Open-Category Classification by Adversarial Sample Generation*

Yang Yu[1], **Wei-Yang Qu**[1], **Nan Li**[2], **Zimin Guo**[3†]
[1]National Key Laboratory for Novel Software Technology, Nanjing University, Nanjing, China
[1]Collaborative Innovation Center of Novel Software Technology and Industrialization, Nanjing, China
[2]Alibaba Group, Hangzhou, China
[3]College of Engineering, UC Berkeley
yuy@nju.edu.cn

Abstract

In real-world classification tasks, it is difficult to collect training samples from all possible categories of the environment. Therefore, when an instance of an unseen class appears in the prediction stage, a robust classifier should be able to tell that it is from an unseen class, instead of classifying it to be any known category. In this paper, adopting the idea of adversarial learning, we propose the ASG framework for open-category classification. ASG generates positive and negative samples of seen categories in the unsupervised manner via an adversarial learning strategy. With the generated samples, ASG then learns to tell seen from unseen in the supervised manner. Experiments performed on several datasets show the effectiveness of ASG.

1 Introduction

As machine learning techniques are adopted in increasing applications, it is appealing that they can be applied in environments that are open and non-stationary, where unseen situations can emerge unexpectedly. For classification, a typical learning task, classical methods implicitly assume that the data is i.i.d. even for the future test ones. This assumption no longer holds in open environments, which drastically weaken the robustness of classical classification methods.

In this work, we consider the open-category classification (OCC) problem, where there are novel classes that none of their instances were observed during the training phase, but in the test phase their instances could be encountered. Classical approaches can only predict instances from unseen classes as one of the seen classes. An open-environment aware classifier, on the contrary, should be able to tell at first if an instance belongs to a seen class.

Different directions have been explored related to the OCC problem. In class incremental-learning [Fink *et al.*, 2006; Muhlbaier *et al.*, 2009; Kuzborskij *et al.*, 2013], new classes are assumed to appear incremental. However, these studies mainly focused on how to enable the system to incorporate later coming training instances from new classes, but did not address the problem of recognizing unseen classes. In learning with a rejection option [Chow, 1970], the classifier rejects to recognize an instance if its confidence is low, which, however, does not take open-environment into consideration. Note that an instance close to the seen class boundary can have a low confidence but belongs to a seen class, while an unseen class instance can have a high confidence if it is far from the seen class boundary. Outlier detection techniques [Hodge and Austin, 2004] could be employed if we treat unseen class instances as outliers. However, samples from seen class could also contains outliers, meanwhile, a cluster of unseen class instances may not be treated as outliers. Zero-shot learning [Palatucci *et al.*, 2009] is close to the OCC problem, but they commonly assume that a high-level attribute set is available for all classes including the unseen one, which provides useful information for recognizing unseen class instances. In our problem, however, we consider learning in an unfamiliar open environment and thus do not assume the availability of such high-level attributes.

Only a few previous studies addressed the OCC problem. For examples, in [Scheirer *et al.*, 2013] the open-set cost is considered, but it is hard to measure that cost without seeing open-set data; in [Da *et al.*, 2014], a set of unlabeled data of all classes is employed, but sufficient unlabeled data from all potential classes may not always available. We are thus more interested in solving the problem without extra information.

In this paper, we propose the adversarial sample generation (ASG) framework for the OCC problem. Inspired by the adversarial learning [Goodfellow *et al.*, 2014], ASG generates negative instances of seen classes by finding data points that are close to the training instances, given that they can be separated from the seen data by a discriminator. When the training data is small, ASG also generates positive instances that cannot be discriminated from the seen class instances, in order to enlarge the dataset. Using the supervision of the generated negative and positive samples of seen classes, it is then straightforward to train an open-category classifier to tell seen from unseen. We conduct experiments on several domains with open categories but no extra training information. The results show that the ASG achieves significant better performance than the compared methods.

*This research was supported by the NSFC (61375061, 61333014), Jiangsu SF (BK20160066), and Foundation for the Author of National Excellent Doctoral Dissertation of China (201451).

†Part of the work was done when Z. Guo was visiting Nanjing University as an undergraduate student.

The rest of this paper is organized in four sections that presents the background, the proposed method, the experiment results, and the conclusion, respectively.

2 Background

2.1 Related Problems

This paper studies the open-category classification problem, where no information about the unseen classes, neither instances nor attributes, is available. Some studies are related to this problem.

