One-Class Remote Sensing Classification From Positive and Unlabeled Background Data

Wenkai Li, Qinghua Guo, and Charles Elkan

***Abstract*—One-class classification is a common situation in remote sensing, where researchers aim to extract a single land type from remotely sensed data. Learning a classifier from labeled positive and unlabeled background data, which is the case-control sampling scenario, is efficient for one-class remote sensing classification because labeled negative data are not necessary for model training. In this study, we propose a novel positive and background learning with constraints (PBLC) algorithm to address this one-class classification problem. With user-specified information of maximum probability as the constraint, PBLC infers the posterior probability of positive class directly in one-step model training. We test PBLC on a synthetic dataset and a real aerial photograph to perform different one-class classification tasks. Experimental results demonstrate that PBLC can successfully train linear and nonlinear classifiers including generalized linear model, artificial neural network, and convolutional neural network. Probabilistic and binary predictions by PBLC are more similar to the gold-standard positive-negative method, outperforming the two-step positive and background learning algorithm that post-calibrates a naïve classifier based on an estimated constant. Hence, the proposed PBLC algorithm has the potential to solve one-class classification problems in the case-control sampling scenario.**

***Index Terms*—Case-control sampling, labeled and unlabeled data, one-class classification, positive and background learning with constraints (PBLC), remote sensing.**

I. Introduction

IN REMOTE sensing classification, there are situations when users are only interested in extracting a single land type, which is the so-called one-class classification in the literature [1]–[3]. The land type of interest is called positive class and other land types of no interest are called negative class. The task of one-class classification can be performed by applying a standard binary classifier given a complete and exhaustively labeled training set [4], [5]. In other words, if one is only interested in mapping urban areas, training samples from all possible land types (e.g., urban, tree, grass, soil, etc.) should be labeled, which is a time-consuming and labor-intensive process. If users fail to identify some land types, the training set is incomplete and hence the classification accuracy may be decreased [5], [6]. Meanwhile, one-class classification is also a common problem in many other applications where users have difficulty in labeling negative data [7]–[9]. In the wildland search and rescue risk assessment, for example, researchers use historical observation data to train a binary classifier to understand the relationship between incident occurrence and environmental variables, but the problem is that negative observation data (locations where incidents have not occurred and will not occur) are usually not available [9]. Hence, it is beneficial to develop classification methods that do not require labeled negative training data.

One-class classification without negative data is more challenging than binary classification with both positive and negative data, as it is difficult to find the optimal decision boundary [10]. In the early stage, positive-only methods were proposed for one-class classification, such as Gaussian domain descriptor (GDD) and one-class support vector machine (OCSVM) [4], [11]. Given positive-only training data, GDD models the density of features for the positive class assuming a Gaussian distribution whereas OCSVM fits a hypersphere to separate positive from negative data [10], [12]. These methods require users to empirically tune a threshold or free parameters to balance overprediction and underprediction, which is difficult when negative data are not available [4], [10], [13]. Later, researchers demonstrated that unlabeled data are also helpful in classifier learning in addition to positive data, and positive-unlabeled methods usually outperform positive-only methods, which make this category of methods become more and more popular in one-class classifications in many fields [7], [14], [15]. One common approach of this category involves heuristically identifying likely negative data (sometimes as well as likely positive data) from unlabeled set iteratively that are combined with the labeled data in the previous step, and then train a standard binary classifier [7], [16]. The biased support vector machine (BSVM) is another approach by treating the unlabeled set as weighted positive and weighted negative data during model training, which has been shown to be superior than the previous heuristic approach [17]. However, the accuracy of BSVM is sensitive to the weights, and users are required to search for the optimal weights iteratively with an independent validation set, which may preclude its usage by nonexpert users due to complicated model selection procedure [7], [15], [17]. The positive and unlabeled learning (PUL) algorithm proposed by Elkan and Noto trains a binary classifier

Manuscript received July 5, 2020; revised August 31, 2020; accepted September 4, 2020. Date of publication September 21, 2020; date of current version January 6, 2021. This work was supported in part by the Guangdong Basic and Applied Basic Research Foundation under Grant 2020A1515010764, in part by National Natural Science Foundation of China under Grant 41401516, and in part by State Key Laboratory of Vegetation and Environmental Change under Grant LVEC-2019kf05. *(Corresponding author: Wenkai Li.)*

Wenkai Li is with the Guangdong Provincial Key Laboratory of Urbanization and Geo-Simulation, School of Geography and Planning, Sun Yat-Sen University, Guangzhou 510275, China (e-mail: liwenk3@mail.sysu.edu.cn).

Qinghua Guo is with the State Key Laboratory of Vegetation and Environmental Change, Institute of Botany, Chinese Academy of Sciences, Beijing 100093, China (e-mail: guo.qinghua@gmail.com).

Charles Elkan is with the Department of Computer Science and Engineering, University of California at San Diego, La Jolla, CA 92093-0404 USA.

Digital Object Identifier 10.1109/JSTARS.2020.3025451

using positive and unlabeled data and estimates a constant using an independent validation set; the original "pseudo" posterior probability predicted by the trained classifier is post-calibrated by the constant to obtain a true posterior probability of positive class, and hence a threshold of 0.5 can be applied to produce binary classification [7]. Actually, PUL is not a specific classifier, but a learning algorithm than can be applied to train standard binary classifiers such as logistic regression, support vector machine (SVM), neural networks, etc. [7], [15], [18]. Please note that there are also other semisupervised learning methods that can generate good performance by combining labeled and unlabeled data, such as transductive SVM [19], context-sensitive semisupervised SVM [20], and semisupervised Laplacian SVM [21], but these methods still require labeled negative data, and hence they are not considered in this research.

The PUL algorithm has good potential in one-class classification, but it requires a "selected completely at random" assumption that may be violated in some real-world applications, i.e., the positive and unlabeled data are randomly collected in a single-training-set, and each positive sample is labeled with the same constant probability. In one-class remote sensing classification, for example, users normally label the first set of positive samples randomly and then extract a second set of random background pixels as the unlabeled data separately, which is referred to as a case-control scenario in contrast to the single-training-set scenario. To address the PUL problem under case-control sampling scenario, the positive (also called presence) and background learning (PBL) algorithm was proposed by Li *et al*. [22], and it has also been successfully applied in one-class remote sensing classification with good performance in different case studies. With space-borne microwave brightness temperature measurements and *in situ* observations, Xu *et al*. [23] applied the PBL algorithm to map global snow cover from 1987 to 2010. Ao *et al*. [24] used PBL to classify a single class such as building, tree, terrain, and power line from airborne light detection and ranging point cloud data. Zhang *et al*. [25] applied the PBL algorithm to map surface water bodies using Sentinel-2 imagery and OpenStreetMap data.

Although the PBL algorithm has shown promise in one-class remote sensing classification, it still has two major drawbacks that can affect its performance. First, PBL relies on a constant to calibrate the posterior probability of positive class, but the constant is consistently underestimated by the existing approach. Second, the classifier is post-processed based on the constant after training, which may produce probabilities greater than one due to inaccurate probability predictions. In order to overcome these problems, we propose a new approach to learn a binary classifier using positive and unlabeled data focusing on the case-control scenario. To investigate its effectiveness in one-class remote sensing classification, we use both synthetic and real datasets to test the performance of the new method. In the following sections, we provide details about the algorithm, experiments, and discussions.

II. Learning a Binary Classifier From Positive and Background Data

Let $y = 1$ denote positive class, $y = 0$ denote negative class, x denote the features (or covariates) of a given pixel, and $\Pr(y = 1)$ denote the prior (or prevalence) of positive class. Our aim is to model the posterior probability of the positive class at a specific pixel: $\Pr(y = 1 \mid x)$. If both positive and negative training data are available, this task can be performed by training a standard binary classifier such as logistic regression, but it becomes a problem when negative data are not available. Let $s = 1$ denote labeled data and $s = 0$ denote unlabeled data. Since we only label positive data, $s = 1$ implies $y = 1$, but either $y = 1$ or $y = 0$ can be true when $s = 0$. In the single-training-set scenario, samples are randomly drawn from the population, and only positive samples are labeled with a constant probability, and the rest of the samples (both positive and negative) are recorded as unlabeled data [7]. In the case-control scenario, the labeled data are randomly sampled from the positive subset, and the unlabeled data are randomly sampled from the entire population (background pixels), separately [22]. A major difference between both scenarios is that the proportion of positive data (labeled positive plus unlabeled positive) in the training set is equal to the class prior $\Pr(y = 1)$ in the single-training-set scenario, but larger than the class prior in the case-control scenario.

If we train a binary classifier using labeled and unlabeled data, we obtain a naïve model: $\Pr(s = 1 \mid x)$ in the single-training-set scenario or $\Pr(s = 1 \mid x, \eta = 1)$ in the case-control scenario. Here $\eta = 1$ is just a notation for the case-control scenario. In the single-training-set scenario, the trained model and desired model have the following relationship [7]:

$$\Pr(y = 1|x) = \Pr(s = 1|x)/c. \tag{1}$$

In the case-control scenario, however, this relationship becomes [22]

$$\Pr(y = 1|x) = \frac{1-c}{c} \times \frac{\Pr(s = 1|x, \eta = 1)}{1 - \Pr(s = 1|x, \eta = 1)}. \tag{2}$$

In both of the above equations, c is the ratio of the number of labeled positive samples to the total number of positive samples in the training set, and it is a fixed constant whose value depends on the class prior. Equivalently, c can be defined as the labeling effort of positive class: $c = \Pr(s = 1 \mid y = 1)$. Suppose that the number of labeled positive samples is n_1 and the number of background samples is n_0, then $c = n_1 / [n_1 + n_0 \times \Pr(y = 1)]$. Assuming that there is a subset of "prototypical positive" samples whose values of $\Pr(y = 1 \mid x)$ are one, the following estimator of c can be derived:

$$c = \frac{1}{k} \sum_{x \in PP} \Pr(s = 1|x) \tag{3}$$

$$c = \frac{1}{k} \sum_{x \in PP} \Pr(s = 1|x, \eta = 1) \tag{4}$$

where k refers to the cardinality of prototypical positive subset PP. (3) and (4) are actually the same, but we use different notations to distinguish two different sampling scenarios. The PUL and PBL algorithms use (1) and (2) to calibrate the naïve models, respectively, both of which share the same estimator of c in (3) or (4). They are two complementary algorithms to address two different sampling scenarios correspondently. Details about

PUL can be found in [7], and details about PBL can be found in [22].

In real-world applications, researchers usually treat part of the observed positive data (e.g., top 50% with higher prediction values) or all the observed positive data in a separate validation set as the prototypical positive subset to estimate c, but the problem is that some of the selected positive samples actually have probabilities smaller than one, so the constant c is consistently underestimated by (3) or (4), and $\Pr(y = 1 \mid x)$ is overestimated consequently. Meanwhile, an artifact of the two-step processing is that a small number of pixels whose predicted probability values are greater than one may be produced due to inaccurate model prediction. Instead of arbitrarily selecting a subset of positive samples to estimate c and post-calibrate the naïve model in two separate steps, here we propose a new approach to infer the desired model $\Pr(y = 1 \mid x)$ directly in one-step model training.

