ICIC Express Letters
Volume **11**, Number **7**, July **2017**

 ISSN 1881-803X
pp. **1159–1165**

A NOVEL FUZZY SUPPORT VECTOR MACHINE BASED ON DIFFERENTIAL EVOLUTION

Mengxuan Zhang[1] and Zhe Ju[2]

[1]School of Control Science and Engineering
Dalian University of Technology
No. 2, Linggong Road, Ganjingzi District, Dalian 116024, P. R. China
zhangmengxuan@mail.dlut.edu.cn

[2]College of Science
Shenyang Aerospace University
No. 37, Daoyi South Avenue, Shenbei New Area, Shenyang 110136, P. R. China
juzhe1120@hotmail.com

Received January 2017; accepted April 2017

Abstract. *This paper proposes a novel method (called DE-FSVM) which combines fuzzy support vector machine (FSVM) with differential evolution. The fuzzy membership function is redefined in consideration of the irregularly distributed dataset, and modified differential evolution is applied to searching the optimal parameters. The simulation and comparisons based on five UCI datasets and five classification approaches demonstrate that the proposed method exhibits the highest average classification accuracy.*
Keywords: Support vector machine (SVM), Fuzzy support vector machine (FSVM), Fuzzy-membership, Differential evolution (DE), Outliers and noises, Class imbalance learning

1. **Introduction.** Support vector machine (SVM) [1,2] is a machine learning method based on the VC-theory and structural risk minimization principle, and it has been applied to a variety of real-world classification problems successfully. However, there are usually some outliers and noises samples in the real datasets. Since the SVM algorithm considers all the training samples identically and gives them uniform weight, it may be not very successful when there are outliers and noises in training set [3,4]. In order to diminish the influence of noisy data, fuzzy support vector machine algorithm (FSVM) has been put forward [5,6]. Besides, the method called class imbalance learning (CIL) has been used to alleviate the impact of imbalance [7] in this paper. Fuzzy support vector machines for class imbalance learning (FSVM-CIL) [8] has emerged to deal with both the problem of noisy data and the problem of imbalance. While FSVM-CIL calculates the fuzzy membership based on the distance between the sample and its class-center, it may not be accurate when the dataset is irregularly-distributed.

It is well known that parameter selection is essential in improving the classification accuracy of SVM. The simple and straightforward way of parameter selection is grid search (GS) [9], but it is time-consuming and difficult. Some numerical optimization methods, which are more efficient than GS, have been proposed [10], while these approaches are instable and easy to trap into local optimum. To overcome the limitations above, evolutionary algorithms have been successfully applied in the selection of SVM parameters because of their high efficiency and global search ability [11-13]. And among the evolutionary algorithms, differential evolution is one of the most powerful stochastic real-parameter optimization algorithms [14]. In consideration of the advantage of differential evolution: simplicity, ease of implementation, reliability and high performance [15,16], it is applied

in this paper to searching for the optimal parameters. However, it should be noticed that the algorithm still needs to be modified to make it more efficient.

In this paper, a novel method is proposed to improve the FSVM for CIL, which can address the problem of both noise/outliers and imbalance, and is more efficient. In DE-FSVM, the fuzzy-membership values are assigned by both the distance between samples and their class center and the affinity around them in consideration of the irregular distribution of datasets. In addition, an adaptive differential evolution algorithm is used to select the optimal parameters to make the training process simple and improve the accuracy of classifier. DE-FSVM method is evaluated on five UCI datasets. Simulation results show that the DE-FSVM method whose parameters selected by adaptive differential evolution has the best performance compared with other SVM methods (SVM, FSVM, DEC, FSVM-CIL).

This paper is organized as follows. Section 2 briefly reviews the theory of SVM. Section 3 presents FSVM algorithm for imbalance. Section 4 describes modified differential evolution. In Section 5, simulation results are showed and illustrated. Section 6 gives the conclusion.

2. Fuzzy Support Vector Machine Learning Theory.

For the binary classification, the basic idea of SVM is to search for an optimal hyper-plane in the sample (kernel) space to maximize the classification interval of samples from two classes. For a given training set $\{(x_1, t_1), (x_2, t_2), \ldots, (x_l, t_l)\}$, where $x_i \in R^N$ represents a sample, and $t_i \in \{-1, 1\}$ denotes the class of that sample, for $i = 1, \ldots, l$. Nonlinear mapping function $\Phi(x)$ is introduced to map the training samples into a high dimensional space. By selecting the proper kernel function $K(x, y) = \Phi(x)^T \Phi(y)$ and introducing a slack variable ξ_i, the general form of SVM can be formulated as follows:

$$\begin{aligned} &\min_{\omega,\xi} \frac{1}{2} \|\omega\|^2 + C \sum_{i=1}^{l} \|\xi_i\|^2 \\ &\text{s.t. } t_i \left(\omega^T \Phi(x_i) + b\right) \geq 1 - \xi_i \\ &\quad \xi_i \geq 0,\ i = 1, 2, \ldots, l \end{aligned} \tag{1}$$

where C is the penalty parameter, ξ_i is the slack variable, and $\Phi(x)$ is nonlinear mapping. The pair (ω, b) is used to define the hyper-plane, and ω is a coefficient vector, b is a threshold.

