Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Iterative Metric Learning for Imbalance Data Classification

Nan Wang, Xibin Zhao*, Yu Jiang, Yue Gao*
BNRist, KLISS, School of Software, Tsinghua University, China
n-wang16@mails.tsinghua.edu.cn
{zxb, jy1989, gaoyue}@tsinghua.edu.cn

Abstract

In many classification applications, the amount of data from different categories usually vary significantly, such as software defect predication and medical diagnosis. Under such circumstances, it is essential to propose a proper method to solve the imbalance issue among the data. However, most of the existing methods mainly focus on improving the performance of classifiers rather than searching for an appropriate way to find an effective data space for classification. In this paper, we propose a method named Iterative Metric Learning (IML) to explore the correlations among the imbalance data and construct an effective data space for classification. Given the imbalance training data, it is important to select a subset of training samples for each testing data. Thus, we aim to find a more stable neighborhood for the testing data using the iterative metric learning strategy. To evaluate the effectiveness of the proposed method, we have conducted experiments on two groups of dataset, *i.e.*, the NASA Metrics Data Program (NASA) dataset and UCI Machine Learning Repository (UCI) dataset. Experimental results and comparisons with state-of-the-art methods have exhibited better performance of our proposed method.

*

1 Introduction

In most real-world classification tasks, the imbalance problem occurs frequently. For example, in software defect prediction field, the data is significantly unbalanced. The number of defect-free samples is far more than defect-prone samples. Another example is in the medical diagnosis field, where the data of illness cases substantially less than the health cases. Moreover, Lack of labeled training data may influence the accuracy of the classification model. However, many classification methods assume that training data is balance and enough, and not take the quality of the data space into account.

In recent years, imbalance data classification attracts more attention [Mesquita *et al.*, 2016; Liu *et al.*, 2015; Sun *et al.*, 2012; Tomar and Agarwal, 2016] from industry and academia. Although many research works concern about improving the performance of learning methods, it is noted that the performances of many state-of-the-art methods are widely depended on the quality of the training data. Thus, sampling methods have been used to solve the imbalance issues[Barua *et al.*, 2014; He and Garcia, 2009; Sobhani *et al.*, 2014]. The sampling methods aim to balance the distribution between the majority class and the minority class. These techniques generally consist of oversampling the minority, undersampling the majority class or a combination of these two methods. Considering that the limited of training data is another challenge for many imbalance classification tasks and the sampling methods will reduce the number of the training data, a more stable and effective data space need to be constructed to improve the performance of the classifiers.

To tackle these issues, in this paper, we propose to find a more stable neighborhood space for the testing data using the iterative metric learning strategy. The framework of our proposed method is exhibited in Figure 1, which consists of three components: iterative metric learning, iterative samples selection and training samples matching. To explore a stable neighborhood space for a testing data under the scenario of imbalance training data. Given the imbalance training data, it is important to select a subset of training samples for each testing data. In this method, due to the limited of the training data, we employ metric learning to embed the original data space into a more effective data space, in which the boundaries between different classes become more clearly. Further more, to solve imbalanced issue among training data, we employ sample selection process to select the most related samples. Considering that conducting metric learning process once may be unable to construct the most effective and stable neighbor for the testing data, we iteratively conduct metric learning process until the selected samples are relatively stable and construct a more stable and effective data space for testing data.

The rest of the paper is organized as follows. Section 2 introduces related work about imbalance data classification. The proposed method is introduced in Section 3. Experimen-

*indicates corresponding authors.

