cdot)$.*

For a high-quality sample $x_i \in C$ with data uncertainty $\sigma_i \approx 0$, since $d_{\mathcal{M}}(z_i^1, z_j^1) \approx d_{\mathcal{M}}(z_i^2, z_j^2)$, $\forall x_j \in C$, then $||t(z_i^1) - t(z_i^2)||_2$ will be a small value approaching 0, hence robustness $r(x_i)$ will be a large value near 1.

For a low-quality sample $x_i \in (X \backslash C)$ with large uncertainty σ_i, the consistency of *Embedding Topology* will be disrupted. See Figure 3(a), the Mahalanobis distance from class-relevant feature ϕ_i to ϕ_1, ϕ_2, ϕ_3 remains similar in Z^1 and Z^2 following above analysis, thus the topology shape among $\phi_{(\cdot)}$ (dashed arrows) keeps unchanged. But z_i^1 and z_i^2 vary evidently caused by σ_i, hence topology shape from z_i to ϕ_1, ϕ_2, ϕ_3 (green solid arrows) shows great distinctions in two space, which results in a reduced $r(x_i)$. Obviously, the larger uncertainty σ_i will trigger larger variation of topology shape, leading to smaller $r(x_i)$.

Candidate Selection. We denote the set of all input x_i in class k as S_k. To generate candidate prototype set C_k for class k, we first find the sample with the highest *embedding topology robustness* score, *i.e.*, $\max\{r(x_i)|x_i \in \mathrm{S}_k\}$. Then samples with $r(\cdot)$ value above $\epsilon \cdot \max\{r(x_i)|x_i \in \mathrm{S}_k\}$ are elected into C_k, where ϵ is a preset threshold.

4.2 Diverse Prototype-Set Filtering

After selecting all the *high-quality* images into the candidate set $C=\{C_i\}_{i=1}^K$, two problems await: (1) C can be *highly redundant*. As in Figure 3(b), samples near x_1 share similar appearance and features. Such redundancy will bring extra computation cost in the next multi-prototype learning step. A straightforward way is to design a filtering for removing the redundant; (2) The multifarious visual appearance of object within the same class leads to distinguished feature representations. For example in Figure 3(b), x_1, x_2 and x_3 appear in different visual looking and their embedding are located at separated positions, symbolizing the diversity of embedding. Such diversity of embedding should be preserved during filtering.

Upon above, the task is turned to generate final prototype set $P=\{P_k\}_{k=1}^K$ from the obtained candidate set C considering both *high-quality* and *diversity*. Specifically for each class k, the method should find samples with local maximum $r(\cdot)$ and large embedding distance to form P_k, like the x_1, x_2 and x_3 in Figure 3(b).

Similar to the NP-hard *coreset selection* (Sener and Savarese 2018) problem, our goal is to choose T prototypes from C_k into P_k for each class k. We implement it by iteratively collecting prototypes by a greedy algorithm as

$$P_k = \bigcup_{i=1}^{T} \{x_i | \max_{x_i \in C_k} \{ \min_{x_j \in C_k} d_{\mathcal{M}}(z_i, z_j) | r(x_j) > r(x_i) \}\}. \quad (8)$$

For initialization, we search candidates with the max $r(\cdot)$ in C_k through $\max\{r(x_i)|x_i \in \mathrm{C}_k\}$ to initialize P_k, then append candidates satisfying Equation 8 into P_k in an iterative way. The detailed implementation is given in Algorithm 1.

Algorithm 1: Filter Candidate Prototype Set with Diversity

Input: Candidate prototype set $C=\{C_i\}_{i=1}^K$; Class number K; Prototype number per class T;
Output: final prototype set $P=\{P_k\}_{k=1}^K$;

```
for k = 1 to K do
    compute Mahalanobis distance matrix D_k ∈ R^{N_k x N_k}
      in Z^1 (or Z^2), N_k is the candidate number in C_k;
    initial a N_k-length array E with max value in D_k;
    for i = 1 to N_k do
        for i-th candidate x_i ∈ C_k, find its closest candidate
          x_j in C_k, where r(x_j) > r(x_i), if exists,
          update E[i]=D_k[i,j];
    end for
    sort E in descending order, E_ind is the sorted index
      array;
    repeat
        add sample whose index is E_ind[0] in C_k into
          P_k, then remove E_ind[0] from E_ind;
    until the number of samples in P_k exceeds T
end for
```