The traditional SVM algorithm considers all the training examples uniformly and assigns the same weight to each sample. However, the real-world dataset is often unbalanced and contains noises, and a reasonable method is to assign different weights to the samples based on their importance. Assume that the first p examples are positive (namely $t_i = 1$, $i = 1, 2, \ldots, p$), and the remaining $l - p$ examples are negative (namely $t_i = -1$, $i = p + 1, p + 2, \ldots, l$). The general form of unbalanced FSVM can be represented as follows:

$$\begin{aligned} &\min_{\omega,\xi} \frac{1}{2} \|\omega\|^2 + C^+ \sum_{i=1}^{p} s_i^+ \xi_i + C^- \sum_{i=p+1}^{l} s_i^- \xi_i \\ &\text{s.t. } t_i \left(\omega^T \Phi(x_i) + b\right) \geq 1 - \xi_i \\ &\quad \xi_i \geq 0,\ i = 1, 2, \ldots, l \end{aligned} \tag{2}$$

where C^+ and C^- are the penalty parameter of positive class and negative class examples respectively. s_i^+ and s_i^- are fuzzy membership functions, which are used to reflect the importance of a sample in its class.

3. **Assigning Fuzzy-Membership Values.** The definition of fuzzy-membership in [5] only takes the distance between the sample and its class center as the measurement of its importance. And the noisy sample may be regarded as the normal sample in the irregularly distributed dataset, thus affecting the accuracy of algorithm. Figure 1 shows the elliptically distributed dataset with a noisy point. The distance between the noisy point x_2 and the class center is the same as that of normal sample x_1. According to Lin and Wang [5], these two points will be assigned to the equal fuzzy membership, which is evidently not in conformity with the reality. It is clear in Figure 1 that the space around the noisy point x_2 is relatively sparser than that of the normal point x_1.

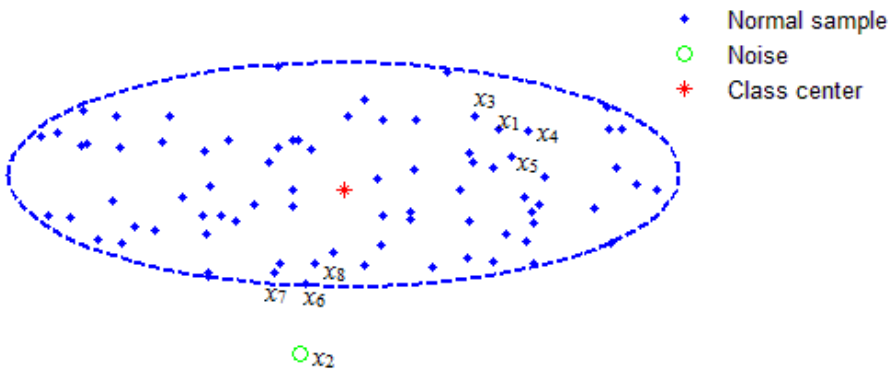

FIGURE 1. Elliptically distribution data with a noisy sample

Based on the observations above, the affinity around a sample should also be taken into account when assigning fuzzy membership. As shown in Figure 1, the average distance between the noisy sample x_2 and its three neighbors $\{x_6, x_7, x_8\}$ is much larger than that of the normal sample x_1. The K-nearest neighbor rule is introduced to measure the affinity around a sample. The affinity around a positive (negative) sample x_i can be defined as:

$$D_i^+ = \frac{1}{K} \sum_{x_j \in N_K^+(x_i)} \|x_i - x_j\|, \quad D_i^- = \frac{1}{K} \sum_{x_j \in N_K^-(x_i)} \|x_i - x_j\| \tag{3}$$

where $N_K^+(x_i)$, $N_K^-(x_i)$ represent the set of K neighbors of x_i in the positive and negative sample set, respectively. The value of K is set as 5 in this paper.