We focus on the case-control scenario as it is more common in one-class remote sensing classification, and "positive and background data" refer to case-control positive and unlabeled data throughout the article. (2) can be rewritten as:

$$\Pr(s = 1|x, \eta = 1) = \frac{\Pr(y = 1|x)}{\Pr(y = 1|x) + (1 - c)/c}. \tag{5}$$

Let $\Pr(y = 1 \mid x) = f(x, \omega)$ and $\Pr(s = 1 \mid x, \eta = 1) = g(x, \beta)$ where f and g are functions, and ω and β are model parameters to be estimated. Consider the maximum likelihood estimation as an example, we can infer the model parameter β by minimizing the following negative log-likelihood function with observed labeled and unlabeled data:

$$L(\beta) = -\sum_{i=1}^{n} \{s_i \log[g(x_i, \beta)] + (1 - s_i)\log[1 - g(x_i, \beta)]\} \tag{6}$$

where n refers to the number of training samples: (x_i, s_i) with $i = 1, 2, 3, \ldots, n$. The above negative log-likelihood function can be replaced by other forms of loss functions as well, such as the mean squared error function [26]. According to (5), we can rewrite (6) as

$$L(\omega, c) = -\sum_{i=1}^{n} \left\{ s_i \log\left[\frac{f(x_i, \omega)}{f(x_i, \omega) + \frac{(1-c)}{c}}\right] + (1 - s_i)\log\left[1 - \frac{f(x_i, \omega)}{f(x_i, \omega) + (1 - c)/c}\right]\right\}. \tag{7}$$

Given observed labeled and unlabeled data, can we infer the model parameter ω by minimizing the loss function in (7)? Generally, if we knew the class prior, then the constant c is also known and hence model $\Pr(y = 1 \mid x)$ is identifiable through (7). In reality, however, the class prior is usually unknown, then model $\Pr(y = 1 \mid x)$ is only identifiable under certain assumptions or conditions [27], [28]. The PUL and PBL algorithms assume that the posterior probability values of selected prototypical positive samples reach one, which, therefore, makes $\Pr(y = 1 \mid x)$ identifiable based on the estimator of c [7], [22]. By contrast, we assume that the maximum value of posterior probabilities of the entire population, denoted as $P_{\max}$, is a priori knowledge, and a constraint term (regularizer) is added to (7), resulting in

$$L(\omega, c) = -\sum_{i=1}^{n} \left\{ s_i \log\left[\frac{f(x_i, \omega)}{f(x_i, \omega) + \frac{(1-c)}{c}}\right] + (1 - s_i)\, log\left[1 - \frac{f(x_i, \omega)}{f(x_i, \omega) + \frac{(1-c)}{c}}\right]\right\} + \lambda|\max[f(x, \omega)] - P_{\max}|^2 \tag{8}$$

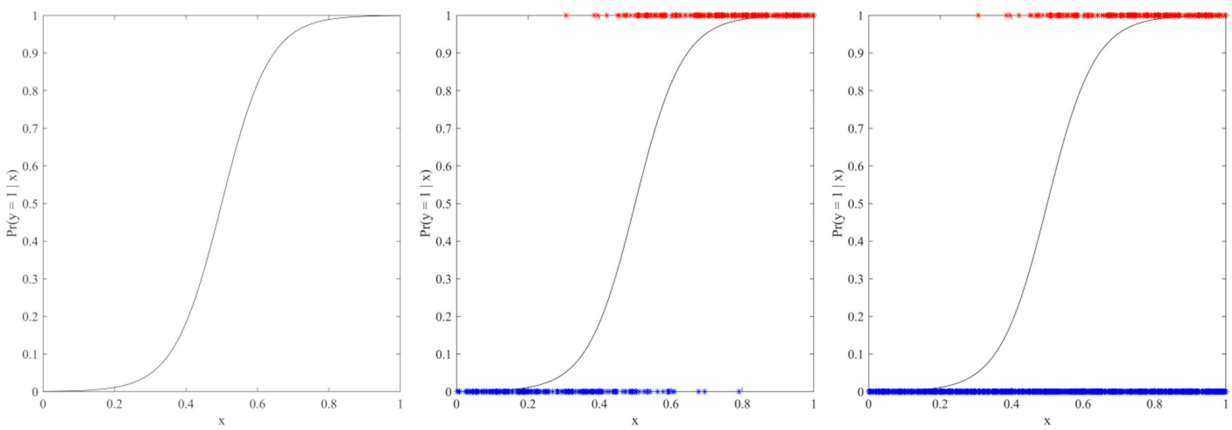
where λ is a regularization parameter. The constant c now becomes a model parameter that will be optimized together with ω during model training. The regularizer in (8) can also be combined with other regularizers such as weight decay to improve generalization [26]. Therefore, we can obtain $\Pr(y = 1 \mid x)$ by minimizing the loss function in (8), assuming that the prior information on $P_{\max}$ is available. In one-class remote sensing classification, it is reasonable to assume that $P_{\max}$ reaches one, but we can instead assume a value smaller than one if the positive class is less separable from the negative class. This model assumption is less strong and more flexible than the previous assumption made by PUL/PBL, and it is not necessary to arbitrarily find a subset of prototypical positive samples in order to estimate c. In summary, (8) provides a new flexible way to train a binary classifier using positive and background data, and the form of $\Pr(y = 1 \mid x)$ can be either linear (e.g., logistic regression) or nonlinear (e.g., neural networks). We name this new algorithm positive and background learning with constraints (PBLC) to distinguish it from the previous PBL algorithm in [22].

III. Experiment

We tested the proposed PBLC algorithm using both synthetic and real datasets to investigate its effectiveness in one-class remote sensing classification. We focused on the case-control sampling scenario, so the most relevant algorithm PBL was selected for comparison. Three typical linear and nonlinear binary classifiers including generalized linear model (GLM) [29], multilayer artificial neural network (ANN) [30], and convolutional neural network (CNN) [31] were used to implement PBL and PBLC, all of which were trained using positive and background data. Compared with learning from positive and background data, a binary classifier trained by both positive and negative data is regarded as the gold-standard model, so we also trained the binary classifiers using positive-negative (PN) data as benchmark models. Therefore, we have three classifiers, each of which was trained by three different learning approaches. For convenience, we use GLM_PN, GLM_PBL, and GLM_PBLC to refer to a GLM classifier trained by standard PN data, PBL, and PBLC, respectively, etc.

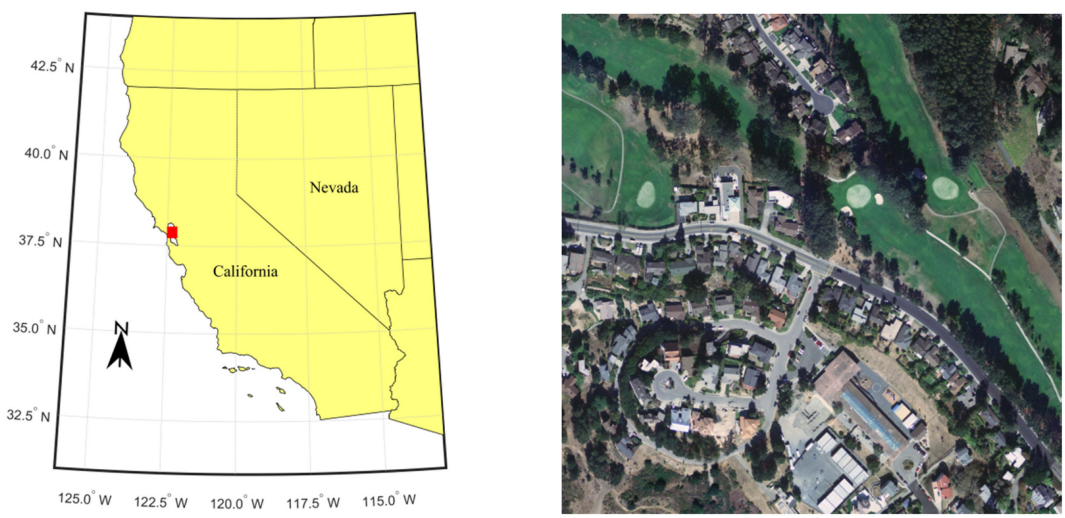
A. Dataset

The performances of PBL and PBLC depend on the ability to predict $\Pr(y = 1 \mid x)$, but we are not able to observe true posterior probability in reality, so we used synthetic data to evaluate the accuracy of predicted probability. The synthetic dataset was generated by a logistic model since it is a commonly



(a) (b) (c)

Fig. 1. (a) Synthetic dataset: true probability curve; (b) positive and negative samples; and (c) positive and background samples. Red asterisk: positive samples. Blue asterisk: negative or background samples. Sample size: $N_p = 200$.



(a) (b)

Fig. 2. Location (red rectangle) of the selected scene in El Cerrito, California (a) and the aerial photograph (b). Spatial resolution: 0.3 m. Extent: 500 m × 500 m. Number of pixels: 1667 × 1667.

used model assumption in binary classification [22], [32]. The posterior probability of positive class was modeled as

$$\Pr(y = 1 \,|x) = \frac{e^{b_0+b_1 x}}{1+e^{b_0+b_1 x}} \tag{9}$$

where b_0 and b_1 are model parameters. We set $b_0 = -7.5$, $b_1 = 15$, and $0 \leq \times \leq 1$. The true probability curve contained 100 001 data points, which was used as the test set [see Fig. 1(a)]. Binary data were realized from the probability curve using the following procedure: at each data point, we generated a random value $0 \leq r < 1$ and assigned that data point as positive ($y = 1$) if $r \leq \Pr(y = 1 \mid x)$ or negative ($y = 0$) otherwise. For PBL and PBLC, we randomly extracted a set of positive samples from the positive subset, and then randomly extracted a second set of background samples from the whole population. A single training set with completely labeled positive and negative data was also extracted by random sampling in order to train the classifiers in a standard PN approach. We tested different sample sizes N_p, including 200, 1000, and 5000. Here N_p only refers to the number of labeled positive data in a training set, and the number of background data was five times of N_p following [22]. Therefore, the true value of constant c is 0.2857 since the class prior $\Pr(y = 1)$ is 0.5. Each training set was repeated by 10 different random realizations. Examples of PN samples and positive-background samples with $N_p = 200$ are shown in Fig. 1(b) and (c), respectively. For this dataset, we produced probabilistic prediction rather than binary classification, and the agreement between predicted and true probabilities of the whole curve was evaluated using Pearson's correlation coefficient (COR) and root mean square error (RMSE).

The second dataset was extracted from an aerial photograph (0.3-m spatial resolution) of the city El Cerrito in California, which was acquired by a Leica ADS40 digital camera, including three visible bands (see Fig. 2). The extent of selected scene is 500 × 500 m, and we tried four examples of one-class classifications, including extraction of urban areas (houses plus

roads), trees, grasses, and soils, respectively. For each band, we calculated the mean, variance, homogeneity, contrast, and second moment using a 3×3 pixel template, and all of these features were rescaled to the range of 0–1 [15]. Please note that these manually designed features were only necessary for GLM and ANN, whereas the features for CNN were learned automatically. For GLM and ANN, a sample was just a single pixel associated with label and features. For CNN, a sample was an image patch (5×5 pixels) whose label was determined by the center pixel, and the window-sliding approach was used to produce pixel-wise classification [32], [33] so that CNN was comparable with GLM and ANN. For each classification task, we collected two types of training sets: one with positive and background data (case-control sampling), and the other with positive and negative data (simple random sampling). Again, we tried different sample sizes N_p, including 200, 1000, and 5000, and the number of background data was five times of N_p [22]. All of the training sets were randomly realized 10 times. The true labels of the entire image via manual interpretation were used as the test set, so the true values of class prior and constant c were also available. Specifically, the image includes 636 826 pixels (22.92%) of urban, 585 171 pixels (21.06%) of tree, 522 546 pixels (18.80%) of grass, and 284 249 pixels (10.23%) of soil. The binary classifications were evaluated by overall accuracy. The producer's accuracy, user's accuracy, and F-score of the positive class were also reported.

B. Model Implementation

We implemented the classifiers GLM, ANN, and CNN in TensorFlow [34]. We set two hidden layers for ANN, and the activation function was logistic sigmoid for the output layer and rectified linear unit (ReLU) for other layers. For CNN, we set two convolutional layers, two fully connected layers, and a final output layer; each convolutional layer was followed by a max-pooling layer and a normalization layer; the activation function was logistic sigmoid for the final layer and ReLU for other layers. We used a negative log-likelihood loss function and trained the classifiers using Adam optimizer [35]. The initial training set was randomly split as two folds: 75% for training and 25% for validation. The validation set was used to determine the learning rate and number of iterations. A common practical guide is to strop training when training error becomes stable but validation error starts to increase [26]. For GLM and ANN, we trained the models 10 times and averaged the outputs. However, it is too expensive to train CNN multiple times, so we used dropout to improve model generalization [26]. Because CNN requires a large training set, so we only applied it to the aerial photograph with the sample size (N_p) of 5000.

We trained the binary classifiers in three different approaches: PN, PBL, and PBLC. The PN approach trained the standard binary classifiers with positive and negative data, which produced the gold-standard models. For PBL, the classifiers were trained similarly to the standard approach but with positive and background data, and the trained classifiers were post-calibrated based on the constant c in a second step according to (2); the constant c was estimated using all positive data in the validation set according to (4). For PBLC, the loss function in (8) was used, and the classifiers were trained by positive and background data. By default, we set the regularization parameter $\lambda = 0$ and tuned it by gradually increasing its value if the maximum predicted probability was smaller than a user-specified threshold. In this study, we empirically specified the threshold as 0.9 rather than one, accounting for the situation where the classes were not separable or model prediction was not accurate due to certain reasons like outliers in the training set. Finally, we applied a threshold of 0.5 to all models to produce binary classifications.

C. Results

1) Synthetic Dataset: Figs. 3 –5 show the predicted probabilities by classifiers trained using PBL, PBLC, and PN. Obviously, the probabilities predicted by PBLC are much closer to the true values than that by PBL, and PN provides the most accurate and stable predictions than the other two. ANN_PBL consistently overestimates the probabilities, whereas the behaviors of GLM_PBL are mixed, and both methods show a common artifact that the maximum predicted probability goes beyond one. In the following accuracy assessment, we cut those extremely large probabilities to one because a probability larger than one makes no sense. Meanwhile, GLM_PBLC consistently produces probabilities in the range of 0–1 with the default regularization parameter $\lambda = 0$, but the maximum probability produced by ANN_PBLC is affected by λ. According to Fig. 6, the maximum predicted probability by ANN_PBLC becomes more and more closer to one when λ increases from 0 to 0.1.

