A Novel Optimization Algorithm Integrating Immunity Clone and Differential Evolution for Parameter Selection of SVM

1,2Li Jun, 1,2Ding Lixin and 3Zou Qilin
1State Key Laboratory of Software Engineering, Wuhan University, Wuhan, 430072, China
2School of Computer, Wuhan University, Wuhan, China
3College of Computer Science and Technology, Wuhan University of Science and Technology, WUST, Wuhan, China

Abstract: The classification accuracy of Support Vector Machine (SVM) depends on parameters strongly. In nature, parameter selection is a search optimization process. A Differential Evolution (DE) algorithm is a real-coding optimal algorithm based on swarm evolution. It has powerful global searching ability. But it gets into premature convergence easily. So a novel hybrid optimization algorithm based on Immunity Clone (IC) and differential evolution is proposed for parameter selection of SVM. In this algorithm, clonal selection and receptor editing mechanism are inserted into the differential evolution process. Thirteen experimental results on UCI datasets distinctly show that compared with default parameters SVM classifier, the differential evolution algorithm, the proposed algorithm has higher classification accuracy.

Key words: Support vector machines, differential evolution algorithm, immunity clone algorithm, clone selection, parameter optimization

INTRODUCTION

Support Vector Machine (SVM) has been successfully applied in many fields such as classification and regression (Vapnik, 2000). In the research work to improve the learning ability and generalization performance of SVM, an important question is how to optimize the parameters of the support vector machine SVM to improve its classification accuracy.

In the related research, the grid search proposed in the literature (Hsu et al., 2003) is an alternative search method that is simple and easy to use, but its search ability is poor. Genetic algorithm proposed in (Zheng and Jiao, 2004) and particle swarm optimization algorithm proposed in (Shao et al., 2006) are for optimization selection of SVM parameters respectively, although these intelligent methods reduce the reliance on initial value selection, yet the algorithm principles and the thought are more complex and they are easy to fall into local optimum. Differential evolution algorithm is a kind of real number encoding optimization algorithm based on population evolution (Storn and Price, 1997; Qin et al., 2009). The algorithm can extract different information from the current population and guide the further search. The principles are relatively simple and it’s algorithm control parameters is relatively less, while the global search ability is far stronger. But it is easy to fall into local optimum; the phenomenon of premature convergence also exists. In order to overcome defects resulted from using the DE algorithm to solve the problem global optimization of SVM parameters, there introduces clonal selection and receptor editing mechanism (Cheng et al., 2012; De Castro and von Zuben, 2000) in the differential evolution process, enhances the diversity of population distribution and enhance the capability of global optimization of population.

The remainder of this study is organized as follows. The basic principle of the Support Vector Machine (SVM) is given in section 2. Section 3 introduces the differential evolution algorithm. Section 4 presents the immune clone algorithm. Section 5 elaborates the proposed SVM parameter selection method based on immune clone and differential evolution algorithm. Section 6 gives the experimental result of comparison of the proposed method with two other methods. Section 7 summaries this study.

SUPPORT VECTOR MACHINE

SVM separates the classes with a decision surface that maximizes the margin between the classes. The surface is often called the optimal hyperplane and the data points closest to the hyperplane are called support vectors.

Corresponding Author: Li Jun, State Key Laboratory of Software Engineering, Wuhan University, Wuhan, 430072, China
Provided sample pairs as \( \{x_i, y_i\}, \{x_i, y_i\}, \ldots, \{x_i, y_i\}\) \(i \times x, y\), among them, \(x_i \in \mathbb{R}^d\) is the input vector, \(y_i \in \{-1, 1\}\) is the corresponding output values for \(x_i, i = 1, 2, \ldots, 1\).

When the sample points approximately linear classification, the problem boils down to the following optimization problem:

\[
\min \frac{1}{2} \|w\|^2 + C \sum \xi_i \\
\text{s.t. } y_i \left(\langle w, x_i \rangle + b \right) + \xi_i \geq 1 (\xi_i \geq 0,i=1,2,\ldots,1)
\]

Among this, \(\|w\|^2 \in [0, 1]\) is called structural risk which is on behalf of the complexity of the model and make the function smoother, thereby can improve the generalization ability:

\[
C \sum \xi
\]

is known as the empirical risk which on behalf of the error of the model, and \(C\) is known as punishment parameter. When the sample points are shown to exhibit non-linear relationship, the original sample set will be mapped to the high dimensional feature space through a nonlinear mapping \(\phi(x)\), then make the linear classification in the high-dimensional feature space. The inner product operation on the high dimensional feature space can be defined as kernel function: \(K(x, x) = \phi(x) \cdot \phi(x)\), only need to use the function operate the variables in the original low dimension.

