Journal of Computers Vol. 32 No. 2, 2021, pp. 23-32
doi:10.3966/199115992021043202003

# Feature Selection Active Learning Algorithm Based on Sparse Subspace

Liang Gong*, Xu-Hong Wang, Yu-Pu Yang

Key Laboratory of Department of Automation, Shanghai Jiao Tong University, Shanghai, China
lianggong@sjtu.edu.cn

Received 4 December 2019; Revised 2 August 2020; Accepted 17 August 2020

Abstract. Active learning has been widely used in many fields, and has proved to be very effective to solve a vast array of machine learning tasks, but it still has some shortcomings. The problem is that it only pays attention to label information of samples and ignores the structural information, so it cannot reach state-of-art performance in some tasks and is sensitive to initial data. In this paper, a new algorithm is proposed by combining Sparse Subspace algorithm with active learning together. Firstly, Sparse Subspace algorithm is used to find the dataset's structure. Then, an active learning algorithm based on feature selection is adopted. In this way, not only the label information but also the distribution information of samples is taken into account. As a result, the efficiency and robustness of active learning are improved, and the labeling cost of samples and the probability of falling into local optimum are reduced.

Keywords: Active learning, feature selection, sparse subspace

## 1 Introduction

In the era of big data, the cost of data acquisition has been significantly reduced, but the cost of labeling these data has not undergone dramatic change. Active learning [1] can reduce the requirement of labeled samples. However, in many fields, the acquisition of labeled samples is very difficult or expensive, such as electroencephalogram (EEG) analysis of schizophrenic patients. So we need to further improve the efficiency of active learning and reduce the cost of labeling samples.

Active learning has been proved to be very effective, but it still has the following problems:

Firstly, the main problem of active learning is that it only focuses on the label information of samples while ignoring the structure information of the whole dataset. This results in the shortcomings, such as sensitivity to initial values, local optimum solution, and lack of robustness.

Secondly, the majority of real-life data, such as text, sound, image, and so on, are high-dimensional and nonlinear. When dealing with these data, active learning is still inefficient, that is to say, it needs more labeled samples to achieve high accuracy. So, how to further reduce the cost of sample labeling is still a very critical issue.

To overcome the abovementioned problems, we propose a new algorithm based on sparse subspace algorithm (SS) [2] and feature selection [3]. SS can effectively exploit low-dimensional structures embedded in high-dimensional data space. Feature selection is also an effective method to project high-dimensional data to low-dimensional space. By using the processed data, active learning becomes more effective and requires fewer labeled samples.

In this paper, we propose a new active learning algorithm to improve the efficiency, so as to reduce the requirement of labeled samples. Our proposed algorithm not only focuses on the label information of samples, but also focuses on the structure information of samples. It uses SS to explore the structure of data set and guide active learning, and also uses feature selection to reduce data dimension and improve classification efficiency. Finally, the algorithm can effectively improve the classification efficiency of active learning, lower the probability of falling into local optimum, and reduce the demand of labeled

* Corresponding Author

samples.

## 2 Related Works

In recent years, the study of active learning has attracted many scholars. To reduce the labeling cost, Goudjil et al. [4] proposed a new support vector machine (SVM) algorithm by intelligently selecting which samples should be labeled. This method selected samples using the posterior probabilities provided by a set of multi-class SVM classifiers. Pérez-Ortiz et al. [5] proposed an ordinal classification algorithm and considered the neighborhood information of unlabelled samples. Sun et al. [6] proposed a SVM model in which parameters could be adjusted automatically. Zhang et al. [7] proposed a semi-supervised learning algorithm that used Gaussian filed and harmonic functions. It used clustering coefficient metric to identify the best instance next to label. Liu et al. [8] proposed an active learning algorithm that used multi-class classification model based on SVM. To solve highly nonlinear problems, Lin-Xiong et al. [9] proposed a novel Kriging algorithm based reliability analysis method. Gupta et al. [10] developed a new method that used sparse recovery and active learning techniques. It boosted accuracy with fewer samples. Hijazi et al. [11] suggested a framework for actively selecting and then propagating constraints for feature selection. It decreased the cost of human-labor. Lei et al. [12] presented an active learning algorithm combining with data balancing to reduce the effort of labeling.