Taking Figure 3(b) for example, x_1 has the max $r(\cdot)$ thus is first elected, then x_3 and x_2 are successively added into final set, as they correspond to 2nd/3rd largest value d_3 and d_2 in E. Note that Mahalanobis distance of high-quality samples remains similar in Z^1 or Z^2, thus computing D_k in either space leads to similar selected prototypes.

5 Embedding Optimization

Generated prototypes as anchors to represent known classes, we enlarge the distance between different embedding regions to reserve larger space for unknowns. Thus the risk of unknowns misclassified as known classes can be reduced.

5.1 Prototype-based Space Optimization

Given sample x_i belonging to known class m and $P_k=\{p_{k,l}\}_{l=1}^T$, we denote the distance from x_i to

prototype set P_k as $d(z_i, z(P_k))$, where $z(P_k)=(z(p_{k,1}), ..., z(p_{k,T}))\in\mathbb{R}^{D\times T}$ is the embedding of T prototypes in P_k. Then we incorporate a prototype-based constraint to optimize a better embedding space for OSR:

$$\begin{aligned}\mathcal{L}_p=&\frac{1}{N}\sum_{i=1}^{N}[d(z_i, z(P_m)) - d(z_i, z(P_u)) + \delta]_+,\\ &P_u= \underset{P_k\in P\setminus P_m}{\arg\min}\ (d(z_i, z(P_k)))\end{aligned} \tag{9}$$

where P_u is the closest prototype set among other classes and δ is a tunable margin. Unlike existed methods (Yang et al. 2018; Chen et al. 2020) that jointly learn sample embedding z_i and prototype representation $z(P_k)$ in training, we update $z(P_k)$ by directly feeding fixed prototype samples in P_k into current embedding model, thus our model can focus on learning a better sample embedding z_i. Such training strategy is more advantageous since we not only avoid the unstable learning of $z(P_k)$, but also ease training difficulty of z_i. Finally, the loss in training phase is a combination:

$$\mathcal{L} = \mathcal{L}_{cls} + \lambda_p\mathcal{L}_p, \tag{10}$$

where $\mathcal{L}_{cls}$ is the SoftMax loss and λ_p is a balancing coefficient. Besides, we design a new point-to-set distance metric with self-attention (Vaswani et al. 2017) mechanism to effectively measure the distance $d(z_i, z(P_k))$.

$$\begin{aligned}d(z_i, z(P_k))=&1-\frac{{z_i}^{\mathrm{T}} z_i^{att}(P_k)}{|{z_i}^{\mathrm{T}}||z_i^{att}(P_k)|},\\ z_i^{att}(P_k)=&\mathit{SoftMax}(\frac{z_i^{\mathrm{T}} z(P_k)}{\sqrt{d}})z(P_k)\end{aligned} \tag{11}$$

where $\sqrt{d}$ is a scale factor(Vaswani et al. 2017) and $|\cdot|$ denotes L2 norm. We use z_i to query embedding in $z(P_k)$ to get its similarity with each prototype, and obtain weighted sum $z_i^{att}(P_k)$. Then distance is computed by referring to similarity between z_i and $z_i^{att}(P_k)$. We jointly consider the correlations between x_i and all diversified prototypes, thus measure the point-to-set distance more comprehensively.

5.2 Rejecting Unknowns

Following the general routine (Yang et al. 2020), two rejection rules are adopted for detecting unknown samples: (1) *Probability based Rejection (PR)*. We directly reject unknowns by thresholding SoftMax probability scores; (2) *Distance based Rejection (DR)*. Unknowns are rejected by thresholding the minimum point-to-set distance, *i.e.*$min\{d(z_i, z(P_k))\}$ where $P_k\in P$, since unknown samples should have larger distance with the closest prototype set than known samples.