Using the duality theory in the optimization theory can get the final decision function:

\[
f(x) = \operatorname{sgn} \left( \sum y_i (x, x_i) + b^* \right)
\]

Several kernel functions help the SVM obtain the optimal solution. The most frequently used such kernel functions are the polynomial, sigmoid and radial basis kernel function (RBF). That is:

\[
K(x, y) = e^{-\frac{||x-y||^2}{\sigma^2}}
\]

or

\[
K(x, y) = e^{-\rho ||x-y||^2}
\]

Generally the RBF is applied most frequently, because it can classify high-dimensional data, unlike the linear kernel function. Additionally, the RBF has fewer parameters to set than a polynomial kernel. Therefore, this study applies an RBF kernel function in the SVM to obtain optimal solution. The major parameters that affect the performance of support vector machine are punished parameter \(C\) and RBF kernel parameter \(\sigma\). They must be set appropriately.

**DIFFERENTIAL EVOLUTION ALGORITHM**

Differential evolution algorithm can be used to solve the global optimization problems which have \(N\) continuous variables. In the population initialization phase, individuals in a population with an average random initial space in the search space. In the evolutionary stage, primarily through mutation, crossover and selection process until the stop condition is met. Three control parameters of original differential evolution algorithm, they are respectively, population size \(NP\), sealing factor \(F\) and the probability of crossover \(CR\).

**The differential mutation:** In the differential evolution algorithm, mutation operator is one of the most important operators. DE/rand/2 is used in this study:

\[
V_i = X_i + F(X_{a} - X_{b}) + F(X_{c} - X_{d})
\]

Among this, \(X_{rand}\) is the best individual in the current population, \(X_i\) is a Father individual, \(i, a, b, c, d\) are five individuals randomly selected from population, \(V_i\) is a variation vector, \(X_{a} - X_{b}\) is a difference vector and \(F \in [0, 1]\) is a zoom factor which used for the difference vector to zoom in, thus to control the search step length.

**Crossover operation**

Differential evolution algorithm use discrete crossover operator, including the Binomial Crossover and Exponential Crossover. Crossover operator let Variable vector \(V_i\) generated by mutation operator and father individual vector \(X_i\) do the discrete crossover and acquire the try vector \(U_j\), Binomial Crossover operator can also be expressed as:

\[
U_j(j) = \begin{cases} 
V_j(j) & \text{if} (\text{rand}[0, 1] < CR \ and \ j = i_{\text{rand}}) \\
X_j(j) & \text{otherwise}
\end{cases}
\]

Among this, \(j = 1, \ldots, D; i_{\text{rand}}\) is a random integer in the range \(1, D\) which ensures that at least one dimension of the attempt vector \(U_i\) from the variation vector \(V_i\) thus to avoid the same as the parent individual vector. In each mutation operator of differential evolution algorithm can be combined with index of crossover operator.

**Select operation:** Through mutation operator and crossover operator, differential evolution algorithm generates subgroup, there adopt the one-to-one selection operator to compare individual to corresponding parent.
individual, then the excellent individual can be saved to the next group. Optimization for minimizing the selection operator can be described as:

$$X_i = \begin{cases} U_i & \text{if}(f(U_i) \leq f(X_i)) \\ X_i & \text{otherwise} \end{cases}$$  \hspace{1cm} (6)

Among this, \(f(X_i)\) is the adaptive value for \(X_i\).

The selection operation of DE algorithm can increase the convergence rate of the algorithm, but the difference among individuals getting smaller and smaller, resulted that population easily converge at one point and the last population in the solution space can’t be re-search, so it is easy to fall into local optimum and appear premature convergence phenomenon. In order to guarantee the diversity of individuals, the study introduces the immune clone algorithm.

**IMMUNITY CLONE ALGORITHM**

Immunity clone can be dated back to the research in the biology immune system area. There are clone, hypermutation, combination of antibody and antigen, generating of memory cell and other process in the clonal selection mechanism. This mechanism can not only ensure the convergence rate, but also could maintain the diversity of antibodies.