Intuitively, the value of D_i^+ (D_i^-) is negatively related to the density around a sample, and if it is denser around a sample, it is more likely to belong to the positive (negative) class and vice versa. Based on the distance between a sample and its class center and the affinity around a sample, the fuzzy membership in the paper is redefined as follows:

$$s_i^+ = \left[1 - \alpha * \frac{d_i^{cen+}}{\max_j \left(d_j^{cen+}\right) + \delta} - (1-\alpha) * \frac{D_i^+ - \min_j \left(D_j^+\right)}{\max_j \left(D_j^+\right) - \min_j \left(D_j^+\right) + \delta} \right]^m, \; i = 1, 2, \ldots, p \tag{4}$$

$$s_i^- = \left[1 - \alpha * \frac{d_i^{cen-}}{\max_j \left(d_j^{cen-}\right) + \delta} - (1-\alpha) * \frac{D_i^- - \min_j \left(D_j^-\right)}{\max_j \left(D_j^-\right) - \min_j \left(D_j^-\right) + \delta} \right]^m, \tag{5}$$

$$i = p+1, p+2, \ldots, l$$

where α and m are fuzzy parameters. δ is a really small positive number to guarantee that the fuzzy-membership is greater than zero and the value of δ is set to 0.0001. $d_i^{cen+} = \left\| x_i - \frac{1}{p} \sum_{j=1}^{p} x_j \right\|$ means the distance between a positive sample and its class center, and

$d_i^{cen-} = \left\| x_i - \frac{1}{l-p} \sum_{j=p+1}^{l} x_j \right\|$ denotes the distance between a negative sample and its class center, similarly.

Therefore, parameters needed to be regulated in DE-FSVM include penalty parameter C, kernel parameter γ and fuzzy parameters α and m. According to the result in [7], SVM algorithm can achieve better performance when C^+/C^- is set as the ratio of majority class number to minority class number. So the penalty factor C^+ is set as $C * (l-p)/p$ and C^- is set as C, $C \in [2^0, 2^{15}]$. The RBF kernel function in the paper is $K(x_i, x_j) = \Phi(x_i)^{\mathrm{T}} \Phi(x_j) = \exp(-\gamma ||x_i - x_j||^2)$ and the value range of γ is $[2^{-15}, 2^0]$. Fuzzy parameters α and m are both within the range $[0, 1]$.

4. **Differential Evolution.** Differential evolution (DE) algorithm is an adaptive global optimization algorithm and belongs to the popular evolutionary algorithm. There are four phases in DE: initialization, mutation, crossover and selection.

4.1. **Initialization.** A population containing NP individuals is generated randomly in the initialization phase and each individual represents an XN-dimensional parameter vector by the following form:

$$x_{ij} = x_{\min} + (x_{\max} - x_{\min}) \cdot rand(0,\ 1) \tag{6}$$

where $i = 1, 2, \ldots, NP$, $j = 1, 2, \ldots, XN$, $x_{\min}$ and $x_{\max}$ are the lower bound and upper bound of individual, respectively.

Then fitness value of each individual is evaluated and the optimal individual is selected.

4.2. **Mutation.** Mutation strategy DE/rand/1 is employed and the mutation vector will be generated by the following equation:

$$v_i = x_k + F \cdot (x_t - x_r) \tag{7}$$

where v_i represents the mutation vector of x_i, and x_k, x_t, x_r $(k \neq t \neq r \neq i)$ are three distinct individuals extracted randomly from the current population. F is the scale factor. In order to keep the diversity of population in early iterations and accelerate convergence at later evolution process, F is redefined as:

$$F = F_{\max} - \left(\frac{I\_iter}{N_{\max}} \right)^2 \cdot (F_{\max} - F_{\min}) \tag{8}$$

where $F_{\max}$ and $F_{\min}$ are the upper bound and lower bound of F, respectively. I_iter is an iterative variable and $N_{\max}$ is the maximum number of iteration.

4.3. **Crossover.** In the crossover phase, each dimension of individual x_i and its mutation vector v_i are exchanged with a certain probability CR and the trial vector u_i is generated:

$$u_{ij} = \begin{cases} v_{ij} & \text{if } \lambda < CR \text{ or } j = j_{rand} \\ x_{ij} & \text{otherwise} \end{cases} \tag{9}$$

where CR is the probability parameter. λ is a number generated in range $(0, 1)$ uniformly. $j = 1, \ldots, XN$, and $j_{rand} \in \{1, 2, \ldots, XN\}$ is a random index to ensure that at least one parameter in trial vector u_i is from the mutation vector v_i. The probability parameter CR is defined as follows to balance the exploration ability and exploitation ability of DE algorithm [17]:

$$CR = CR_{\min} + \frac{I\_iter}{N_{\max}} \cdot (CR_{\max} - CR_{\min}) \tag{10}$$

where $CR_{\min}$ and $CR_{\max}$ are the lower bound and upper bound of CR, respectively.