The *incremental learning* requires a proper adaptation of traditional machine learning approaches to deal with the dynamic and open environment, in addition, class-incremental learning(C-IL) [Zhou and Chen, 2002] is an important branch of it, which mainly concerned with the addition of new classes. In [Fink *et al.*, 2006; Kuzborskij *et al.*, 2013], each new class has a binary classifier which distinguishes between existing categories and new categories, and the new category in the classifier shares the hypothesis with the existing category to train. However, the method still need a few instances of new categories, when no instances of new categories are available in the training phase, these methods can not be applied. To address this limitation, a new method is introduced in [Da *et al.*, 2014], which processes new categories by exploring unlabeled instances, this requires reliable unlabeled data, but the data availability and quality is difficult to guarantee in practice.

The *open set recognition* problem is mainly concerned by the community of pattern recognition, and has been applied to face recognition and speech recognition, etc. In [Phillips *et al.*, 2011], the main concern is the operation of the threshold, only instances where the confidence is above the threshold are classified as seen classes. In [Scheirer *et al.*, 2013; Bendale and Boult, 2015], both the new decision boundary and the risk over open space are considered to limit the regions for seen categories.

The *outlier detection* problem [Hodge and Austin, 2004] requires the identification of anomaly instances from a given data set. The outlier detection methods can be applied in open-category problems as long as the abnormal instances is predicted as novel classes. However, the difference is that outlier detection is only concerned with the discovery of abnormal instances, which is limited to specific applications. Besides, it does not take into account the classification error of existing categories

The *class discovery* problem tries to identify the instances of rare categories which are not known in advance, but are known to exist in the training data [Pelleg and Moore, 2005; Hospedales *et al.*, 2013]. The open-category problem is different from class discovery problem, because the unseen class is not necessarily a rare class. On the other hand, it is possible to find examples of a rare class in the training data, while the instance of novel classes only appear in the test data.

The *zero-shot* problem tries to identify the unseen classes with the assumption that some high-level attributes of all classes are known as a prior. For example, in [Xian *et al.*, 2016], a latent embedding model was presented which learns a compatibility function in the high-level attributes space considered between image and class embeddings. In this work, we do not assume the availability of such information.

2.2 Adversarial Learning

The adversarial learning employs a generative model and a discriminative model, where the generative model learns to generate instances that can fool the discriminative model as a non-generated instance, which is also called Turing Learning in [Li *et al.*, 2016]. Besides, the Generative adversarial nets (GAN) combines the two models as a whole neural network for end-to-end training, resulting in a model that is consistent with the original data distribution. Further improvements include studies on the stability of GAN. For example, in [Nowozin *et al.*, 2016], the F-divergence is introduced from the perspective of distance measurement, and GAN has been shown to be a special case of F-divergence when it comes to a particular metric.

In our work, we will employ the adversarial learning principle that a generative model fights to generate instances judged by a discriminator. Different with the studies on GAN, our framework can apply to various learning models besides neural networks. Moreover, to solve the open-category classification problem, we do not only need to generate seen class data, but more importantly need to generate unseen class data.

2.3 Derivative-free Optimization

Previously learning approaches commonly employed gradient-based optimization methods. However, the optimization problems may not always simple enough to fit the gradient-based methods. Often, a complex optimization has to be relaxed to a convex problem, sacrificing the faithfulness to the original problem.

Ancient derivative-free optimization methods include representatives such as genetic algorithms [Goldberg, 1989], which are mostly heuristic methods. Recently, the derivative-free optimization methods have made significant progress in both theoretical foundation and practical usage, including Bayesian optimization methods [Brochu *et al.*, 2010], optimistic optimization methods [Munos, 2014], and model-based optimization [Yu *et al.*, 2016]. A derivative-free optimization method considers an optimization formalized as $\arg\max_{x \in X} f(x)$, where X is domain. The method, instead of calculating gradients of f, samples solutions x and learns from their feedbacks $f(x)$ for finding better solutions. Therefore, derivative-free optimization methods can be more suitable for problem with bad mathematical properties, including non-convexity, non-differentiability, and having many local optima. We thus employ the state-of-the-art derivative-free methods in our approach.

3 Proposed Method

The open-category classification (OCC) problem can be described as follows. Given a training dataset $D = \{(x_i, y_i)\}_{i=1}^{L}$, where $x_i \in R^d$ is a training instance and $y_i \in Y = \{1, 2, ..., K\}$ is the corresponding category label. In the test phase, we need to predict the categories of an open dataset $D_o = \{(x_i, y_i)\}_{i=1}^{\infty}$, where $y_i \in Y_o = \{1, 2, ..., K, K +$

$1, ..., M\}$ with $M > K$. Since there are classes which are not observed in the training phase, the goal of OCC is to learn a model $f(x) : X \to Y' = \{1, 2, ..., K, novel\}$, where the option $novel$ indicates that the category was unseen in the training phase, so as to minimize the expected risk as follows

$$f^* = \arg\min_{f \in \mathcal{H}} \mathbb{E}_{(x,y)\sim D_o} err(y, f(x)) \quad (1)$$

where $\mathcal{H}$ is the hypnosis space and err is defined as follows

$$err(y, f(x)) = \begin{cases} I(f(x) \neq y), & y \in Y \\ I(f(x) \neq novel), & y \notin Y \end{cases} \quad (2)$$

The I(expression) is an indicator function, it equals 1 when the expression holds and 0 otherwise.

