SN Computer Science (2020) 1:77
https://doi.org/10.1007/s42979-020-0086-9

ORIGINAL RESEARCH

Deep Open Set Recognition Using Dynamic Intra-class Splitting

Patrick Schlachter[1] · **Yiwen Liao**[1] · **Bin Yang**[1]

Received: 19 December 2019 / Accepted: 27 February 2020 / Published online: 11 March 2020

Abstract
This paper provides a generic deep learning method to solve open set recognition problems. In open set recognition, only samples of a limited number of known classes are given for training. During inference, an open set recognizer must not only correctly classify samples from known classes, but also reject samples from unknown classes. Due to these specific requirements, conventional deep learning models that assume a closed set environment cannot be used. Therefore, special open set approaches were taken, including variants of support vector machines and generation-based state-of-the-art methods which model unknown classes by generated samples. In contrast, our proposed method models unknown classes by atypical subsets of training samples. The subsets are obtained through intra-class splitting (ICS). Based on a recently proposed two-stage algorithm using ICS, we propose a one-stage method based on alternating between ICS and the training of a deep neural network. Finally, several experiments were conducted to compare our proposed method with conventional and other state-of-the-art methods. The proposed method based on dynamic ICS showed a comparable or better performance than all considered existing methods regarding balanced accuracy.

Keywords Open set recognition · Dynamic intra-class splitting · Deep learning · End-to-end

Introduction

Over recent years, complex classification tasks such as natural image classification have been solved with high accuracy [33]. One major step toward a high classification accuracy was deep learning, since it enabled to train large models in an end-to-end manner without requiring manual feature engineering [16]. However, most of the research has been limited to closed set problems in which the number of occurring classes is known in advance. This is not the case in many real-world applications. For example, in face recognition, new faces may occur during inference, which were not known during training [8]. In such a case, open set recognition is necessary.

In open set recognition (OSR), samples of K known classes are given during training. During inference, samples of both K known and U unknown classes may occur. An open set recognizer is able to classify samples from known classes and to reject samples from unknown classes [27]. Figure 1 visualizes the difference between a closed set classifier and an open set recognizer.

Since its goal is closed and tight decision boundaries, OSR requires special methods. Conventional OSR methods are mostly based on decision scores obtained by closed set classifiers, such as support vector machines (SVMs) [34]. By selecting a threshold, they compare decision scores with this threshold to decide whether to reject a test sample. In contrast, state-of-the-art methods often utilize generative models to generate fake samples which model unknown classes. More details about related work are presented in the next section.

Recently, we proposed a novel approach toward OSR based on intra-class splitting (ICS) [30]. Its idea is to split given training samples into two subsets: typical and atypical samples. Then, the atypical samples are used to model unknown classes. This enables to transform a K-class open

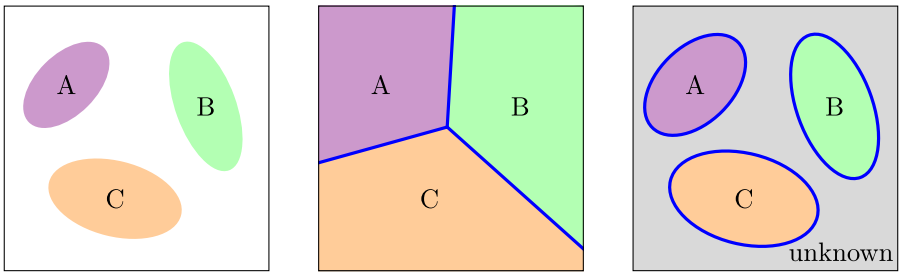
Patrick Schlachter and Yiwen Liao have contributed equally to this work.

✉ Patrick Schlachter
patrick.schlachter@fs-ing.de

Yiwen Liao
yiwen.liao@iss.uni-stuttgart.de

Bin Yang
bin.yang@iss.uni-stuttgart.de

[1] Institute of Signal Processing and System Theory, University of Stuttgart, Pfaffenwaldring 47, 70550 Stuttgart, Germany



(a) Training **(b)** Closed set classification **(c)** Open set recognition

Fig. 1 Exemplary comparison between closed set classification and open set recognition based on **a** three known classes A, B and C. **b** A closed set classifier can only learn decision boundaries that divide the feature space into three parts and thus cannot be used to detect unknown samples. **c** In contrast, in open set recognition, tight decision boundaries around the known classes are desired. Therefore, the gray space represents unknown classes

set problem into a $K+1$-class closed set problem as shown in Fig. 2. This transformation is also common for generation-based methods. However, the ICS-based method achieved better results. A possible reason is that atypical subsets of training samples are closer to real unknown classes compared to generated samples.

Although the ICS-based method showed a good performance, it requires two stages, the ICS and the training of an open set recognizer. As the splitting is so far done by training another model, the method requires the training of two different deep neural networks. Therefore, the training procedure of the first stage implicitly becomes a general hyperparameter, which is not user-friendly enough. Is it possible to combine both training steps?

In this paper, we answer this question by compressing the ICS-based method into a one-stage method. Furthermore, we provide an insight into open set recognition and the new method. To this end, many experiments on image datasets were conducted.

Related Work

Many previous studies summarize the algorithms toward OSR problems into two groups: conventional machine learning methods and deep learning methods. However, this grouping hides the key ideas of different approaches in solving OSR problems. Therefore, in this paper, we propose to group prior work into two categories: threshold-based methods and generation-based methods. In particular, threshold-based approaches use predicted scores or re-calibrated probabilities from a conventional classifier with a predefined threshold to reject samples from unknown classes. In contrast, generation-based methods try to model the unknown classes and therefore transform an OSR problem into a classification problem by discriminating generated samples from given known classes.

Threshold-based Open Set Recognizers Open set recognition was first formally formulated by Scheirer et al. [27]. In this work, a 1-vs-set support vector machine was proposed by adding an additional hyperplane between a learned decision boundary and non-matching data. Thus, samples located in the space between the two hyperplanes were considered to belong to unknown classes. However, this method was only a linear model and resulted in a loose decision boundary.