The general algorithm is given as follows:

- **Step 1:** Initialize candidate possible solutions according to the problem to be solved
- **Step 2:** Calculate echo individual Affinities and extract the n antigens with high individual affinity
- **Step 3:** Clone these n-best individuals extracted to generate a temporary clone population \(C\), the number of clones of antibody-antigen affinity proportional to the number of cloning:

$$N_C = \sum_{i=1}^{n} \text{round}\left(\frac{\beta \cdot n}{i} + b\right)$$  \hspace{1cm} (7)

(\(\beta \in (0,1), \beta \) is the coefficient cloning, \(n\) is population size, \(b\) is a constant coefficient, a minimum number of cloning, Assume that \(b = 2\))

- **Step 4:** Hypermutate individuals in the temporary clone population \(C\) to obtain population \(V_i\), the degree of variation is inversely proportional to individual affinity (fitness) the formula as follows:

$$x_{i}^{v^{t+1}} = x_{i}^{v^{t}} + \alpha \cdot \eta \cdot x_{i}^{v^{t}} \cdot \text{rand}(0,1) - \alpha \cdot \eta \cdot x_{i}^{v^{t}} \cdot \text{rand}(0,1)$$

$$\alpha = \begin{cases} 1, \text{rand}(0,1) \leq P_n \\ 0, \text{else} \end{cases}$$  \hspace{1cm} (8)

\(X_{n}^{v^{t}}, X_{n}^{v^{t}}\) is a random number between 0 and 1, \(P_n = 0.5\):

$$\eta(t) = 1 - r \left[ \frac{\text{rand}(0,1)}{b} \right]$$  \hspace{1cm} (9)

\(b\) is a constant which represents the index variation of the control system of space, \(r \in (0,1)\), represents the base variation of the control system of space. In the early evolution, when \(r\) takes a smaller value, \(\eta(t)\) is 1, the variation of range become bigger, rather late in the evolution, \(g\) is close to GEN, local search is limited in the small range

- **Step 5:** Receptor editing (Ge et al., 2008), extract individuals which have the best degree after antigen stimulation in the mutated population and the remaining will be replaced by these individuals receptor editing formula as follows:

$$x_{i}^{v^{t}} = x_{i}^{v^{t}} + \frac{X_{n}^{v^{t}} - X_{n}^{v^{t}}}{m} + U_{r_{ed}}$$  \hspace{1cm} (10)

\(X_{n}^{v^{t}}, X_{n}^{v^{t}}\) is the upper and lower bounds of the location, \(m\) is the number of normal which represents individual position correct scaling factor, \(U_{r_{ed}}\) is the Logistic chaotic sequence and \(U_{r_{ed}}\) sequence is as follows:

$$U_{r_{ed}} = u \cdot U_i \cdot (1 - U_i), r = 0.1, 0.2, ..., 0 < U_i < 1$$  \hspace{1cm} (11)

\(u\) is the state control parameters, when \(u\) equals 4, initial value of \(U_i\) is \(U_0 = (0.25,0.5,0.75,1)\), the system is fully represented on the chaotic state. \(U_{r_{ed}}\) will traverse from 0 to 1

- **Step 6:** Replace the original antibody population with the antibodies extracted, individuals with low affinity to be eliminated

Cloning of biology immune mechanism system selectively generates the most reactive with the antigen antibody clone variation; thereby generate antibody diversity in the sample. Under this guiding ideology, the immune algorithm avoids falling into local extremum to the maximum extent possible and has a fast convergence to the global optimum.

**DE-CLONE-SVM**

This research proposes a novel hybrid optimization algorithm based on immunity clone and differential evolution, denoted as de-clone-svm model, to obtain the optimal parameter settings for SVM kernel parameters which result in a better classification accuracy rate.
Fig. 1: The flow chart of the de-clone-svm model

So the research uses the SVM classification accuracy as fitness function.

In optimization algorithm based on differential evolution (denoted as de-svm model), the difference among individuals getting smaller and smaller, resulted that population easily converge at one point. But clone, mutation, selection, suppression and replacement can be executed on each individual with Clone Algorithm, then these individual cross-validation accuracy obtained by svm training determine whether an individual into a local solution, if individual trapped in local solutions with high fitness clones from particle to be updated its suppression and replacement, thus escape from the local extremum interval.

Figure 1 shows the flow chart of the developed de-clone-svm model.