The OCC problem is difficult to solve because no information about the $novel$ class is available. Our overall idea is that, if we can generate the instances of the class $novel$ and put them into the training set (denoted the augmented training set as $\tilde{D}$), then the problem will be easily solved by standard supervised learning

$$f^* = \arg\min_{f \in \mathcal{H}} \mathbb{E}_{(x,y)\sim \tilde{D}} I(y \neq f(x)) \quad (3)$$

The label of instance x can be predicted as $\arg\max_{k=1,...,K,novel} f_k(x)$. The problem is then how to generate data of the $novel$ class.

The idea of adversarial learning [Goodfellow *et al.*, 2014] is employed for data generation. In the adversarial learning, a generative model is trained to generate samples that are thought to be appropriate according to a discriminator. For example, to generate data consisting with the training data, the objective is that the discriminator cannot distinguish the generated data from the training data. Using this idea, we directly generate the data of unseen class for the OCC problem.

Generate Unseen Class Instances

To generate an instance of the unseen classes, ASG searches in the instance space, such that the instance should not be recognized as the seen class by the discriminator, which is a classifier trained to separate generated samples and seen class instances. However, there are too many such instances. For the purpose of distinguishing seen from unseen, we only need the samples that are around the boundary between seen and unseen classes. Therefore, ASG tries to find an instance that is close to the seen class instances, but is recognized as unseen class by the discriminator.

ASG considers each class separately, and for each class, it generates samples one by one. Let $P_D(x; D, D^-)$ denote the probability of x to be positive by the discriminator, trained with positive data D and negative data D^-. For class k, denote the current generated samples as D_k^-, which is empty initially. The objective that a generated sample does not belong to the seen class is

$$\arg\min_x P_D(x; D_k, D_k^- \cup \{x\}) \quad (4)$$

Intuitively, we evaluate a generated sample by adding it to the negative data set and train the model to see if the generated sample is not classified as positive (seen class).

Eq.(4) alone cannot generate all the boundary samples of the seen class, but only data samples that do not belong to the seen class. To generate boundary samples, we further require that the generated samples are close to the seen class data. Therefore, it is natural to consider that the distance between the generated sample and the original data set D_k should be small in a measure, which is enforced by the penalty term as

$$P_1(x, D_k) = \max\{0, \arg\min_{x' \in D_k} dist(x, x') - C_1\} \quad (5)$$

where the radius parameter C_1 is a positive constant, and $dist(x, x')$ is a distance measure that can be Euclidean distance or other distance measures.

Furthermore, since ASG generates samples one by one, Eq.(4) and Eq.(5) cannot prevent it from generating many identical samples. However, we want the generated samples to be scattered around the boundary. Therefore, we force the generated samples to be different, i.e., a newly generated sample should be far away from the previously generated samples, which is expressed as

$$P_2(x, D_k^-) = \max\{0, C_2 - \arg\min_{x' \in D_k^-} dist(x, x')\} \quad (6)$$

Here the radius parameter C_2 is a positive constant.

Combining the loss and the penalty terms, the overall objective function is

$$\arg\min_x P_D(x; D_k, D_k^- \cup \{x\}) + \lambda_1 P_1(x, D_k) + \lambda_2 P_2(x, D_k^-) \quad (7)$$

where λ_1 and λ_2 are hyper-parameters of the penalty terms. Note that our penalty terms aims only at are pushing samples out of the radius distance, the hyper-parameters will not have a great impact on the learning result.

Note that Eq.(7) is non-convex, particularly when models other than neural networks are considered. It is hard to rely on the gradient-based method. Thus, we employ the derivative-free method to solve the optimization. Since most derivative-free methods are generally applicable, we denote such an algorithm as Opt in Algorithm 1. The concrete algorithm will be disclosed in the experiment section.

Algorithm 1 shows the overall procedure of generating negative instances. The algorithm takes input of labeled training dataset D and parameter T to indicate the number of instances to be generated per class. For each class

Algorithm 1 Generation of negative instances of seen classes

Input:
D: Training instances$\{(x_i, y_i)\}_{i=1}^L$
T: Number of generated instances per class
$\mathcal{L}$: Learning algorithm for the discriminant model
Opt: A derivative-free optimization method
Output:
D^-: Negative samples of all class $1, 2, ..K$

```
for each k ∈ {1, ..., K} do
    D_k^- = ∅
    for t = 1, 2, ..., T do
        x^- = solve Eq.(7) by Opt with discriminator L
        Update D_k^- = D_k^- ∪ {x^-}
    end for
end for
return D^- = {D_1^-, D_2^-, ..., D_K^-}
```

$k \in 1, 2, ..., K$, it generates a set of instances, D_k^-, in turn, and the instances are obtained by optimizing Eq.(7).