Because many scholars have carried out in-depth research on active learning algorithm, it is very difficult to further improve the efficiency, so in this paper we focus on the data structure of samples and make full use of the structure information to improve the efficiency of active learning. SS is an algorithm based on spectral graph theory and it can exploit a smooth low-dimensional manifold embedded in the high-dimensional space. We also propose an improved feature selection algorithm. At the beginning of iteration, it selects several key features to build the model, so that it can grasp the overall structure of the whole dataset and reduce the probability of falling into local optimum. As iteration goes on, it gradually increases the detail features to further improve the accuracy of active learning.

The experimental results show that the algorithm we proposed can improve the efficiency and robustness of active learning.

## 3 Sparse Subspace Algorithm

SS algorithm is a research hotspot in recent years. Its main idea is that a high-dimensional space is composed of multiple low-dimensional subspaces, which can be expressed by linear combination of low-dimensional space, and this linear expression can also be used to describe the similarity of different low-dimensional subspaces. As shown in Fig. 1, given a three-dimensional data space, the 3D dataset shown in the figure can be regarded as a combination of two one-dimensional subspaces and one two-dimensional subspace. So in essence, the 3D dataset is composed of several low dimensional subspaces.

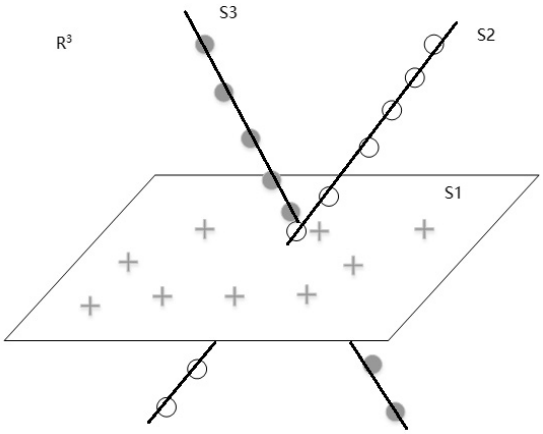


**Fig. 1.** 3D dataset is composed of three low-dimensional subspaces S1-S3

Because low-dimensional subspace can better reflect the internal relationship of dataset, it is very important for data processing and mining, pattern recognition and so on. It is widely used in image processing, computer vision, system identification and other fields. Former researcher [13-14] try to obtain the coefficient matrix by sparse representation, then the segmentation of subspace can be obtained by spectral clustering methods such as NCut (Normalized Cuts) [15].

Let a set of sample data set $X=\left[x_1, x_2, \ldots, x_N\right] \in R^{D\times N}$ belong to the union of K linear subspaces $\{S_i\}_{i=1}^{k}$, and express the sample $x_i \in S_a$ with the linear combination of other sample data:

$$\mathrm{x}_i = \sum_{j\neq i} Z_{ij} x_j . \tag{2.1}$$

In order to find the K subspaces, a certain constraint condition is added to the expression coefficient $Z_{ij}$, so that: for all $x_i \notin S_a$, $Z_{ij}=0$. That is, $x_i$ can only be expressed by the samples of the same subspace. Then rearrange $Z_{ij}$ in a certain way, so the formula (2.1) is equivalent to:

$$X = XZ, \tag{2.2}$$

Where $Z \in R^{N\times N}$ and $Z_{ij}=0$ when $x_i$ and $x_j$ belong to different subspaces. If we get the coefficient matrix Z, it can be arranged into the following structure:

$$Z=\begin{bmatrix} \mathrm{Z}_1 & \cdots & 0 \\ 0 & \mathrm{Z}_2 & 0 \\ \vdots & \ddots & \vdots \\ 0 & \cdots & \mathrm{Z}_\mathrm{k} \end{bmatrix}. \tag{2.3}$$