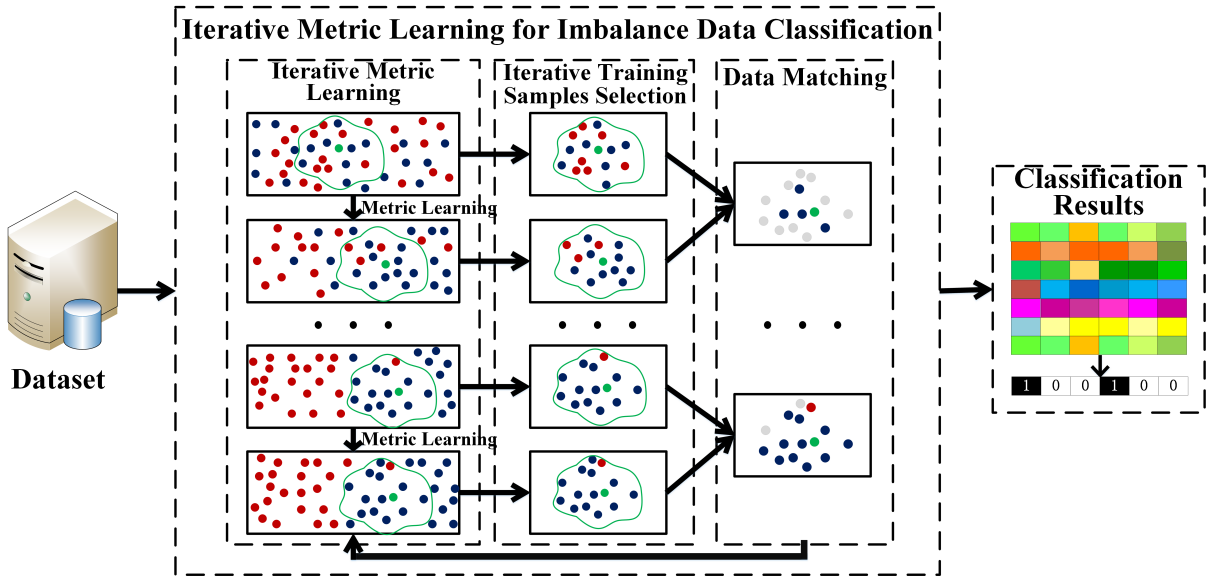


Figure 1: Illustration of the framework of our proposed iterative metric learning for imbalance data classification.

tal results and discussions are provided in Section 4. Finally, we conclude this paper in Section 5.

2 Related Works

Most of traditional classification works assume that the data space is balance and sufficient. Thus, many traditional machine learning algorithms have been widely employed in constructing classifier, such as Support Vector Machines (SVM) [Yan *et al.*, 2010], Bayesian network [Jeet *et al.*, 2011] and so on. However, in fact, the imbalance issue widely exists in many classification applications and the amount of minor class is limited, such as medical diagnosis, fraud defection and so on. Due to the specific of imbalance data, the traditional classifiers need to be further modified.

In order to increase the accuracy of unbalance data classifier, many research works concerned about imbalance classification field. For instance, Mesquita *et al.* [Mesquita *et al.*, 2016] employed reject option into weighted extreme learning machine and proposed rejoELM. Based on reioELM, they further proposed its variant, IrejoELM, and made the model possible to postpone the final decision of non-classified modules. Moreover, Sun *et al.* [Sun *et al.*, 2012] proposed a method which transformed the binary classification work into a multi-classification work, and the method can effectively handle the highly imbalanced data. Recently, cost-sensitive learning is widely applied to solve the imbalanced problem. Divya *et al.* [Tomar and Agarwal, 2016] proposed a method named WLSTSVM to improve the performance by assigning higher misclassification cost to minority. Zhang *et al.* [Zhang *et al.*, 2017] developed a nonnegative sparse graph by label propagation, and they also took information from copious unlabeled data into consideration.

Considering that the performances of classifiers largely depend on the quality of the data space and some research works concerned about data distribution demonstrate that multi-metrics limits the performance of the classifier due to its abundantly unrelated metrics. Thus, it is meaningful to construct a more effective data space for classification. Wu *et al.* [Wu *et al.*, 2017] introduced the idea of MFL into imbalanced learning and provided a multiple sets construction strategy, incorporated the cost-sensitive factor into MFL. Mohamed *et al.* [Bader-El-Den *et al.*, 2016] took advantage of the local neighborhood of the minority instances to identify and treated difficult examples belonging to the same classes to solve the imbalance issue. Kwabena *et al.* [Bennin *et al.*, 2017] interpreted two distinct sub-classes as parents and generated a new instance that inherits different traits from each parent and contributed to the diversity within the data distribution.

Although there are many works concentrating on imbalance data classification, most of existing methods fail to take the limited of the training data into account. The limited training data may influence the development of the unbalance data classifier. Therefore, many research works focus on obtaining more information from limited training data for classification models. For instance, a semi-supervised learning method was introduced in [Zhang *et al.*, 2017], which utilized the relationships among training data, and then constructed a label propagation method based on these relationships. Yang et al. [Yang *et al.*, 2015] proposed a learning-to-rank method which directly optimized the ranking performance and constructed a software defect prediction models according to the ranking information.

3 Iterative Metric Learning for Imbalance Data Classification

3.1 Overview of the Method

This section, we introduce our iterative metric learning method for imbalance data classification. The main objective of our work is to explore a stable neighborhood space for a testing data under the scenario of imbalance training data. In

our method, we propose to find a more stable neighborhood space for the testing data using the iterative metric learning strategy, which consists of three components: iterative metric learning, iterative samples selection and training samples matching.