Generate Seen Class Instances

On the other hand, if the class k is a rare class that has only a small number of samples, the trained classification model may be inaccurate. Our sample generation method can also be used to generate positive/seen class data, in order to improve the prediction accuracy. We only need to change the optimization function in step 6 of Algorithm 1 to complete this goal.

For the goal of generating a seen class instance, the loss is

$$\arg\max_x P_D(x; D_k, D_k^+ \cup \{x\}) \quad (8)$$

where D_k^+ is the previously generated positive data set. Again, we want to generate scattered samples, thus we force the generated instance to be distant to the previously generated ones, by the penalty:

$$P_3(x, D_k^+) = \max\{0, C_3 - \arg\min_{x' \in D_k^+} dist(x, x')\} \quad (9)$$

where the $dist$ is a distance measure function, and C_3 is a positive constant. Then the overall objective is:

$$\arg\max_x P_D(x; D_k, D_k^+ \cup \{x\}) - \eta P_3(x, D_k^+) \quad (10)$$

where η is the coefficient of regular entry P_3. Let the Eq.(10) replace the Eq.(7) in the step 6 of Algorithm 1, the D_k^+ set can be obtained for each class k.

Overall Procedure

After the data of unseen and seen classes are generated, it is straightforward to train a classifier for distinguishing between them. In ASG, we prefer to train such classifier for each class, i.e., train f_k^{ooc} from $D_k \cup D_k^+$ and D_k^-.

Another issue needs to be considered is the learning capacity of the discriminator $\mathcal{L}$. Note that when the capacity is too high, every generated sample can be discriminated from the original data; while when the capacity is too low, the boundary of the seen classes cannot be well captured. Unlike unsupervised learning, in the OCC problem we have some data of the seen classes. Using this data, we can fine tune the hyper-parameter (e.g. the structure of a neural network) of the learning algorithm for a proper capacity.

Overall, given any learning algorithm $\mathcal{L}$, the procedure of the ASG framework is as follows:

(1) On the seen data D, fine tune the hyper-parameters of $\mathcal{L}$;

(2) Generate negative instances D^- of the seen classes by Algorithm 1, as well as positive instances D^+ if necessary;

(3) For each class k, train a classifier f_k^{ooc} from positive data $D_k \cup D_k^+$ and negative data D_k^- by $\mathcal{L}$;

In the prediction stage, for a test instance x, it is tested by each of f_k^{ooc}. If all f_k^{ooc} classify x as negative, then x is predicted as a $novel$ class. Otherwise, x is predicted as the class k with f_k^{ooc} has the highest confident.

4 Experiments

4.1 Comparison Methods

As discussed above, the hyper-parameters of the learning algorithm should be determined at the first step. In the experiment, we employ SVM with RBF kernel as the learning algorithm and for the discriminant model too. And the binary classifier trained after the generation is also set as RBF-SVM. In order to verify the validity of the ASG framework, we conducted experiments on several benchmark datasets, comparing with:

OC-SVM: One-Class SVM [Schölkopf *et al.*, 2001] is a state-of-the-art outlier detector [Ma and Perkins, 2003], which computes a binary function supposed to capture regions in the input space where the probability density lives.

MOC-SVM: The OC-SVM can hardly find local outliers since it seeks for a hyperplane to separate the data and the origin in essence. Therefore, for this comparison method, we train multiple one-class SVMs, one OC-SVM for each seen class in the training set, for outlier detection.

1-vs-Set: 1-vs-Set Machine [Scheirer *et al.*, 2013] introduces extra decision boundaries for the seen classes in order to minimize the risk over open space.

OVR-SVM: One-vs-rest SVM is a powerful multi-class classification scheme [Rifkin and Klautau, 2004]. In the original OVR-SVM, an instance x is predicted as category y, where $y = \arg\max_{k=1,...,K} f_k(x)$ where f_k is a binary SVM trained for class k. In order to adapt OVR-SVM for the prediction of open-category problem, we let the x be predicted as class y only when $max_k f(x) > 0$, otherwise x is considered to be the instance of $novel$ class.