Detailed accuracy assessment is provided in Table I. Overall, PBLC and PN produce low RMSE values and high COR values, and the estimated class prior $\Pr(y = 1)$ and constant c are close to the true values. By contrast, PBL shows larger discrepancies between predictions and true values, with $\Pr(y = 1)$ being overestimated and c underestimated consistently. With the sample size of 1000, the average RMSE values by GLM_PBL, GLM_PBLC, and GLM_PN are 0.1227, 0.0192, and 0.0070, respectively, and the corresponding average COR values are 0.8421, 0.9992, and 0.9999, respectively; the estimated values of class prior by GLM_PBL, GLM_PBLC, and GLM_PN are 0.5066, 0.5013, and 0.5011, respectively, whereas the estimated values of c by GLM_PBL and GLM_PBLC are 0.2416 and 0.2903, respectively. Meanwhile, increasing the sample size generally results in increased COR and decreased RMSE. For example, the RMSE and COR values by ANN_PBLC are 0.0479 and 0.9973 with $N_p = 200$, but these values become 0.0271 and 0.9996 with $N_p = 5000$.

2) Aerial Photograph: In Figs. 7 –10, we present part of the binary classification maps of different land types. Since the classification maps of 10 random realizations of the same training set are similar, here we only present the average prediction. For classes of urban and grass, the classification results by different methods are quite similar visually, but the differences become more obvious for classes of tree and soil. Generally, we can observe that PN produces the best binary classification results, and PBLC provides better results than PBL. For different classifiers, CNN produces the best classification results whereas GLM

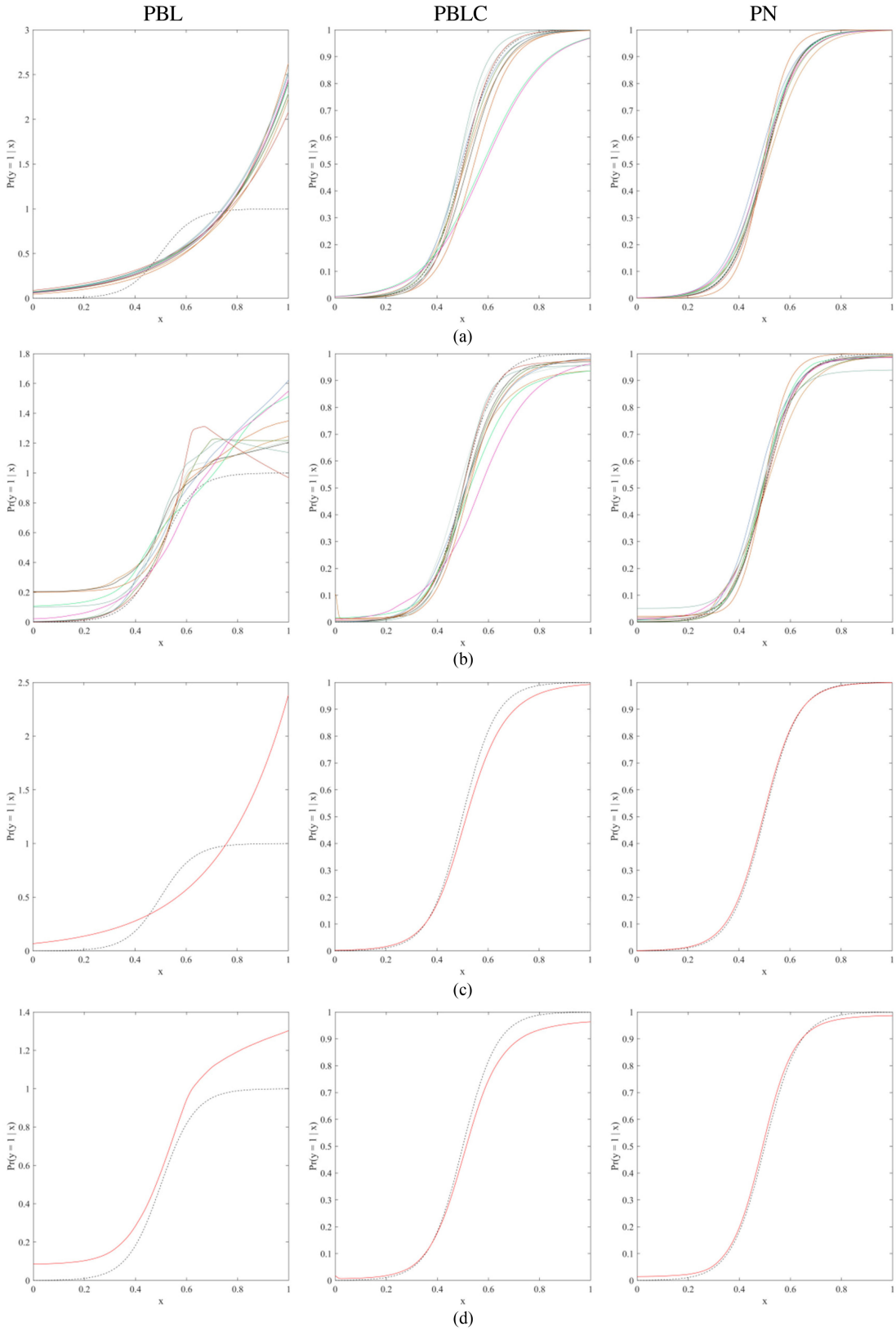


Fig. 3. Probabilistic predictions of synthetic dataset. Classifiers were trained by PBL, PBLC, and PN. Predictions of ten training sets: (a) GLM and (b) ANN. Average prediction over ten training sets: (c) GLM and (d) ANN. GLM: $\hat{\lambda} = 0$. ANN: $\hat{\lambda} = 0.1$. Colored solid line: predicted probability. Black dashed line: true probability. Sample size: $N_p = 200$.

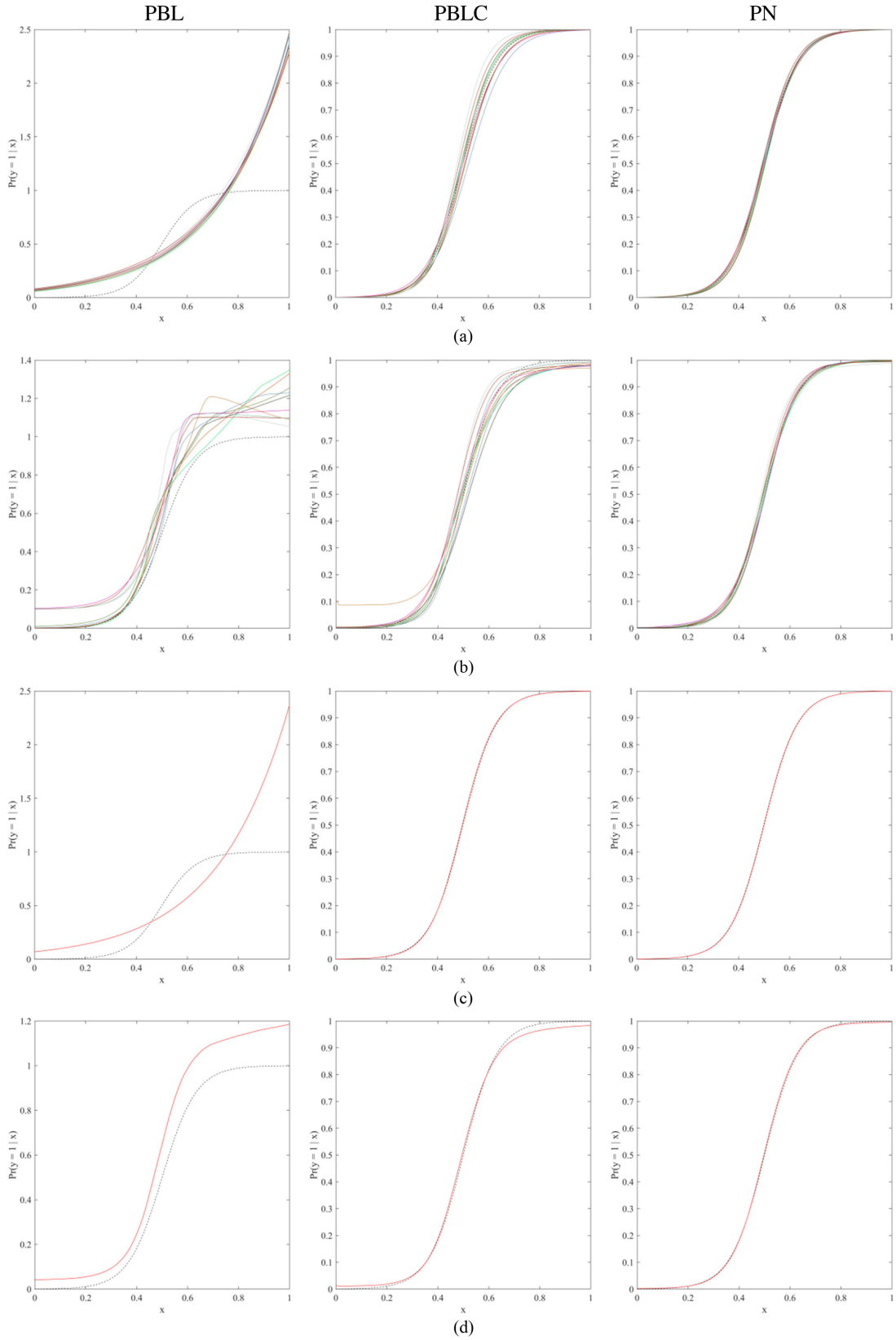


Fig. 4. Probabilistic predictions of synthetic dataset. Classifiers were trained by PBL, PBLC, and PN. Predictions of ten training sets: (a) GLM and (b) ANN. Average prediction over ten training sets: (c) GLM and (d) ANN. GLM: $\hat{\lambda} = 0$. ANN: $\hat{\lambda} = 0.1$. Colored solid line: predicted probability. Black dashed line: true probability. Sample size: $N_p = 1000$.

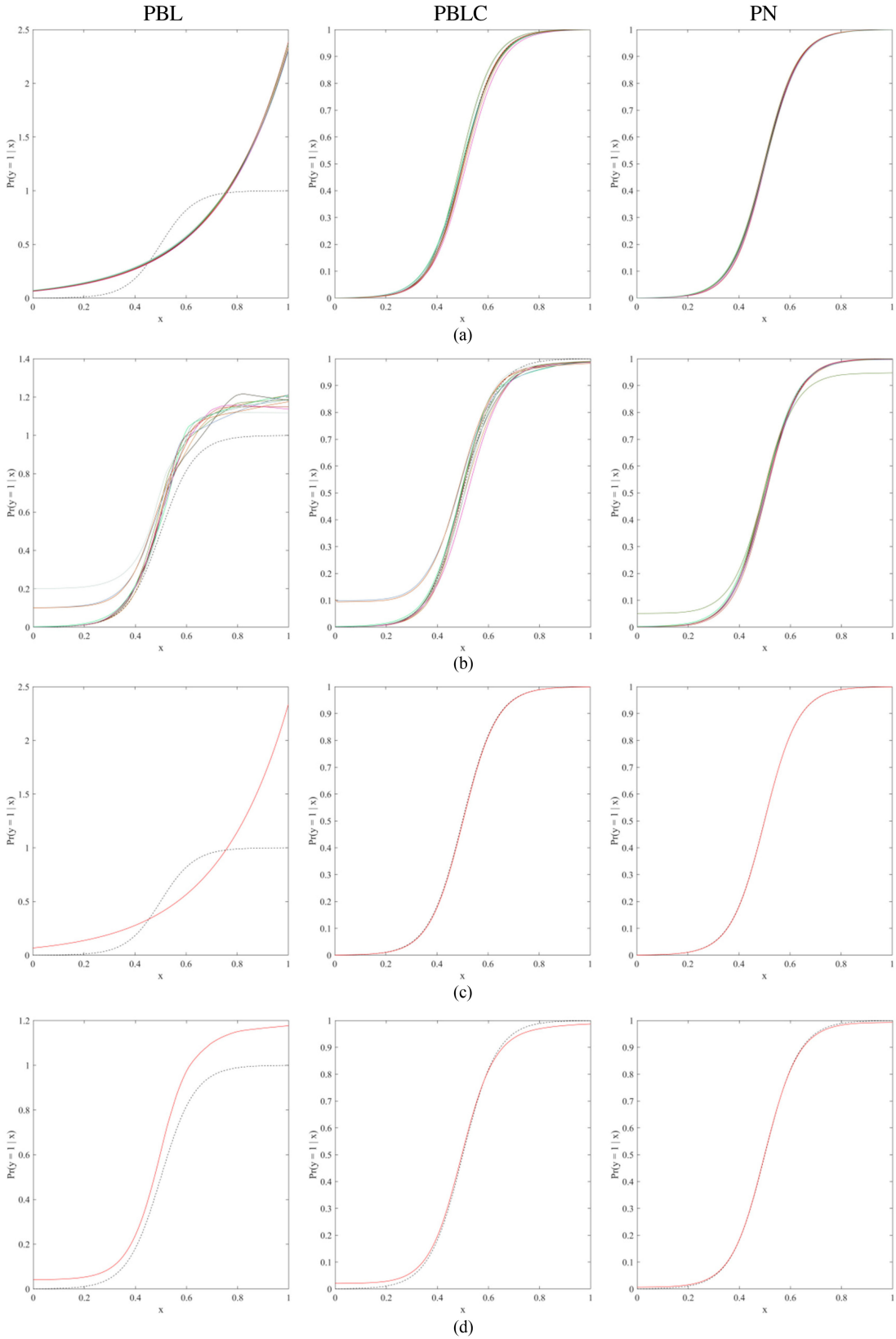


Fig. 5. Probabilistic predictions of synthetic dataset. Classifiers were trained by PBL, PBLC, and PN. Predictions of ten training sets: (a) GLM and (b) ANN. Average prediction over ten training sets: (c) GLM and (d) ANN. GLM: $\hat{\lambda} = 0$. ANN: $\hat{\lambda} = 0.1$. Colored solid line: predicted probability. Black dashed line: true probability. Sample size: $N_p = 5000$.