In the first stage, due to the absence of prior knowledge, most classifiers just like KNN, used the Euclidean distance to measure the similarities between samples and constructed their models. But actually, Euclidean distance metric is incapable of describing the statistical regularities of the training data [Weinberger and Saul, 2009a]. To solve this problem, we utilize the benefit of metric learning, which has the ability to reduce distances between same categories samples and increase distances between different categories samples. Then, in the new data space, the nearby samples are more likely to have the same labels.

In the second stage, considering that the training data is imbalanced and usually unrelated, we conduct sample selection process to select the most relevant data according to the testing data. After samples selection process, a subset of training data which locates nearest to the testing data are selected to construct the training data space. In this way, the neighborhood training samples are more relevant to the testing data, and the imbalanced issue can be solved in this step.

Considering that single metric learning process may not transform the data into the most effective data space, we optimize the neighborhood space for testing samples by iteratively executing metric learning process and sample selection process. We stop the iteration procedure by comparing the current selected training samples with the previous selected samples. If the selected samples are relatively stable, for instance, the samples selected in two adjacent iteration process are similar. We interrupt the iterative procedure, otherwise, we repeat the above steps.

3.2 Iterative Metric Learning

In order to improve the accuracy of imbalance data classification model, we employ metric learning methods to modify structure of data space. The objective of metric learning is to separate the samples from different classes by a large margin and minimize pairwise distances between samples from the same class, simultaneously. As a result, the newly transformed data space is more efficient for label classification. The process is demonstrated in Figure 2. It is noted that only conducting metric learning process once may not find the most effective and stable neighbor for the testing data. Thus, we propose an iterative metric learning method for data space modification. As we can see in Figure 2, by repeating metric learning process, the training data space becomes more related to the testing data.

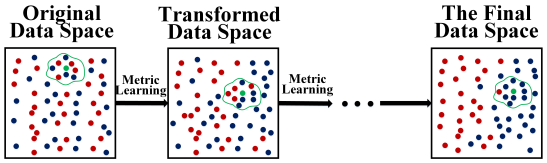


Figure 2: A example of finding a stable neighborhood space for testing sample .

As for the selection of metric learning algorithm, it is not vital for our method, and we select one of the most benchmark metric learning method, *i.e.*, Large Margin Nearest Neighbor (LMNN) method [Weinberger and Saul, 2009b] to transform the data space. In order to select the appropriate training data space, LMNN constructs a loss function to pull the samples with similar labels close to the testing sample and push training samples with different labels far away from the testing sample. The loss function of LMNN is defined as $\phi(L) = (1-\mu)\phi_{pull}(L)+\mu\phi_{push}(L)$. The first component of the loss function is defined as $\phi_{pull}(L) = \sum_{i,j} \|L(x_i - x_j)\|^2$, which penalizes the large distance between the testing samples and the similarly labeled training samples. In this function, L is the linear transformation of the input space, and x_i is the testing sample, and x_j is the training samples which have similar labels with the testing sample. The second component of the loss function penalizes the small distances between the testing samples and training samples with different labels, and the function is defined as $\phi_{push}(L) = \sum_{i,j}\sum_{m}(1-y_{im})[1+\|L(x_i-x_j)\|^2 - \|L(x_i-x_l)\|^2]_+$. The term $[q]_+ = \max(q, 0)$ denotes the standard hinge loss. In this formula, y_{il} =1 only if $y_i = y_l$, and in other cases, $y_i l = 0$. LMNN utilizes positive semidefinite matrices to optimize this formula. Then the formula can be rewritten as $\phi(M) = (1-\mu)\sum_{i,j}\mathcal{D}_M(\overrightarrow{x_i},\overrightarrow{x_j})+\mu\sum_{i,j}\sum_{l}(1-y_{il})$. In this formulation, $M = L^T L$ and $\mathcal{D}_M(\overrightarrow{x_i},\overrightarrow{x_j}) = (\overrightarrow{x_i}-\overrightarrow{x_j})^T M(\overrightarrow{x_i}-\overrightarrow{x_j})$. The parameter μ is used to balance the two terms. After getting the distance matrix M, The data space can be transformed into a new distinguishable data space. In the new data space, the neighborhood samples among the testing samples become more related and the nearby samples are more like to have the same labels.