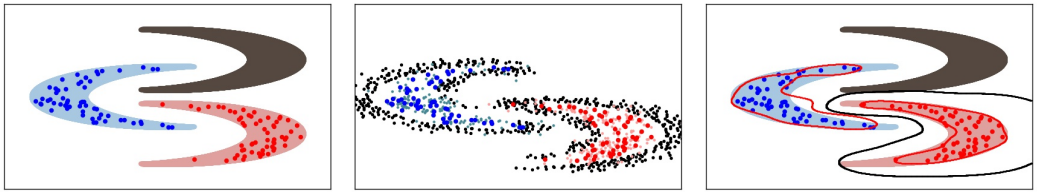
NNO: Nearest Non-Outlier(NNO) [Bendale and Boult, 2015] is a theoretical guaranteed method which deal with the open world recognition by introduce a measurable function in Nearest Class Mean method.

In experiments, we use the implementations of OC-SVM, MOC-SVM and OVR-SVM in the LIBSVM software [Chang and Lin, 2011], and the implementation of 1-vs-Set Machine from the code released by the authors. And we implement the NNO by ourselves due to the miss of the code. The coefficient C in SVM is determined by cross validation on the training dataset using the original OVR-SVM. The width of Gaussian kernel γ is fixed to $1/d$, where d is the size of the feature. For the ASG algorithm, the distance measure $dist$ is set to be the Euclidean distance. When generating negative samples of class k, the parameter C_1,C_2 is set as $\min_{x_1,x_2 \in D_k; x_1 \neq x_2} dist(x_1, x_2)$, λ_1 and λ_2 are both set to 0.1, and $T = 200$. When generating positive samples for class k, the parameter C_3 is also set to $\min_{x_1,x_2 \in D_k; x_1 \neq x_2} dist(x_1, x_2)$, η is set to 0.3, and also $T = 200$. For the derivative-free optimization method, we employ the recently developed RACOS algorithm[1] [Yu *et al.*, 2016] with default parameters.

4.2 Results

Three Moons: We first illustrate the behaviors of ASG using a 2-dimensional synthetic data as in Figure 1. The dataset consists of three classes, but only two of them are seen in the training stage, as in Figure 1 (a). With the generated data, the boundary of the seen classes can be well characterized, as in Figure 1 (b). The decision boundaries in Figure 1 (c) show that ASG can correctly exclude the unseen moon as seen data, while the classical SVM will classify the unseen moon as blue.

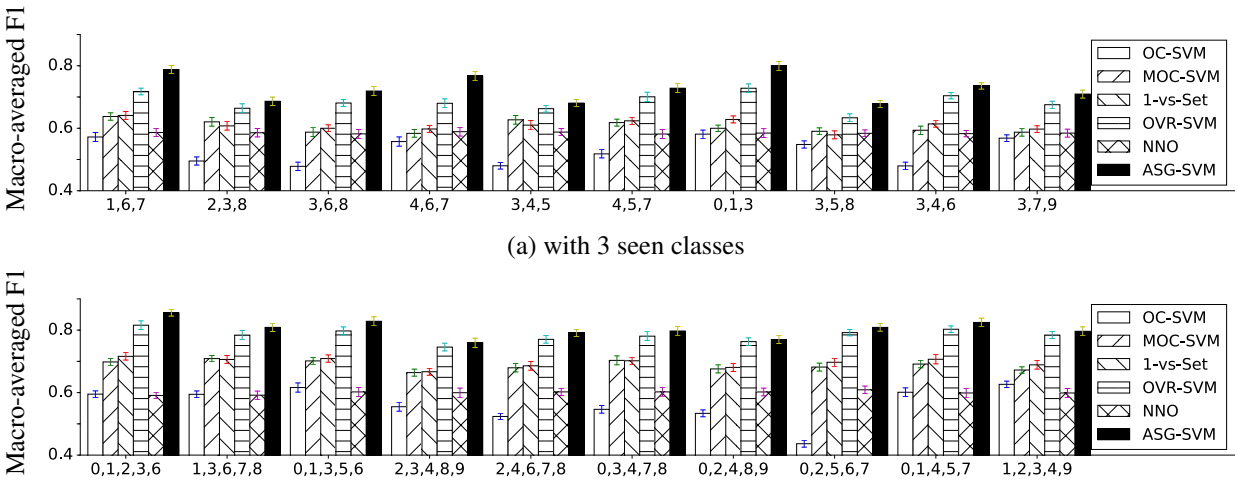
[1] using the implementation in https://github.com/eyounx/ZOOpt



(a) true classes and training data (b) original and generated data (c) decision boundaries

Figure 1: An illustration of ASG framework on the synthetic data. (a) shows the origin distribution of the three moons, where the blue and red moons are seen classes and the gray moon is unseen, and the dots are the training instances; (b) shows the generated data of seen classes (small colored dots) as well as that of unseen classes (gray dots); (c) shows the classification boundaries of a well tuned SVM (in grey line) and the ASG (in red lines).