6 Experiments

6.1 Experiments on Small-Scale Benchmarks

Datasets. Following (Neal et al. 2018), we first conduct comparisons with state-of-the-arts on 6 standard datasets including (1) MNIST (LeCun et al. 1998), SVHN (Netzer et al. 2011), CIFAR10 (Krizhevsky and et al 2009): 4 classes are randomly selected as known classes and the rest 6 classes are unknowns; (2) CIFAR+10, CIFAR+50: 4 non-animal classes from CIFAR10 are chosen to be known classes, then 10 and 50 animal classes are respectively sampled from CIFAR100 (Krizhevsky and et al 2009) to be unknowns; (3) TinyImageNet (TINY) (Ya and Xuan 2015): 20 classes are randomly sampled as knowns and the left 180 classes are unknowns.

Implementations. Two backbones are adopted to implement our method. The light-weighted backbone OSCRI (Neal et al. 2018) with parameters less than $1M$ is used to validate our performance when equipped on applications with limited resources. The larger-scale backbone Wide-ResNet (WRN) (Chen et al. 2020) with $9M$ parameters is implemented for a fair comparison with previous methods, whose parameter is still less than most existed methods (Yoshihashi et al. 2019; Oza and Patel 2019; Sun et al. 2020). We adopt Adam optimizer to train our model on each dataset for 600 epochs with batchsize 128. The learning rate starts at 0.01 and is dropped by 0.1 every 120 epochs, momentum is set to 0.9 and weight decay is 5e-4. The same optimization strategy is used for obtaining pre-trained models and for embedding space optimization. For all datasets, the margin δ is fixed to 0.5, λ_p is set to 1 and prototype number T=10.

Evaluation Protocols. Following (Neal et al. 2018), the evaluation includes 2 parts: (1) close set performance of known classes is reported by classification accuracy ACC on test set of knowns, and (2) unknown detection performance is evaluated by the most adopted metric AUROC (Area Under ROC Curve) (Neal et al. 2018) on the test set of both known and unknown classes. Reported results are averaged over 5 random splits. We observe two rejection rules lead to similar results, thus we simply report the results of *DR*.

Result Comparison. (1) **Close Set Recognition**: Table 1 shows we obtain the best ACC on all datasets, especially the gain reaches 2%$\sim$3% on three CIFAR datasets and TINY. We attribute it to the fact that PMAL learns more compact intra-class embedding compared to other methods (shown in Figure 6(e)$\sim$(h), thus the classification decision boundaries among classes can be more correctly drawn. (2) **Open Set Recognition**: PMAL achieves the best AUROC on all benchmarks in Table 1, especially on the most complex TINY, PMAL-WRN achieves 2.2% gain compared to previous best RPL-WRN. The superiority of PMAL is more obvious when equipped on light-weight 'OSCRI'. Compared to RPL-OSCRI, ARPL and CPN with the same backbone, PMAL-OSCRI outperforms them by a larger margin over 3.6%. Besides, the light PMAL-OSCRI only falls slightly behind PMAL-WRN, still holding top performance among all the reported results (even those with larger networks).

6.2 Experiments on Larger-Scale Benchmarks

Datasets. We further validate our method on more challenging large-scale datasets including (1) ImageNet-100, ImageNet-200 (Yang et al. 2020): the first 100 and 200 classes from ImageNet (Deng et al. 2009) are selected as known classes and the rest are treated as unknowns; (2) ImageNet-LT (Liu et al. 2019): a long-tailed dataset with 1000 known classes from ImageNet-2012 (Deng et al. 2009), and additional classes in ImageNet-2010 are as un-

Table 1: Close set ACC and Open set AUROC on small datasets. '*' denotes implemented results and 'C' is short for 'CIFAR'.