3.3 Iterative Training Samples Selection

The second step of our method is iterative training samples selection. In this procedure, considering that the neighborhood samples are more likely to have the same labels, for each testing sample, we calculate the distance between the training samples and the testing sample, and choose the closest training samples. Giving a set of testing and training samples $\{\zeta_{train_1}, \zeta_{train_3}, \ldots, \zeta_{train_n}, \zeta_{test_1}, \zeta_{test_2}, \ldots, \zeta_{test_n}\}$, the feature of samples can be represented as feature vectors $\{\mathbf{x}_{train_1}, \mathbf{x}_{train_3}, \ldots, \mathbf{x}_{train_n}, \mathbf{x}_{test_1}, \mathbf{x}_{test_2}, \ldots, \mathbf{x}_{test_n}\}$. As shown in Figure 3, we uses ζ to donate one of the testing samples which is represented as green circle. Firstly, we calculate the distances between the testing sample ζ and all of the training samples. For the positive and negative training data classes, we both choose the top n_p and top n_f training samples, and the selected positive samples are represented as red circles and the negative samples are represented as blue circles in Figure 3. By this step, we can only focus on the samples which are more likely to influence the classification results of testing data, and the training samples selection process can significantly reduce the size of training samples, especially the defect-prone training samples.

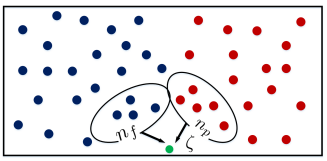


Figure 3: Training samples selection.

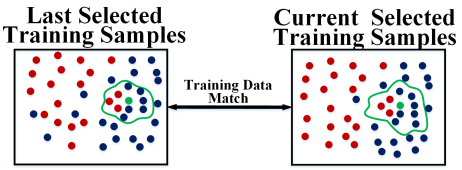


Figure 4: A example of training samples matching.

3.4 Selected Samples Matching

Considering that conducting metric learning process once may not discover the most effective training samples, we iteratively repeat the above processes and stop iteration when the selected samples are relatively stable. In order to find the appropriate termination point, in each iteration, we record the selected training samples and compare the selected training samples with the previously selected training samples. If the selected training samples are relatively stable, *e.g.*, as shown in Figure 4, the selected training data in two adjacent iterations are similar, then we stop the iteration and use the current selected samples for classification. Otherwise, we repeat the above steps until the most appropriate training samples are found.

After getting the optimal training samples, we can use classifier such as KNN and SVM to classify the testing sample. In our experiment, we choose KNN to classify testing sample. KNN is one of the most popular algorithms for classification and it classifies each testing data by the majority label among its k-nearest neighbors in the training dataset [Aci *et al.*, 2010].

4 Experiments

4.1 The Testing Dataset

In our experiments, we employ the widely used seven data from NASA Metrics Data Program (NASA) dataset [Menzies *et al.*, 2007], including CM1, KC3, MC2, MW1, PC1, PC3, PC4 and nine data from binary UCI Machine Learning Repository (UCI)[Lichman, 2013], including australian, haberman, heartstatlog, ionosphere, LiverDisorders, sonar, SPET, SPECTF, wdbc to evaluate the performance of our method.

	Predict as Positive	Predict as Negative
Positive Sample	True positive (TP)	False negative (FN)
Negative Sample	False positive (FP)	True negative (TN)

Table 1: Classification confusion matrix.

4.2 Evaluation Criteria

We use probability of detection (PD), the area under Receiver Operating Characteristic (AUC) and F_1-measure to evaluate our method, which have been commonly used in imbalance data classification field. The confusion matrix for the model is demonstrated in Table 1.

1. Probability of Detection (PD): PD is the ratios between the positive modules correctly classified as positive and the number of positive modules, which is represented as $PD = \frac{TP}{TP+FN}$.
2. F_1-measure: In order to achieve high value of PD and precision, we utilize F_1-measure to take Precision and PD into account simultaneously, which is defined as F_1-measure$= \frac{2\times PD\times Precision}{PD+Precision}$. Precision is the percentage of positive modules correctly classified as positive compared with the number of modules which are classified as positive, which is represented as $Precision = \frac{TN}{TN+FN}$.
3. AUC: AUC measures the area under the ROC curve. The ROC curve plots PF on the x-axis and the PD on the y-axis. ROC curve pass through points (0,0) and (1,1) by definition.

4.3 Compared Methods

To evaluate the effectiveness of the proposed method, we compare our algorithm with other two state-of-the-art methods.

