Learning Network Architecture for Open-Set Recognition

Xuelin Zhang[1,3], Xuelian Cheng[1,3], Donghao Zhang[2,3], Paul Bonnington[2], Zongyuan Ge[2,3]

[1]Monash University,
[2]Monash eResearch Centre,
[3]Monash Medical AI
{xuelin.zhang,xuelian.cheng,donghao.zhang,paul.bonnington,zongyuan.ge}@monash.edu

Abstract

Given the incomplete knowledge of classes that exist in the world, Open-set Recognition (OSR) enables networks to identify and reject the unseen classes after training. This problem of breaking the common closed-set assumption is far from being solved. Recent studies focus on designing new losses, neural network encoding structures, and calibration methods to optimize a feature space for OSR relevant tasks. In this work, we make the first attempt to tackle OSR by searching the architecture of a Neural Network (NN) under the open-set assumption.In contrast to the prior arts, we develop a mechanism to both search the architecture of the network and train a network suitable for tackling OSR.Inspired by the compact abating probability (CAP) model, which is theoretically proven to reduce the open space risk, we regularize the searching space by VAE contrastive learning.To discover a more robust structure for OSR, we propose Pseudo Auxiliary Searching (PAS), in which we split a pretended set of know-unknown classes from the original training set in the searching phase, hence enabling the super-net to explore an effective architecture that can handle unseen classes in advance. We demonstrate the benefits of this learning pipeline on 5 OSR datasets, including MNIST, SVHN, CIFAR10, CIFARAdd10, and CIFARAdd50, where our approach outperforms prior state-of-the-art networks designed by humans. To spark research in this field, our code is available at https://github.com/zxl101/NAS_OSR.

Introduction

Deep learning has achieved great success in classification and recognition tasks (He et al. 2016; Krizhevsky, Sutskever, and Hinton 2012; Simonyan and Zisserman 2015). However, due to the assumption that the test classes are consistent with the classes from the training set, models often fail in real-world scenarios when unexplored unknown classes appear during the inference stage, e.g. clinical diagnosis (Tian et al. 2020) and autonomous driving (Wong et al. 2020). The key question to ask is – Without enough knowledge about the world with open space risk (Cevikalp and Serhan Yavuz 2017), can our models still perform well when facing challenging unseen scenarios?

Open-set Recognition setting by introducing novel unknown classes into the model testing provides a new evaluation criterion to verify the robustness of models by breaking the conventional closed-set assumption. More specifically, under the open-set assumption, data can be split into three categories, *known known classes* (KKCs), *known unknown classes* (KUCs), and *unknown unknown classes* (UUCs) (Scheirer et al. 2013). The goal of Open-set research is to make sure the model can successfully distinguish all known classes from the training set while rejecting unknown classes in the inference phase.

OSR methods can be simply classified into two main streams, traditional machine learning based methods and deep learning based methods. A representative traditional machine learning based OSR method is the compact abating probability (CAP) model proposed in (Scheirer, Jain, and Boult 2014) to explicitly reduce the open space risk. Deep learning based methods use deep learning models with different losses and recognition functions to tackle OSR. Early deep learning OSR models (Bendale and Boult 2016) calibrate softmax scores and use extreme value theorem to detect outliers.

CAP models: An OSR model should be able to reduce the open space risk, which is proposed in (Scheirer et al. 2013). It is defined as the relative measure of open space compared to the overall measure space.According to (Scheirer, Jain, and Boult 2014), the compact abating probability (CAP) model is proven to reduce the open space risk by the idea that a CAP model ensures the recognition function is decreasing away from the training data. A Weibull-calibrated SVM with standard RBF kernels is a CAP model. The properties of the CAP model ensure a smooth boundary of the training data, and thus thresholding can be used to limit the labeled region and classify unknowns. An advantage of the CAP model is by adjusting the threshold, one can reduce the amount of open space that can be labeled positive and control the open space risk. Theoretically, a CAP model should be able to reduce the open space risk to zero. However, as mentioned in (Scheirer, Jain, and Boult 2014), the performance of the CAP model still depends on how well the model can learn positive regions of known classes using probabilities. Under the deep learning setting, this is equivalent to whether the model has the ability to learn feature representations with smooth boundaries. This ability is affected by the model architecture, as the operations and connections in a model determine the projection of input data to the corresponding feature representation.

Model architecture: Prior OSR works have exploited popular neural networks, such as VGG-16 (Simonyan and Zisserman 2015), and RESNET50 (He et al. 2016). However, these neural networks are designed based on philosophy and observations under the closed-set assumption. There is no clear evidence showing that such tailored human-designed networks for closed-set problems are optimal for open-set problems. Taken the single layer as an example, recent work (Yao et al. 2021) shows that while batch normalization is a popular layer used in closed-set problems, it is not optimal for open-set problems. Furthermore, the complexity of a model determines the upper boundary classification ability of the neural network (Chen, Gong, and Wang 2021). A model with low complexity is unable to address the OSR task. Meanwhile, a model with high complexity may overfit known class distributions and fails on the OSR task.