Handwritten Digit Image Classification: We conducted the second experiment on the MNIST handwritten digit dataset. First, seen categories were randomly selected from 10 total categories and the test data contains all 10 categories. The number of training data, generated data and test data are 100, 300 and 100 for each class. We have 10 random selections for the seen categories and repeat 10 times for each selection, both mean and standard deviation are recorded. Figure 2 shows the results on 3 and 5 seen classes in training. It can be observed that ASG-SVM achieves the best performance on both configurations, while the OVR-SVM is the runner up. Besides, the 1-vs-set machine shows to outperform the OC-SVM and MOC-SVM, and the outlier detection methods, OC-SVM and MOC-SVM, produce the worst per-



(a) with 3 seen classes

(b) with 5 seen classes

Figure 2: Comparisons of different methods on MNIST dataset

Table 1: F1, Precesion and Recall for seen classes 2,3,5 in MNIST dataset (NIPC means number of instances per class)

	Measure	NIPC	OC-SVM	MOC-SVM	1-vs-Set	OVR-SVM	NNO	ASG-SVM
Seen Class	F1	100	.398±.033	.446±.028	.488±.046	.552±.021	.424±.013	**.570±.029**
		1000	.389±.026	.437±.027	.542±.043	.612±.013	.421±.010	**.624±.024**
	Precision	100	.330±.037	.414±.033	.336±.051	.398±.019	.367±.012	**.539±.027**
		1000	.314±.027	.374±.031	.388±.048	.534±.017	.379±.011	**.566±.022**
	Recall	100	.502±.031	.484±.030	.882±.046	**.898±.024**	.501±.014	.605±.031
		1000	.511±.029	.525±.027	**.897±.044**	.716±.014	.474±.012	.697±.025
Unseen Class	F1	100	.645±.035	.918±.031	.882±.048	.895±.023	.702±.012	**.933±.030**
		1000	.617±.031	.905±.029	.897±.044	.930±.016	.722±.011	**.935±.026**
	Precision	100	.736±.036	.941±.034	.983±.049	**.987±.020**	.763±.014	.957±.028
		1000	.726±.027	.944±.033	**.986±.043**	.968±.018	.763±.012	.966±.024
	Recall	100	.577±.032	.897±.032	.752±.053	.818±.025	.650±.013	**.911±.034**
		1000	.538±.030	.870±.028	.789±.048	.896±.013	.685±0.01	**.907±.026**

Table 2: Improvement ratio of ASG-SVM to the comparing methods on MNIST dataset (NIPC means number of instances per class)

#classes -NIPC	Comparison Methods				
	OC-SVM	MOC-SVM	1-vs-set	OVR-SVM	NNO
3-1000	.2391	.1720	.1782	.0624	.1624
3-100	.3816	.2061	.1957	.0650	.2469
5-1000	.3995	.0755	.1016	.0108	.3877
5-100	.4283	.1691	.1552	.0259	.3403

formance. Besides, the NNO seems not perform well in this dataset.

To break down the performance to seen and unseen data separately, we set the class $2, 3, 5$ as seen categories, and training data size for each category is set to 100 and 1000, respectively. The experiments are repeated for 10 times, and both mean and standard variance of the measure are reported. The results of F1, precision and recall for both seen classes and unseen classes are shown in Table 1. For seen classes, the ASG-SVM gets the best performance on F1 and precision when the instance number for each category is set to 100 and 1000, while the recall of OVR-SVM and 1-vs-set is the highest when the number of instance is set to 100 and 1000. For unseen classes, the ASG-SVM obtains the highest performance on F1 and recall, while the precision of OVR-SVM and 1-vs-set is the best when the instance number of each class is 100 and 1000.

Table 2 shows the improvement ratio of ASG-SVM over the comparing methods in 4 configurations of the number of seen classes and training set size. The cases in the first column represent different situations, for example, 3-100 represent there are 3 seen classes in the training data and 100 instances for each class. It can be observed that, when the training data has a smaller size, ASG-SVM has larger improvements. This mainly due to that ASG also generates samples of seen classes to improve the prediction accuracy.

Document Classification: In the third experiment, we conduct experiments on the 20 Newsgroups dataset, which is a popular text dataset consists of documents from 20 different topics. Note that this data set has some topics that are highly similar, such as $comp.sys.ibm.pc.hardware$ and $comp.sys.mac.hardware$. When one belongs to the seen classes and the other is unseen, the classification will be quite difficult. We set the number of seen classes as 5. The number of training data, generated positive data, generated negative data and test data are set to 3000, 2000, 2000 and 3000, respectively. We randomly sample seen classes and the training data to form 20 data sets, and for each data sets, the experiment is repeat for 10 times.