In order to obtain the most sparse representation of samples, we need minimize the $l_0 - norm$ of Z. Because the minimization problem is NP-hard, it is usually replaced by the norm of $l_1$:

$$\begin{aligned} &\min_{z} \| Z \|_1 \\ &s.t.\ \ X = XZ, diag\left(Z\right)=0, \end{aligned} \tag{2.4}$$

Where $diag\left(Z\right)=0$ is to prevent each sample from being expressed by itself, which will make the coefficient matrix Z become a unit matrix. This optimization problem can be solved by alternating direction method of multipliers (ADMM) [16].

ADMM algorithm is as follows:

Suppose the optimization problem is

$$\begin{aligned} &\min_{x} f\left(x\right)+g\left(z\right) \\ &s.t.\ Ax+Bz=c, \end{aligned} \tag{2.5}$$

Where $x \in \mathrm{R}^{\mathrm{n}}, z \in \mathrm{R}^{\mathrm{m}}, A \in \mathrm{R}^{\mathrm{p}\times\mathrm{n}}, B \in \mathrm{R}^{\mathrm{P}\times\mathrm{m}}, c \in \mathrm{R}^{\mathrm{p}}$. $x$ and $z$ are variables to be optimized, $f\left(x\right)+g\left(z\right)$ is the objective function to be minimized, $Ax+Bz=c$ is the combination of $p$ equality constraints. Using augmented Lagrangian function to decompose formula (2-5):

$$L_p\left(x,z,y\right)=f\left(x\right)+g\left(z\right)+y^T\left(Ax+Bz-c\right)+\frac{\rho}{2}\| Ax+Bz-C \|_2^2, \tag{2.6}$$

Where $y$ A is the Lagrange multiplier, $\rho>0$ is the penalty parameter. Then the formula (2.6) is scaled. By scaling the $y$ in the augmented Lagrangian function, the quadratic term and the linear term of the equation constraint in the function are combined. The scaled function is as follows:

$$L_p\left(x,z,y\right)=f\left(x\right)+g\left(z\right)+\frac{\rho}{2}\| Ax+Bz-C+\mu \|_2^2, \tag{2.7}$$

Where

$$\mu = \frac{1}{\rho} y .$$

ADMM algorithm optimizes $x$ and $z$ alternately. First fix $x$, optimize $z$; then fix $z$, optimize $x$; finally update the scaling dual variable. This is done alternately. By solving, the coefficient matrix Z is obtained, from which the adjacency matrix $W$ of graph $G$ can be constructed:

$$W = |Z| + |Z^T| . \tag{2.8}$$

At last, using adjacency matrix W, we can get Laplace matrix L of graph G and calculate its eigenvalues.

The specific algorithm is as follows:

**Table 1.** Sparse subspace algorithm

| |
|---|
| **Input:** dataset X<br>**Output:** K eigenvalues |
| **Step 1:** Solving coefficient matrix Z with ADMM algorithm;<br>**Step 2:** $W = \|Z\| + \|Z^T\|$;<br>**Step 3:** Construct Laplacian matrix $L$ from $W$;<br>**Step 4:** Calculating K eigenvalues of Laplace matrix $L$. |

## 4 Feature Selection Active Learning Based on Sparse Subspace

An important reason for the poor robustness of the traditional active learning algorithms is that they consider only the label information of samples, and ignore the structure information of the dataset when selecting new samples.

In order to improve the efficiency of active learning and reduce the labeling cost, a feature selection active learning algorithm based on sparse subspace is proposed. SS algorithm can obtain the subspace structures of dataset, and then guide active learning algorithm to classify. In this paper, we choose Support Vector Machines (SVM) [17] as the active learning algorithm.

First of all, the eigenvalues of subspaces are obtained by SS algorithm. Then, the eigenvalues are sorted from large to small. At the beginning of active learning, several key features of the most importance are selected to build the model, which can promote the active learning algorithm to grasp the overall structure of the whole dataset, and reduce the probability of falling into local optimum. As iteration goes on, the detail features are added gradually to further improve the accuracy.