Based upon consideration, we would like to explore an optimal network that can fit open-set problems with the underlying principle of the CAP model. We propose to automate the process of learning architecture using NAS under the open-set assumption for OSR. To ensure the searched model can reduce open space risk in a tractable manner, our method cooperates with VAE Contrastive Learning. We use a variational autoencoder with each class having a multivariate Gaussian prior distribution. Since a class is a multivariate Gaussian distribution, samples far from the class mean have lower probabilities belonging to that class and higher probabilities of being from unknown classes. This makes our searched architecture a CAP model. Combining with the uniqueness of the OSR problem setting, we further introduce a novel training strategy named Pseudo Auxiliary Searching, which allows our model to exploit pseudo KUCs in the architecture search phase. Specifically, we split partial sub-classes from the KKCs as pseudo KUCs for validation, while the rest of the original sub-classes in KKCs are for searching and training. PAS allows us to choose the best network architecture from an existing pool of architectures. All architectures in the pool are trained using the same set of KKCs and tested on the same set of pseudo KUCs. Our contributions are the following:

- We propose an end-to-end hierarchical NAS pipeline for Open-set Recognition. By using VAE contrastive learning, we ensure our searched model is equivalent to a CAP model and can reduce open space risk in a tractable manner.

- The network architecture is searched using the novel PAS strategy. By using part of the known classes as pseudo unknown classes, we can search for a more robust network architecture for OSR during the search phase while utilizing the whole training set during the training phase.

- We evaluated our model on 5 common OSR datasets and the model is able to outperform state-of-the-art OSR methods on all datasets while having fewer parameters than the baseline model. The ablation studies show the benefits of different components designed for the open-set problem in our method.

Related Work

Open-Set Recognition Deep learning-based OSR methods can be categorized into two groups: discriminative model-based and generative model-based methods. Discriminative model-based methods calibrate the classification logistics to detect UUCs. OpenMax (Bendale and Boult 2016) uses an OpenMax layer and fits output probabilities with Weibull distributions. Generative model-based methods, on the other hand, learn distributions of known classes. G-OpenMax (Ge et al. 2017) uses a conditional GAN to synthesize mixtures of unknown classes and predict explicit probability estimations over unknown classes. GCM-CF (Yue et al. 2021) disentangles sample attributes and class attributes. Previous methods do not focus on designing networks and use existing popular networks for closed-set problems. Different from GCM-CF, our work focuses on a method to find a better architecture. With the searched architecture and a simplified network, our work outperforms GCM-CF and discovers the ability to use NAS to solve OSR problems.

Neural Architecture Search Early NAS methods (Zoph and Le 2017; Real et al. 2017) search in a discrete global search space from scratch, and each network architecture candidate needs to be trained fully. This results in tremendous training time and only part of the model can be optimized using back propagation (Ren et al. 2020). DARTS (Liu, Simonyan, and Yang 2019) relaxes the discrete search space and treats the search as a bilevel optimization problem. This relaxation allows the usage of gradient descent to efficiently optimize the architecture. Despite the success of NAS in closed-set tasks (Tan and Le 2019; Mei et al. 2020; Li et al. 2020; Wang et al. 2020; Chen et al. 2019), related studies regarding how to design a good architecture for Open-set Recognition still remain absent in the open-set community. As the task itself is quite challenging and in its early stages, there is not a clear principle for how the automated algorithm can optimize network architectures to benefit open space learning. There are only two recent works that are related to this topic. NADS (Ardywibowo et al. 2020) searches the block architecture based on reinforcement learning, but it only tackles the binary classification problem. (Sun et al. 2019) uses NAS to solve OSR, but the search of network architecture is only performed under the closed-set setting.

Our Method

Problem Definitions For a fair comparison, our work follows the OSR problem setting in (Scheirer et al. 2013). Assuming classes of all objects belong to $K \cup U$, where K are known classes and U are unknown classes, every image of object can be processed into a d-dimensional vector $v \in \chi \subset R^d$, where χ is the whole feature space of all objects. The training set is $D_{train} = \{v^i_{train}, y^i_{train}\}$, where v^i_{train} is the feature vector of the i-th sample and $y^i_{train} \in K$ is the corresponding label. The testing set is $D_{test} = \{v^i_{test}, y^i_{test}\}$, where $v^i_{test} \in \chi$ and $y^i_{test} \in K \cup U$. Given a sample from the test set, the model needs to distinguish between samples from K and U and correctly predict its class if it belongs to K.

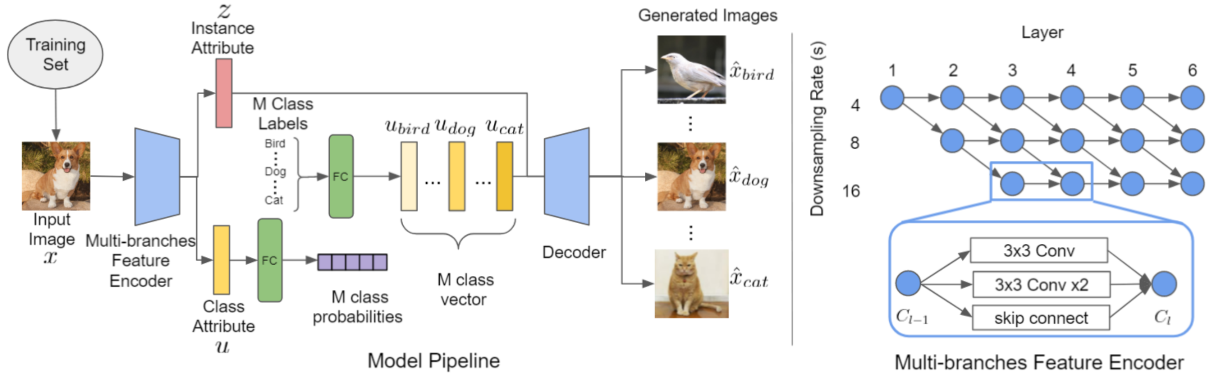

Figure 1: The left figure shows the overall encoder-decoder pipeline. The right figure shows the searchable space for the Multi-branches Feature Encoder and its three candidate operations.