TABLE I
MODEL PERFORMANCES ON SYNTHETIC DATASET WITH DIFFERENT SAMPLE SIZES (N_p)

Model	$N_p = 200$				$N_p = 1000$				$N_p = 5000$			
	RMSE	COR	Pr(y = 1)	c	RMSE	COR	Pr(y = 1)	c	RMSE	COR	Pr(y = 1)	c
GLM_PBL	**0.1242**	**0.8396**	**0.5030**	**0.2435**	**0.1227**	**0.8421**	**0.5066**	**0.2416**	**0.1244**	**0.8414**	**0.5014**	**0.2409**
	0.0071	*0.0112*	*0.0159*	*0.0098*	*0.0034*	*0.0064*	*0.0126*	*0.0060*	*0.0015*	*0.0030*	*0.0050*	*0.0048*
GLM_PBLC	**0.0501**	**0.9941**	**0.4770**	**0.2960**	**0.0192**	**0.9992**	**0.5013**	**0.2903**	**0.0097**	**0.9998**	**0.4976**	**0.2855**
	0.0456	*0.0083*	*0.0346*	*0.0200*	*0.0117*	*0.0010*	*0.0124*	*0.0086*	*0.0062*	*0.0003*	*0.0064*	*0.0083*
GLM_PN	**0.0178**	**0.9993**	**0.5048**	**-**	**0.0070**	**0.9999**	**0.5011**	**-**	**0.0045**	**1.0000**	**0.4996**	**-**
	0.0115	*0.0008*	*0.0089*	-	*0.0027*	*0.0001*	*0.0039*	-	*0.0022*	*0.0000*	*0.0029*	-
ANN_PBL	**0.0883**	**0.9864**	**0.5646**	**0.2443**	**0.0812**	**0.9950**	**0.5569**	**0.2498**	**0.0753**	**0.9992**	**0.5529**	**0.2508**
	0.0397	*0.0122*	*0.0385*	*0.0044*	*0.0284*	*0.0039*	*0.0290*	*0.0066*	*0.0320*	*0.0005*	*0.0327*	*0.0072*
ANN_PBLC	**0.0479**	**0.9973**	**0.4716**	**0.3013**	**0.0309**	**0.9987**	**0.4974**	**0.2876**	**0.0271**	**0.9996**	**0.5027**	**0.2868**
	0.0285	*0.0044*	*0.0188*	*0.0091*	*0.0117*	*0.0013*	*0.0172*	*0.0093*	*0.0209*	*0.0003*	*0.0214*	*0.0068*
ANN_PN	**0.0254**	**0.9989**	**0.5063**	**-**	**0.0102**	**0.9997**	**0.5007**	**-**	**0.0092**	**0.9999**	**0.4997**	**-**
	0.0119	*0.0010*	*0.0091*	-	*0.0052*	*0.0004*	*0.0047*	-	*0.0112*	*0.0001*	*0.0033*	-

RMSE: Root mean square error. COR: Pearson's correlation coefficient. True class prior: Pr($y = 1$) = 0.5. True constant: $c = 0.2857$. Bold values refer to averages. Italic values refer to standard deviations; GLM_PBL: A GLM model trained from positive and background data using PBL; GLM_PBLC: A GLM model trained from positive and background data using PBLC; GLM_PN: A GLM model trained from positive and negative data using standard approach; ANN_PBL: An ANN model trained from positive and background data using PBL; ANN_PBLC: An ANN model trained from positive and background data using PBLC; ANN_PN: An ANN model trained from positive and negative data using standard approach.

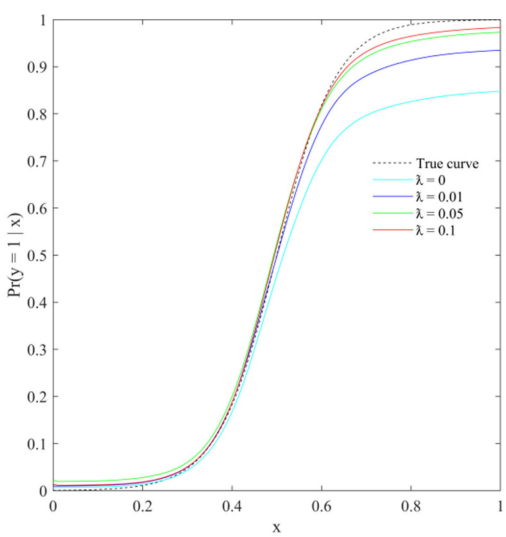


Fig. 6. Average prediction over 10 training sets of synthetic dataset by ANN_PBLC with varied regularization parameter λ. Sample size: $N_p = 1000$.

produces relatively poor results. For the tree class, in particular, GLM_PBL and GLM_PBLC show obvious over-predictions whereas GLM_PN shows obvious under-prediction.

The accuracies of different models for each land type are reported in Tables II–V. Again, we can see that the overall rank of classification accuracy by different learning methods is PN > PBLC > PBL, and the overall rank of classification accuracy by different classifiers is CNN > ANN > GLM. With the sample size $N_p = 5000$ of urban class, for example, the F values by GLM_PBL, GLM_PBLC, and GLM_PN are 0.7848, 0.8084, and 0.8092; the F values by ANN_PBL, ANN_PBLC, and ANN_PN are 0.8590, 0.8627, and 0.8669, respectively; the F values by CNN_PBL, CNN_PBLC, and CNN_PN are 0.8697, 0.8720, and 0.8867, respectively. The classification accuracy generally increases as the sample size increases, but this effect is more obvious for ANN compared with GLM.

The PBLC algorithm also provides more accurate estimates of class prior Pr($y = 1$) and constant c compared with PBL. For the urban class, the true class prior to Pr($y = 1$) is 0.2292 and the true constant c is 0.4660. With $N_p = 1000$, the estimated Pr($y = 1$) by GLM_PBL and ANN_PBL are 0.2633 and 0.2893, whereas the estimated values by GLM_PBLC and ANN_PBLC are 0.2558 and 0.2297; the estimated c by GLM_PBL and ANN_PBL are 0.3614 and 0.3816, whereas the estimated values by GLM_PBLC and ANN_PBLC are 0.4410 and 0.4823, respectively (see Table II).

IV. DISCUSSION

In this study, we investigate the problem of one-class remote sensing classification using positive and unlabeled data, with a focus on the case-control sampling scenario. The positive data are randomly sampled from the target class, and the unlabeled data come from randomly sampled background data without label information, both of which are sampled separately. Given a training set, the ratio of the number of positive data in the labeled set to the total number of positive data in the entire training set is a fixed constant, namely $c = \Pr(s = 1 \mid y = 1)$, which is a key parameter that should be estimated in order to infer the desired model $\Pr(y = 1 \mid x)$ [7], [22]. Assuming that the posterior probability $\Pr(y = 1 \mid x)$ of a prototypical positive reaches the maximum value one, the PBL algorithm arbitrarily selects a prototypical positive subset to estimate the constant c, but this approach usually underestimates c due to the fact that an

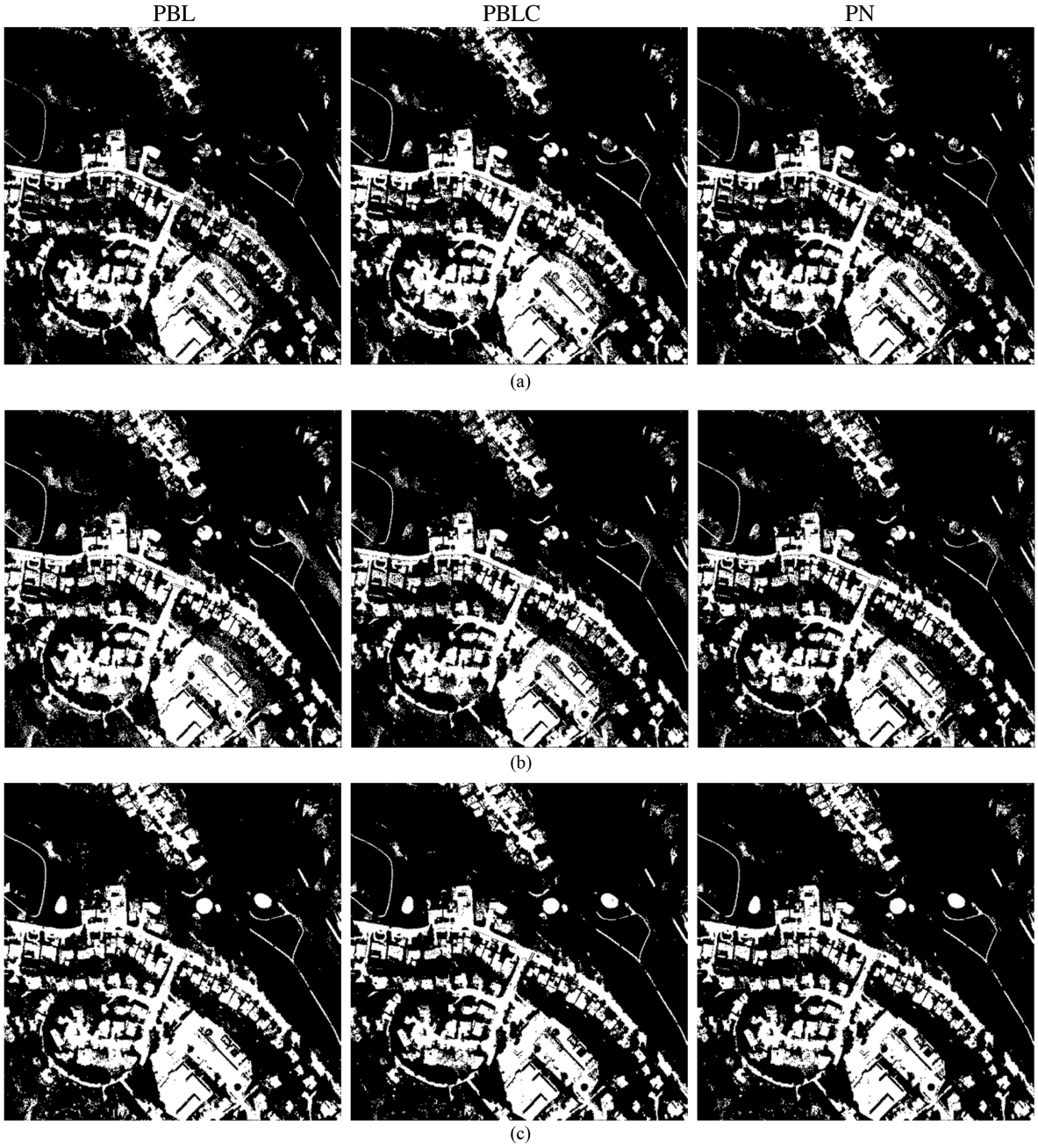


Fig. 7. Average binary classification maps of urban over ten random realizations of training sets. Classifiers were trained by PBL, PBLC, and PN. (a) GLM with $N_p = 1000$. (b) ANN with $N_p = 1000$. (c) CNN with $N_p = 5000$.

observed positive sample may actually have a probability value smaller than one [22]. According to (2), the posterior probability $\Pr(y = 1 \mid x)$ and hence the class prior $\Pr(y = 1)$ are overestimated with an underestimated constant c. By contrast, PBLC treats the constant c as a model parameter whose value is automatically estimated during model training, and this approach provides more accurate estimates of c, $\Pr(y = 1 \mid x)$, and $\Pr(y = 1)$. Unlike PBL that obtains $\Pr(y = 1 \mid x)$ indirectly in two-step processing, PBLC infers $\Pr(y = 1 \mid x)$ directly in one-step training without the artifact that estimated probabilities are larger than one. For these reasons, PBLC outperforms PBL in our experiments, providing more accurate probabilistic and binary predictions.