The specific process is as follows:

(1) Find the eigenvalues of subspaces by SS algorithm, and sort them according to the importance of features;

(2) Use the main part of "information" to train iteratively with the active learning algorithm;

(3) After reaching the designated stop conditions, increase the percentage of the "information" and carry out the next stage of training.

(4) Repeat Step 3, gradually increase the percentage of the "information" until it reaches 100%.

To measure the amount of "information" in the above process, "information variance" is defined as follows in the paper:

***Definition 3.1 Information variance***

Eigenvalue decomposition is carried out on Matrix A to obtain the diagonal matrix:

$$\mathbf{\Sigma} = diag(\sigma_1, \sigma_2, \cdots, \sigma_n), \text{ where } \sigma_1 \geq \sigma_2 \geq \cdots \geq \sigma_n .$$

S is written as the information variance:

$$S = \sigma_1^2 + \sigma_1^2 + \ldots + \sigma_n^2 .$$

Selecting Dimension d to retain 60% of the "information" means the value of d must satisfy:

$$\sigma_1^2 + \sigma_1^2 + \ldots + \sigma_{d-1}^2 < 60\% S \text{ and } \sigma_1^2 + \sigma_1^2 + \ldots + \sigma_d^2 \geq 60\% S .$$

In order to determine the stop condition, the definition of vector difference degree is proposed to compare the difference between two vectors.

***Definition 3.2 Vector difference degree***

Two n-dimensional vectors $\hat{x}$ and $\hat{y}$ are set. Their vector difference degree function $diff(\hat{x}, \hat{y})$ is defined as the percentage between the number of elements in $\hat{x}$ and $\hat{y}$ with different values and same subscript and the vector dimension n, namely:

$$diff(\hat{x}, \hat{y}) = \frac{\| \hat{x} - \hat{y} \|_0}{n} \times 100\% .$$

The stop condition of the above algorithm can be measured by the vector difference degree. When the vector difference degree is less than 0.5%, the change of the vector is considered to be very little, so it can go to the next iteration stage.

In this paper, the amount of information is selected with reference to Pareto's law, or twenty-eight law. Finally, four stages of 30%, 60%, 80% and 100% are determined according to the law as well as several experiments and simulations.

Thus, the specific steps of the active learning algorithm based on feature selection of sparse subspace are as follows:

**Table 2.** Feature selection active learning based on sparse subspace

| |
|---|
| **Input:** $m \times n$ dimensional data matrix X (samples were arranged in columns, a total of $n$ samples), Set $I_L$ composed of subscripts with labeled samples, Number of samples $k_1$ selected at each iteration. |
| **Output:** classification model $f : X \mapsto Y$ |
| **Step 1:** Find the eigenvalues of subspaces by SS algorithm, and sort them from large to small; according to retain the information variance of 30%, 60%, 80% and 100%, calculate the dimensions to obtain the transformed data matrixes $Z_1$ 、 $Z_2$ 、 $Z_3$ and $Z_4$ ; |
| **Step 2:** Create the set $I_U = \{0, 1, 2, \cdots, n-1\} - I_L$ to contain the subscripts of all unlabeled samples; make the number of iterations $t = 0$ ;<br>For i = 1 to 4: |
| **Step 3:** Make $L = \{\mathbf{z}_j \in \mathrm{Z}_i \mid j \in I_L\}$ ; learn from the labeled sample set $L$ to get the classification hyperplane $f^{(t)}$ , and predict the classification of all samples with $f^{(t)}$ to get $\hat{y}^{(t)}$ ; |
| **Step 4:** Select $k_1$ values from $I_U$ by the selection engine according to $f^{(t-1)}$ , written as $I_Q$ ; Label the samples in the set $Q = \{\mathbf{z}_j \in \mathrm{Z}_i \mid j \in I_Q\}$ by experts; $I_L = I_L \cup I_Q$ , $I_U = I_U - I_Q$ ; |
| **Step 5:** $t = t + 1$ ; |
| **Step 6:** Repeat Step 3 to 5 until the vector difference $diff(\hat{\mathbf{y}}^{(t-1)}, \hat{\mathbf{y}}^{(t)})$ <0.5%;<br>End For |
| **Step 7:** Repeat Step 3, 4 and 5 until it meets the stop condition; |
| **Step 8:** Return to $f^{(t)}$ . |