OSR-NAS Framework At a high level, we propose an end-to-end hierarchical NAS pipeline for Open-set Recognition. In particular, the entire process of architecture searching is formulated as bilevel optimization. The network weights and the network architecture parameters are updated alternately. Prior work NADS (Ardywibowo et al. 2020) only searches a single block structure on the proxy task and progressively conducts an ensemble of blocks to optimize entropy estimation. Different from NADS (Ardywibowo et al. 2020), we formulate the searchable super-network as a multi-branches encoder and optimize the entire network by embedding OSR guidance. As shown in Figure 1, we formulate our model as a Variational Autoencoder (VAE), which consists of two modules: (i) a multi-branches feature encoder, to perform the network-level and cell-level search; and (ii) a decoder network to reconstruct images. VAE allows us to use recognition functions based on probabilities and makes our model a CAP model. To learn a better architecture for OSR, we exploit the output from the encoder, class attribute vector u, and instance attribute vector z, which connect the encoder and the decoder.

Architecture Search Space

Cell-level Search Space A cell is defined as the basic searchable unit in NAS. Our cell structure contains one input node, one intermediate node, and one output node. For a layer $l \in L$, the output node is N_l and the input node is the output from the previous layer N_{l-1}. During the architecture search, functions in an intermediate node are defined as below:

$$N_l = \sum_{i=1}^{d_l} \frac{exp(\alpha_{i,l})}{\sum_{j=1}^{d_l} exp(\alpha_{i,l})} o_{i,l}(N_{l-1}) \qquad (1)$$

where $o_{i,l}$ is the i-th candidate operation of the layer l and d_l is the number of candidate operations in the layer l. $\alpha_{i,l}$ is the weight vector for the i-th operation in layer l, and a softmax function is used alongside to limit the sum of the weight vectors in one intermediate node to be 1. After the architecture search is completed, the operation with the highest weight is chosen to formulate the task-optimized architecture. A different intermediate node is also searched in each layer instead of a global cell structure.

Network-level Search Space At a higher level, our network contains candidate paths that connect cells and control the spatial resolution variations. Inspired by (Chen et al. 2020), our search space is defined by a set of downsampling rates S and the number of layers L, as shown in Figure 1. Since the size of images in the dataset is 32×32, we choose downsampling rates of $1/4, 1/8$, and $1/16$. We follow the common practice and double the number of filters when halving the height and width of a feature map. We pre-define the number of layers L as 6. This allows the depth of the encoder to be in the range of the number of layers in GCM-CF (10 layers) and the popular VGG-16 backbone (16 layers). We use a set of search parameters β to search over network paths in order to find a path that minimizes the loss. Taken a cell $C_{l-1,s}$ as input, each path has two options: downsampling $O_{l,\frac{s}{2}}$ or keeping the resolution $O_{l,s}$. The input $I_{l,s}$ of a cell $C_{l,s}$ is:

$$I_{l,s} = \beta_{l,2s} O_{l-1,2s} + \beta_{l,s} O_{l-1,s} \qquad (2)$$

where the sum of $\beta_{l,2s}$ and $\beta_{l,s}$ equals to one. $O_{l-1,2s}$ and $O_{l-1,s}$ are calculated by Equation 1. At most one path is searched at each downsampling rate. The searched path is called a branch. We name branches with downsampling rates 4, 8, 16 as branch 0, branch1, and branch 2 correspondingly. Branch 2 is always selected since it generates the final feature vector to both a classifier and a decoder.

OSR Architecture Learning

Optimized architecture for OSR should have the capability to encode samples to a feature space with the following properties according to the CAP model.Samples from the unknown classes among various known class clusters should bear a clear margin. Meanwhile, samples from the same known class should tightly cluster together and there exists a margin to make different known class clusters separable. These two properties can be re-formulated as the two research questions,
I. *Can the network learn the ability to identify unknown classes in the searching phase?*
II. *How does the network disentangle the intra-class information thus separating unknown classes and known classes in a single search epoch?*

Algorithm 1: SEARCHING PIPELINE

```
Input: input image $x$
       input class label $y$
       All $M$ class labels $Y$
       training datasets $\mathcal{D}_{train\_p}$ and $\mathcal{D}_{train\_w}$
       validation dataset $\mathcal{D}_{val}$
       network weights $w$
       searchable architecture network parameters
       $\{\alpha, \beta\}$
       Encoder module $\mathcal{E}$ and decoder module $\mathcal{D}$
       Classifier module $F$
       Label encoding module $Le$
Split $\mathcal{D}_{train\_p}$ into $\mathcal{D}_{train\_p\_kkc}$ and $\mathcal{D}_{train\_p\_pkuc}$;
Split $\mathcal{D}_{train\_w}$ into $\mathcal{D}_{train-w\_kkc}$ and $\mathcal{D}_{train-w\_pkuc}$;
repeat Bi-level optimization
    On $\mathcal{D}_{train-w\_kkc}$:
        Forward$(x, y, Y)$;
        Update $w$ by $\bigtriangledown_w \mathcal{L}(w, \alpha, \beta)$ ;
    On $\mathcal{D}_{train\_p\_kkc}$:
        Forward$(x, y, Y)$;
        update $\alpha$ and $\beta$ by $\bigtriangledown_{\alpha,\beta} \mathcal{L}(w, \alpha, \beta)$;
    On $\mathcal{D}_{train\_p\_pkuc}, \mathcal{D}_{train-w\_pkuc}, \mathcal{D}_{val}$:
        Forward$(x, y, Y)$;
    predict label $y_{pred}$ ;
    Calculate macro-averaged F1 score
until converged;
Select the best architecture based on macro-averaged
  F1 scores;
```

Therefore, we propose two searching strategies **Pseudo Auxiliary Searching (PAS)** and **VAE Contrastive Learning** to adapt the network architecture search to the open-set scenario.