Figure 3 shows the Macro-F1 performance with the increase of test classes. It can be observed that the curves in general decreases as expected. Among these methods, ASG-SVM consistently achieves the best performance, and the OVR-SVM is the runner-up. The outlier detection methods, OC-SVM and MOC-SVM, fail in this data, which further confirms that the instance of unseen classes should not be simply treated as outliers in the OCC problem. The win/tie/loss table on all algorithm pairs is in the appendix due to the space limit, which will show that ASG framework is significantly better than the other methods.

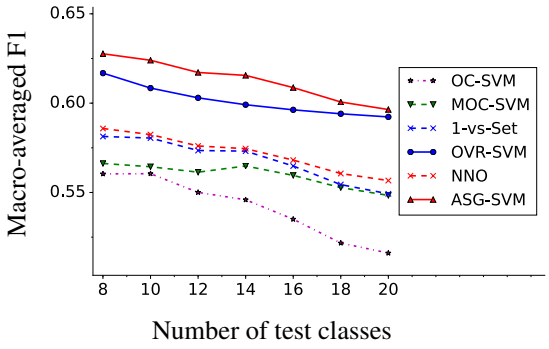


Figure 3: Performance with different number of test classes on 20News dataset

Table 3: Comparisons of classification performance (Macor-F1 score, mean±std.), the best performance on each dataset is in bold.

Dataset	artificial	flag	glass	letter	page_blocks
OC-SVM	.419±.036	.309±.052	.482±.039	.658±.024	.505±.049
MOC-SVM	.555±.022	.504±.050	.678±.037	.721±.019	.536±.042
1-vs-Set	.562±.042	.462±.057	.609±.046	.739±.030	.513±.046
OVR-SVM	.667±.020	.501±.042	.646±.032	.885±.015	.579±.037
NNO	.564±.013	**.581±.045**	.659±.014	.938±.016	.519±.036
ASG-SVM	**.703±.035**	.522±.043	**.706±.042**	**.944±.026**	**.611±.043**

Small Datasets: We also conducted experiments on five small datasets, where the size for each seen class is no more than 100 in the training datasets, which makes the classification tasks become more challenging.

As the result shown in Table 3, ASG-SVM obtains the best performance in four datasets compared to the other five methods; NNO has the best performance in the $flag$ dataset, but can be worse than some of the comparing methods. In addition to ASG-SVM, NNO and OVR-SVM are runner-ups that have better performance in several cases, but have lower Macro-F1 in the $glass$ dataset than MOC-SVM, where the number of each class is just 15 in the training set. Sort the algorithms with their average ranks on all data sets, the best is ASG-SVM (average rank 1.2), and the remaining are NNO (2.6), OVR-SVM (3.0), MOC-SVM (3.6), 1-vs-Set (4.6), and OC-SVM (6).

5 Conclusion

Open-category classification problem often occurs in practical problems, where a system needs to predict the data in an open environment. In this paper, we propose the ASG framework to address the problem by adversarial data generation. In experiments, we demonstrate that ASG can successfully generate boundary data around the seen classes, which makes the recognition of unseen classes can be easily done by supervised learning. On several datasets, ASG shows to be more effective than several state-of-the-art methods. In the future, besides the current SVM model, we would like to apply ASG with state-of-the-art multi-class learning method, and develop a theoretical grounded method for the OCC problem.

References

[Bendale and Boult, 2015] Abhijit Bendale and Terrance E. Boult. Towards open world recognition. In *CVPR'15*, pages 1893–1902, Boston, MA, 2015.

[Brochu *et al.*, 2010] Eric Brochu, Vlad M. Cora, and Nando de Freitas. A tutorial on bayesian optimization of expensive cost functions, with application to active user modeling and hierarchical reinforcement learning. *CoRR*, abs/1012.2599, 2010.

[Chang and Lin, 2011] C.C. Chang and C.J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2(3):27, 2011.

[Chow, 1970] C Chow. On optimum recognition error and reject tradeoff. *IEEE Transactions on Information Theory*, 16(1):41–46, 1970.

[Da *et al.*, 2014] Qing Da, Yang Yu, and Zhi-Hua Zhou. Learning with augmented class by exploiting unlabeled data. In *AAAI'14*, pages 1760–1766, Quebec, Canada, 2014.

[Fink *et al.*, 2006] Michael Fink, Shai Shalev-Shwartz, Yoram Singer, and Shimon Ullman. Online multiclass learning by interclass hypothesis sharing. In *ICML'06*, pages 313–320, Pittsburgh, PA, 2006.

[Goldberg, 1989] David E. Goldberg. *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley, 1989.