The outline of the proposed algorithm lists as follows:

- **Step 1:** Initialize population size NP, population X and algorithm parameters
- **Step 2:** Train individuals in the population X with svm to get each individual cross-validation accuracy, then find the best cross-validation accuracy BestAccuracy and its corresponding c and g, so Bestc = c, Bestg = g
  - **Step 3:** Determine whether the current accuracy and number of iterations reaches the end of the condition, if reached, go to step 14, otherwise the next step into the next generation iterative process
  - **Step 4:** Extract Pm (such as Pm = X*1/4) optimal individuals from the population X
  - **Step 5:** Clone Pm individuals to generate temporary populations C
  - **Step 6:** Hypermutate individuals in the population C to obtain populations V, the degree of variation is inversely proportional to individual fitness (individual cross-validation accuracy)
  - **Step 7:** If g(the current algebra) divides n(a fixed constant, such as n = 10), perform a receptor editing on inactive individuals in the population V, in order to improve their fitness (i.e., cross-validation accuracy), to obtain population S. Otherwise go to the next step
  - **Step 8:** Train individuals in the population X with svm to get each individual cross-validation accuracy, then find the best cross-validation accuracy temp_BestAccuracy and its corresponding temp_c and temp_g, If BestAccuracy > temp_BestAccuracy, then BestAccuracy = temp_BestAccuracy, Bestc = temp_c, Bestg = temp_g
  - **Step 9:** Extract NP optimal individuals from the population S to form population X
  - **Step 10:** Execute differential mutation in the population X to obtain population Y
  - **Step 11:** Execute crossover operation in the population Y to obtain populations Z
  - **Step 12:** Train individuals in the population Z with svm to get each individual cross-validation accuracy, then find the best cross-validation accuracy temp_BestAccuracy and its corresponding temp_c and temp_g, If BestAccuracy > temp_BestAccuracy, then BestAccuracy = temp_BestAccuracy, Bestc = temp_c, Bestg = temp_g
  - **Step 13:** Execute choose operation in the population Z, the offspring population in Z and X in the parent population individuals compete to select the fitness of individuals for outstanding replace X. Go to step 3
  - **Step 14:** Utilize Bestc and Bestg to train the training data set to obtain learning model, use the model to predict the test data set to get the prediction accuracy of the test data set
EXPERIMENT RESULTS

The platform adopted to develop the DE-Clone-SVM approach is Dell desktop with the following features: Intel Pentium Cores duo CPU E7200 2.53 GHz main frequency, 2GB RAM, a Windows XP sp3 operating system. Our implementation was carried out on the Matlab2010a development environment by extending the LIBSVM which is originally designed by Chang and Lin (2001).

In order to measure the performance of the developed DE-Clone-SVM approach, the following datasets in UCI Irvine Machine Learning Repository[11] are used: balance, diabetes, glass, haberman, iris, kr_V_kp, liver, parkinsons, sonar_all_data, splice, wdbc, wine, zoo (Hettich et al., 1998). Table 1 presents the properties of these datasets.

To guarantee valid results for making predictions regarding new data, the dataset is further randomly partitioned into training sets and independent test sets via a k-fold cross validation. Each of the k subsets acts as an independent holdout test set for the model trained with the remaining k-1 subsets. The advantages of cross validation are that all of the test sets were independent and the reliability of the results could be improved. This study used k = 10. Each of the 10 subset is used as test data sets in turn, so the program runs 10 times. The final classification accuracy is expressed in the form “mean±standard deviation”.

The searching range of parameters C and δ of SVM is between $10^{-7}$ and $10^3$. In the de-clone-algorithm, the parameters are initially set as follows. Iterations GEN is 200, population size NP is 40, scaling factor F is 0.8, crossover probability CR is 0.5, Clone proportion P_c is 0.5, index of spatial variation b is 2, base of spatial variation r is 0.6, individual position correct scaling factor m is 20.

To verify the excellence of DE-Clone-SVM for parameters optimization, we use the approach for comparing the performance of default parameter SVM classifier algorithms and DE algorithm. The experimental result is shown in Table 2.