The algorithm is implemented in four stages with 30%, 60%, 80% and 100% eigenvalues for active learning. Since the eigenvalues are sorted from large to small, in the first stage with 30% of the eigenvalues, the most important eigenvalues are selected to learn. After a certain number of iterations, if the change of the classification model is very small, which is judged by the vector difference degree above, it can enter the next stage.

Assuming that the model $f^{(t)}$ is used to predict the class of all samples, the label vector $\hat{y}^{(t)}$ is obtained. If $\hat{y}^{(t)}$ is very close to that of the last iteration, that is,

$$diff(\hat{y}^{(t-1)}, \hat{y}^{t}) \leq 0.5\% .$$

It means that the updated model $f^{(t)}$ has very little difference from the previous model $f^{(t-1)}$, which indicates that it tends to be stable after several iterations. At this time, the model obtained by the algorithm may not have very high accuracy, but it has mastered the overall structure of the dataset. Continuous iteration will cause low efficiency, so it is necessary to enter the next stage, namely, adding more detailed features to further improve the classification accuracy. Of course, the process of active learning can be divided into more stages according to the actual problem.

## 5 Evaluations

In order to evaluate the performance of the proposed algorithm, we conduct experiments on several widely used datasets in the paper. All of these datasets come from the open dataset LIBSVM [18]. A popular algorithm is used for comparison, meanwhile, in order to find the influence of different information variances on the proposed algorithm, the comparison between 80% and 90% information variances are added. The four algorithms are introduced as follows:

(1) Active Learning based on Principal Component Analysis (ALPCA) [19]: a similar active learning method is selected for comparison. It extracts features by the principal component method and then carries on the active learning process.

(2) 80%: retain the fixed 80% information variance in the proposed algorithm.

(3) 90%: retain the fixed 90% information variance in the proposed algorithm.

(4) Proposed: the algorithm in this paper.

### 5.1 USPS Dataset Validation

With the USPS dataset as the test target, a group of handwritten characters “5” are randomly selected. The samples of the dataset have 256 dimensions, and 5 samples are randomly selected as the initial labeled samples in the initial stage. In order to convert the problem to binary classification which can be processed by the support vector machine, Character 5 is labeled as +1, and other characters as -1.

The hyperparameters of the support vector machines are set as $C = 10$, $\gamma = 0.01$.

To objectively evaluate the advantages and disadvantages of the algorithm, a total of 100 runs are carried out, and the statistical results are analyzed. The average results of 100 runs are shown in Fig. 2 below.