Afterward, Scheirer et al. proposed the Weibull support vector machine (*WSVM*) [28]. It utilizes decision scores obtained by a one-class support vector machine (OCSVM) [32] and a binary support vector machine [5] to fit a Weibull distribution [23]. Subsequently, the fitted cumulative density function was used to calibrate scores for

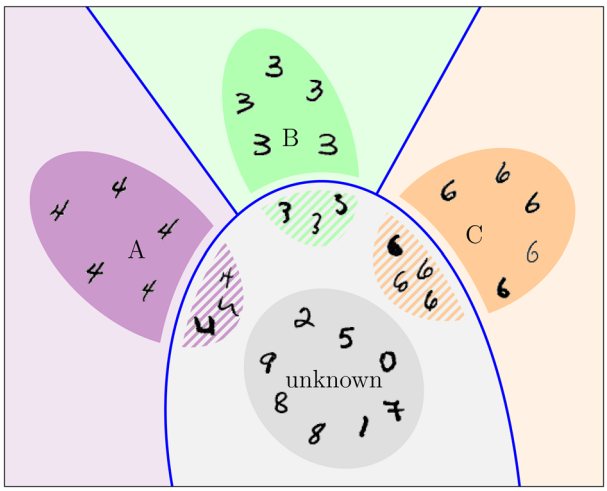


Fig. 2 Basic idea of the ICS method: Split given data from K known classes, here A, B and C, into typical and atypical subsets. By discriminating the K typical subsets and one additional class combining all atypical subsets from each other, a trained $K+1$-class classifier is expected to reject samples from unknown classes (gray) (color figure online)

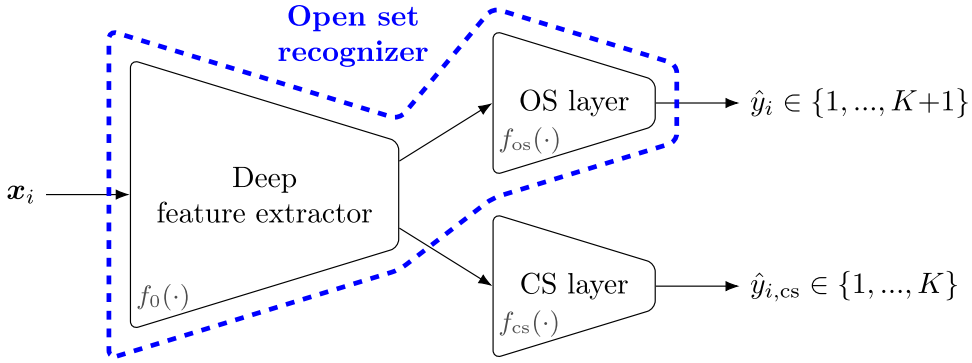


Fig. 3 The DICS network architecture consisting of a deep feature extractor, an open set (OS) layer with K+1 outputs and a closed set (CS) layer with K output neurons. During inference, the deep feature extractor and OS layer form the final open set recognizer

classification. Although this model could outperform previous approaches toward OSR problems, it was thresholded and thus had sensitive hyperparameters. Similarly, Jain et al. [11] proposed a P_I-*SVM* based on WSVM by introducing an automatic threshold estimation.

Moreover, Rudd et al. [24] proposed the extreme value machine (*EVM*) which used distances among training samples to fit a Weibull distribution. Given a new sample, an EVM estimates inclusion probabilities for each known class. According to a threshold, a sample is then either rejected or assigned to the class with the highest inclusion probability.

As the first deep-learning-based method, Bendale et al. [3] proposed *OpenMAX* to be used as an output layer in a deep neural network, enabling the rejection of unknown classes. However, it introduced three additional hyperparameters, which should be carefully selected based on unknown classes or validation sets. This is not always feasible in practice. Moreover, the OpenMAX layer was not used during training. In other words, the OpenMAX method can be considered as an extended EVM for a feature space learned by a deep neural network.

Generation-based Open Set Recognizers One recent generation-based method is the counterfactual image generation method (*CF*) proposed by Neal et al. [19]. It models unknown classes by generated samples which enable to reformulate an OSR problem into a classification task. Indeed, the authors first trained a modified Wasserstein-GAN [7, 18, 35] to obtain an encoder and a decoder. Then, the encoder transformed samples of known classes into latent representations. With a pretrained multi-class neural network, they sought latent representations in the feature space which were close to known classes but had low decision scores. Subsequently, these representations were decoded into counterfactual images. Accordingly, an OSR problem could be reformulated into a classification problem by discriminating among known classes and the counterfactual image class. Although this method achieved state-of-the-art performance in the literature, it suffers from common problems with GAN-based methods such as mode collapse. Furthermore, the numbers of training steps for a GAN and optimization steps for the counterfactual image generation are difficult to determine, because there are few quantity metrics to evaluate such generation qualities. Beyond that, there are other studies based on generation-based methods. For example, Jo et al. generated fake unknown data by modeling a noisy distribution in a latent space [12].

Intra-class Splitting Intra-class splitting (ICS) is a strategy to model unknown classes [30, 31]. More precisely, given training samples are split into typical and atypical subsets. The atypical subsets are then used to model unknown classes. This is because samples with less frequent patterns are often less important to model known classes and thus are not representative for them. For example, in the field of image classification, training a classifier on images with many redundant details may mislead the classifier and decrease the performance as shown in [29, 30]. Similarly, Li et al. [17] stated that misclassified samples from a training dataset can be considered as outliers or hard examples, which are not representative.

Proposed Method

Similar to our previous approach [30], the proposed method transforms a K-class open set problem into a K+1-class closed set problem. Thereby, intra-class splitting is used to find atypical samples which then serve as an additional class representing unknown classes. In addition, the proposed method shares the same neural network structure [30] as shown in Fig. 3. Equal to the previous approach, the closed set (CS) layer with K outputs is used as a closed set regularization to increase the closed set classification performance of atypical samples. During inference, only the combination of the deep feature extractor and the open set (OS) layer with K+1 outputs is used as an open set recognizer.

In contrast to the previous approach, intra-class splitting is now performed by directly using the outputs of the CS layer instead of training a separate neural network. Moreover, the splitting is not performed once before training, but dynamically epoch by epoch. We call this *dynamic*

intra-class splitting (DICS). Based on DICS, the original two-stage method is transformed into a one-stage method in which only one neural network must be trained. In the following, DICS and the training procedure are described in more detail.

Dynamic Intra-class Splitting (DICS)

Let $\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_N, y_N)\}$ denote a training dataset with N samples affiliated to K known classes. Correspondingly, each sample $\boldsymbol{x}_i \in \mathcal{X} = \{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$ has an individual class label $y_i \in \{1, 2, \ldots, K\}$. After the e-th training epoch, the score $s_i^{(e)}$ of an input sample $\boldsymbol{x}_i$ depends on its predicted class label $\hat{y}_{i,\text{cs}}^{(e)}$:

$$\forall \boldsymbol{x}_i \in \mathcal{X} : s_i^{(e)} = \begin{cases} \hat{P}(y_i|\boldsymbol{x}_i), & \text{if } \hat{y}_{i,\text{cs}}^{(e)} = y_i \\ 0, & \text{otherwise} \end{cases} \tag{1}$$

where $\hat{P}(y_i|\boldsymbol{x}_i)$ is the conditional class probability modeled by the closed set classifier, i.e., the concatenation of deep feature extractor and CS layer in Fig. 3. Note that a higher score means a more typical sample. Per e-th training epoch, scores for all training samples are collected as a score set $\mathcal{S}^{(e)} = \{s_1^{(e)}, s_2^{(e)}, \ldots, s_N^{(e)}\}$ for the e-th training epoch. Let $0 < \rho < 1$ be a predefined intra-class splitting ratio and $\mathcal{S}_\rho^{(e)}$ be the ρ-th fraction of $\mathcal{S}^{(e)}$ with the lowest scores. Then, $\tau_\rho^{(e)} = \max \mathcal{S}_\rho^{(e)}$ acts as a threshold between atypical and typical samples. Hence, the training dataset is split according to:

$$\forall \boldsymbol{x}_i \in \mathcal{X} : \boldsymbol{x}_i \in \begin{cases} \mathcal{X}_{\text{typical}}, & \text{if } s_i^{(e)} > \tau_\rho^{(e)} \\ \mathcal{X}_{\text{atypical}}, & \text{else} \end{cases} \tag{2}$$

Thereby, the goal of the scoring procedure is to find those samples which are either incorrectly classified or correctly classified but with a low confidence. As a result, ρ shows how many samples from known classes are allowed to be incorrectly rejected as unknown classes, similar to [31].