Pseudo Auxiliary Searching To enable the network to obtain the ability to recognize unknown classes, we propose this strategy which could optimize the network to be aware of the risk from the unknown open space during each searching epoch updating. As illustrated in Algorithm 1, given a dataset with M known classes, labels for all these classes are embedded as a set of vectors Y, the distribution of which can guide the distribution of feature vectors for each known class. To avoid over-fitting, we first use two disjoint training sets $\mathcal{D}_{train\_p}$ and $\mathcal{D}_{train\_w}$ for network weights w and network architecture parameters $\{\alpha, \beta\}$ optimization respectively. Then for each training set, we pick several classes as pseudo KUCs, namely $D_{train\_p\_pkuc}$ and $D_{train\_w\_pkuc}$. The rest of original classes are denoted as $D_{train\_p\_kkc}$ and $D_{train\_w\_kkc}$. We do alternating optimization for w and $\{\alpha, \beta\}$. Once the optimization convergences, we decode the discrete network structures by finding a path with maximum probability. Noting that $D_{train\_p\_pkuc}$ and $D_{train\_w\_pkuc}$ are only available during searching the optimal architecture. We term this searching strategy for OSR as Pseudo Auxiliary Searching. By using the Pseudo Auxiliary Searching strategy, the architecture is able to perceive the signal from the pseudo unknown space and generalize to the open space.

VAE Contrastive Learning for robust class attributes By using VAE with class conditioned Gaussian prior distributions, a sample farther away from a class mean would have a decreasing probability of belonging to that class. By setting a threshold on class probabilities, the model is a CAP model and can reduce open space risk and recognize unknown class samples. However, VAE alone is not suitable for OSR, even with a classifier applied to feature vectors. In a closed-set scenario, VAE exploits both the class attributes and instance attributes together[1] to fully reconstruct the input image. However, in the open-set scenario, according to the Counterfactual Faithfulness theory (Yue et al. 2021), unseen cases would become more sensible when every attribute is disentangled. For example, the class attributes can be used as anchor information (Miller et al. 2021) to infer the unknown samples. Knowing instance attributes from class attributes would allow regularisation (Higgins et al. 2017; Suter et al. 2019) to prevent the model from inevitable mapping anything to the known idiosyncrasies. Therefore, we propose using VAE Contrastive Learning during each single searching step to guide the searching of our architecture to disentangling class attributes and instance attributes and fitting to the OSR while make our architecture a CAP model.

Assuming there are M known classes, given an input image x in the i-th known class, we use encoded instance attribute z and class mean vectors $CMVs = \{\mu_1, \mu_2, ..., \mu_m\}$ to generate images $X = \{\hat{x}_1, \hat{x}_2, ..., \hat{x}_m\}$. The formula to calculate the VAE contrastive loss!(Yue et al. 2021) is:

$$\mathcal{L}_{cr} = -log \frac{exp(-distance(x, \hat{x}_i))}{\sum_{j=1}^{M} exp(-distance(x, \hat{x}_j))} \quad (3)$$

where $distance$ is the $L2$ distance function.

Optimization

Objective function Our model is penalized by four loss functions. During training, we exploit Kullback–Leibler divergence ($\mathcal{L}_{kl}$) to force class attribute u to be close to the mean of the corresponding class. Reconstruction loss ($\mathcal{L}_{re}$) is calculated by the $L2$ distance between an input image x and the reconstructed image $\hat{x}$. We use a fully connect layer as a classifier on u and calculate cross-entropy loss ($\mathcal{L}_{ce}$). The last loss is the VAE contrastive loss. Details of obtaining u, z, $\hat{x}$, and generated images are shown in the appendix. The loss functions used in our model are summarized as follows:

$$\mathcal{L} = \gamma_{ce}\mathcal{L}_{ce} + \gamma_{kl}\mathcal{L}_{kl} + \gamma_{re}\mathcal{L}_{re} + \gamma_{cr}\mathcal{L}_{cr} \quad (4)$$

where $\gamma_{ce}, \gamma_{kl}, \gamma_{re}, \gamma_{cr}$ are coefficients of different losses. We fix the values of γ_{kl} and γ_{re}, and have only tried γ_{ce} and γ_{cr} from a small range of numbers.

Training procedure The training of the network includes two phases: the searching phase and the finetuning phase. Detailed steps of the searching phase are shown in Algorithm 1. After the optimization of w, α, and β converges, we determine the architecture structure by choosing the operation

[1] The class attributes are defined as the information required to distinguish a class. Any other variations in the image are considered as instance attributes.

with the highest α in each cell and paths with the highest probabilities based on β. The weight of the searched model is re-initialized and trained on the whole training set and optimized using the loss $\mathcal{L}$.