1. Non-negative sparse graph based label propagation (NSGLP) [Zhang *et al.*, 2017]. In NGSLP, the authors employed Laplacian score and negative sparse algorithm to label the testing data.
2. Cost-sensitive discriminative dictionary learning (CDDL) [Jing *et al.*, 2014]. CDDL learned multiple dictionaries and sparse representation coefficients to classify the testing samples and took the misclassification cost into consideration.

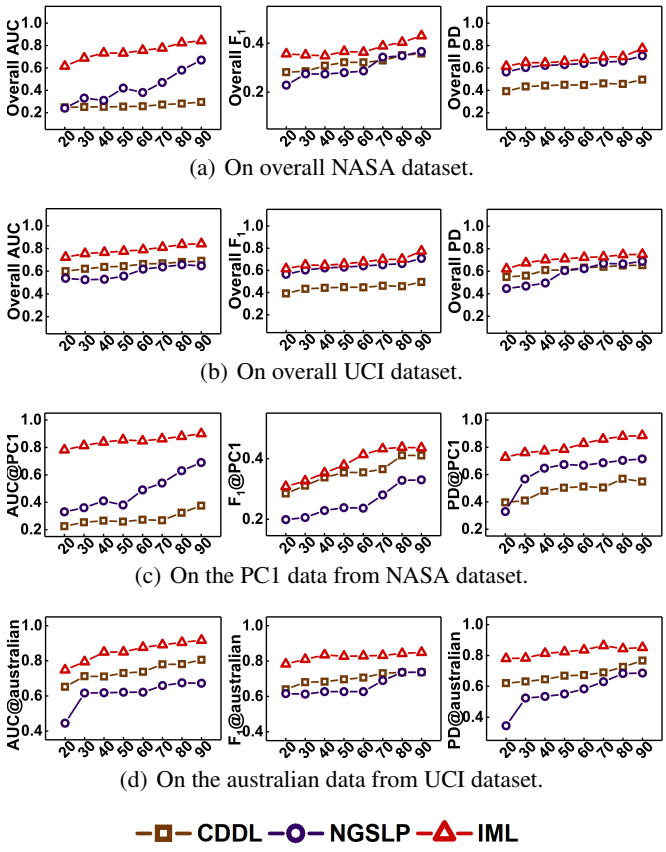
4.4 Experimental Settings and Results

We randomly divide the datasets into training datasets and testing datasets and use eight differently labeled rates from 20% to 90%. On account of the influence of the random division, we repeat the experiment 20 times and report the average classification results. Moreover, we choose the k-nearest neighbors (KNN) algorithm as classifier, and compare the performance of our method with KNN using data space which is handled by different times of iterations.

Considering the sparsity of data space, it is difficult to determine the matching rate for training samples matching process. In order to select the most effective matching ratio, we vary K from 0.5 to 0.9, and set matching ratio as 0.8 by experimental results. The comparison results are presented below.

Comparison with Other State-of-the-art Methods

We compare our proposed method with several representative methods. The experimental results are shown in Figure 5 and our method achieves the best performance both in NASA dataset and UCI dataset.



(a) On overall NASA dataset.

(b) On overall UCI dataset.

(c) On the PC1 data from NASA dataset.

(d) On the australian data from UCI dataset.

Figure 5: Experimental results compared with other methods. X-axis shows different labeled rates (%)

(i) Comparison with NGSLP. In terms of F_1-measure, IML achieves gains of 20.4%, 29.8%, 31.5%, 31%, 23.3%, 22.3%, 18.3%, 19.2% when compared with NGSLP using eight differently labeled rates as $20, 30, 40, 50, 60, 70, 80$ and 90% respectively on the overall UCI dataset. On separate data, our method achieves average gains of 25.4% and 45.5% on PC1 dataset from NASA dataset and australian data from UCI dataset in terms of PD. Experiment results observed from other datasets and evaluation criteria also show our method achieves better performance when compared with NGSLP.

(ii) Comparison with CDDL. When compared with CDDL, IML also achieves better performance. In terms of AUC, our method achieves gains of 140.2%, 143.2%, 146.5%, 157.3%, 162.4%, 180.4%, 184.5%, 210.3% compared with CDDL from 20% to 90% respectively on the overall NASA dataset. On separate data, IML achieves average gains of 5.7% and 27.4% on PC1 data from NASA dataset and australian data from UCI dataset in terms of F_1. These results demonstrate that the proposed method performs better both in software defect prediction field and other imbalance data classification fields like breast cancer identification and so on.