Compared with learning from positive and negative data in a standard approach, learning a classifier from positive and background data is more challenging since the training set contains less information, which is the reason why PN produces the best results in our experiments [22]. In order to make $\Pr(y = 1 \mid x)$ identifiable from positive and background data, additional information such as $\Pr(y = 1)$ or model assumptions are necessary [27], [28], [36]. In reality, it is difficult to know $\Pr(y = 1)$ without negative data, so we only consider the situation that $\Pr(y = 1)$ is unknown in this study. Previous studies have revealed that assuming $\Pr(y = 1 \mid x)$ to be certain parametric forms (such as linear logistic model) can make the model identifiable, but it is

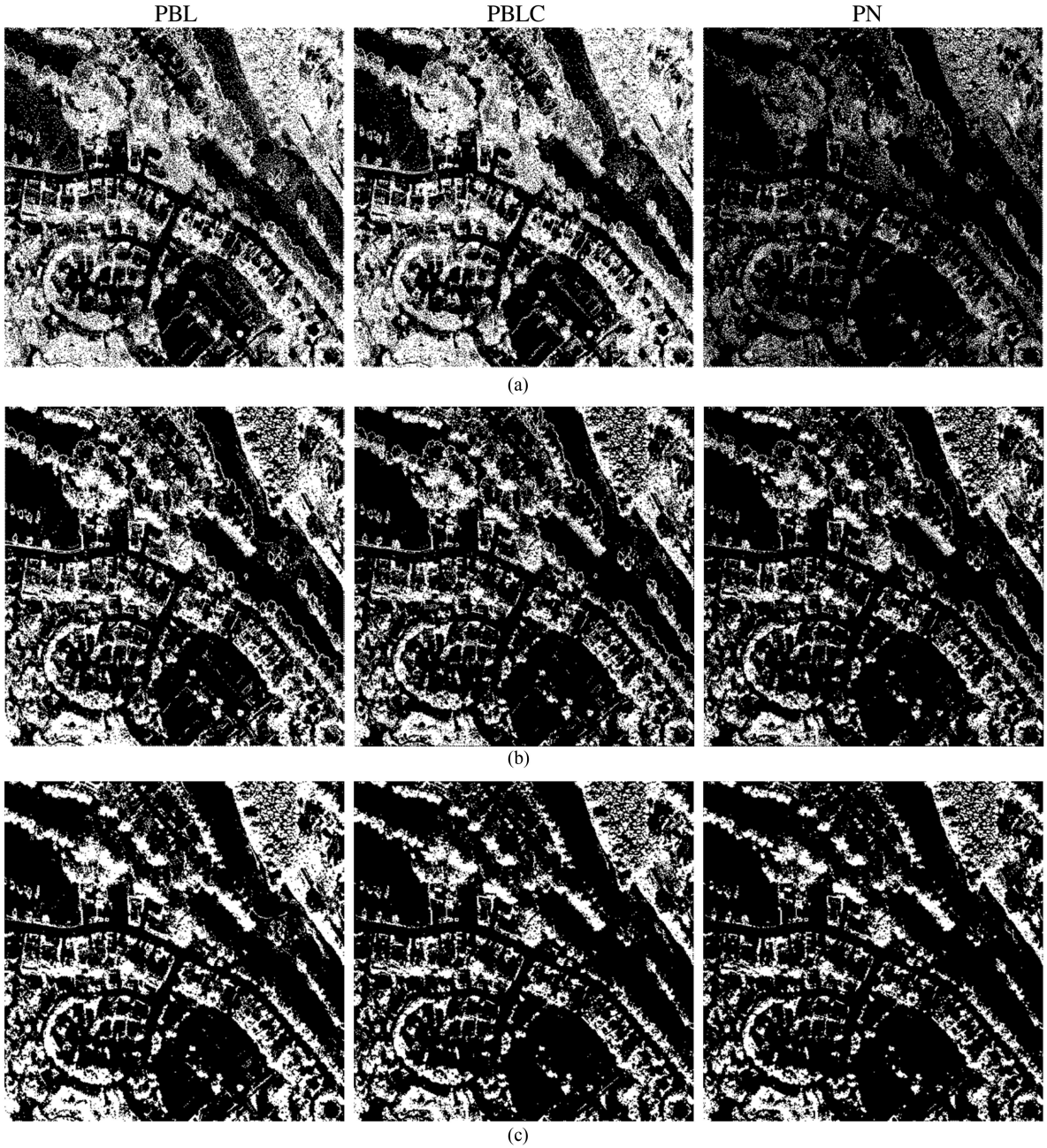


Fig. 8. Average binary classification maps of tree over ten random realizations of training sets. Classifiers were trained by PBL, PBLC, and PN. (a) GLM with $N_p = 1000$. (b) ANN with $N_p = 1000$. (c) CNN with $N_p = 5000$.

not recommended because the linear assumption can be violated easily in real-world applications [27], [28]. Instead of relying on certain parametric assumptions, PBL assumes that the posterior probabilities of selected prototypical positive data are one, but it might be difficult to select such pure positive data in practice. By contrast, PBLC makes a less strong and more flexible assumption than PBL, i.e., prior information on the maximum value of $\Pr(y = 1 \mid x)$, namely $P_{\max}$, is available. Like PBL, setting $P_{\max} = 1$ is a reasonable default choice as classes are normally separable with appropriate features in remote sensing classification, but we can further relax the assumption by setting a smaller value of $P_{\max}$ when classes are less separable in the feature space.

The regularization coefficient λ is a free parameter related to the model assumption of $P_{\max}$. According to our experiments, GLM_PBLC with $\lambda = 0$ consistently produces $P_{\max} = 1$. The reason might be that the parametric structure of GLM (i.e., linear logistic model) is sufficient to make $\Pr(y = 1 \mid x)$ identifiable as mentioned previously, so additional model assumption of $P_{\max}$ seems not necessary. On the synthetic dataset, GLM_PBLC produces high accuracies since the dataset matches the parametric form, but it produces lower accuracies

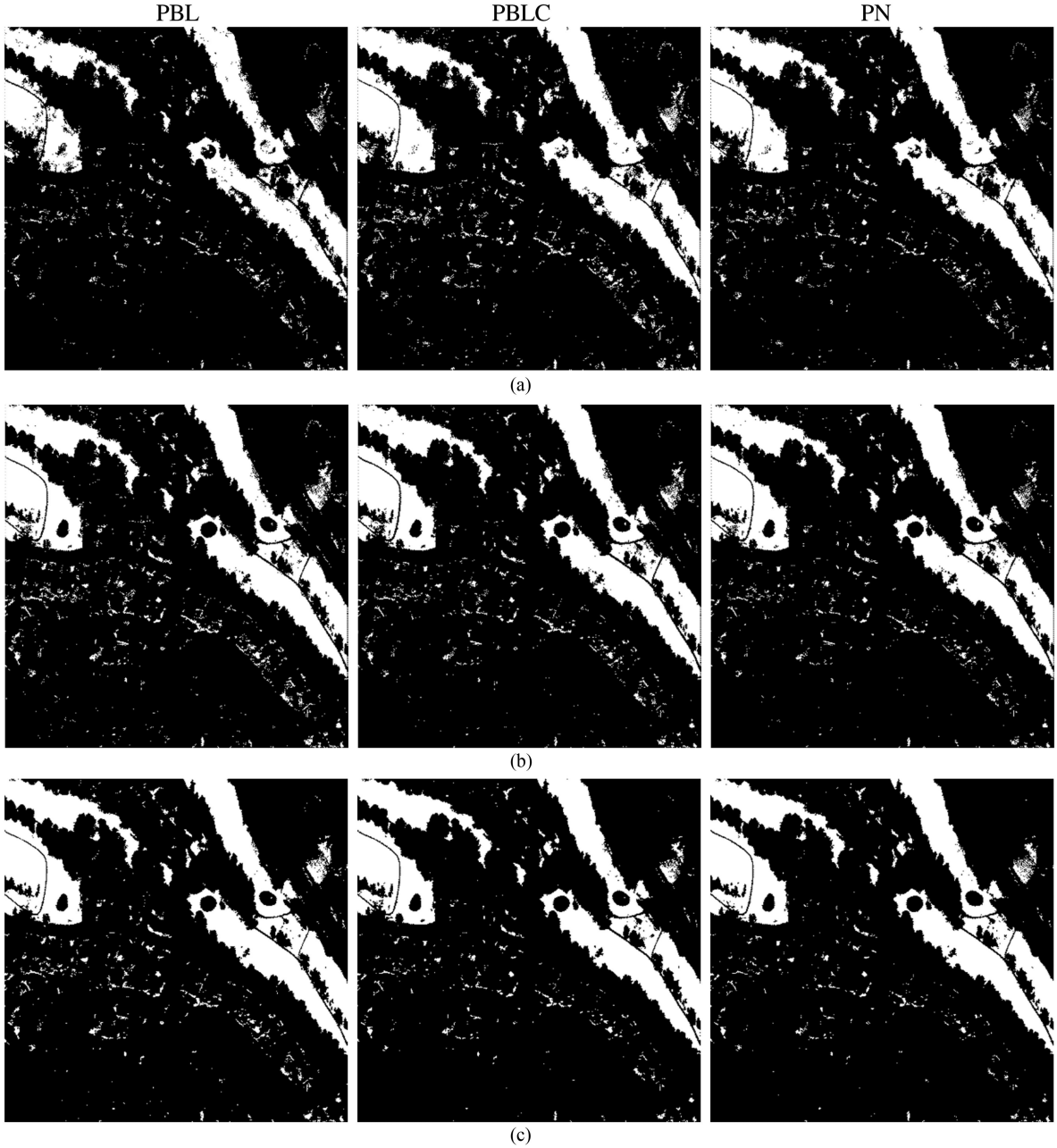


Fig. 9. Average binary classification maps of grass over ten random realizations of training sets. Classifiers were trained by PBL, PBLC, and PN. (a) GLM with $N_p = 1000$. (b) ANN with $N_p = 1000$. (c) CNN with $N_p = 5000$.

on the real aerial photograph probably because the parametric form is misspecified [7], [28]. Please also be aware that the model structure of GLM matches the assumption of $\Pr(y = 1 \mid x)$ on the synthetic dataset, but the naïve model $\Pr(s = 1 \mid x, \eta = 1)$ does not satisfy the assumption of linear logistic model anymore, so GLM is misspecified when it is used to fit the naïve model, which is the reason why GLM_PBL produces poor results on the synthetic dataset. Clearly, the advantage of GLM is that it can produce relatively high accuracies for linearly separable data, which might be the reason that GLM outperforms ANN on classifying urban and grass with a small sample size ($N_p = 200$). By contrast, nonlinear models such as ANN and CNN can deal with much more complex relationships, but they usually require a larger training set, and it is necessary to tune the regularization parameters as well. According to our test, it is not necessary to strictly force $P_{\max} = 1$ by setting a large value of $\hat{\lambda}$. Instead, allowing $P_{\max}$ to vary within a range (e.g., $0.9 \leq P_{\max} \leq 1$) using a small value of $\hat{\lambda}$ is sufficient to produce good results, and the default value $\hat{\lambda} = 0$ is fine most of the time in our experiments.