Testing procedure When training is completed, we model each class distribution $f_m(u) = \mathcal{N}(u; \mu_m, var_m)$ based on latent space representations of all correctly classified samples in the training set. For an image in the test set, we obtain u and z through our encoder. We generate M images using the same v and different μ_m, where m is the corresponding class. We calculate $L2$ distances, $Dist = \{dist_1, dist_2, dist_3, ..., dist_M\}$ between generated images and the original one. We choose a reconstruction error threshold that ensures 95% of training samples are predicted as known, and any image with $min(Dist)$ larger than the threshold is recognized as unknown.

After filtering test samples by reconstruction error threshold, we calculate the probabilities $P_M = \{p_1, p_2, p_3, ..., p_m\}$ a sample belongs to different classes based on the class attribute u. The probability a sample belongs to known class m is:

$$P_m(x) = 1 - \int_{-|v_0|}^{|v_0|} \int_{-|v_1|}^{|v_1|} ... \int_{-|v_r|}^{|v_r|} f_m(v) dv \tag{5}$$

where v_r is the r-th dimension of class attribute v (Yue et al. 2021). We classify samples with $min(P_M)$ smaller than a predefined threshold, as in the CAP model (Scheirer, Jain, and Boult 2014), as unknown. The rest of the samples are recognized as from known classes and their class labels are determined based on their softmax values from the classifier.

Experiments

For a fair comparison, we adopt the same dataset splitting protocols in (Oza and Patel 2019; Yue et al. 2021) to analyze the results. We follow the experimental settings and evaluation metrics in (Neal et al. 2018) and provide comparison results with other state-of-the-art OSR networks. We conduct ablation studies to demonstrate the effectiveness of each searching strategy.

Experimental Setup

Dataset We conduct the architecture search on CIFAR10 dataset (Krizhevsky and Hinton 2009). Once the search is completed, we adopt our searched best architecture to conduct evaluations on MNIST (LECUN 2012), SVHN (Netzer et al. 2011), CIFAR10, CIFARAdd10 (C+10), and CIFARAdd50 (C+50). Noting that we only perform the architecture search on CIFAR10 dataset (Krizhevsky and Hinton 2009), and fine-tune the searched architecture weights on other datasets. As defined in (Yue et al. 2021), the C+10 dataset is generated by choosing the four animal classes from CIFAR10 as KKCs and randomly sampling 10 non-animal classes as UUCs. C+50 follows the same procedure and contains 50 UUCs. We provide detailed dataset information in the appendix.

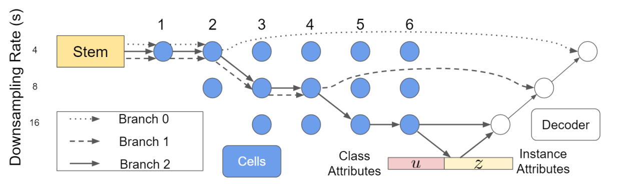

Figure 2: We show three branches with different downsampling rates in different lines from the searched architecture. The operations in branch 2 are shown in Table 3.

Implementation Our found architecture for OSR is shown in Figure 2 and its corresponding operations are displayed in Table 3. We perform the architecture search for a total of 60 epochs. The first 30 epochs are to pretrain the network weights w and the rest 30 epochs targets for alternatively updating α, β and w. For implementation details including data augmentation techniques, learning rates, optimizers, coefficients of different weights, and training times, please refer to the appendix.

Evaluation metrics Following the setting in (Neal et al. 2018), we evaluate all methods on three OSR metrics: 1) *Area Under the Receiver Operating Characteristics (AUROC)*; 2) *Macro-averaged F1 scores*; 3) *Openness-F1 plot*, which shows the macro-averaged F1 scores under various openness levels (Scheirer et al. 2013).

Comparison With Prior Arts

Architectures for comparison We compare our searched model with the following methods: (1) Softmax, a CNN model using a softmax layer with a cutoff threshold for unknown samples in its classifier; (2) OpenMax (Bendale and Boult 2016) using mean class vectors from the penultimate layer to calibrate predicted probabilities; (3) G-OpenMax using GAN to generate unknown samples to train the network; (4) OSRCI using GAN to synthesizes unknown samples close to known classes and treat unknown samples as an extra class (5) CROSR utilizing supervised prediction and unsupervised reconstruction of latent space representations (6) C2AE using extreme value theory (EVT) on reconstruction errors to get decision boundaries (7) CGDL (Sun et al. 2020) combining class conditional VAE with probabilistic ladder structure to extract high-level abstract features; (8) GCM-CF (Yue et al. 2021) using counterfactual images to disentangle features.

Performance We compare the proposed method with other relevant methods on metrics AUROC, macro F1 score, closed-set accuracy, and openness plot. As illustrated in Table 1, our model outperforms prior state-of-the-art networks on almost all datasets, especially for CIFAR10 with an improved margin of 12% on AUROC than the previous SOTA method GCM-CF. The performance drop on MINST can be explained by the domain gap between the searching dataset and the testing dataset. MNIST is a dataset that only contains black and white digit images, while our architecture is searched on CIFAR10 which includes full-colour RGB images and classes that are more divergent. Our searched architecture

Method	MNIST	SVHN	CIFAR10	C+10	C+50
SoftMax †	0.978	0.886	0.677	0.816	0.805
OpenMax † (Bendale and Boult 2016)	0.981	0.894	0.695	0.817	0.796
G-OpenMax † (Ge et al. 2017)	0.984	0.896	0.675	0.827	0.819
OSRCI † (Neal et al. 2018)	0.988	0.910	0.699	0.838	0.827
CROSR ‡ (Yoshihashi et al. 2019a)	**0.991**	0.899	-	-	-
C2AE § (Oza and Patel 2019)	-	0.892	0.711	0.810	0.803
CGDL ¶ (Sun et al. 2020)	0.977	0.896	0.681	0.794	0.794
GCM-CF (Yue et al. 2021)	0.830	0.708	0.720	0.815	0.817
Ours	0.961	**0.949**	**0.843**	**0.840**	**0.871**

Table 1: AUROC scores (the higher the better) on different dataset. † are provided by (Neal et al. 2018), ‡ from (Yoshihashi et al. 2019b), § from (Perera et al. 2020), ¶ from (Guo et al. 2021). We obtain values of GCM-CF by running the code of the original paper.