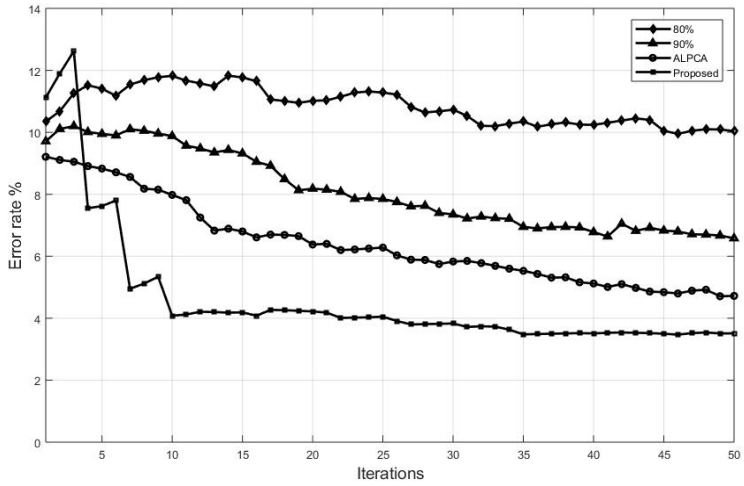


**Fig. 2.** Average Error Rates on USPS 5 VS Others

According to Fig. 2, the effect of the proposed algorithm is outstanding. Although the error rate is high in the first stage, after several iterations, it decreases rapidly. Especially when entering a new stage, the error score rate decreases significantly. This is consistent with our design conception. In the first stage, we use the main features to classify. At this time, the accuracy is not very high, but we have mastered the overall structure of the dataset. As iteration goes on, more features are added gradually, the accuracy is

improved rapidly. Finally, it has better classification accuracy and stability than the other algorithms.

Four statistics indexes of error rates after fiftieth iteration are shown in Table 3, these results are calculated from the final error rates after 100 runs. Bold face means the best performance. MAX means the maximum of the final error rates, MIN means the minimum of the final error rates, MEAN means the mean of the final error rates, STDEV means the standard deviation of the final error rates.

**Table 3.** Error Rates Statistic of Four Algorithms on USPS 5 over 100 Runs

| | ALPCA | Proposed | 80% | 90% |
|---|---|---|---|---|
| MAX | 0.3541 | 0.2245 | 0.3253 | 0.3466 |
| MIN | 0.0158 | 0.0146 | 0.0193 | 0.0130 |
| MEAN | 0.0462 | 0.0347 | 0.0984 | 0.0625 |
| STDEV | 0.0418 | 0.0266 | 0.0552 | 0.0465 |

Table 3 shows that the proposed algorithm has the best performance in average error and standard deviation, except for the minimum error rate. The smallest average error and standard deviation indicate the high classification accuracy and robustness.

### 5.2 Letter Dataset Validation

In order to further validate the effectiveness of the proposed algorithm, another dataset is used for validation. As a widely used character recognition dataset, Letter dataset is a handwritten characters recognition dataset with 20000 samples of 26 characters.

The tests are carried out on Letter dataset, with selected characters ‘A’, ‘B’ and ‘C’. The selected characters are labeled as +1, and the rest characters as -1. We run the proposed algorithm 100 times. In each run, we select 10 samples as initial labeled set. In each iteration, we select 5 samples from unlabeled data pool to label.

The hyperparameters of the support vector machines are set as $C = 10$, $\gamma = 1.0$.

#### 5.2.1 Letter A VS Others

According to Fig. 3, the proposed algorithm has good performance. The figure also shows that the proposed algorithm has the fastest convergence speed and lowest error rate.

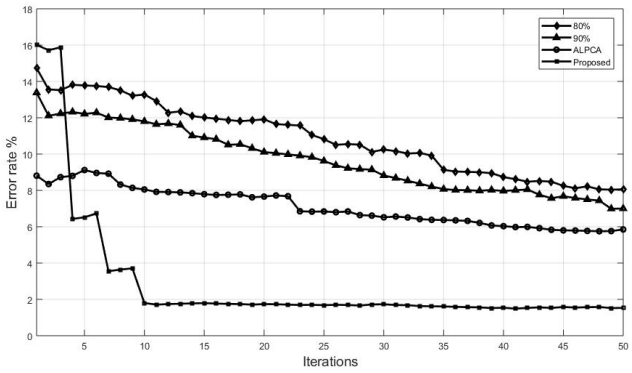


**Fig. 3.** Average Error Rates on Letter A VS Others

The statistic indexes of error rates on Letter B VS others over 100 runs are shown as Table 4:

**Table 4.** Error Rates Statistic of Four Algorithms on Letter A over 100 Runs

| | ALPCA | Proposed | 80% | 90% |
|---|---|---|---|---|
| MAX | 0.4862 | **0.0478** | 0.5339 | 0.4578 |
| MIN | 0.0058 | **0.0049** | 0.0105 | 0.0079 |
| MEAN | 0.0535 | **0.0139** | 0.0739 | 0.0683 |
| STDEV | 0.0809 | **0.0106** | 0.0988 | 0.0935 |

Table 4 shows that the proposed algorithm has the best performance in all indexes. The smallest average error and standard deviation means the proposed algorithm has higher classification accuracy and robustness.