Methods	Close set ACC						Open set AUROC					
	MNIST	SVHN	C10	C+10	C+50	TINY	MNIST	SVHN	C10	C+10	C+50	TINY
SoftMax	99.5	94.7	80.1	-	-	-	97.8	88.6	67.7	81.6	80.5	57.7
CPN (Yang et al.)	99.7	96.7	92.9	94.8*	95.0*	81.4*	99.0	92.6	82.8	88.1	87.9	63.9
PROSER (Zhou, Ye, and Zhan)	-	96.5	92.8	-	-	52.1	94.3	-	89.1	96.0	95.3	69.3
CGDL (Sun et al.)	99.6	94.2	91.2	-	-	-	99.4	93.5	90.3	95.9	95.0	76.2
OpenHybrid (Zhang et al.)	94.7	92.9	86.8	-	-	-	99.5	94.7	95.0	96.2	95.5	79.3
RPL-OSCRI (Chen et al.)	99.5*	95.3*	94.3*	94.6*	94.7*	81.3*	99.3	95.1	86.1	85.6	85.0	70.2
ARPL (Chen et al.)	99.5	94.3	87.9	94.7	92.9	65.9	99.7	96.7	91.0	97.1	95.1	78.2
RPL-WRN (Chen et al.)	99.6*	95.8*	95.1*	95.5*	95.9*	81.7*	99.6	96.8	90.1	97.6	96.8	80.9
PMAL-OSCRI	99.6	96.5	96.3	96.4	96.9	84.4	99.5	96.3	94.6	96.0	94.3	81.8
PMAL-WRN	**99.8**	**97.1**	**97.5**	**97.8**	**98.1**	**84.7**	**99.7**	**97.0**	**95.1**	**97.8**	**96.9**	**83.1**

knowns. The image number per class ranges from 5 to 1280, thus it can well simulate the problem of long-tailed data distribution.

Implementations. Similar to (Chen et al. 2020), ResNet-50 (He et al. 2016) is used as network backbone and SGD optimizer is adopted with learning rate 0.2, which drops by 0.1 every 30 epochs. Other detailed setups are the same with experiments on small-scale datasets in Section 6.1.

Result Comparison. The same metrics (ACC and AUROC) are used for evaluation. See Table 2, our method improves close set ACC by 3.2% and 2% on ImageNet-LT and ImageNet-200. Moreover, open set AUROC is significantly enhanced by 16.5%, 12.6% and 13.7% compared to state-of-the-art. Such performance gains are much more evident than the improvements on small-scale datasets, reflecting that our method possesses larger advantage in more challenging large-scale tasks. It can be observed that the parameter number (apart from the adopted same backbone) of previous methods increases along with known class number, which exceeds a non-negligible cost 2M on ImageNet-LT. The increased prototype parameters may aggravate the difficulty of model training, thus deteriorate their performance. Instead, since PMAL brings no parameters for prototypes, its performance is invariant to the number of known classes, which explains the promotion on more complicated datasets.

Besides, the huge promotion on *long-tailed* dataset shows PMAL can better handle the problem than the existing. Previous ones tend to learn unreliable prototypes for those few-sample classes, due to unbalanced training of various classes. While PMAL directly mines prototypes from training data, thus produces stable prototypes with fewer samples.

6.3 Detailed Analysis

Ablation of Each Component in PMAL. As shown in Table 3, we perform each experiment of the proposed components for ablation study, including the components regarding the two properties, *i.e.*, *high-quality* and *diversity* (see Sec. 4), and the embedding learning procedure using the *point-to-set* distance metric (see Sec. 5.1).

- In prototype mining (PM) phase, both *high-quality* and *diversity* matters in prototypes, where *high-quality* is the more crucial factor validated by our experiments. Jointly combining the two further boosts the performance.
- In embedding optimization (EO) phase, no '✓' denotes the commonly adopted way: computing the distance from sample x_i to its nearest prototype. Obviously, our proposed *point-to-set* metric has 1.2% gain, comparing (a)/(c),(b)/(d) or (e)/(f).