Learning a classifier without requiring labeled negative data is beneficial in many applications [7], [8], [37]. Compared with the positive class, the negative class is usually a mixture of many diverse classes, and exhaustively labeling all of the classes is labor-intensive and sometimes impossible. In the example of mapping global snow cover, the *in situ* snow cover observation

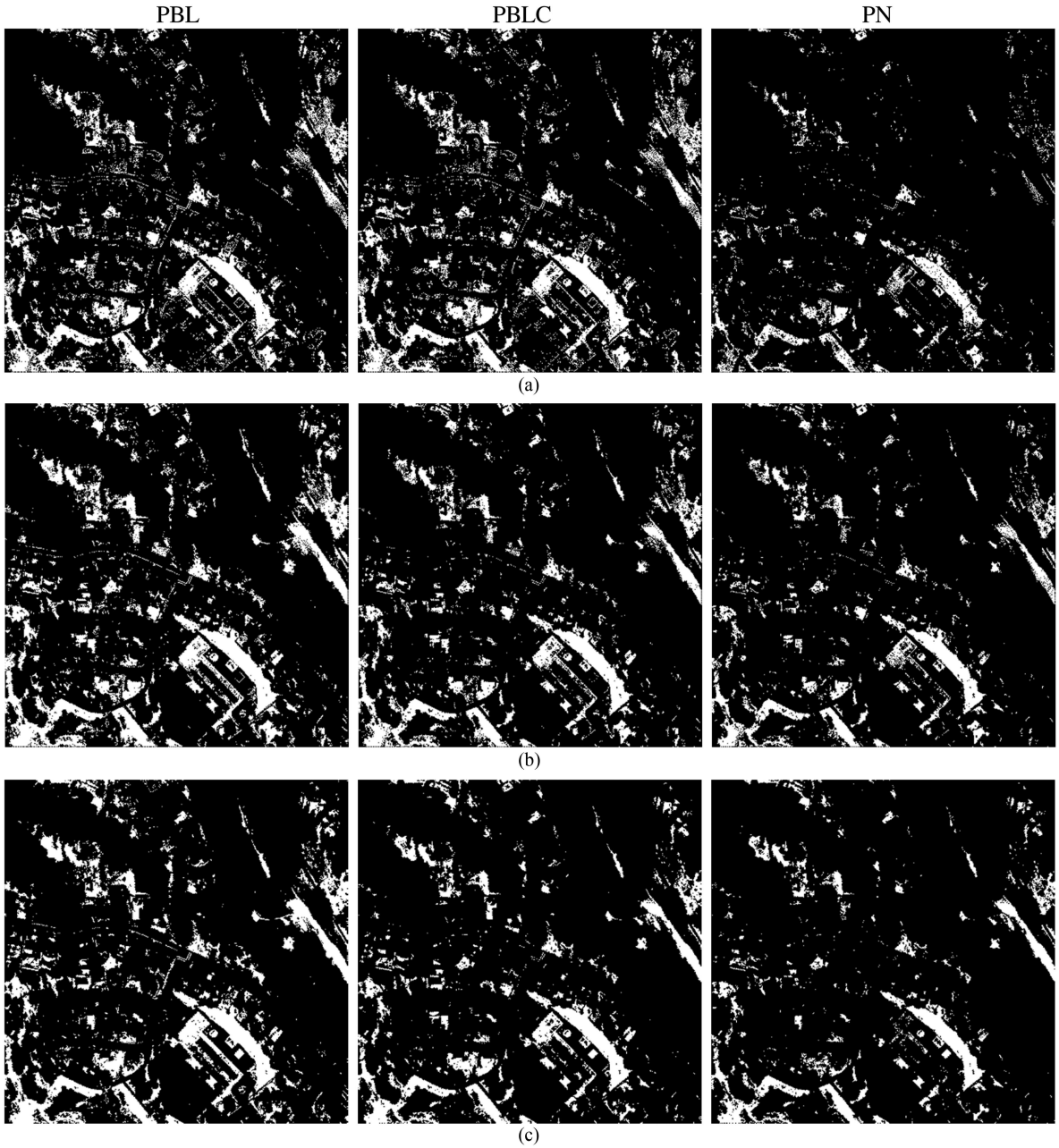


Fig. 10. Average binary classification maps of soil over ten random realizations of training sets. Classifiers were trained by PBL, PBLC, and PN. (a) GLM with $N_p = 1000$. (b) ANN with $N_p = 1000$. (c) CNN with $N_p = 5000$.

data provided by Global Surface Summary of Day product only records the observed snow cover (labeled positive) without recording observed snow absence (labeled negative) [23]. Although it is still possible to manually collect labeled negative data in order to train a standard binary classifier, learning a classifier from positive and background data is much more efficient for this one-class classification problem on a global scale. There is another category of applications where users are not able to observe reliable negative data, such as the wildland search and rescue risk assessment [9]. In the brownfield redevelopment assessment, historical observation data are used to train a model to understand the relationship between redevelopment suitability and relevant indicators (e.g., elevation, landscape fragmentation, educational facility, transport accessibility, population, etc.), but only positive samples are available in this urban renewal practice [38]. Although we only test it using a synthetic dataset and aerial photograph in this study, the proposed PBLC has the potential to be applied in other one-class classification scenarios. And users should be cautious about the validity of the model assumption when it is applied in practice.

In the current work, we only implement PBLC using GLM, ANN, and CNN as the first attempt, but other classifiers such as SVM and random forest are also possible, which should be investigated in the future. Meanwhile, random sampling is a common requirement for many statistical models, but observation data are usually biased toward areas of high spatial accessibility,

TABLE II
MODEL PERFORMANCES OF URBAN WITH DIFFERENT SAMPLE SIZES (N_{p})

	NP = 200						nT = 1000						nT = 5000					
Model	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c
GLM_PBL	**90.06**	**0.7736**	**0.7415**	**0.8105**	**0.2681**	**0.3489**	**90.58**	**0.7824**	**0.7395**	**0.8309**	**0.2633**	**0.3614**	**90.64**	**0.7848**	**0.7448**	**0.8295**	**0.2651**	**0.3585**
	0.38	*0.0100*	*0.0296*	*0.0248*	*0.0187*	*0.0232*	*0.19*	*0.0064*	*0.0152*	*0.0091*	*0.0080*	*0.0209*	*0.08*	*0.0011*	*0.0040*	*0.0054*	*0.0023*	*0.0135*
GLM_PBLC	**89.78**	**0.7905**	**0.8378**	**0.7552**	**0.2650**	**0.4433**	**90.84**	**0.8065**	**0.8324**	**0.7828**	**0.2558**	**0.4410**	**90.82**	**0.8084**	**0.8447**	**0.7754**	**0.2614**	**0.4355**
	1.43	*0.0163*	*0.0534*	*0.0579*	*0.0346*	*0.0332*	*0.24*	*0.0021*	*0.0188*	*0.0160*	*0.0093*	*0.0132*	*0.20*	*0.0017*	*0.0128*	*0.0120*	*0.0069*	*0.0179*
GLM_PN	**91.39**	**0.8037**	**0.7694**	**0.8416**	**0.2300**	**-**	**91.69**	**0.8094**	**0.7706**	**0.8527**	**0.2287**	**-**	**91.69**	**0.8092**	**0.7691**	**0.8538**	**0.2275**	**-**
	0.29	*0.0059*	*0.0137*	*0.0156*	*0.0068*	-	*0.08*	*0.0038*	*0.0129*	*0.0082*	*0.0067*	-	*0.04*	*0.0015*	*0.0063*	*0.0050*	*0.0030*	-
ANN_PBL	**85.37**	**0.7258**	**0.8430**	**0.6393**	**0.3851**	**0.3279**	**92.15**	**0.8377**	**0.8838**	**0.7963**	**0.2893**	**0.3816**	**93.34**	**0.8590**	**0.8862**	**0.8337**	**0.2667**	**0.3897**
	1.27	*0.0136*	*0.0306*	*0.0356*	*0.0321*	*0.0190*	*0.37*	*0.0067*	*0.0104*	*0.0124*	*0.0266*	*0.0104*	*0.14*	*0.0021*	*0.0107*	*0.0099*	*0.0057*	*0.0048*
ANN_PBLC	**88.16**	**0.7300**	**0.7020**	**0.7715**	**0.2264**	**0.4876**	**92.73**	**0.8389**	**0.8271**	**0.8528**	**0.2297**	**0.4823**	**93.76**	**0.8627**	**0.8552**	**0.8709**	**0.2377**	**0.4703**
	0.76	*0.0181*	*0.0673*	*0.0568*	*0.0311*	*0.0364*	*0.18*	*0.0061*	*0.0311*	*0.0226*	*0.0337*	*0.0226*	*0.16*	*0.0045*	*0.0167*	*0.0121*	*0.0259*	*0.0167*
ANN_PN	**89.23**	**0.7487**	**0.7009**	**0.8046**	**0.2320**	**-**	**93.47**	**0.8529**	**0.8262**	**0.8816**	**0.2284**	**-**	**94.08**	**0.8669**	**0.8418**	**0.8937**	**0.2275**	**-**
	0.64	*0.0169*	*0.0274*	*0.0194*	*0.0113*	-	*0.07*	*0.0022*	*0.0122*	*0.0106*	*0.0058*	-	*0.11*	*0.0027*	*0.0067*	*0.0051*	*0.0034*	-
CNN_PBL	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**93.73**	**0.8697**	**0.9124**	**0.8317**	**0.2642**	**0.4112**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.3606*	*0.0051*	*0.0185*	*0.0210*	*0.0121*	*0.0115*
CNN_PBLC	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**94.17**	**0.8720**	**0.8667**	**0.8795**	**0.2388**	**0.4550**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.4068*	*0.0090*	*0.0348*	*0.0290*	*0.0199*	*0.0129*
CNN_PN	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**94.80**	**0.8867**	**0.8887**	**0.8852**	**0.2331**	**-**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.08*	*0.0016*	*0.0157*	*0.0138*	*0.0093*	-

OA: Overall accuracy. F: F-score. PA: Producer's accuracy of positive class. UA: User's accuracy of positive class. True class prior: $\Pr(y = 1) = 0.2292$. True constant: $c = 0.4660$. Bold values refer to averages. Italic values refer to standard deviations; GLM_PBL: A GLM model trained from positive and background data using PBL; GLM_PBLC: A GLM model trained from positive and background data using PBLC; GLM_PN: A GLM model trained from positive and negative data using standard approach; ANN_PBL: An ANN model trained from positive and background data using PBL; ANN_PBLC: An ANN model trained from positive and background data using PBLC; ANN_PN: An ANN model trained from positive and negative data using standard approach; CNN_PBL: A CNN model trained from positive and background data using PBL; CNN_PBLC: A CNN model trained from positive and background data using PBLC; CNN_PN: A CNN model trained from positive and negative data using standard approach.

TABLE III
MODEL PERFORMANCES OF TREE WITH DIFFERENT SAMPLE SIZES (N_{p})

	nT = 200						nT = 1000						nT = 5000					
Model	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c
GLM_PBL	**75.02**	**0.5771**	**0.8088**	**0.4492**	**0.4338**	**0.2710**	**75.22**	**0.5847**	**0.8285**	**0.4519**	**0.4415**	**0.2731**	**74.98**	**0.5845**	**0.8357**	**0.4495**	**0.4465**	**0.2704**
	1.39	*0.0071*	*0.0284*	*0.0137*	*0.0234*	*0.0121*	*0.54*	*0.0024*	*0.0124*	*0.0058*	*0.0094*	*0.0089*	*0.41*	*0.0022*	*0.0068*	*0.0043*	*0.0065*	*0.0104*
GLM_PBLC	**73.43**	**0.5626**	**0.8132**	**0.4376**	**0.4021**	**0.3574**	**74.68**	**0.5816**	**0.8348**	**0.4469**	**0.3975**	**0.3476**	**75.45**	**0.5848**	**0.8208**	**0.4546**	**0.3844**	**0.3484**
	3.22	*0.0153*	*0.1029*	*0.0375*	*0.0615*	*0.0301*	*1.34*	*0.0046*	*0.0308*	*0.0139*	*0.0316*	*0.0206*	*0.97*	*0.0027*	*0.0251*	*0.0105*	*0.0297*	*0.0142*
GLM_PN	**80.41**	**0.4160**	**0.3372**	**0.5585**	**0.2089**	**-**	**80.44**	**0.4124**	**0.3264**	**0.5612**	**0.2093**	**-**	**80.50**	**0.4080**	**0.3193**	**0.5657**	**0.2087**	**-**
	0.34	*0.0541*	*0.0697*	*0.0122*	*0.0111*	-	*0.05*	*0.0131*	*0.0171*	*0.0029*	*0.0046*	-	*0.08*	*0.0128*	*0.0159*	*0.0030*	*0.0044*	-
ANN_PBL	**76.29**	**0.6059**	**0.8572**	**0.4712**	**0.4545**	**0.2784**	**84.49**	**0.6977**	**0.8495**	**0.5924**	**0.3469**	**0.3345**	**85.50**	**0.7151**	**0.8642**	**0.6100**	**0.3277**	**0.3441**
	3.95	*0.0261*	*0.0399*	*0.0381*	*0.0537*	*0.0171*	*0.70*	*0.0062*	*0.0152*	*0.0154*	*0.0335*	*0.0143*	*0.27*	*0.0023*	*0.0080*	*0.0066*	*0.0053*	*0.0046*
ANN_PBLC	**82.33**	**0.5755**	**0.5839**	**0.5900**	**0.2625**	**0.4730**	**86.49**	**0.7007**	**0.7510**	**0.6591**	**0.2676**	**0.4551**	**87.85**	**0.7160**	**0.7283**	**0.7058**	**0.2284**	**0.4792**
	0.95	*0.0510*	*0.1267*	*0.0421*	*0.0655*	*0.0468*	*0.60*	*0.0057*	*0.0381*	*0.0260*	*0.0461*	*0.0208*	*0.22*	*0.0061*	*0.0316*	*0.0189*	*0.0231*	*0.0144*
ANN_PN	**83.25**	**0.4928**	**0.3899**	**0.6799**	**0.2048**	**-**	**87.67**	**0.6991**	**0.6805**	**0.7193**	**0.2089**	**-**	**88.77**	**0.7204**	**0.6871**	**0.7576**	**0.2091**	**-**
	0.60	*0.0430*	*0.0533*	*0.0242*	*0.0075*	-	*0.12*	*0.0060*	*0.0175*	*0.0090*	*0.0045*	-	*0.15*	*0.0052*	*0.0148*	*0.0102*	*0.0044*	-
CNN_PBL	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**86.96**	**0.7356**	**0.8600**	**0.6437**	**0.3042**	**0.3651**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.81*	*0.0079*	*0.0239*	*0.0236*	*0.0215*	*0.0119*
CNN_PBLC	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**88.76**	**0.7382**	**0.7539**	**0.7270**	**0.2453**	**0.4559**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.44*	*0.0082*	*0.0467*	*0.0306*	*0.0273*	*0.0122*
CNN_PN	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**89.29**	**0.7366**	**0.7130**	**0.7652**	**0.2104**	**-**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.20*	*0.0138*	*0.0438*	*0.0251*	*0.0181*	-