Training

Let the deep feature extractor from Fig. 3 be denoted as $f_0(\cdot)$, the OS layer be named $f_{\text{os}}(\cdot)$ and the CS layer be denoted as $f_{\text{cs}}(\cdot)$. Then, the resulting open set recognizer is defined as

$$f_{\text{osr}}(\cdot) = (f_{\text{os}} \circ f_0)(\cdot)\,, \tag{3}$$

while the conventional closed set regularization is denoted as

$$f_{\text{csr}}(\cdot) = (f_{\text{cs}} \circ f_0)(\cdot)\,. \tag{4}$$

Based on these definitions, the objective of the proposed method at each training epoch can be defined as:

$$\min_{f_0, f_{\text{os}}} \left(\mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\mathcal{L}_{\text{os}}(f_{\text{osr}}(\boldsymbol{x}), \zeta^{(e)}(\boldsymbol{x}) \cdot y)] + \lambda \cdot \mathbb{E}_{(\boldsymbol{x},y)\sim\mathcal{D}}[\mathcal{L}_{\text{cs}}(f_{\text{csr}}(\boldsymbol{x}), y)] \right), \tag{5}$$

where the OS loss $\mathcal{L}_{\text{os}}$ and CS loss $\mathcal{L}_{\text{cs}}$ are the learning objectives for regular K+1- and K-class classification problems, respectively. Moreover, the hyperparameter λ controls the trade-off between both losses.

In this work, the categorical entropy loss is used for both terms in the objective function. Note that $\zeta(\cdot)^{(e)}$ is an indicator function that returns 1 if a given sample is affiliated to the typical subset and otherwise returns 0. This means that typical samples maintain their original ground truths while atypical samples are assigned to a new label of zero during the optimization. The superscript (e) is used to emphasize that the outputs of $\zeta(\cdot)^{(e)}$ may change epoch by epoch because of the *dynamic* ICS.

Consequently, a minimization of the first term in Eq. 5 equals forcing the decision boundary to be between the typical and atypical samples, i.e., minimizing the open risk [27]. On the contrary, a minimization of the second term corresponds to minimizing the empirical risk on the training data from the known classes. Hence, the decision boundary is forced to enclosure the known classes.

Evaluation

Setup

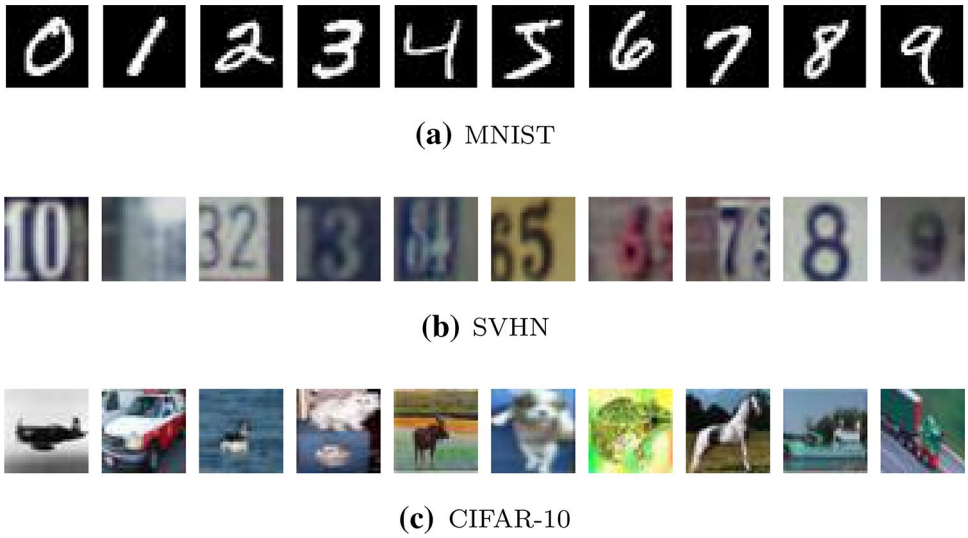
In an open set recognition scenario, K known classes and U unknown classes are present. During the conducted experiments, to be consistent with previous studies, K was equal to six while U varied depending on different datasets.

To evaluate the performance of an open set recognizer, the balanced accuracy (BACCU) [4] was used as the fundamental metric. In order to be consistent with prior work [19, 26–28], known classes were also denoted as positive classes, while unknown classes were considered as negative classes. Accordingly, BACCU is defined as

$$\text{BACCU} = \frac{1}{2} \cdot \left(\frac{TN}{\#\text{ negative samples}} + \frac{TP}{\#\text{ positive samples}} \right), \tag{6}$$

where TN (true negative) is the number of correctly rejected negative samples and TP (true positive) is the number of correctly classified positive samples. The BACCU gives the same weights to both rejecting negative samples and correctly classifying positive samples. Finally, in order to be consistent with prior work [19, 28], the area under curve (AUC) and closed set accuracy (CSACCU) were taken into consideration, too.

The backbone neural network architecture of the DICS method shared the same settings with [31]. The batch size



(a) MNIST

(b) SVHN

(c) CIFAR-10

Fig. 4 Exemplary images from the datasets

was set to 64 and the network was trained for 80 epochs in each experiment.

Baseline Methods

We selected seven baselines including state-of-the-art methods from the literature for comparison.

Multi-Class Neural Network with Rejection Option (CRO) A multi-class classifier was trained on the known classes in a closed set configuration. Then, a rejection threshold δ was selected by assuring up to 10% of the training samples were incorrectly rejected as unknown-class samples. During inference, samples with predicted scores lower than δ were rejected as from unknown classes. Note that this multi-class classifier shared the same architecture and hyperparameters as the proposed method.

Extreme Value Machine (EVM) EVM was implemented based on [24] with the default suggested hyperparameters. δ was set as 0.99 according to a grid search in the set $\{0.01, 0.05, 0.1, 0.5, 0.9, 0.99, 0.999\}$.

One-Class Support Vector Machine with Multi-Class Classifier (OCMC) An OCSVM [32] was trained with $\nu = 0.01$ and a kernel with radial basis function (RBF) [6] on the given known samples to reject unknown samples during inference. Then a multi-class classifier was trained on the known classes for a closed set prediction. Note that this multi-class classifier shared the same architecture and hyperparameters with the proposed method.