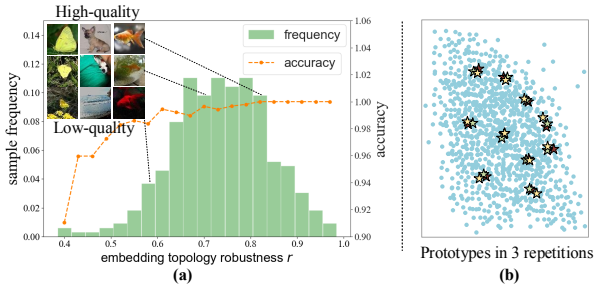

Figure 4: (a) Distribution of r; (b) Prototypes in embedding space (visualized by T-SNE) under 3 repetitions. Star in different color denotes prototypes in different repetition.

Effect of Embedding Topology Robustness. (1) Distribution of $r(\cdot)$: We further analyze the distribution of $r(\cdot)$ on TinyImageNet in Figure 4(a). It shows different samples correspond to various robustness, which decreases along with quality degradation caused by occlusion or background interference *etc.* Besides, we summarize classification accuracy in each interval of $r(\cdot)$, which shows a monotonously ascending trend. It means high-quality samples have larger probabilities to be correctly identified. **(2) Comparison with other methods**: Furthermore, we compare with mainstream methods to prove the advantage of *embedding topology robustness* $r(\cdot)$ for selecting high-quality samples on TinyImageNet: (a) *Probability*: samples with the highest predicted probability from each class are chosen as prototypes; (b) *Deep Ensembles* (Lakshminarayanan, Pritzel, and Blundell 2017): samples with the lowest probability variance between two models are used as prototypes; (c) *MC-dropout* (Gal and Ghahramani 2016): data uncertainty is modeled by MC-dropout with ratio 0.5 and samples with the lowest probability variance are selected as prototypes.

Table 2: Comparisons on 3 large-scale datasets. We denote 'ImageNet' as 'IN' for simplicity.

Method	Close Set ACC			Open Set AUROC			Additional Params		
	IN-LT	IN-100	IN-200	IN-LT	IN-100	IN-200	IN-LT	IN-100	IN-200
Softmax	37.8	81.7	79.7	53.3	79.7	78.4	0	0	0
CPN	37.1	86.1	82.1	54.5	82.3	79.5	2M	0.2M	0.4M
RPL	39.0	81.8*	80.7*	55.1	81.2*	80.2*	2M	0.2M	0.4M
RPL++	39.7	-	-	55.2	-	-	4M	-	-
PMAL	**42.9**	**86.2**	**84.1**	**71.7**	**94.9**	**93.9**	0	0	0

Table 3: Ablations of each module on TinyImageNet.

Components		(a)	(b)	(c)	(d)	(e)	(f)
PM	High-Quality	✓		✓		✓	✓
	Diversity		✓		✓	✓	✓
EO	Point-to-Set			✓	✓		✓
AUROC		80.3	78.1	81.6	80.2	81.9	**83.1**

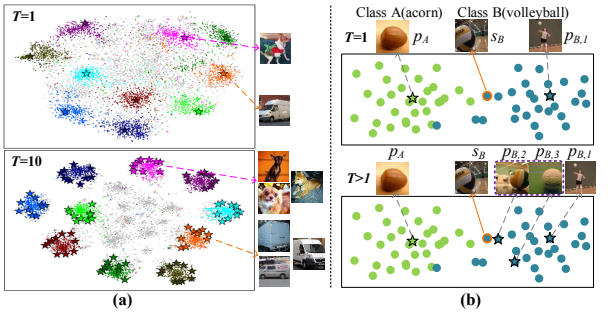

Figure 5: (a)Embedding space (visualized by 2-D T-SNE) of different prototype number; (b)Illustration for the advantage of multiple prototypes. Classes are distinguished by color, star denotes prototype and circle denotes other samples.

For a fair comparison, we fix ϵ to 0.7 and replace $r(\cdot)$ with above 'probability' or 'probability variance', and then adopt the same diversity strategy to produce final prototypes. Table 4 verifies proposed *embedding topology robustness* is better than other widely-used data uncertainty modeling methods.