### 5.2.2 Letter B VS Others

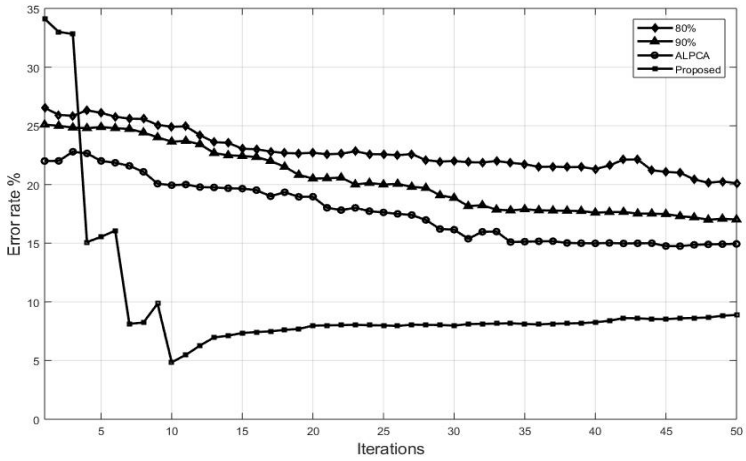


**Fig. 4.** Average Error Rates on Letter B VS Others

The statistic indexes of error rates on Letter B VS others over 100 runs are shown below:

**Table 5.** Statistic Indexes of Error Rates on Letter B over 100 Runs

| | ALPCA | Proposed | 80% | 90% |
|---|---|---|---|---|
| MAX | 0.5022 | **0.1958** | 0.4911 | 0.5192 |
| MIN | **0.0302** | 0.0334 | 0.0546 | 0.0496 |
| MEAN | 0.1489 | **0.0877** | 0.2013 | 0.1489 |
| STDEV | 0.1098 | **0.0343** | 0.0959 | 0.0977 |

From Table 5, the proposed algorithm gives attention to the structure of the whole dataset, so the classification results are very stable with little fluctuation.

### 5.2.3 Letter C VS Others

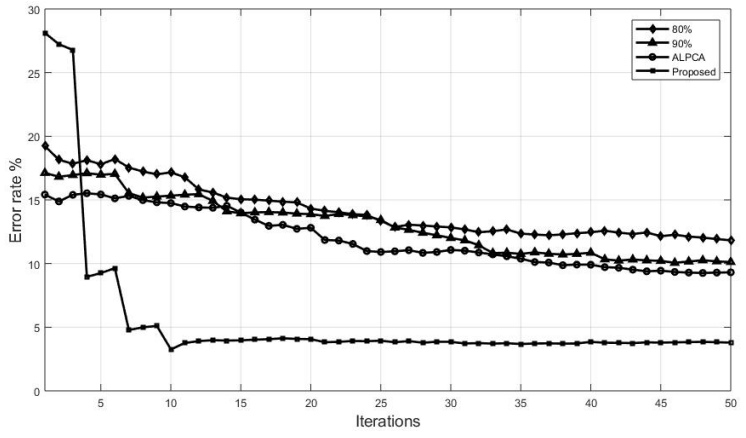


**Fig. 5.** Average Error Rates on Letter C VS Others

**Table 6.** Statistic Indexes of Error Rates on Letter C over 100 Runs

| | ALPCA | Proposed | 80% | 90% |
|---|---|---|---|---|
| MAX | 0.5880 | **0.1017** | 0.4255 | 0.4927 |
| MIN | 0.0140 | **0.0134** | 0.0237 | 0.0193 |
| MEAN | 0.0943 | **0.0361** | 0.1109 | 0.1015 |
| STDEV | 0.0960 | **0.0180** | 0.0805 | 0.0939 |