Weibull Support Vector Machine (WSVM) The OCSVM and binary SVM were implemented using [22]. Both SVMs utilized an RBF kernel. We selected the hyperparameters as follows: $\nu = 0.01$ for OCSVM, $C = 2, \gamma = 0.03125$ for the binary SVM as suggested in [28]. A Weibull distribution was fitted according to [26]. The decision thresholds were set as $\delta_\tau = 0.001, \delta_R = 0.5$.

OpenMAX We modified the codes from [3] as little as possible to satisfy our datasets. In order to have a fair comparison, the backbone network shared the same architecture of our method. As suggested in [3], we used $\alpha = 1, \eta = 20$ for all experiments.

Counterfactual Image Generation for OSR (CF) We translated the original codes [19] from PyTorch [21] into Keras [1] in order to maintain a consistent experimental environment for all baselines. All hyperparameters were maintained the same as in [19].

Intra-class Splitting (ICS) We kept all hyperparameters as in [30].

Datasets

We used three image datasets to validate the effectiveness of our method and to evaluate the sensitivity to the key hyperparameters. The first dataset MNIST [15] contains images of handwritten digits from 0 to 9 in gray scale. The number of training samples is around 6000 per class, while the number of test samples is around 1000 per class. The second dataset SVHN [20] consists of color digit images from 0 to 9 obtained in the real world. Thereby, most classes contain around 5000 training samples and 2000 test samples. The third dataset CIFAR-10 [13] is the most difficult considered dataset as it contains images of real-world objects such as airplanes, dogs and trucks. Each class in CIFAR-10 consists of 5000 training and 1000 testing samples. Figure 4 shows exemplary images from the three datasets.

Comparison

First, the proposed DICS method was compared to the other baselines on all three datasets. In each experiment on a dataset, 6 classes from the training set were randomly selected

Table 1 Results with performance metrics (std.) in %

Dataset	CRO	EVM	OCMC	WSVM	OpenMAX	CF	ICS	DICS
(a) BACCU								
MNIST	91.9 (± 1.0)	49.3 (± 1.6)	68.9 (± 0.6)	86.8 (± 2.6)	87.6 (± 0.2)	88.0 (± 0.5)	**94.4 (± 0.6)**	93.6 (± 1.2)
SVHN	78.8 (± 0.8)	44.5 (± 7.4)	58.1 (± 1.5)	63.7 (± 1.1)	76.1 (± 0.3)	76.2 (± 0.4)	82.1 (± 0.9)	**82.6 (± 0.7)**
CIFAR-10	65.0 (± 1.2)	47.3 (± 0.7)	48.4 (± 4.4)	53.3 (± 2.8)	54.9 (± 2.3)	52.7 (± 0.2)	72.0 (± 1.6)	**73.8 (± 0.9)**
(b) AUC								
MNIST	97.0 (± 1.1)	72.0 (± 5.5)	74.0 (± 2.2)	89.9 (± 5.1)	81.8 (± 0.7)	96.8 (± 0.2)	**98.3 (± 0.2)**	98.0 (± 0.8)
SVHN	88.0 (± 0.7)	61.3 (± 11.6)	71.6 (± 3.1)	76.4 (± 2.2)	80.6 (± 0.7)	79.7 (± 2.6)	**89.8 (± 0.9)**	**89.8 (± 0.7)**
CIFAR-10	74.0 (± 1.3)	55.4 (± 1.6)	64.7 (± 8.4)	66.7 (± 2.2)	61.1 (± 1.9)	71.7 (± 3.3)	80.2 (± 1.6)	**82.3 (± 2.1)**
(c) CSACCU								
MNIST	90.0 (± 0.8)	40.0 (± 1.3)	94.5 (± 0.0)	87.5 (± 2.0)	**98.3 (± 0.2)**	97.1 (± 0.0)	92.8 (± 0.5)	91.6 (± 0.3)
SVHN	70.5 (± 0.9)	34.2 (± 7.0)	86.1 (± 0.0)	65.5 (± 2.1)	**91.6 (± 1.4)**	79.4 (± 7.4)	82.2 (± 2.1)	84.2 (± 1.0)
CIFAR-10	63.3 (± 1.0)	38.1 (± 0.6)	79.0 (± 0.5)	53.1 (± 2.6)	**82.6 (± 1.2)**	76.8 (± 5.1)	76.2 (± 2.1)	74.8 (± 1.2)

as the known classes for training. Subsequently, we used all samples from the test set for the evaluation, i.e., 6 known classes and 4 unknown classes. The experiment was repeated five times for each dataset, and the results were reported by means and standard deviations (std.).

Table 1 shows the resulting BACCU on the three datasets mentioned above. DICS achieved a comparable or better performance than the original ICS method. On the datasets SVHN and CIFAR-10, the DICS method outperformed the original ICS. We argue that the DICS adds stochastic behavior to the selection of atypical samples. This means that at each epoch, a small number of typical samples were wrongly labeled as atypical, which leads to a higher robustness of the entire open set recognizer and thus a better performance on more complex datasets.

In an open set configuration, the AUC is a measure for the ability of an open set recognizer to correctly reject a sample from unknown classes based on manually selected thresholds. As shown in Table 1, the AUC shows a similar trend as the BACCU. Thereby, bold values mark the best result. Both ICS and DICS outperformed other considered baselines. Considering ICS and DICS, the dynamic splitting procedure seems to be more superior for complex datasets such as CIFAR-10.

The major weakness of both ICS and DICS is the performance regarding the closed set accuracy as shown in Table 1. In order to achieve a tight decision boundary, the key idea of all ICS-based methods is to use atypical samples to shrink the resulting decision boundaries. Therefore, samples from known classes that are located near the decision boundaries are sometimes rejected as unknown classes. Nevertheless, in practice, a slightly lower closed set accuracy is tolerable [2, 19].

Openness

Openness is used to describe how "open" an OSR problem is [28]. It is defined as:

$$\text{openness} = 1 - \sqrt{\frac{2 \cdot K}{K + C}} \in [0, 1) \,, \tag{7}$$

where K equals the number of known classes. Furthermore, $C = K + U$ where U is the number of unknown classes encountered during testing. The more unknown classes are encountered during inference, the more open an OSR problem is. Thereby, an OSR problem with a higher openness typically requires a more advanced OSR algorithm. In other words, an optimal OSR algorithm should have consistent performance over different openness.