Effect of Diversity-based Filtering. **(1) Number of prototypes:** We vary prototype number T to quantify the effect of diversity on TinyImageNet. As in Table 5, diverse prototypes lead to much better performance than a single prototype, we obtain the best result when T=10. When T increases over 10, performance gradually stabilized. To interpret the advantage of diversity, we visualize embedding space when T=1 or T=10 in Figure 5(a). Evidently, the chosen 10 prototypes appear in diverse visual looking and their embeddings are located at separate positions. Moreover, using 10 prototypes learns more compact intra-class embedding regions, where unknown sample points are farther from the embedding region of known classes. The reason can be illustrated by Figure 5(b): when T=1, sample s_B from class B looks even more like the prototype p_A of class A than the prototype $p_{B,1}$ of class B, but its embedding will still be forced to be closer to $p_{B,1}$ than p_A, which is hard to be optimized. Instead when T>1, embedding of s_B is mainly pulled closer to $p_{B,2}$ and $p_{B,3}$ with similar appearance and adjacent embedding, which eases training difficulty. Hence it learns more compact intra-class embedding. **(2) Comparison with other methods**: We compare with 2 strategies to select multiple (T=10) prototypes from same candidates: (a) *Randomization*: prototypes are randomly selected from candidates. (b) *Clustering*: K-Means clustering is used and samples whose embedding nearest to cluster centers are used as prototypes. Table 4 shows the obvious advantage using our diversity-based method above randomization and clustering.

Table 4: Comparisons with other methods on the *quality* and *diversity* property.

Method	ACC	AUROC
(a)Probability	81.9	79.3
(b)Deep Ensembles	82.3	80.5
(c)MC-dropout	81.6	78.8
(a)Randomization	81.5	79.1
(b)Clustering	81.8	79.6
Ours	**84.7**	**83.1**

Table 5: AUROC under different hyper-parameters, including T,ϵ,U and δ.

T	1	5	10	20	30
AUROC	79.9	81.1	83.1	82.6	83.1
ϵ	0.1	0.3	0.5	0.7	0.9
AUROC	73.6	78.1	82.6	83.1	81.2
U	2	3	4	5	6
AUROC	83.1	83.3	83.2	83.0	83.3
δ	0.1	0.3	0.5	0.8	1.
AUROC	80.9	82.8	83.1	82.1	80.5

Fluctuation of Prototypes. The selected prototypes are affected by pre-trained embedding model M^1 and M^2. We repeat the mining process for 3 times using different embedding models, and give the results of a class from TinyImageNet in Figure 4(b). We observe prototypes chosen in different repetitions only fluctuate very slightly in embedding space, which validates the stability of prototype mining.

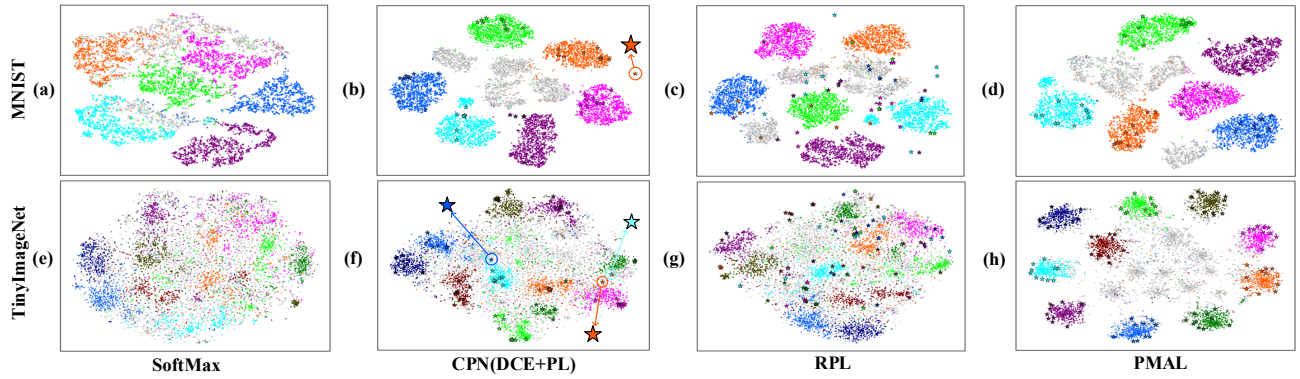

Figure 6: Learned embedding visualized by 2-D T-SNE. CPN and RPL are optimized to achieve reported results. We visualize 10 from the 20 known classes on TINY for better clarity. Each color denotes different classes and 'gray' denotes unknowns, prototypes are marked as stars. Better viewed by zooming in.