## 6 Conclusion

In this algorithm, we use SS to explore the subspace structure of the dataset. Then, a few key features are selected in the first several iterations, which promote the classification model to grasp the overall structure of the dataset and reduce the risk of falling into local optimum. Thirdly, the detailed features are added gradually, which help to improve the classification accuracy of the active learning model. Thus, it has good classification performance, and reduces the cost of sample labeling. After the validation with different datasets, the algorithm can effectively improve the stability and classification accuracy of active learning, and reduce the iteration number and labeling costs.

Different from other algorithms, our algorithm not only uses the label information of samples, but also uses the structure information of samples, so it can effectively reduce the cost of labeled samples. However, our algorithm increases the cost of computation, and its time complexity is $O(n^3)$. Therefore, the algorithm is suitable for the case that the cost of sample labeling is expensive or the labeled samples are rare, and it is not suitable for the case that there are a large number of labeled samples.

Future work will entail investigating other sparse algorithms, to lower the computational complexity of our algorithm, and further explore how to utilize the information of unlabeled data.

## References

[1] B. Settles, Active learning, Synthesis Lectures on Artificial Intelligence & Machine Learning (2012) 1-114.

[2] E. Elhamifar, R. Vidal, Sparse subspace clustering: algorithm, theory and applications, IEEE Transactions on Pattern Analysis and Machine Intelligence 35(11)(2013) 2765-2781.

[3] J.R. Adhikary, M.N. Murty, Feature selection for unsupervised learning, International Conference on Neural Information Processing, Springer-Verlag (2012).

[4] M. Goudjil, et al., A novel active learning method using SVM for text classification, International Journal of Automation and Computing 15(3)(2018) 290-298.

[5] M. Pérez-Ortiz, P.A. Gutiérrez, M. Carbonero-Ruz, C. Hervás-Martínez, Semi-supervised learning for ordinal Kernel Discriminant Analysis, Neural Networks: the Official Journal of the International Neural Network Society 84(2016) 57-66.

[6] F. Sun, Y. Xu, J. Zhou, Active learning SVM with regularization path for image classification, Multimedia Tools and Applications 75(3)(2016) 1427-1442.

[7] J. Zhang, G. Liao, N. Li, Combining active learning and local patch alignment for data-driven facial animation with fine-grained local detail, Neurocomputiong 398(2020) 431-441.

[8] D. Liu, Y. Liu, An active learning algorithm for multi-class classification, Pattern Analysis and Applications 22(3)(2019) 1051-1063.

[9] L.-X. Hong, et al., A novel kriging based active learning method for structural reliability analysis, Journal of Mechanical Science and Technology 34(4)(2020) 1545-1556.

[10] M. Gupta, S.A. Beckett, E.B. Klerman, On-line EEG denoising and cleaning using correlated sparse signal recovery and active learning, International Journal of Wireless Information Networks 24(2)(2017) 109-123.

[11] S. Hijazi, et al., Active learning of constraints for weighted feature selection, Advances in Data Analysis and Classification, 2020.

[12] H. Lei, et al., Improving active learning by data balance to reduce annotation efforts, Journal of Engineering (23)(2019) 8650-8653.

[13] J. Wright, A.Y. Yang, A. Ganesh, et al., Robust Face Recognition via Sparse Representation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2008.

[14] Y.C. Eldar, M. Mishali, Robust Recovery of Signals from a Union of Subspaces, Computer Research Repository, 2008.

[15] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions Pattern Analysis Machine Intelligence 22(8)(2000) 888-905.

[16] R. Glowinski, ADMM and non-convex variational problems, Splitting Methods in Communication, Imaging, Science and Engineering, Springer International Publishing, 2016.

[17] C.J. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (1998) 121-167.

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

[19] W. Huizinga, et al., PCA-based groupwise image registration for quantitative MRI, Medical Image Analysis (2016) 65-78.