Comparison with KNN with Different Metric Learning Iterations

In order to further evaluate the effectiveness of IML, we compare the proposed method with KNN using data space handled by different times of iterations, *i.e.*, the original data space, the data space conducted iteration once, the data space conducted iteration twice. Experimental results are shown in Figure 6.

1. Comparison with KNN using original data space. As shown in Figure 6, the experimental results of IML significantly perform better than KNN using original data space. When the labeling rate is set as 20, 40, 60 and 80%, for instance, the proposed method achieves gains of 6.8%, 12.7%, 9.3%, 10.7% in terms of AUC and 38.1%, 32.7%, 22.4%, 22.5% in terms of F_1-measure on the overall NASA dataset. More specifically, on SPECTF data from UCI dataset, IML achieves 44%, 51.3%, 35% and 32.7% in terms of F_1-measure and 62.8%, 81.3%, 61.3% and 25.8% in terms of PD when the labeling rate is set as 20, 40, 60 and 80%. Similar results are observed from other datasets and evaluation criteria.

2. Comparison with KNN using data space conducted iteration once. When the labeling rate is set as 20, 40, 60 and 80%, IML achieves gains of 4.7%, 2.6%, 3.2%, 10.01% in terms of AUC on the CM1 data from NASA dataset. Moreover, on the overall UCI dataset, IML achieves gains of 2.1%, 1.2%, 1.3% and 2.2% in terms of PD and 0.6%, 1.7%, 2.7% and 2.6% in terms of F_1-measure. The experimental results of IML are better than KNN using data space conducted iteration once.

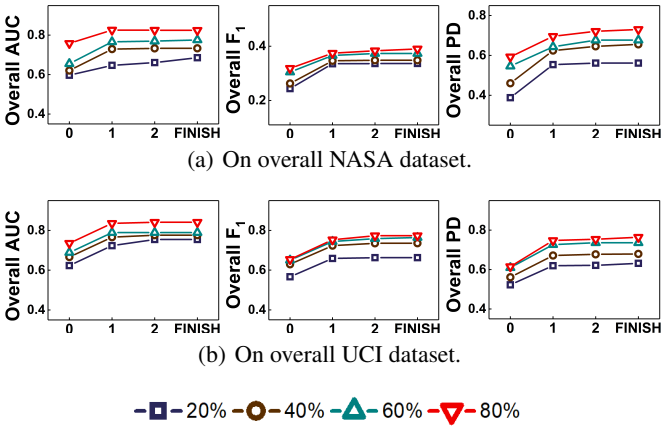
3. Comparison with KNN using data space conducted iteration twice. The experimental results of IML are slightly better than KNN using data space conducted iterations twice. When the labeling rate is set as 20, 40, 60 and 80%, in terms of AUC, our method achieves gains of 1.2%, 0.2%, 0.8% and 0.1% on the overall NASA dataset. The reason for these experimental results is that most of the learning processes stop after twice iterations and the selected training samples after twice iterations are almost the most effective training samples for classification. The statistical results of iteration times are shown in Figure 7

Obviously, the results on each evaluation criterion get better and better over the times of iteration, and most of the experimental results become stable after conducting metric learning twice. The results show the superiority of IML in imbalance data classification field.

4.5 Discussions

Comparison with Other Methods

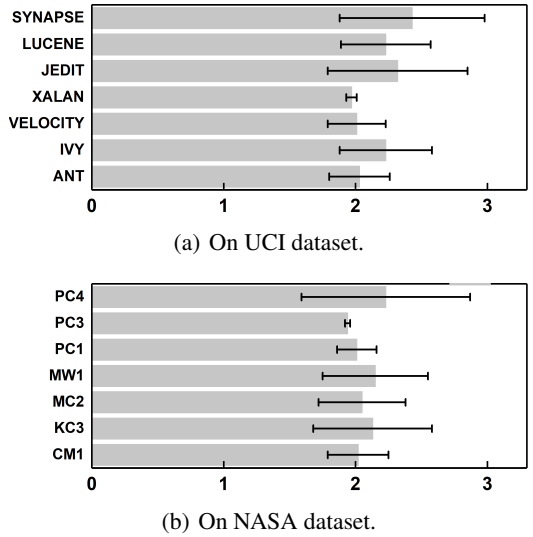
Compared with state-of-the-art methods, *i.e.*NGSLP, CDDL, our method achieves the best performance. The results can be explained by two advantages of IML. First, we employ metric learning method to transform the original data space structure into a more effective data space. By metric learning, the boundaries between different classes become clearer.



(a) On overall NASA dataset.

(b) On overall UCI dataset.