In this work, we compared the performance of WSVM, CF, ICS and the proposed DICS method under different openness. These four methods were trained on six randomly selected classes from the dataset CIFAR-10. During

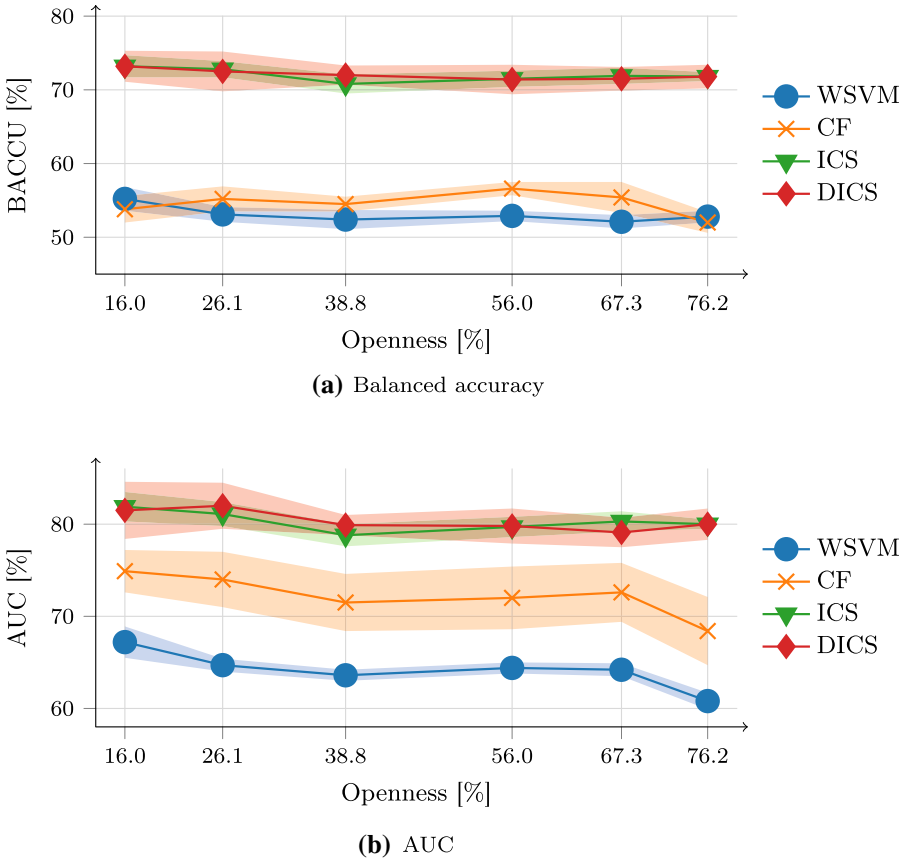


Fig. 5 Performance metrics over different openness

(a) Balanced accuracy

(b) AUC

inference, randomly selected images from the datasets CIFAR-100 [13] and Tiny ImageNet [25] were used as unknown classes. Thereby, seven different numbers of unknown classes were considered in this work: 5, 10, 20, 50, 75, 100 and 200.[1] The images for the last case were randomly sampled from Tiny ImageNet, while the images for the other six cases were randomly selected from CIFAR-100. Here we only report the resulting balanced accuracy in Fig. 5a and AUC in Fig. 5b, because the closed set accuracy did not change in respect of variable numbers of unknown classes.

The ICS and DICS methods outperformed the other two models regarding BACCU and AUC. Figure 5 shows the performance over different openness with corresponding standard deviations of the four methods. In average, the proposed DICS method achieved a balanced accuracy comparable to the original ICS method as depicted in Fig. 5a. As discussed before, this improvement is caused by the better robustness of DICS which was achieved by the dynamic splitting at each epoch.

Splitting Ratio

Similar to the original ICS method, the splitting ratio ρ plays a crucial role in the proposed DICS method. As shown in Fig. 6a, the balanced accuracy on all three datasets first increased and then decreased with ascending splitting ratios. Indeed, a small ρ such as $\rho = 1\%$, means that almost all training data maintain their original ground truths. Such a training procedure is similar to closed set classification. Therefore, the trained model cannot well reject samples from unknown classes. On the contrary, a large splitting ratio $\rho = 75\%$ means that the majority of the given training data has new labels differing from the original ground truths. In this case, the proposed closed set regularization cannot guarantee maintaining a high closed set accuracy. This comparison can also be observed by the closed set accuracy shown in Fig. 6c. The proposed DICS achieved a high closed set accuracy with a small splitting ratio and vice versa. Interestingly, as shown in Fig. 6b, AUC does not have much variance over different splitting ratios. In a wide range of splitting ratios, such as $1\% \leq \rho \leq 50\%$, the AUC is on a similar level.

[1] The corresponding openness values are: 16.0%, 26.1%, 38.8%, 56.0%, 62.9%, 67.3%, 76.2%.

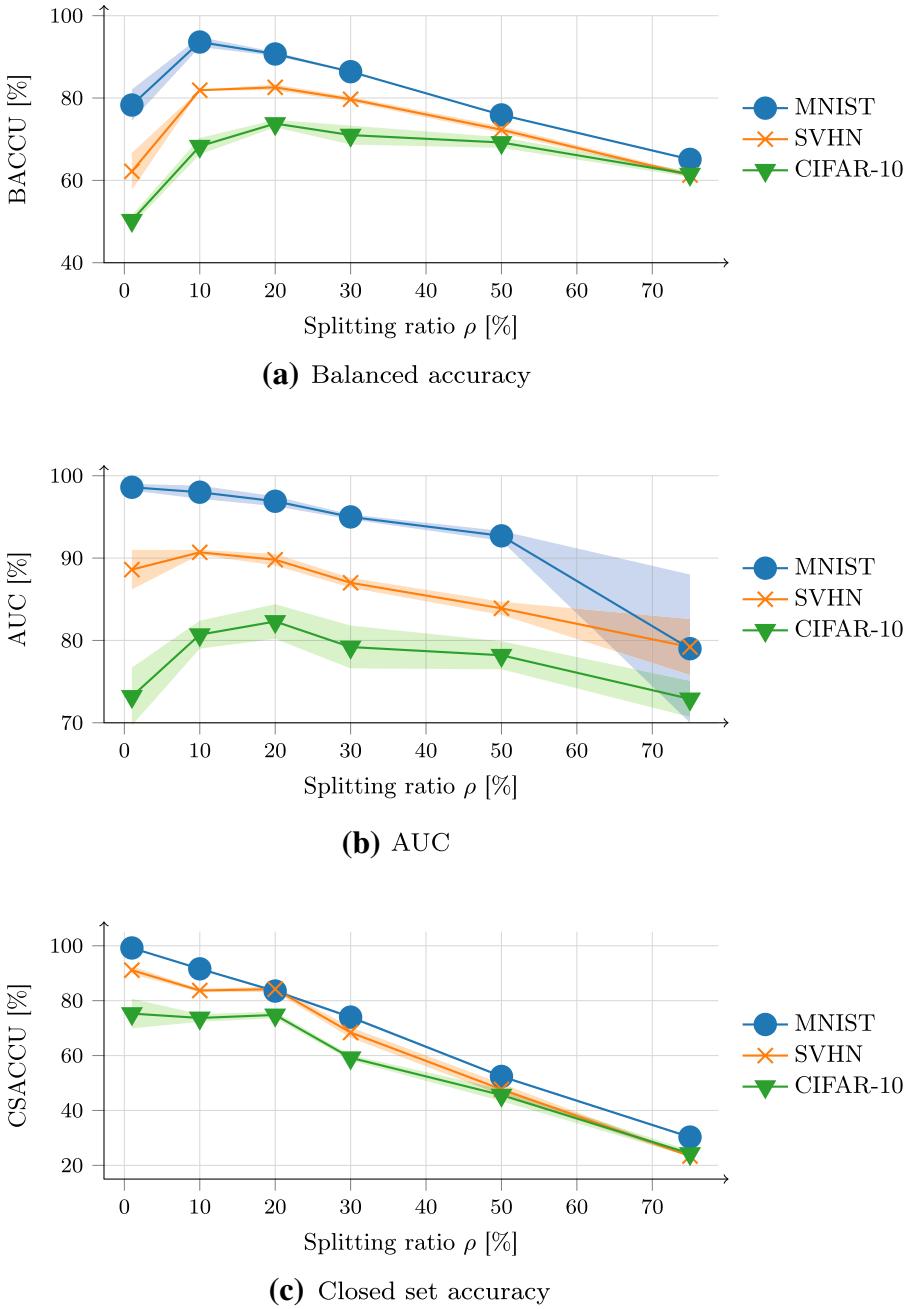


Fig. 6 Performance metrics over different splitting ratios

Initialization

In the DICS method, the given training data from the known classes are first split based on a randomly initialized network, meaning the deep feature extractor and CS layer. Hence, the splitting results in a first training epoch may not be reliable because the atypical samples are randomly selected at the beginning.