Method	MNIST	SVHN	CIFAR10	C+10	C+50
Softmax	76.82	76.16	70.39	77.82	65.96
OpenMax (Bendale and Boult 2016)	85.93	77.95	71.38	78.68	67.68
CGDL (Sun et al. 2020)	88.95	76.31	71.03	77.92	70.96
GCM-CF (Yue et al. 2021)	91.37	79.25	72.63	79.38	74.60
Ours	**92.04**	**82.49**	**75.29**	**80.96**	**76.71**

Table 2: Macro F1 score (the higher the better) comparisons on different datasets. We report the scores averaged over 6 experiments. Values other than the proposed method are taken from (Yue et al. 2021).

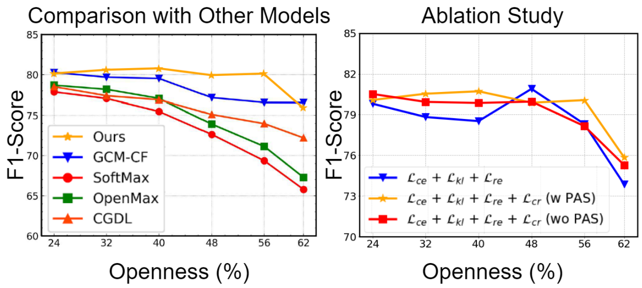

Figure 3: Openness plot. The left figure shows the macro-averaged F1 scores of our model and other models on various openness levels. For other models, we directly report the results in the original paper (Yue et al. 2021). The right figure shows F1 scores of our models with different searching strategies.

may be overcomplex for MNIST. As the reflection of network robustness, our model achieves the highest macro-averaged F1 scores on all datasets (Refer to Table 2).

According to the Macro F1 score shown in Table 2, our model improves macro F1 score by 0.67% on MNIST, 3.24% on SVHN, 2.66% on CIFAR10, 1.58% on C+10, and 2.11% on C+50. Moreover, the left image in Figure 3 shows the openness plot of our model against other methods. Our model has a superior performance over other methods at openness levels of 32, 40, 48, and 56. The previous SOTA method GCM-CF only achieves a better performance than our method by a small margin at openness levels of 24 and 62. To further ensure the improvement stems from known/unknown class classification, we also compare the closed-set accuracy of our model to that of a plain CNN (see Appendix). Overall, our searched model achieves significant results improvement on different datasets and is robust with different openness levels.

Ablation Study

In this section, we conduct three ablation studies on SVHN and CIFAR10 datasets, to demonstrate the effectiveness of proposed components including Pseudo Auxiliary Searching, VAE Contrastive Learning during the search phase, and different number of branches.

VAE Contrastive Learning We evaluate two architectures here: the architecture searched w/wo the VAE contrastive loss. As seen in Table 4, the results of found architecture with VAE contrastive loss supervision has a 1.26% increase in macro-averaged F1 score on SVHN. Also, in the right image of Figure 3, this model outperforms the other variant models at all openness levels except at 48. In Table 3, we find that the model searched with VAE contrastive loss has more convolutional layers. One explanation is without the proper guidance from the open-set oriented loss during searching, the training objective tends to find the best architecture towards the closed-set setting. As under the closed-set setting, a network does not need to consider the sufficient margins between KKCs and UUCs, so the searched architecture is oversimplified.

Psuedo auxiliary searching To analyze the effect of Pseudo Auxiliary Searching, we compare models searched

Method	Layer1	Layer2	Layer3	Layer4	Layer5	Layer6
$\mathcal{L}_{ce} + \mathcal{L}_{kl} + \mathcal{L}_{re}$	3×3 *conv*	3×3 *conv*×2	3×3 *conv*	3×3 *conv*	3×3 *conv*×2	3×3 *conv*×2
$\mathcal{L}_{ce} + \mathcal{L}_{kl} + \mathcal{L}_{re} + \mathcal{L}_{cr}$ (w PAS)	3×3 *conv*×2	3×3 *conv*	3×3 *conv*×2	3×3 *conv*	3×3 *conv*×2	3×3 *conv*×2
$\mathcal{L}_{ce} + \mathcal{L}_{kl} + \mathcal{L}_{re} + \mathcal{L}_{cr}$ (wo PAS)	3×3 *conv*×2	3×3 *conv*×2	3×3 *conv*	3×3 *conv*×2	3×3 *conv*×2	3×3 *conv*×2

Table 3: Searched model architectures. The network-level architecture is shown in Figure 2.