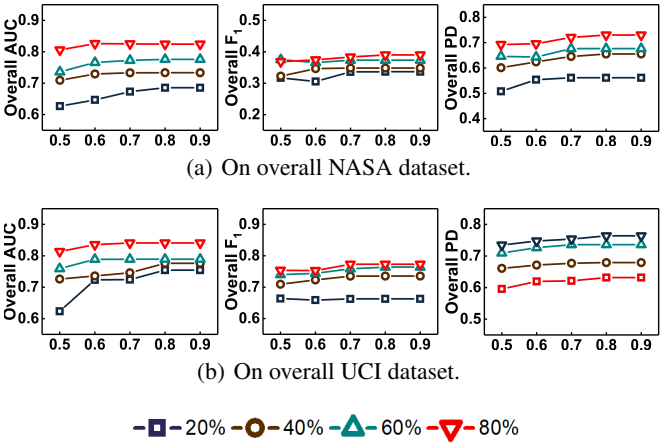
Figure 6: The experimental results with respect to different iteration times. X-axis is the number of iterations.



(a) On UCI dataset.

(b) On NASA dataset.

Figure 7: The number of iterations on UCI dataset and NASA dataset



(a) On overall NASA dataset.

(b) On overall UCI dataset.

Figure 8: The experimental results with respect to different selections of selected samples matching ratios. X-axis is the ratio for training data matching.

Second, in order to explore a stable neighborhood space for a testing data under the scenario of imbalance training data. Given the imbalance training data, it is important to select a subset of training samples for each testing data. Thus, we propose to find a more stable neighborhood for the testing data using the iterative metric learning strategy. After iteratively handling the original data space, the correlations among the samples are clearer and the training samples around the testing sample are more likely to have the same label with the testing sample. Moreover, due to the imbalanced and the sparsity of the training samples, in the learning procedure, we utilize training samples selection to select the most relevant data. As we can see in Figure 7, most of the iteration process stops after twice handling. This phenomenon shows that our method can quickly converge and find the effective training samples.

Moreover, as shown in Figures 5, our method performs better than NGLSP, CDDL. Considering that the datasets used in our experiment are from multi-domains and the relationships among the data are difficult to explore, NGSLP which uses graph structure to describe the correlation among the data and CDDL which combines cost information into the learning process are both superior to KNN classifier in imbalance data classification. As shown by the experimental results, our method which employs traditional KNN as the classifier achieves better performance than these two methods. The results show that our method is effective to handle the imbalance data, and it can be used as data space reconstruction method for other classification methods which use original metric distance for classification.

On Training Samples Matching Ratio

In our method, we incorporate the influence of different training samples matching ratios, which is defined as the ratios between the number of matching selected training data in adjacent two iterations. Because it is difficult to determine the ratios for training samples matching process, we vary the ratios in $[0.5\ 0.6\ 0.7\ 0.8\ 0.9]$, and show the comparison results in Figure 6. Based on the experimental results, it is obvious that the proposed method achieves stable and better performance when the matching ratio is set around 0.8. So in our experiments, we choose 0.8 as the matching ratio.

5 Conclusion

Imbalance data classification attracts more and more attention from industry and academia. This paper proposes an iterative metric learning method for imbalance data classification. During the learning process, the method mainly contains three phases: iterative metric learning, iterative training samples selection, and selected samples matching. By iteratively executing the above steps, we find the most effective training samples for imbalance data classification.

Experimental results on the NASA and UCI datasets show the superiority of our method when compared with state-of-the-art methods on all evaluation criteria. Additionally, in order to evaluate the iterative handling process, we compare our

method with KNN classifier using data space handled by different times of iterations. The experimental results show that iterative metric learning has the superiority in finding a more stable neighborhood space for the testing data, and almost all the selected samples become stable after twice iterations.

Although the performance of the IML shows its advantage in classification, there are still several limitations. On the one hand, considering that misclassification of positive and negative modules is associated with different costs in real application, the proposed method should add the cost information in the learning process. On the other hand, it is better to modify the weight of training data according to the structure of the data space.

Acknowledgments

This work was supported by National Key R and D Program of China (Grant No. 2017YFC0113000), National Natural Science Funds of China (U1701262, 61671267), National Science and Technology Major Project (No. 2016ZX01038101), MIIT IT funds (Research and application of TCN key technologies) of China, and The National Key Technology R and D Program (No. 2015BAG14B01-02).