We can distinctly see that in most of the datasets, the mean test accuracy of the de-clone-svm algorithm is better than the original svm and the de-svm and the standard deviation of the de-clone-svm algorithm is smaller.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>No. of instances</th>
<th>No. of features</th>
<th>No. of classes</th>
</tr>
</thead>
<tbody>
<tr>
<td>balance</td>
<td>625</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>diabetes</td>
<td>768</td>
<td>8</td>
<td>2</td>
</tr>
<tr>
<td>glass</td>
<td>214</td>
<td>9</td>
<td>6</td>
</tr>
<tr>
<td>haberman</td>
<td>306</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>iris</td>
<td>150</td>
<td>4</td>
<td>3</td>
</tr>
<tr>
<td>kr_V_kp</td>
<td>3196</td>
<td>36</td>
<td>2</td>
</tr>
<tr>
<td>liver</td>
<td>327</td>
<td>6</td>
<td>2</td>
</tr>
<tr>
<td>parkinsons</td>
<td>195</td>
<td>22</td>
<td>2</td>
</tr>
<tr>
<td>sonar_all_data</td>
<td>208</td>
<td>60</td>
<td>2</td>
</tr>
<tr>
<td>splice</td>
<td>1000</td>
<td>60</td>
<td>2</td>
</tr>
<tr>
<td>wdbc</td>
<td>569</td>
<td>30</td>
<td>2</td>
</tr>
<tr>
<td>wine</td>
<td>178</td>
<td>12</td>
<td>3</td>
</tr>
<tr>
<td>zoe</td>
<td>101</td>
<td>16</td>
<td>7</td>
</tr>
</tbody>
</table>

Table 2: Experimental results

<table>
<thead>
<tr>
<th>Dataset</th>
<th>SVM (%)</th>
<th>de-svm (%)</th>
<th>de-clone-svm (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>balance</td>
<td>98.7±2.12</td>
<td>99.05±2.01</td>
<td>99.68±0.67</td>
</tr>
<tr>
<td>diabetes</td>
<td>76.3±3.34</td>
<td>77.27±2.47</td>
<td>80.91±1.38</td>
</tr>
<tr>
<td>glass</td>
<td>45.2±4.05</td>
<td>54.09±7.86</td>
<td>54.54±13.71</td>
</tr>
<tr>
<td>haberman</td>
<td>68.0±9.65</td>
<td>83.87±4.56</td>
<td>83.87±0.00</td>
</tr>
<tr>
<td>iris</td>
<td>92.00±1.62</td>
<td>100.00±0.00</td>
<td>96.25±3.23</td>
</tr>
<tr>
<td>kr_V_kp</td>
<td>99.40±0.54</td>
<td>98.78±0.93</td>
<td>99.38±0.00</td>
</tr>
<tr>
<td>liver</td>
<td>66.88±11.04</td>
<td>52.43±6.24</td>
<td>76.97±6.42</td>
</tr>
<tr>
<td>parkinsons</td>
<td>82.63±11.91</td>
<td>85.00±4.71</td>
<td>90.50±7.25</td>
</tr>
<tr>
<td>sonar_all_data</td>
<td>84.00±9.37</td>
<td>93.34±2.46</td>
<td>91.43±2.01</td>
</tr>
<tr>
<td>splice</td>
<td>88.10±1.52</td>
<td>82.68±1.34</td>
<td>91.09±0.00</td>
</tr>
<tr>
<td>wdbc</td>
<td>87.68±8.89</td>
<td>88.24±3.51</td>
<td>94.92±0.55</td>
</tr>
<tr>
<td>wine</td>
<td>87.06±14.36</td>
<td>93.89±0.95</td>
<td>94.71±5.85</td>
</tr>
<tr>
<td>zoe</td>
<td>93.00±8.23</td>
<td>90.91±0.00</td>
<td>100.00±0.00</td>
</tr>
</tbody>
</table>

higher than it. De-svm-clone has less variance than de-svm in 8 datasets, with one almost the same and 4 a little higher than it. The smaller the standard deviation, the smaller the rise and fall of the solution will be. It means that the solution of de-clone-svm is more stable than others.

The obtained results clearly confirm the superiority of the DE-Clone-SVM algorithm compared to default parameters SVM classifier and the de-svm algorithm.

CONCLUSION

In this study, aiming at the shortcomings of differential evolution based parameters selection for support vector machine, combined with the clonal selection algorithm, a novel hybrid optimization algorithm based on immune clone and differential evolution is proposed. With clonal selection and receptor editing, the algorithm improves the diversity of population and avoids premature convergence. Simulation results show that the proposed algorithm greatly enhances classification accuracy of SVM.

AKNOWLEDGMENT

This research was supported by National Natural Science Foundation of China under Grant No. 61273303.
REFERENCES