4.4. **Selection.** The fitness value of individual x_i and its trial vector u_i are compared and the winner will be a member in next generation. In order to find the parameters of maximum classification accuracy, the selection phase can be described in the following form:

$$x_i = \begin{cases} u_i & \text{if } f(u_i) > f(x_i) \\ x_i & \text{otherwise} \end{cases} \tag{11}$$

where $f(\cdot)$ denotes the fitness function.

The common classifier performance evaluation G-mean (Gm), which is the geometric mean of sensitivity Sn and specificity Sp, is adopted to assess the performance of a classifier in this paper. The definition of Sn, Sp and Gm are as follows:

$$Sn = \frac{TP}{TP + FN}, \quad Sp = \frac{TN}{TN + FP}, \quad Gm = \sqrt{Sn \cdot Sp} \tag{12}$$

where TP, TN, FP and FN represent the number of true positives, true negatives, false positives, and false negatives, respectively.

Ten times ten-fold cross-validation were implemented to ensure the reliability of simulation results, and the fitness function is the mean of these results:

$$f(C, \gamma, \alpha, m) = \frac{1}{10} \sum_{k=1}^{10} Gm_k \tag{13}$$

Then the parameters which result in the highest cross-validation accuracy are the best, so the objective of DE is to search the optimal parameters to maximize the fitness function.

5. **Experiment Results.** The proposed algorithm is evaluated by comparing it with SVM [2], DEC [7], FSVM [5] and FSVM-CIL [8] algorithms. Ten-fold cross-validation is carried out on each of the datasets to evaluate the performance of all classifiers. The wildly used LibSVM package [18] is used to train the SVM model and LibSVM-weights-3.20 package is employed to train all the FSVM models. All the numerical experiments are performed on 3.20 GHz/4.00 GB PC by using matlabR2014a software. Five datasets are selected from the UCI machine learning repository as the experimental datasets. The details of these datasets are shown in Table 1, which contains the number of positive examples (Pos_num), the number of negative examples (Neg_num), the total number of examples (Tot_num), the imbalance ratio (Im_ratio), the total number of classes (Tot_class), and the number of classes selected as positive classes (Pos_class).

Table 1. UCI datasets and related properties

Dataset	Pos_num	Neg_num	Tot_num	Im_ratio	Tot_class	Pos_class
Pima-Indians	268	500	768	1.87	2	1
Haberman	81	225	306	2.78	2	2
Ecoli	77	259	336	3.36	8	2
Glass	13	201	214	15.46	7	5
Yeast	51	1433	1484	28.10	10	5

The experimental results of SVM, DEC, FSVM, FSVM-CIL, DE-FSVM on five UCI datasets are shown in Table 2. It can be seen from Table 2 that the performance of FSVM is not necessarily better than that of traditional SVM algorithm even on dataset with small unbalance such as Pima-Indians and Haberman. It is because these datasets may be not standard spherical distributed. So it is reasonable and necessary to consider the affinity around samples in the calculation of fuzzy membership in the paper. The design of fuzzy membership in FSVM_CIL is the same as that in FSVM, and only the distance between the sample and its class center is considered in these algorithms, which causes the importance of a sample not to be well reflected in the irregularly-distributed data. In

TABLE 2. Comparison of the classification results $Gm(\%)$ on five UCI datasets

Dataset	Pima-Indians	Haberman	Ecoli	Glass	Yeast
SVM	67.41±0.63	52.80±2.01	87.16±1.05	92.65±3.49	42.45±3.64
FSVM	67.27±0.90	52.85±2.08	87.98±1.23	86.38±1.34	48.28±1.61
DEC	73.62±0.65	64.54±1.09	90.30±1.01	94.68±0.15	84.90±0.57
FSVM-CIL$_{lin}^{cen}$	73.95±0.70	64.76±1.95	90.20±0.99	93.51±0.26	84.91±0.45
FSVM-CIL$_{\exp}^{cen}$	73.59±0.57	65.23±1.14	90.36±0.68	94.80±0.28	84.98±0.48
DE-FSVM	**79.97±0.62**	**65.34±1.15**	**90.71±0.16**	**95.07±0.15**	**85.22±0.40**