Therefore, the proposed method was evaluated under different network initializations to examine their impact. In this subsection, the proposed DICS method was tested on the datasets MNIST, SVHN and CIFAR-10 with a randomly selected combination of known classes. After fixing the known classes for training, we repeated the experiment for five times with randomly initialized network weights and reported the averaged results and their standard deviations.

Figure 7 shows the resulting performance. On the three datasets, the DICS method achieved a consistent performance for different initializations of the neural networks. In particular, DICS had a low standard deviation regarding BACCU and AUC. As discussed before, the introduction of dynamic splitting enabled a higher robustness to reject unknown classes. On the contrary, we noticed that CSACCU had a slightly higher standard deviation than BACCU. A

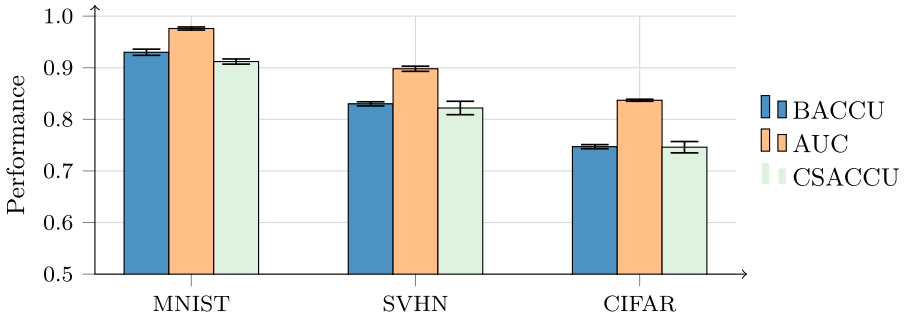


Fig. 7 Impact of different initializations

possible reason is that DICS forces the classifier to be confused with the true labels of some samples from the known classes for the early training epochs.

Discussion

Although one reason to develop a new method for OSR was the dependence of existing methods on hyperparameters, the proposed DICS method also depends on one crucial hyperparameter, the splitting ratio ρ. Why is this better than other methods?

In OSR, there is a trade-off between correctly rejecting unknown classes and identifying known classes. Hence, if no prior information at all about unknown classes is available, there must be at least one inevitable hyperparameter that sets the trade-off between the two contradictory objectives. Regarding the proposed DICS method, this hyperparameter corresponds to the splitting ratio ρ which sets the fraction of training samples to be considered as atypical. In practice, the inevitable hyperparameter in OSR can often be set based on given regulations or experience.

In contrast to the proposed method, existing methods do not only depend on one inevitable hyperparameter, but also on several algorithm-dependent hyperparameters. For example, WSVM depends on ν, δ_τ and δ_R [28]. Likewise, CF also has many algorithm-dependent hyperparameters. For example, CF utilizes the generated counterfactual images to represent unknown classes. Therefore, the quality of generated images plays a key role in this algorithm [19]. Accordingly, the entire training procedure can be considered as a general hyperparameter for CF, including the number of optimization steps and ratios among all losses.

In fact, such algorithm-dependent hyperparameters complicate the usage of open set recognizers, because the choice of these hyperparameters often requires a full understanding of the algorithm. Furthermore, algorithm-dependent hyperparameters are often sensitive and hard to fine-tune for new datasets or domains. Some of these hyperparameters even depend on the number of unknown classes, which is not feasible in practice. In contrast, the splitting ratio ρ of DICS is inevitable and easy to interpret.

Finally, it should be noted that the concrete network architectures are also hyperparameters in DICS. However, they can be considered as inevitable since they are common in almost all deep learning-based approaches.

Conclusion

We proposed a new method for open set recognition. By applying dynamic intra-class splitting (DICS), the method enables to use an arbitrary deep neural network as a one-stage end-to-end open set recognizer. Experiments on several image datasets showed the superiority over state-of-the-art methods regarding a compromise between closed set accuracy and rejection capability. In addition, the proposed method achieves a comparable or better performance than a former proposed two-stage method using ICS. Thereby, DICS still depends on a hyperparameter. However, we argue that this hyperparameter is inevitable and easier to choose than algorithm-dependent hyperparameters of existing methods due to its easy interpretability. The experiments indicated that DICS did not have the best closed set accuracy, although it might be tolerable in specific cases. Therefore, further research could focus on improving the closed set accuracy, for example by combining DICS with generative models or by choosing a more sophisticated network architecture.

Acknowledgments Open Access funding provided by Projekt DEAL.

Compliance with Ethical Standards

Conflict of interest On behalf of all authors, the corresponding author states that there is no conflict of interest.

Ethical approval This article does not contain any studies with human participants performed by any of the authors.