References

[Aci *et al.*, 2010] Mehmet Aci, Cigdem Inan Aci, and Mutlu Avci. A hybrid classification method of k nearest neighbor, bayesian methods and genetic algorithm. *Expert Systems with Applications*, 37(7):5061–5067, 2010.

[Bader-El-Den *et al.*, 2016] Mohamed Bader-El-Den, Eleman Teitei, and Adda Mo. Hierarchical classification for dealing with the class imbalance problem. In *International Joint Conference on Neural Networks*, pages 3584–3591, 2016.

[Barua *et al.*, 2014] Sukarna Barua, Md. Monirul Islam, Xin Yao, and Kazuyuki Murase. Mwmote–majority weighted minority oversampling technique for imbalanced data set learning. *IEEE Transactions on Knowledge and Data Engineering*, 26(2):405–425, 2014.

[Bennin *et al.*, 2017] K. Ebo Bennin, J. Keung, P. Phannachitta, A. Monden, and S. Mensah. Mahakil:diversity based oversampling approach to alleviate the class imbalance issue in software defect prediction. *IEEE Transactions on Software Engineering*, pages 1–1, 2017.

[He and Garcia, 2009] Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284, 2009.

[Jeet *et al.*, 2011] Kawal Jeet, Nitin Bhatia, and Rajinder Singh Minhas. A bayesian network based approach for software defects prediction. *ACM SIGSOFT Software Engineering Notes*, 36(4):1–5, 2011.

[Jing *et al.*, 2014] Xiao-Yuan Jing, Shi Ying, Zhi-Wu Zhang, Shanshan Wu, and Jin Liu. Dictionary learning based software defect prediction. In *Proceedings of International Conference on Software Engineering*, pages 414–423, 2014.

[Lichman, 2013] M. Lichman. UCI machine learning repository, 2013.

[Liu *et al.*, 2015] Wei Liu, Cun Mu, Rongrong Ji, Shiqian Ma, John R. Smith, and Shih-Fu Chang. Low-rank similarity metric learning in high dimensions. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, pages 2792–2799, 2015.

[Menzies *et al.*, 2007] Tim Menzies, Jeremy Greenwald, and Art Frank. Data mining static code attributes to learn defect predictors. *IEEE Transactions on Software Engineering*, 33(1):2–13, 2007.

[Mesquita *et al.*, 2016] Diego Parente Paiva Mesquita, Lincoln S. Rocha, João P. P. Gomes, and Ajalmar R. da Rocha Neto. Classification with reject option for software defect prediction. *Applied Soft Computing*, 49:1085–1093, 2016.

[Sobhani *et al.*, 2014] Parinaz Sobhani, Herna Viktor, and Stan Matwin. *Learning from Imbalanced Data Using Ensemble Methods and Cluster-Based Undersampling*. Springer International Publishing, 2014.

[Sun *et al.*, 2012] Zhongbin Sun, Qinbao Song, and Xiaoyan Zhu. Using coding-based ensemble learning to improve software defect prediction. *IEEE Transactions on Systems, Man, and Cybernetics*, 42(6):1806–1817, 2012.

[Tomar and Agarwal, 2016] Divya Tomar and Sonali Agarwal. Prediction of defective software modules using class imbalance learning. *Proceedings of Annual Computer Software and Applications Conference*, 2016:7658207:1–7658207:12, 2016.

[Weinberger and Saul, 2009a] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[Weinberger and Saul, 2009b] Kilian Q. Weinberger and Lawrence K. Saul. Distance metric learning for large margin nearest neighbor classification. *Journal of Machine Learning Research*, 10:207–244, 2009.

[Wu *et al.*, 2017] Fei Wu, Xiao-Yuan Jing, Shiguang Shan, Wangmeng Zuo, and Jing-Yu Yang. Multiset feature learning for highly imbalanced data classification. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence,*, pages 1583–1589, 2017.

[Yan *et al.*, 2010] Zhen Yan, Xinyu Chen, and Ping Guo. Software defect prediction using fuzzy support vector regression. In *Proceedings of the 7th International Conference on Advances in Neural Networks*, pages 17–24, 2010.

[Yang *et al.*, 2015] Xiaoxing Yang, Ke Tang, and Xin Yao. A learning-to-rank approach to software defect prediction. *IEEE Transactions on Reliability*, 64(1):234–246, 2015.

[Zhang *et al.*, 2017] Zhi-Wu Zhang, Xiao-Yuan Jing, and Tiejian Wang. Label propagation based semi-supervised learning for software defect prediction. *Automated Software Engineering*, 24(1):47–69, 2017.