Visualization of Embedding Space. We compare learned embedding space of PL methods in Figure 6. On the easier MNIST dataset, compared to the naive 'SoftMax', the latter three can enlarge the distance among known classes.But on the more complex TinyImageNet, our method pushes away the embedding region of known classes to a much larger extent compared to CPN and RPL, so the overlap between known and unknowns is much more evidently reduced.

For CPN, we observe undesired prototypes in the circle of Figure 6(f), arising from the unstable prototype learning fooled by low-quality samples. For RPL, known and unknown samples also can not be well separated by referring to learned prototypes in Figure 6(g). This reveals existed methods endanger from learning sub-optimal prototypes or embedding space especially in complicated tasks. Instead, PMAL is capable of mining trustworthy prototypes and optimizing satisfying embeddings in more challenging tasks.

Hyper-parameters. (1) **Threshold** ϵ. ϵ controls the balance of prototype quality and diversity. A larger ϵ implies more strict condition to ensure quality, but less samples will be elected as candidates, thus diversity among candidates is reduced. A smaller ϵ is on the contrary. As in Table 5, setting ϵ too small or large both results in AUROC declined. We find ϵ in the range [0.6, 0.8] leads to stable performance. (2) **Initial model number** U. Optionally we can adopt more than 2 models to compute $r(\cdot)$, that is, we first compute $r(\cdot)$ with Equation 7 between two arbitrary models then average the results. AUROC remains similar as we increase U, shown in Table 5, which implies 2 models are already sufficient to extract effective prototypes. Note that we adopt 2 models for prototype mining, but only one model is employed for prototype learning, thus no extra model parameters are added during inference. (3) **Margin** δ **and** λ_p. δ decides the separability of different embedding regions. When δ is too small, different regions can not be well separated. But if δ becomes too large, $\mathcal{L}_p$ will grow to a large value overwhelming $\mathcal{L}_{cls}$, causing $\mathcal{L}_{cls}$ hard to converge. See Table 5, a value between 0.3 and 0.5 for δ can produce better AUROC results. For loss weight λ_p, when we vary it among 0.5, 0.8 and 1, the resulted AUROC are 82.6, 82.8 and 82.9, which implies PMAL is not very sensitive to λ_p. For all the validations in various tasks, we adopt the universal hyper-parameter setting: T=10, ϵ=0.7, U=2, δ=0.5 and λ_p=1.

7 Conclusion

This paper proposes a novel prototype mining and learning algorithm. It directly discovers high-quality and diversified prototype sets from training samples. Then based on generated prototypes, the OSR model can focus on optimizing a better embedding space in which known and unknown classes are separated. Extensive experiments on various benchmarks show that our method outperforms the state-of-the-art approaches. In future work, we will explore our prototype mining mechanism in broader tasks other than OSR.

References

Bendale, A.; and Boult, T. E. 2016. Towards Open Set Deep Networks. In *CVPR*, 1563–1572.

Chang, J.; Lan, Z.; Cheng, C.; and Wei, Y. 2020. Data Uncertainty Learning in Face Recognition. In *CVPR*, 5709–5718.

Chen, G.; Peng, P.; Wang, X.; and Tian, Y. 2021. Adversarial Reciprocal Points Learning for Open Set Recognition. *IEEE TPAMI*, 1–1.

Chen, G.; Qiao, L.; Shi, Y.; Peng, P.; Li, J.; Huang, T.; Pu, S.; and Tian, Y. 2020. Learning Open Set Network with Discriminative Reciprocal Points. In *ECCV*, volume 12348, 507–522.

Deng, J.; Dong, W.; Socher, R.; Li, L.; Li, K.; and Li, F. 2009. ImageNet: A large-scale hierarchical image database. In *CVPR*, 248–255.