$\mathcal{L}_{cr}$	PAS	branch0	branch1	Params	F1
	✓			63.1M	81.76
✓				73.2M	79.72
✓	✓	✓		65.4M	80.44
✓	✓		✓	72.6M	82.30
✓	✓	✓	✓	73.7M	81.67
✓	✓			63.7M	83.02

Table 4: Ablation studies using different strategies. The parameters, FLOPS, and macro F1 scores are calculated on the SVHN dataset.

under the setting w/wo this strategy. We keep the principle that only KKCs are allowed for training, and pseudo KUCs are only accessible for calculating open-set evaluation metrics and selecting architectures. Thus there is an imbalance in the number of samples seen by the models during searching. We do not balance the number of training samples on purpose in the two settings because having fewer training samples is a trade-off for using Pseudo Auxiliary Searching. Despite the disadvantage, as seen in Table 4 and Figure 3, the model searched with PAS outperforms the one without on all datasets, except at an openness level of 5. In Table 3, the architecture searched without PAS is deeper than the one searched with PAS.

Number of branches We compare the architecture using only branch 2 to connect the encoder and the decoder with other architectures having more than one branch in the connection. In Table 4, the model using only branch 2 has the highest F1 score. We believe the reason is behind the logic of applying VAE contrastive loss. As the VAE contrastive loss is calculated by generating images using instance attributes and class attributes, it is necessary to control the information received by the decoder. Connecting several branches between the encoder and the decoder allows extra information about the original image to be transferred. This leads to instance attribute and class attribute fail to be disentangled and the generated counterfactual images being close to the original ones.

Discussion

Our searched network $vs.$ handcrafted network for OSR Sharing a similar mechanism of disentangling instance attributes and class attributes for OSR, our method is different from GCM-CF by adopting the VAE contrastive loss during the searching phase to guide the network architecture search. Also, we apply Pseudo Auxiliary Searching to select the best architecture among our dynamically changing architectures. With the searched architecture, our model has 78% fewer parameters and 42.7% fewer FLOPS than GCM-CF. At the same time, our model outperforms GCM-CF on AUROC and macro-averaged F1 scores on all datasets as illustrated in Table 5.

Method	Params	FLOPS	SVHN	CIFAR10	C+50
GCM-CF	291M	1.92G	79.25	72.63	74.60
Ours	64M	1.1G	83.02	75.87	75.58

Table 5: Our model $vs.$ GCM-CF. We compare the number of parameters, FLOPS, and macro F1 scores on three datasets.

Analysis of the found architecture We have several observations from the found architectures: 1) A suitable architecture for OSR needs a moderate depth to achieve good performance. As shown in Table 4, architectures with one more and one less convolutional layers than our found architecture have lower performance; 2) The architecture searched on one dataset is robust to be applied on other datasets when the domain gap is small, see Table 1 and Table 2; 3) Yet the found architecture shows the best improvement on the searching dataset 4) Limiting the information flow between the encoder and decoder is essential for OSR task. Even though skip connection is a widely-used layer in many SOTA architecture designs, the benefit may not apply to all imaging recognition tasks.

Conclusion

In this paper, we propose the first end-to-end hierarchical NAS framework for Open-set Recognition. In particular, our search framework incorporates Pseudo Auxiliary Searching strategy that enables itself to generalize to the open space and VAE Contrastive Learning strategy for robust class attributes. The network is a deep learning CAP model searched under the open-set setting. Our found network outperforms previous state-of-the-art OSR methods. In the future, we plan to extend the searching space and search the encoder and the decoder at the same time to further reduce the time consumption on human design. We will take advantage of the large amount of data in TinyImageNet (Le and Yang 2015) or ImageNet (Deng et al. 2009) to search for a more robust architecture for OSR in the future.

Acknowledgements

The study is partially supported by the Airdoc Monash donation grant.

References

Ardywibowo, R.; Boluki, S.; Gong, X.; Wang, Z.; and Qian, X. 2020. NADS: Neural Architecture Distribution Search for Uncertainty Awareness. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Bendale, A.; and Boult, T. 2016. Towards Open Set Deep Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Cevikalp, H.; and Serhan Yavuz, H. 2017. Fast and accurate face recognition with image sets.

Chen, W.; Gong, X.; Liu, X.; Zhang, Q.; Li, Y.; and Wang, Z. 2020. FasterSeg: Searching for Faster Real-time Semantic Segmentation. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Chen, W.; Gong, X.; and Wang, Z. 2021. Neural Architecture Search on ImageNet in Four GPU Hours: A Theoretically Inspired Perspective. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Chen, Y.; Meng, G.; Zhang, Q.; Xiang, S.; Huang, C.; Mu, L.; and Wang, X. 2019. RENAS: Reinforced Evolutionary Neural Architecture Search. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Deng, J.; Dong, W.; Socher, R.; Li, L.-J.; Li, K.; and Fei-Fei, L. 2009. Imagenet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Ge, Z.; Demyanov, S.; Chen, Z.; and Garnavi, R. 2017. Generative OpenMax for multi-class open set classification. In *Proceedings of the British Machine Vision Conference Proceedings (BMVC)*.

Guo, Y.; Camporese, G.; Yang, W.; Sperduti, A.; and Ballan, L. 2021. Conditional Variational Capsule Network for Open Set Recognition. *arXiv preprint arXiv:2104.09159*.

He, K.; Zhang, X.; Ren, S.; and Sun, J. 2016. Deep Residual Learning for Image Recognition. In *Proceedings of the IEEE Conference Computer Vision and Pattern Recognition (CVPR)*.