OA: Overall accuracy. F: F-score. PA: Producer's accuracy of positive class. UA: User's accuracy of positive class. True class prior: $\Pr(y = 1) = 0.2106$. True constant: $c = 0.4871$. Bold values refer to averages. Italic values refer to standard deviations; GLM_PBL: A GLM model trained from positive and background data using PBL; GLM_PBLC: A GLM model trained from positive and background data using PBLC; GLM_PN: A GLM model trained from positive and negative data using standard approach; ANN_PBL: An ANN model trained from positive and background data using PBL; ANN_PBLC: An ANN model trained from positive and background data using PBLC; ANN_PN: An ANN model trained from positive and negative data using standard approach; CNN_PBL: A CNN model trained from positive and background data using PBL; CNN_PBLC: A CNN model trained from positive and background data using PBLC; CNN_PN: A CNN model trained from positive and negative data using standard approach.

TABLE IV
MODEL PERFORMANCES OF GRASS WITH DIFFERENT SAMPLE SIZES (N_{p})

	nT = 200						nT = 1000						nT = 5000					
Model	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c
GLM_PBL	**93.83**	**0.8192**	**0.7447**	**0.9110**	**0.1959**	**0.4276**	**94.27**	**0.8321**	**0.7556**	**0.9262**	**0.1971**	**0.4220**	**94.33**	**0.8335**	**0.7547**	**0.9307**	**0.1953**	**0.4226**
	0.48	*0.0158*	*0.0250*	*0.0142*	*0.0124*	*0.0345*	*0.13*	*0.0047*	*0.0112*	*0.0086*	*0.0052*	*0.0128*	*0.07*	*0.0022*	*0.0043*	*0.0038*	*0.0029*	*0.0128*
GLM_PBLC	**94.55**	**0.8479**	**0.8131**	**0.8893**	**0.1816**	**0.5361**	**95.01**	**0.8660**	**0.8576**	**0.8750**	**0.1954**	**0.5046**	**95.09**	**0.8681**	**0.8590**	**0.8776**	**0.1951**	**0.5021**
	0.54	*0.0211*	*0.0510*	*0.0239*	*0.0169*	*0.0426*	*0.13*	*0.0031*	*0.0132*	*0.0133*	*0.0056*	*0.0150*	*0.05*	*0.0014*	*0.0104*	*0.0092*	*0.0057*	*0.0114*
GLM_PN	**95.09**	**0.8657**	**0.8412**	**0.8920**	**0.1912**	**-**	**95.30**	**0.8700**	**0.8370**	**0.9058**	**0.1870**	**-**	**95.33**	**0.8712**	**0.8410**	**0.9038**	**0.1885**	**-**
	0.17	*0.0032*	*0.0105*	*0.0154*	*0.0047*	-	*0.05*	*0.0022*	*0.0088*	*0.0063*	*0.0041*	-	*0.04*	*0.0014*	*0.0050*	*0.0042*	*0.0030*	-
ANN_PBL	**90.11**	**0.7723**	**0.8728**	**0.6983**	**0.3680**	**0.3422**	**96.05**	**0.8952**	**0.8980**	**0.8928**	**0.2143**	**0.4524**	**96.35**	**0.9032**	**0.9051**	**0.9014**	**0.2202**	**0.4540**
	2.98	*0.0471*	*0.0199*	*0.0809*	*0.0653*	*0.0189*	*0.14*	*0.0026*	*0.0097*	*0.0131*	*0.0239*	*0.0148*	*0.06*	*0.0010*	*0.0064*	*0.0080*	*0.0306*	*0.0131*
ANN_PBLC	**93.55**	**0.7930**	**0.6678**	**0.9851**	**0.1673**	**0.5820**	**96.28**	**0.8982**	**0.8729**	**0.9255**	**0.1739**	**0.5429**	**96.50**	**0.9044**	**0.8811**	**0.9291**	**0.1787**	**0.5303**
	1.30	*0.0530*	*0.0757*	*0.0099*	*0.0344*	*0.0480*	*0.09*	*0.0033*	*0.0163*	*0.0137*	*0.0114*	*0.0208*	*0.05*	*0.0012*	*0.0066*	*0.0071*	*0.0048*	*0.0105*
ANN_PN	**93.51**	**0.7912**	**0.6634**	**0.9884**	**0.1807**	**-**	**96.50**	**0.9034**	**0.8710**	**0.9384**	**0.1858**	**-**	**96.67**	**0.9081**	**0.8766**	**0.9421**	**0.1895**	**-**
	1.30	*0.0516*	*0.0751*	*0.0082*	*0.0061*	-	*0.06*	*0.0021*	*0.0087*	*0.0074*	*0.0034*	-	*0.02*	*0.0008*	*0.0065*	*0.0062*	*0.0033*	-
CNN_PBL	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**96.43**	**0.9060**	**0.9134**	**0.8994**	**0.2034**	**0.4672**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.32*	*0.0066*	*0.0111*	*0.0214*	*0.0074*	*0.0102*
CNN_PBLC	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**96.55**	**0.9045**	**0.8724**	**0.9407**	**0.1839**	**0.5136**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.23*	*0.0090*	*0.0316*	*0.0197*	*0.0101*	*0.0126*
CNN_PN	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**96.86**	**0.9143**	**0.8900**	**0.9404**	**0.1849**	**-**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.08*	*0.0028*	*0.0152*	*0.0127*	*0.0069*	-

OA: Overall accuracy. F: F-score. PA: producer's accuracy of positive class. UA: User's accuracy of positive class. True class prior: $\Pr(y = 1) = 0.1880$. True constant: $c = 0.5154$. Bold values refer to averages. Italic values refer to standard deviations; GLM_PBL: A GLM model trained from positive and background data using PBL; GLM_PBLC: A GLM model trained from positive and background data using PBLC; GLM_PN: A GLM model trained from positive and negative data using standard approach; ANN_PBL: An ANN model trained from positive and background data using PBL; ANN_PBLC: An ANN model trained from positive and background data using PBLC; ANN_PN: An ANN model trained from positive and negative data using standard approach; CNN_PBL: A CNN model trained from positive and background data using PBL; CNN_PBLC: A CNN model trained from positive and background data using PBLC; CNN_PN: A CNN model trained from positive and negative data using standard approach.

TABLE V
MODEL PERFORMANCES OF SOIL WITH DIFFERENT SAMPLE SIZES (N_{p})

	nT = 200						nT = 1000						nT = 5000					
Model	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c	OA(%)	F	PA	UA	Pr(y = 1)	c
GLM_PBL	**90.30**	**0.5922**	**0.6888**	**0.5204**	**0.1979**	**0.4103**	**90.58**	**0.6018**	**0.6956**	**0.5310**	**0.2011**	**0.4160**	**90.73**	**0.6054**	**0.6951**	**0.5363**	**0.1995**	**0.4132**
	0.42	*0.0068*	*0.0290*	*0.0151*	*0.0140*	*0.0296*	*0.40*	*0.0052*	*0.0190*	*0.0162*	*0.0106*	*0.0170*	*0.11*	*0.0021*	*0.0064*	*0.0050*	*0.0035*	*0.0139*
GLM_PBLC	**90.20**	**0.5906**	**0.6899**	**0.5191**	**0.1580**	**0.5657**	**90.62**	**0.6040**	**0.6991**	**0.5344**	**0.1644**	**0.5622**	**90.87**	**0.6077**	**0.6912**	**0.5434**	**0.1538**	**0.5700**
	0.86	*0.0106*	*0.0411*	*0.0294*	*0.0133*	*0.0323*	*0.60*	*0.0043*	*0.0393*	*0.0274*	*0.0268*	*0.0288*	*0.38*	*0.0021*	*0.0258*	*0.0172*	*0.0085*	*0.0130*
GLM_PN	**91.98**	**0.5168**	**0.4218**	**0.6720**	**0.1037**	**-**	**92.07**	**0.5099**	**0.4041**	**0.6928**	**0.1002**	**-**	**92.07**	**0.5087**	**0.4021**	**0.6944**	**0.1001**	**-**
	0.24	*0.0336*	*0.0439*	*0.0123*	*0.0069*	-	*0.07*	*0.0168*	*0.0235*	*0.0094*	*0.0043*	-	*0.07*	*0.0166*	*0.0232*	*0.0087*	*0.0052*	-
ANN_PBL	**87.17**	**0.5711**	**0.8270**	**0.4380**	**0.2379**	**0.4101**	**91.14**	**0.6540**	**0.8172**	**0.5457**	**0.1895**	**0.4656**	**91.65**	**0.6700**	**0.8282**	**0.5628**	**0.1863**	**0.4719**
	2.00	*0.0333*	*0.0245*	*0.0407*	*0.0208*	*0.0210*	*0.50*	*0.0083*	*0.0174*	*0.0186*	*0.0201*	*0.0168*	*0.31*	*0.0049*	*0.0126*	*0.0125*	*0.0267*	*0.0125*
ANN_PBLC	**92.08**	**0.5541**	**0.5053**	**0.6611**	**0.1194**	**0.6882**	**92.62**	**0.6616**	**0.7058**	**0.6251**	**0.1272**	**0.6272**	**93.07**	**0.6770**	**0.7107**	**0.6475**	**0.1279**	**0.6182**
	0.48	*0.0922*	*0.1381*	*0.0545*	*0.0334*	*0.0586*	*0.37*	*0.0039*	*0.0370*	*0.0252*	*0.0100*	*0.0205*	*0.18*	*0.0038*	*0.0261*	*0.0168*	*0.0119*	*0.0180*
ANN_PN	**92.65**	**0.5516**	**0.4513**	**0.7320**	**0.1011**	**-**	**93.44**	**0.6578**	**0.6166**	**0.7059**	**0.1036**	**-**	**93.98**	**0.6783**	**0.6206**	**0.7487**	**0.1026**	**-**
	0.37	*0.0650*	*0.0817*	*0.0286*	*0.0043*	-	*0.05*	*0.0081*	*0.0227*	*0.0129*	*0.0044*	-	*0.15*	*0.0102*	*0.0195*	*0.0147*	*0.0033*	-
CNN_PBL	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**91.94**	**0.6793**	**0.8291**	**0.5778**	**0.1696**	**0.4699**
	-	-	-	-	-	-	-	-	-	-	-	-	*1.17*	*0.0265*	*0.0236*	*0.0449*	*0.0174*	*0.0390*
CNN_PBLC	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**93.41**	**0.6878**	**0.7153**	**0.6737**	**0.1322**	**0.6007**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.41*	*0.0256*	*0.0837*	*0.0470*	*0.0181*	*0.0167*
CNN_PN	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**-**	**94.35**	**0.6996**	**0.6490**	**0.7738**	**0.1025**	**-**
	-	-	-	-	-	-	-	-	-	-	-	-	*0.30*	*0.0274*	*0.0785*	*0.0549*	*0.0149*	-

OA: Overall accuracy. F: F-score. PA: Producer's accuracy of positive class. UA: User's accuracy of positive class. True class prior: $\Pr(y = 1) = 0.1023$. True constant: $c = 0.6616$. Bold values refer to averages. Italic values refer to standard deviations; GLM_PBL: A GLM model trained from positive and background data using PBL; GLM_PBLC: A GLM model trained from positive and background data using PBLC; GLM_PN: A GLM model trained from positive and negative data using standard approach; ANN_PBL: An ANN model trained from positive and background data using PBL; ANN_PBLC: An ANN model trained from positive and background data using PBLC; ANN_PN: An ANN model trained from positive and negative data using standard approach; CNN_PBL: A CNN model trained from positive and background data using PBL; CNN_PBLC: A CNN model trained from positive and background data using PBLC; CNN_PN: A CNN model trained from positive and negative data using standard approach.

and spatial uncertainties and outliers may exist as well [20], [39]–[43]. For example, the wildland search and rescue incident occurrence data of Yosemite National Park are georeferenced from textual locality descriptions that are associated with large uncertainties, so the georeferenced coordinate of a positive sample point might be actually attached to a negative point [44]. These issues may affect the model performance and how to debias/denoise the observation data requires future research as well. Furthermore, future research could consider introducing a spatiotemporal autocorrelation of environmental variables to improve model performance [45].