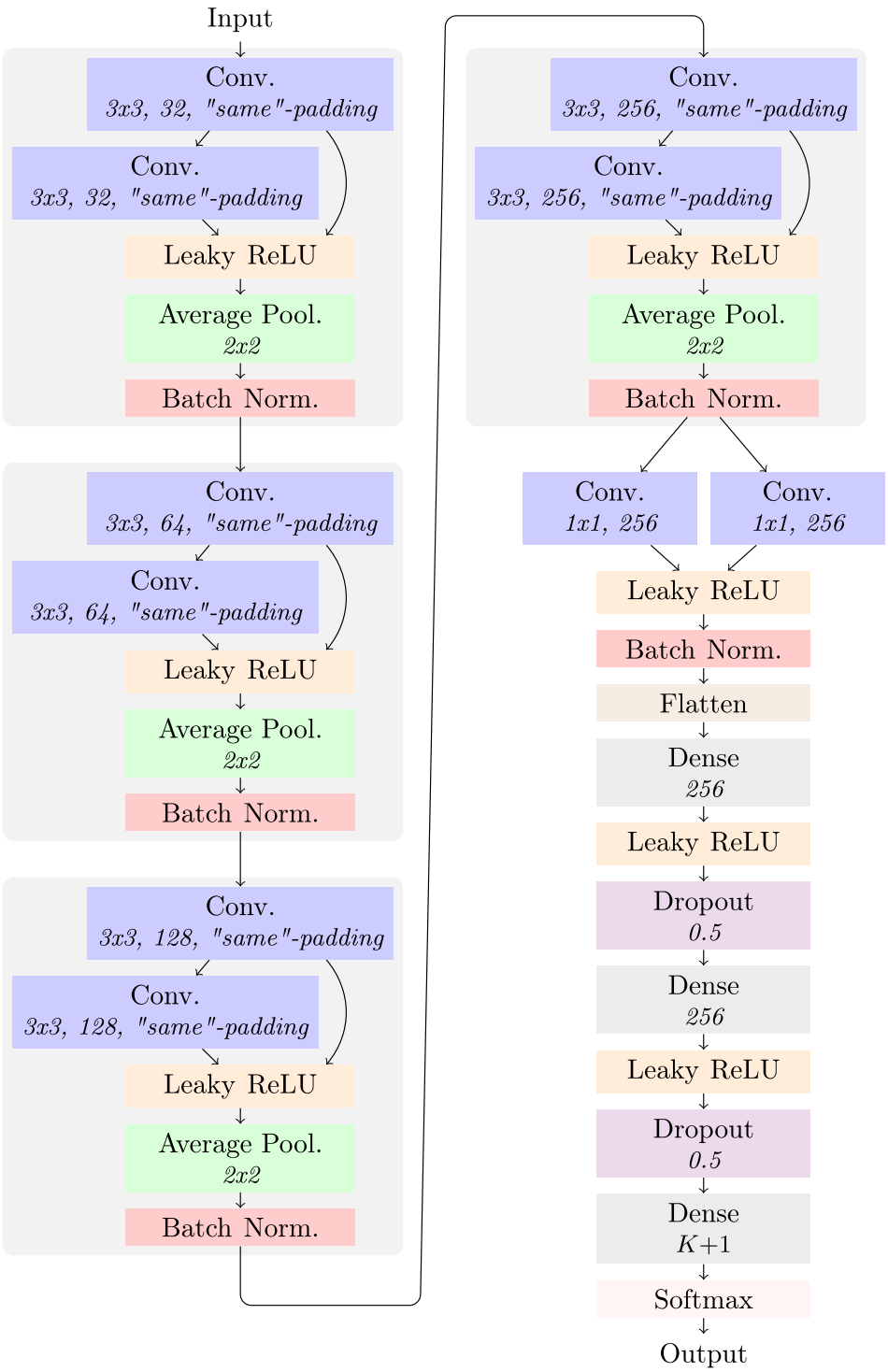


Fig. 8 Architecture of the proposed DICS method

Appendix 1: Architecture

The proposed method is a generic methodology toward OSR problems and not limited to a concrete network architecture. In this work, we implemented a VGG-like neural network [33] with residual blocks [9] to evaluate the proposed method. The detailed network architecture is presented in Fig. 8. It consists of four convolutional residual blocks, each with two convolutional layers ("Conv."), an activation layer with a leaky rectified linear unit (Leaky ReLU), an average pooling and a batch normalization layer [10]. Thereby, whenever two blocks

point into one block, a concatenation layer is used. At the end of the architecture, three dense layers with intermediate dropout layers [14] are utilized.

Appendix 2: Evaluation

The following tables contain the exact results for the corresponding figures in the main part of this paper (Tables 2, 3, 4, 5, 6 and 7). Thereby, bold values mark the best result.

Table 2 BACCU (std.) versus openness in %

Openness	WSVM	CF	ICS	DICS
16.0% (5)	55.2 (± 1.6)	53.8 (± 0.8)	**73.2 (± 1.5)**	**73.2 (± 2.1)**
26.1% (10)	53.1 (± 0.0)	55.2 (± 0.7)	**72.8 (± 1.1)**	72.5 (± 2.7)
38.8% (20)	52.4 (± 0.2)	54.5 (± 0.3)	70.8 (± 1.3)	**72.0 (± 1.3)**
56.0% (50)	52.9 (± 0.1)	56.6 (± 0.0)	**71.5 (± 1.1)**	71.4 (± 2.0)
67.3% (100)	52.1 (± 0.0)	55.4 (± 0.0)	71.9 (± 1.1)	71.5 (± 1.6)
76.2% (200)	52.8 (± 0.3)	52.0 (± 1.0)	**71.8 (± 0.6)**	**71.8 (± 1.6)**
Average	53.1 (± 1.0)	54.6 (± 1.4)	72.0 (± 0.8)	**72.1 (± 0.6)**

Table 3 AUC (std.) versus openness in %

Openness	WSVM	CF	ICS	DICS
16.0% (5)	67.2 (± 1.7)	74.9 (± 2.3)	**81.9 (± 1.6)**	81.5 (± 3.1)
26.1% (10)	64.7 (± 0.7)	74.0 (± 3.0)	81.1 (± 1.3)	**82.0 (± 2.5)**
38.8% (20)	63.6 (± 0.6)	71.5 (± 3.1)	78.8 (± 1.2)	**79.9 (± 1.1)**
56.0% (50)	64.4 (± 0.6)	72.0 (± 3.4)	79.7 (± 1.1)	**79.8 (± 1.9)**
67.3% (100)	64.2 (± 0.7)	72.6 (± 3.2)	**80.3 (± 1.1)**	79.1 (± 1.6)
76.2% (200)	60.8 (± 0.9)	68.4 (± 3.7)	**80.0 (± 0.5)**	**80.0 (± 1.7)**
Average	64.2 (± 1.9)	72.2 (± 2.1)	**80.3 (± 1.0)**	**80.4 (± 1.0)**

Table 4 BACCU (std.) versus splitting ratio ρ in %

ρ	MNIST	SVHN	CIFAR-10
1	78.3 (± 3.8)	62.2 (± 4.5)	50.3 (± 1.2)
10	93.6 (± 1.2)	81.9 (± 0.1)	68.3 (± 2.0)
20	90.7 (± 0.5)	82.6 (± 0.7)	73.8 (± 0.9)
30	86.4 (± 0.1)	79.7 (± 0.6)	71.0 (± 2.3)
50	75.9 (± 0.1)	72.3 (± 0.8)	69.2 (± 1.3)
75	65.1 (± 0.2)	61.3 (± 0.5)	61.5 (± 0.7)

Table 5 AUC (std.) versus splitting ratio ρ in %

ρ	MNIST	SVHN	CIFAR-10
1	98.6 (± 0.4)	88.6 (± 2.4)	73.2 (± 3.5)
10	98.0 (± 0.8)	90.7 (± 0.3)	80.7 (± 1.7)
20	96.9 (± 0.6)	89.8 (± 0.7)	82.3 (± 2.1)
30	95.0 (± 0.3)	87.0 (± 0.6)	79.2 (± 2.6)
50	92.7 (± 0.6)	83.9 (± 0.8)	78.2 (± 1.7)
75	79.0 (± 9.0)	79.2 (± 3.4)	72.9 (± 2.2)