Higgins, I.; Matthey, L.; Pal, A.; Burgess, C.; Glorot, X.; Botvinick, M.; Mohamed, S.; and Lerchner, A. 2017. betavae: Learning basic visual concepts with a constrained variational framework. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Krizhevsky, A.; and Hinton, G. 2009. Learning multiple layers of features from tiny images. *Master's thesis, Department of Computer Science, University of Toronto*.

Krizhevsky, A.; Sutskever, I.; and Hinton, G. E. 2012. ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the Conference on Neural Information Processing Systems (NeurIPS)*.

Le, Y.; and Yang, X. 2015. Tiny ImageNet Visual Recognition Challenge. Available: http://tiny-imagenet.herokuapp.com.

LECUN, Y. 2012. THE MNIST DATABASE of handwritten digits. *http://yann.lecun.com/exdb/mnist/*.

Li, C.; Peng, J.; Yuan, L.; Wang, G.; Liang, X.; Lin, L.; and Chang, X. 2020. Block-wisely Supervised Neural Architecture Search with Knowledge Distillation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Liu, H.; Simonyan, K.; and Yang, Y. 2019. DARTS: Differentiable Architecture Search. *arXiv preprint arXiv:1806.09055*.

Mei, J.; Li, Y.; Lian, X.; Jin, X.; Yang, L.; Yuille, A.; and Yang, J. 2020. AtomNAS: Fine-Grained End-to-End Neural Architecture Search. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Miller, D.; Suenderhauf, N.; Milford, M.; and Dayoub, F. 2021. Class Anchor Clustering: A Loss for Distance-Based Open Set Recognition. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*.

Neal, L.; Olson, M.; Fern, X.; Wong, W.-K.; and Li, F. 2018. Open Set Learning with Counterfactual Images. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Netzer, Y.; Wang, T.; Coates, A.; Bissacco, A.; Wu, B.; and Ng, A. Y. 2011. Reading Digits in Natural Images with Unsupervised Feature Learning. *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*.

Oza, P.; and Patel, V. M. 2019. C2AE: Class Conditioned Auto-Encoder for Open-Set Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Perera, P.; Morariu, V. I.; Jain, R.; Manjunatha, V.; Wigington, C.; Ordonez, V.; and Patel, V. M. 2020. Generative-Discriminative Feature Representations for Open-Set Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Real, E.; Moore, S.; Selle, A.; Saxena, S.; Suematsu, Y. L.; Tan, J.; Le, Q. V.; and Kurakin, A. 2017. Large-Scale Evolution of Image Classifiers. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Ren, P.; Xiao, Y.; Chang, X.; Huang, P.-Y.; Li, Z.; Chen, X.; and Wang, X. 2020. A Comprehensive Survey of Neural Architecture Search: Challenges and Solutions. *arXiv preprint arXiv:2006.02903*.

Scheirer, W. J.; de Rezende Rocha, A.; Sapkota, A.; and Boult, T. E. 2013. Toward Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Scheirer, W. J.; Jain, L. P.; and Boult, T. E. 2014. Probability Models for Open Set Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*.

Simonyan, K.; and Zisserman, A. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *Proceedings of the International Conference on Learning Representations (ICLR)*.

Sun, L.; Yu, X.; Wang, L.; Sun, J.; Inakoshi, H.; Kobayashi, K.; and Kobashi, H. 2019. Automatic Neural Network Search Method for Open Set Recognition. In *IEEE International Conference on Image Processing (ICIP)*.

Sun, X.; Yang, Z.; Zhang, C.; Peng, G.; and Ling, K.-V. 2020. Conditional Gaussian Distribution Learning for Open Set Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Suter, R.; Miladinovic, D.; Schölkopf, B.; and Bauer, S. 2019. Robustly Disentangled Causal Mechanisms: Validating Deep Representations for Interventional Robustness. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Tan, M.; and Le, Q. V. 2019. EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks. In *Proceedings of the International Conference on Machine Learning (ICML)*.

Tian, Y.; Maicas, G.; Pu, L. Z. C. T.; Singh, R.; Verjans, J. W.; and Carneiro, G. 2020. Few-Shot Anomaly Detection for Polyp Frames from Colonoscopy.

Wang, X.; Xiong, X.; Neumann, M.; Piergiovanni, A.; Ryoo, M. S.; Angelova, A.; Kitani, K. M.; and Hua, W. 2020. AttentionNAS: Spatiotemporal Attention Cell Search for Video Classification. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

Wong, K.; Wang, S.; Ren, M.; Liang, M.; and Urtasun, R. 2020. Identifying unknown instances for autonomous driving. In *Conference on Robot Learning (CoRL)*.

Yao, Z.; Cao, Y.; Zheng, S.; Huang, G.; and Lin, S. 2021. Cross-Iteration Batch Normalization. *arXiv preprint arXiv:2002.05712*.

Yoshihashi, R.; Shao, W.; Kawakami, R.; You, S.; Iida, M.; and Naemura, T. 2019a. Classification-Reconstruction Learning for Open-Set Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yoshihashi, R.; Shao, W.; Kawakami, R.; You, S.; Iida, M.; and Naemura, T. 2019b. Classification-Reconstruction Learning for Open-Set Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Yue, Z.; Wang, T.; Zhang, H.; Sun, Q.; and Hua, X.-S. 2021. Counterfactual Zero-Shot and Open-Set Visual Recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Zoph, B.; and Le, Q. V. 2017. Neural Architecture Search with Reinforcement Learning. In *Proceedings of the International Conference on Learning Representations (ICLR)*.