V. Conclusion

In this study, we propose a novel PBLC algorithm to address the one-class classification problem in the case-control sampling scenario. The algorithm trains a binary classifier from positive and background data, with user-specified prior information on the maximum value of posterior probability. Unlike PBL that post-calibrates the trained model in a separate process, PBLC infers the desired model directly in the one-step training with more accurate estimates of model parameters. Using both synthetic and real aerial photograph datasets, we show that PBLC can successfully train linear and nonlinear classifiers including GLM, ANN, and CNN, providing more accurate probabilistic and binary predictions than the PBL algorithm. Without requiring labeled negative data, the proposed PBLC algorithm has the potential to solve one-class classification problems in relevant fields.

Acknowledgment

The authors would like to thank the Editors and three anonymous Reviewers for their constructive comments that significantly strengthened this article.

References

[1] G. M. Foody *et al.*, "Training set size requirements for the classification of a specific class," *Remote Sens. Environ.*, vol. 104, no. 1, pp. 1–14, Sep. 2006.

[2] B. Song, P. Li, J. Li, and A. Plaza, "One-class classification of remote sensing images using kernel sparse representation," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 9, no. 4, pp. 1613–1623, Apr. 2016.

[3] X. Wang and P. Li, "Extraction of earthquake-induced collapsed buildings using very high-resolution imagery and airborne Lidar data," *Int. J. Remote Sens.*, vol. 36, no. 8, pp. 2163–2183, Apr. 2015.

[4] J. Muñoz-Marí, L. Bruzzone, and G. Camps-Vails, "A support vector domain description approach to supervised classification of remote sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 8, pp. 2683–2692, Aug. 2007.

[5] Q. Guo *et al.*, "A framework for supervised image classification with incomplete training samples," *Photogram. Eng. Remote Sens.*, vol. 78, no. 6, pp. 595–604, 2012.

[6] G. M. Foody, "Estimation of sub-pixel land cover composition in the presence of untrained classes," *Comput. Geosci.*, vol. 26, no. 4, pp. 469–478, 2000.

[7] C. Elkan and K. Noto, "Learning classifiers from only positive and unlabeled data," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2008, pp. 213–220.

[8] C. Xiao *et al.*, "A new method of pseudo absence data generation in landslide susceptibility mapping with a case study of Shenzhen," *Sci. China Technol. Sci.*, vol. 53, no. 1, pp. 75–84, May 2010.

[9] P. J. Doherty *et al.*, "Space-time analyses for forecasting future incident occurrence: A case study from yosemite national park using the presence and background learning algorithm," *Int. J. Geograph. Inf. Sci.*, vol. 28, no. 5, pp. 910–927, May 2014.

[10] D. M. J. Tax, "One-class classification, concept-learning in the absence of counter-examples," Ph.D. dissertation, Delft Univ. Technol., Delft, The Netherlands, 2001.

[11] C. Sanchez-Hernandez, D. S. Boyd, and G. M. Foody, "One-class classification for mapping a specific land-cover class: SVDD classification of fenland," *IEEE Trans. Geosci. Remote Sens.*, vol. 45, no. 4, pp. 1061–1073, Apr. 2007.

[12] B. Schölkopf *et al.*, "Estimating the support of a high-dimensional distribution," *Neural Comput.*, vol. 13, no. 7, pp. 1443–1471, Jul. 2001.

[13] W. Li and Q. Guo, "A new accuracy assessment method for one-class remote sensing classification," *IEEE Trans. Geosci. Remote Sens.*, vol. 52, no. 8, pp. 4621–4632, Aug. 2014.

[14] V. Castelli and T. M. Cover, "The relative value of labeled and unlabeled samples in pattern recognition with an unknown mixing parameters," *IEEE Trans. Inf. Theory*, vol. 42, no. 6, pp. 2102–2117, Nov. 1996.

[15] W. Li, Q. Guo, and C. Elkan, "A positive and unlabeled learning algorithm for one-class classification of remote-sensing data," *IEEE Trans. Geosci. Remote Sens.*, vol. 49, no. 2, pp. 717–725, Feb. 2011.

[16] H. Yu, "Single-class classification with mapping convergence," *Mach. Learn.*, vol. 61, no. 1, pp. 49–69, Nov. 2005.

[17] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu, "Building text classifiers using positive and unlabeled examples," in *Proc. 3rd IEEE Int. Conf. Data Mining*, 2003, pp. 179–186.

[18] X. Chen *et al.*, "Effect of training strategy for positive and unlabelled learning classification: Test on Landsat imagery," *Remote Sens. Lett.*, vol. 7, no. 11, pp. 1063–1072, Nov. 2016.

[19] L. Bruzzone, M. Chi, and M. Marconcini, "A novel transductive SVM for semisupervised classification of remote-sensing images," *IEEE Trans. Geosci. Remote Sens.*, vol. 44, no. 11, pp. 3363–3373, Nov. 2006.

[20] L. Bruzzone and C. Persello, "A novel context-sensitive semisupervised SVM classifier robust to mislabeled training samples," *IEEE Trans. Geosci. Remote Sens.*, vol. 47, no. 7, pp. 2142–2154, Jul. 2009.

[21] L. Gómez-Chova *et al.*, "Semisupervised image classification with Laplacian support vector machines," *IEEE Geosci. Remote Sens. Lett.*, vol. 5, no. 3, pp. 336–340, Jul. 2008.

[22] W. Li, Q. Guo, and C. Elkan, "Can we model the probability of presence of species without absence data?" *Ecography*, vol. 34, no. 6, pp. 1096–1105, 2011.

[23] X. Xu *et al.*, "Global snow cover estimation with microwave brightness temperature measurements and one-class in situ observations," *Remote Sens. Environ.*, vol. 182, pp. 227–251, Sep. 2016.

[24] Z. Ao *et al.*, "One-class classification of airborne LiDAR data in urban areas using a presence and background learning algorithm," *Remote Sens.*, vol. 9, no. 10, 2017, Art. no. 1001.

[25] Z. Zhang *et al.*, "Automated surface water extraction combining Sentinel-2 imagery and openstreetmap using presence and background learning (PBL) algorithm," *IEEE J. Sel. Topics Appl. Earth Observ. Remote Sens.*, vol. 12, no. 10, pp. 3784–3798, Oct. 2019.

[26] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.

[27] G. Ward *et al.*, "Presence-only data and the EM algorithm," *Biometrics*, vol. 65, no. 2, pp. 554–563, 2009.

[28] T. Hastie and W. Fithian, "Inference from presence-only data; the ongoing controversy," *Ecography*, vol. 36, no. 8, pp. 864–867, 2013.

[29] A. Guisan, T. C. Edwards, and T. Hastie, "Generalized linear and generalized additive models in studies of species distributions: Setting the scene," *Ecological Model.*, vol. 157, no. 2, pp. 89–100, Nov. 2002.

[30] M. D. Richard and R. P. Lippmann, "Neural network classifiers estimate Bayesian a posteriori probabilities," *Neural Comput.*, vol. 3, no. 4, pp. 461–483, 1991.

[31] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proc. IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov. 1998.

[32] L. Zhang, L. Zhang, and B. Du, "Deep learning for remote sensing data: A technical tutorial on the state-of-the-art," *IEEE Geosci. Remote Sens. Mag.*, vol. 4, no. 2, pp. 22–40, Jun. 2016.

[33] Q. Yuan *et al.*, "Deep learning in environmental remote sensing: Achievements and challenges," *Remote Sens. Environ.*, vol. 241, May 2020, Art. no. 111716.

[34] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," in *Proc. 12th USENIX Conf. Oper. Syst. Des. Implementation*, 2015, pp. 265–283.

[35] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. 3rd Int. Conf. Learn. Representations*, 2015, pp. 1–15.

[36] K. Jaskie and A. Spanias, "Positive and unlabeled learning algorithms and applications: A survey," in *Proc. 10th Int. Conf. Inf., Intell., Syst. Appl.*, 2019, pp. 1–8.
[37] L. Gao *et al.*, "One-class classification for highly imbalanced medical image data," *Proc. SPIE*, vol. 11318, Mar. 2020, Art. no. 11318C.
[38] Y. Liu *et al.*, "Land-use decision support in brownfield redevelopment for urban renewal based on crowdsourced data and a presence-and-background learning (PBL) method," *Land Use Policy*, vol. 88, Nov. 2019, Art. no. 104188.
[39] M. Pal and P. M. Mather, "Support vector machines for classification in remote sensing," *Int. J. Remote Sens.*, vol. 26, no. 5, pp. 1007–1011, 2005.
[40] J. J. Heckman, "Sample selection bias as a specification error," *Econometrica*, vol. 47, no. 1, pp. 153–161, 1979.
[41] B. Zadrozny, "Learning and evaluating classifiers under sample selection bias," in *Proc. 20st Int. Conf. Mach. Learn.*, , 2004, pp. 903–910.
[42] A. T. Smith and C. Elkan, "Making generative classifiers robust to selection bias," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, 2007, pp. 657–666.
[43] J. Wieczorek, Q. Guo, and R. Hijmans, "The point-radius method for georeferencing locality descriptions and calculating associated uncertainty," *Int. J. Geograph. Inf. Sci.*, vol. 18, no. 8, pp. 745–767, Dec. 2004.
[44] P. Doherty *et al.*, "Georeferencing incidents from locality descriptions and its applications: A case study from Yosemite national park search and rescue," *Trans. GIS*, vol. 15, no. 6, pp. 775–793, 2011.
[45] Q. Yuan *et al.*, "Deep learning in environmental remote sensing: Achievements and challenges," *Remote Sens. Environ.*, vol. 241, 2020, Art. no. 111716.

Wenkai Li received the B.S. degree in environmental science from Sun Yat-Sen University, Guangzhou, China, in 2005, the M.S. degree in environmental engineering from Peking University, Beijing, China, in 2008, and the Ph.D. degree in environmental systems from University of California, Merced, CA, USA, in 2013.

He is currently an Associate Professor with the School of Geography and Planning, Sun Yat-Sen University, Guangzhou, China. His research interests include image classification, ecological niche modeling, and radiative transfer modeling.

Qinghua Guo received the B.S. degree in environmental science and the M.S. degree in remote sensing and geographic information system from Peking University, Beijing, China, in 1996 and 1999, respectively, and the Ph.D. degree in environmental science from the University of California, Berkeley, CA, USA, in 2005.

He is a Professor with the Institute of Botany, Chinese Academy of Sciences, Beijing. He is also an Adjunct Professor and a Member of the founding faculty in the School of Engineering, University of California at Merced, Merced, CA, USA. He has published more than 100 journal articles, and has developed an integrated platform for modeling geographic one class data. His recent research interests include GIS and remote sensing algorithm development and their environmental applications, such as object-based image analysis, geographic one-class data analysis, and LiDAR data processing.

Charles Elkan received the B.S. degree from Cambridge University, Cambridge, U.K., in 1984, and the Ph.D. degree from Cornell University, Ithaca, NY, USA, in 1990.

In 2005 and 2006, he was on sabbatical with the Massachusetts Institute of Technology, Cambridge, and in 1998 and 1999, he was a Visiting Associate Professor with Harvard University, Cambridge. He is currently a Professor with the Department of Computer Science and Engineering, University of California, San Diego. He is known for his research in machine learning, data mining, and computational biology. The MEME algorithm he developed with his Ph.D. student T. Bailey has been used in over 1000 publications in biology.

Dr. Elkan was the recipient of several best paper awards and data mining contests awards.