[Goodfellow *et al.*, 2014] I.J. Goodfellow, Jean P.A., Mehdi Mirza, Bing Xu, W. F. David, Sherjil Ozair, Courville A. C., and Yoshua Bengio. Generative adversarial nets. In Zoubin Ghahramani, Max Welling, Corinna Cortes, Neil D. Lawrence, and Kilian Q. Weinberger, editors, *NIPS 27*, pages 2672–2680, 2014.

[Hodge and Austin, 2004] Victoria J Hodge and Jim Austin. A survey of outlier detection methodologies. *Artificial Intelligence Review*, 22(2):85–126, 2004.

[Hospedales *et al.*, 2013] Timothy M Hospedales, Shaogang Gong, and Tao Xiang. Finding rare classes: Active learning with generative and discriminative models. *IEEE Transactions on Knowledge and Data Engineering*, 25(2):374–386, 2013.

[Kuzborskij *et al.*, 2013] I. Kuzborskij, F. Orabona, and B. Caputo. From N to N+1: Multiclass transfer incremental learning. In *CVPR'13*, pages 3358–3365, Portland, OR, 2013.

[Li *et al.*, 2016] Wei Li, Melvin Gauci, and Roderich Gro. Turing learning: a metric-free approach to inferring behavior and its application to swarms. *Swarm Intelligence*, 10(3):211–243, 2016.

[Ma and Perkins, 2003] Junshui Ma and Simon Perkins. Time-series novelty detection using one-class support vector machines. In *IJCNN'03*, volume 3, pages 1741–1745, 2003.

[Muhlbaier *et al.*, 2009] M.D. Muhlbaier, A. Topalis, and R. Polikar. Learn++ NC: Combining ensemble of classifiers with dynamically weighted consult-and-vote for efficient incremental learning of new classes. *IEEE Transactions on Neural Networks*, 20(1):152–168, 2009.

[Munos, 2014] Rémi Munos. From bandits to monte-carlo tree search: The optimistic principle applied to optimization and planning. *Foundations and Trends in Machine Learning*, 7(1):1–129, 2014.

[Nowozin *et al.*, 2016] Sebastian Nowozin, Botond Cseke, and Ryota Tomioka. f-gan: Training generative neural samplers using variational divergence minimization. In Daniel D. Lee, Masashi Sugiyama, Ulrike V. Luxburg, Isabelle Guyon, and Roman Garnett, editors, *NIPS 29*, pages 271–279, 2016.

[Palatucci *et al.*, 2009] Mark Palatucci, Dean Pomerleau, Geoffrey E. Hinton, and Tom M. Mitchell. Zero-shot learning with semantic output codes. In Yoshua Bengio, Dale Schuurmans, John D. Lafferty, Christopher K. I. Williams, and Aron Culotta, editors, *NIPS 22*, pages 1410–1418, 2009.

[Pelleg and Moore, 2005] Dan Pelleg and Andrew W. Moore. Active learning for anomaly and rare-category detection. In L.K. Saul, Y. Weiss, and L. Bottou, editors, *NIPS 17*, pages 1073–1080. 2005.

[Phillips *et al.*, 2011] P Jonathon Phillips, Patrick Grother, and Ross Micheals. Evaluation methods in face recognition. In *Handbook of Face Recognition*, pages 551–574. 2011.

[Rifkin and Klautau, 2004] Ryan Rifkin and Aldebaro Klautau. In defense of one-vs-all classification. *Journal of Machine Learning Research*, 5:101–141, 2004.

[Scheirer *et al.*, 2013] Walter J Scheirer, Terrance E Boult, Anderson de Rezende Rocha, and Archana Sapkota. Toward open set recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(7):1757–1772, 2013.

[Schölkopf *et al.*, 2001] Bernhard Schölkopf, John C Platt, John Shawe-Taylor, Alex J Smola, and Robert C Williamson. Estimating the support of a high-dimensional distribution. *Neural computation*, 13(7):1443–1471, 2001.

[Xian *et al.*, 2016] Yongqin Xian, Zeynep Akata, Gaurav Sharma, Quynh N. Nguyen, Matthias Hein, and Bernt Schiele. Latent embeddings for zero-shot classification. In *CVPR'16*, pages 69–77, Las Vegas, NV, 2016.

[Yu *et al.*, 2016] Yang Yu, Hong Qian, and Yi-Qi Hu. Derivative-free optimization via classification. In *AAAI'16*, pages 2286–2292, Phoenix, AZ, 2016.

[Zhou and Chen, 2002] Zhi-Hua Zhou and Zhao-Qian Chen. Hybrid decision tree. *Knowledge-based systems*, 15(8):515–528, 2002.