Gal, Y.; and Ghahramani, Z. 2016. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *ICML*, volume 48, 1050–1059.

Ge, Z.; Demyanov, S.; and Garnavi, R. 2017. Generative OpenMax for Multi-Class Open Set Classification. arXiv:1707.07418.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *CVPR*, 770–778.

Kendall, A.; Badrinarayanan, V.; and Cipolla, R. 2016. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. arXiv:1511.02680.

Krizhevsky, A.; and et al, G. H. 2009. Learning multiple layers of features from tiny images. Technical report.

Lakshminarayanan, B.; Pritzel, A.; and Blundell, C. 2017. Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *NeurIPS*, 6402–6413.

LeCun, Y.; Bottou, L.; Bengio, Y.; and Haffner, P. e. a. 1998. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324.

Liu, Z.; Miao, Z.; Zhan, X.; Wang, J.; Gong, B.; and Yu, S. X. 2019. Large-Scale Long-Tailed Recognition in an Open World. In *CVPR*, 2537–2546.

Ma, M.; Shao, M.; Zhao, X.; and Fu, Y. 2013. Prototype based feature learning for face image set classification. In *FG*, 1–6.

Neal, L.; Olson, M. L.; Fern, X. Z.; Wong, W.; and Li, F. 2018. Open Set Learning with Counterfactual Images. In *ECCV*, volume 11210, 620–635.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. 2011. Reading digits in natural images with unsupervised feature learning. In *NeurIPS workshop*.

Oza, P.; and Patel, V. M. 2019. C2AE: Class Conditioned Auto-Encoder for Open-Set Recognition. In *CVPR*, 2307–2316.

Rosch, E. 1973. Natural categories. *Cognitive psychology*, 4(3): 328–350.

Scheirer, W. J.; de Rezende Rocha, A.; Sapkota, A.; and Boult, T. E. 2013. Toward Open Set Recognition. *IEEE TPAMI*, 35(7): 1757–1772.

Sener, O.; and Savarese, S. 2018. Active Learning for Convolutional Neural Networks: A Core-Set Approach. arXiv:1708.00489.

Shi, Y.; and Jain, A. K. 2019. Probabilistic Face Embeddings. In *ICCV*, 6901–6910.

Snell, J.; Swersky, K.; and Zemel, R. S. 2017. Prototypical Networks for Few-shot Learning. In *NeurIPS*, 4077–4087.

Sun, X.; Yang, Z.; Zhang, C.; Ling, K. V.; and Peng, G. 2020. Conditional Gaussian Distribution Learning for Open Set Recognition. In *CVPR*, 13477–13486.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, L.; and Polosukhin, I. 2017. Attention Is All You Need. In *NeurIPS*, 6000–6010.

Wang, W.; Wang, R.; Shan, S.; and Chen, X. 2016. Prototype Discriminative Learning for Face Image Set Classification. In *ACCV*, volume 10113, 344–360.

Ya, L.; and Xuan, Y. 2015. Tiny imagenet visual recognition challenge. *CS 231N*, 7(7): 3.

Yang, H.; Zhang, X.; Yin, F.; and Liu, C. 2018. Robust Classification With Convolutional Prototype Learning. In *CVPR*, 3474–3482.

Yang, H.-M.; Zhang, X.-Y.; Yin, F.; Yang, Q.; and Liu, C.-L. 2020. Convolutional Prototype Network for Open Set Recognition. *IEEE TPAMI*, 1–1.

Yoshihashi, R.; Shao, W.; Kawakami, R.; You, S.; Iida, M.; and Naemura, T. 2019. Classification-Reconstruction Learning for Open-Set Recognition. In *CVPR*, 4016–4025.

Zhang, H.; Li, A.; Guo, J.; and Guo, Y. 2020. Hybrid Models for Open Set Recognition. In *ECCV*, volume 12348, 102–117.

Zhou, D.; Ye, H.; and Zhan, D. 2021. Learning Placeholders for Open-Set Recognition. In *CVPR*, 4401–4410.