Table 6 CSACCU (std.) versus splitting ratio ρ in %

ρ	MNIST	SVHN	CIFAR-10
1	99.2 (± 0.1)	91.1 (± 1.4)	75.3 (± 5.4)
10	91.6 (± 0.3)	83.7 (± 0.6)	73.7 (± 1.4)
20	83.5 (± 0.4)	84.2 (± 1.0)	74.8 (± 1.2)
30	74.0 (± 0.5)	68.4 (± 1.9)	59.2 (± 1.0)
50	52.4 (± 0.4)	47.7 (± 2.1)	45.6 (± 2.2)
75	30.3 (± 0.4)	23.4 (± 0.4)	24.3 (± 1.1)

Table 7 Impact of initialization

Datasets	BACCU (std.)	AUC (std.)	CSACCU (std.)
MNIST	93.0 (± 0.6)	97.6 (± 0.3)	91.2 (± 0.5)
SVHN	83.0 (± 0.4)	89.8 (± 0.5)	82.2 (± 1.3)
CIFAR-10	74.7 (± 0.4)	83.7 (± 0.2)	74.6 (± 1.1)

References

1. Abadi M, Barham P, Chen J, Chen Z, Davis A, Dean J, Devin M, Ghemawat S, Irving G, Isard M, et al. Tensorflow: a system for large-scale machine learning. In: Proceedings of the 12th USENIX conference on Operating Systems Design and Implementation, USENIX Association. 2016. pp. 265–83.
2. Bartler A, Mauch L, Yang B, Reuter M, Stoicescu L. Automated detection of solar cell defects with deep learning. In: 2018 26th European signal processing conference (EUSIPCO). 2018. pp. 2035–39. https://doi.org/10.23919/EUSIPCO.2018.8553025.
3. Bendale A, Boult TE. Towards open set deep networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. pp. 1563–72.
4. Brodersen KH, Ong CS, Stephan KE, Buhmann JM. The balanced accuracy and its posterior distribution. In: 2010 20th international conference on pattern recognition (ICPR). IEEE; 2010. pp. 3121–24.

5. Burges CJ. A tutorial on support vector machines for pattern recognition. Data Min Knowl Discov. 1998;2(2):121–67.
6. Cho Y, Saul LK. Kernel methods for deep learning. In: Advances in neural information processing systems. 2009. pp. 342–50.
7. Goodfellow I, Shlens J, Szegedy C. Explaining and harnessing adversarial examples. In: International conference on learning representations. 2015.
8. Günther M, Cruz S, Rudd E, Boult T. Toward open-set face recognition. 2017. https://doi.org/10.1109/CVPRW.2017.85.
9. He K, Zhang X, Ren S, Sun J. Deep residual learning for image recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. pp. 770–8.
10. Ioffe S, Szegedy C. Batch normalization: accelerating deep network training by reducing internal covariate shift. arXiv:150203167. 2015
11. Jain LP, Scheirer WJ, Boult TE. Multi-class open set recognition using probability of inclusion. In: European conference on computer vision. Springer; 2014. pp. 393–409.
12. Jo I, Kim J, Kang H, Kim YD, Choi S. Open set recognition by regularising classifier with fake data generated by generative adversarial networks. In: 2018 IEEE international conference on acoustics. IEEE: Speech and Signal Processing (ICASSP); 2018. pp. 2686–90.
13. Krizhevsky A. Learning multiple layers of features from tiny images. Technical report, Citeseer. 2009.
14. Krizhevsky A, Sutskever I, Hinton GE. Imagenet classification with deep convolutional neural networks. In: Advances in neural information processing systems. 2012. pp. 1097–105.
15. LeCun Y, Bottou L, Bengio Y, Haffner P. Gradient-based learning applied to document recognition. Proc IEEE. 1998;86(11):2278–324.
16. LeCun Y, Bengio Y, Hinton G. Deep learning. Nature. 2015;521(7553):436–44. https://doi.org/10.1038/nature14539.
17. Li B, Liu Y, Wang X. Gradient harmonized single-stage detector. arXiv:181105181. 2018.
18. Moosavi-Dezfooli SM, Fawzi A, Frossard P. DeepFool: a simple and accurate method to fool deep neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition. 2016. pp. 2574–82.
19. Neal L, Olson M, Fern X, Wong WK, Li F. Open set learning with counterfactual images. In: Proceedings of the European conference on computer vision (ECCV). 2018. pp. 613–28.
20. Netzer Y, Wang T, Coates A, Bissacco A, Wu B, Ng A. Reading digits in natural images with unsupervised feature learning. NIPS. 2011.
21. Paszke A, Gross S, Chintala S, Chanan G, Yang E, DeVito Z, Lin Z, Desmaison A, Antiga L, Lerer A. Automatic differentiation in PyTorch. In: NIPS-W. 2017.
22. Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E. Scikit-learn: machine learning in Python. J Mach Learn Res. 2011;12:2825–30.
23. Rinne H. The Weibull distribution: a handbook. Boca Raton: Chapman and Hall/CRC; 2008.
24. Rudd E, Jain LP, Scheirer WJ, Boult T. The extreme value machine. IEEE Trans Pattern Anal Mach Intell. 2017;40(3):762–8.
25. Russakovsky O, Deng J, Su H, Krause J, Satheesh S, Ma S, Huang Z, Karpathy A, Khosla A, Bernstein M, Berg AC, Fei-Fei L. ImageNet large scale visual recognition challenge. Int J Comput Vis. 2015;115(3):211–52. https://doi.org/10.1007/s11263-015-0816-y.
26. Scheirer WJ, Rocha A, Michaels R, Boult TE. Meta-recognition: the theory and practice of recognition score analysis. IEEE Trans Pattern Anal Mach Intell. 2011;33:1689–95.
27. Scheirer WJ, Rocha A, Sapkota A, Boult TE. Towards open set recognition. IEEE Trans Pattern Anal Mach Intell. 2013;35:1757–1772.
28. Scheirer WJ, Jain LP, Boult TE. Probability models for open set recognition. IEEE Trans Pattern Anal Mach Intell. 2014;36:2317–24.
29. Schlachter P, Liao Y, Yang B. One-class feature learning using intra-class splitting. CoRR arXiv:abs/1812.08468, 2018.
30. Schlachter P, Liao Y, Yang B. Deep one-class classification using intra-class splitting. In: IEEE Data Science Workshop (DSW). 2019.
31. Schlachter P, Liao Y, Yang B. Open set recognition using intra-class splitting. In: IEEE EUSIPCO. 2019.
32. Schölkopf B, Platt JC, Shawe-Taylor J, Smola AJ, Williamson RC. Estimating the support of a high-dimensional distribution. Neural Comput. 2001;13(7):1443–71.
33. Simonyan K, Zisserman A. Very deep convolutional networks for large-scale image recognition. In: International conference on learning representations. 2015.
34. Suykens JAK, Vandewalle J. Least squares support vector machine classifiers. Neural Process Lett. 1999;9(3):293–300. https://doi.org/10.1023/A:1018628609742.
35. Zhu JY, Park T, Isola P, Efros AA. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE international conference on computer vision. 2017. pp. 2223–32.