TABLE 3. Optimal parameter of DE-FSVM on five UCI datasets

Parameters	Pima-Indians	Haberman	Ecoli	Glass	Yeast
$\log_2^C$	3.1275	10.8915	13.9095	10.8946	10.2135
$\log_2^\gamma$	−11.9835	−13.0504	−13.7835	−6.5673	−14.2515
α	0.6403	0.5128	0.3359	0.1034	0.7148
m	0.7025	0.6987	0.2820	0.0986	0.1978

addition, the proposed algorithm achieves the maximum Gm on datasets with different unbalance ratios, which means that the proposed algorithm is superior to SVM, DEC, FSVM, FSVM-CIL algorithms on imbalanced datasets in the presence of outliers and noises. Though three additional parameters (m, K and α) are introduced in calculating the fuzzy membership, it is not more time-consuming in search of the optimal parameters, because the application of differential evolution enables the method to find better solution with fewer iterations. Table 3 shows the optimal parameters of these algorithms on each UCI dataset.

6. **Conclusion.** This paper presents a novel fuzzy SVM algorithm used to process unbalanced data with noises/outliers and a modified differential evolution is used to optimize the parameters. Numerical experiments on five UCI datasets demonstrate the effectiveness of the proposed method. The next step is to evaluate the proposed method on more UCI datasets and try to apply other evolutionary algorithms in the parameter selection for better performance.

REFERENCES

[1] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer, New York, 1995.
[2] N. Cristianini and J. Schawe-Taylor, *An Introduction to Support Vector Machines*, Cambridge University Press, Cambridge, 2000.
[3] X. Zhang, Using class-center vectors to build support vector machines, *Proc. of the 1999 IEEE Signal Processing Society Workshop*, Madison, pp.3-11, 1999.
[4] Y. Liu and Y. Chen, Face recognition using total margin-based adaptive fuzzy support vector machines, *IEEE Trans. Neural Networks*, vol.18, no.1, pp.178-192, 2007.
[5] C. F. Lin and S. D. Wang, Fuzzy support vector machines, *IEEE Trans. Neural Networks*, vol.13, no.2, pp.464-471, 2002.
[6] X. Jiang, Z. Yi and J. Lv, Fuzzy SVM with a new fuzzy membership function, *Neural Computing & Applications*, vol.15, no.3, pp.268-276, 2006.
[7] R. Batuwita and V. Palade, Class imbalance learning methods for support vector machines, *Imbalanced Learning: Foundations, Algorithms, and Applications*, pp.83-99, 2013.
[8] R. Batuwita and V. Palade, FSVM-CIL: Fuzzy support vector machines for class imbalance learning, *IEEE Trans. Fuzzy Systems*, vol.18, no.2, pp.558-571, 2010.
[9] C. Hsu and C. Lin, A comparison of methods for multiclass support vector machines, *IEEE Trans. Neural Networks*, vol.13, no.2, pp.415-425, 2002.

[10] O. Chapelle, V. Vapnik and O. Bousquet, Choosing multiple parameters for support vector machines, *Machine Learning*, vol.46, no.1, pp.131-159, 2002.

[11] L. M. Saini, S. K. Aggarwal and A. Kumar, Parameter optimization using genetic algorithm for support vector machine-based price-forecasting model in national electricity market, *IET Generation, Transmission & Distribution*, vol.4, no.1, pp.36-49, 2010.

[12] X. Zhang, D. Qiu and F. Chen, Support vector machine with parameter optimization by a novel hybrid method and its application to fault diagnosis, *Neurocomputing*, pp.641-651, 2015.

[13] L. Demidova, E. Nikulchev and Y. V. Sokolova, The SVM classifier based on the modified particle swarm optimization, *International Journal of Advanced Computer Science and Applications*, vol.7, no.2, 2016.

[14] S. Das and P. N. Suganthan, Differential evolution: A survey of the state-of-the-art, *IEEE Trans. Evolutionary Computation*, vol.15, no.1, pp.4-31, 2011.

[15] K. Price, R. M. Storn and J. A. Lampinen, Differential evolution: A practical approach to global optimization, *Springer Science & Business Media*, 2006.

[16] S. Das, S. S. Mullick and P. N. Suganthan, Recent advances in differential evolution – An updated survey, *Swarm and Evolutionary Computation*, vol.27, pp.1-30, 2016.

[17] W. Xiang, X. Meng and M. An, An enhanced differential evolution algorithm based on multiple mutation strategies, *Computational Intelligence and Neuroscience*, vol.2, 2015.

[18] C. C. Chang and C. J. Lin, LIBSVM: A library for support vector machines, *ACM Trans. Intelligent Systems and Technology (TIST)*, vol.2, no.3, p.27, 2011.