A NEW APPROACH FOR STATIC SECURITY ASSESSMENT OF POWER SYSTEMS USING CORE VECTOR MACHINE

R. Thamizh selvan, S. Ganapathy

Address for Correspondence

1 Assistant Professor, 2 Professor, Department of Electrical Engineering, Annamalai University, Annamalainagar-608002, India.

ABSTRACT

In this paper, a new methodology is presented for classification of power system security assessment using a machine learning algorithm. Here, Core Vector Machine (CVM) is utilized as a data classifier to evaluate of power system Static Security Assessment (SSA). One of the most important aspects for an efficient power system security evaluation is the proper selection of features. Rather than using all the features as inputs of a CVM classifier, a statistical based feature selection technique, F-Value feature selection has been used to reduce features from the larger data. The reduced feature set is then used as inputs for learning and validating the CVM classifier. The proposed algorithm is implemented in a standard IEEE 30 bus system. Simulation results reveal that, the implementation of F-Value feature selection along with CVM reduces the dimension of data space, which leads to reduced complexity and enhanced accuracy in power system static security assessment.

KEYWORDS: Core Vector Machine, F-Value Feature Selection, Minimum Enclosing Ball, Static Security Assessment.

1. INTRODUCTION

Security considerations have long been familiar as an essential part of power system planning, design, operation and control. It is imperative to meet the load changes or demands without violating the system operating constraints. The probable most common operational troubles arise, due to faults or disturbances which may cause trouble of deterioration of power quality. These situations can articulate a power system to unstable conditions, so it is vital to monitor the power system security status with operating condition changes and contingencies. The prediction of such problems would be exceptionally helpful in preventing them. Hence, there has been considerable attention in developing tools to identify, predict and control these security constraints. Thus, the operators need different computational tools to ensure system security. These tools must be accurate and fast to allow in real time with safer, reliable and economical behaviour for security assessment of power systems [1,2,3].

For understanding the static security assessment, the system should be designed to operate within normal operating ranges for credible load and generation patterns for base case operation and be designed to withstand the more probable contingencies without widespread system failure and instability, maintaining power quality within specified voltage and frequency fluctuation ranges and maintaining voltage and thermal loadings within operating limits [2]. The more probable contingencies include single contingency, overlapping single contingency and generator outage and trip-maintenance disturbances. Conventional methods for static security analysis involve solving full AC load flow for each contingency scenario. This method of security analysis is very much time engrossing and reproduces voluminous results, making it inflexible for real time applications [2,3]. To concentrate on this negative aspect, many Artificial Intelligence (AI) techniques have been presented in the literature to solve static security assessment problems. Self-Organizing Feature Map has been applied for the problem of static security assessment in [12] and Multilayer Feed forward with back propagation algorithm has been applied for the problem of static security assessment in [17]. The use of ANN based pattern recognition (PR) approach [2,3,18]Decision tree based security classifier [19], genetic based neural network, fuzzy logic combined with neural network [16] query-based learning approach in neural networks for static security evaluation process have also been reported. All these AI techniques pertain to some inconveniences similar to that in conventional methods [13].

Hence, to overcome this, Machine learning based classifier has been introduced for solving all the security assessment related problems. Till now, Support Vector Machine (SVM) has been widely used for solving many data classification problems and power system security related problems [15]. The SVM problem is usually solved as a Quadratic Programming (QP) problem. The existing solution strategies for this problem have an associated time and space complexity. This makes the application on SVM limited on data sets having a few thousands of elements. To overcome the draw backs of SVM in handling large data sets, in this paper, Core Vector Machine (CVM) algorithm [15-18] has been used for solving the Static Security Assessment problem of a standard IEEE 30 bus system.

Evaluating the similar patterns in the dataset will enhance the performance of the proposed classifier. So, finding a finest feature selection algorithm is an additional apprehension in this paper. Many feature selection algorithms are available in literature, such as Fisher Discrimination Analysis, Entropy Maximization, and Divergence Maximization etc [21]. The major problem with the existing feature selection algorithms is that they are mainly adopted for linearly separable classes and not well established on non-linearly separable classes [3]. In this paper, the process of feature selection is performed by a simple approach called F-value selection method [2]. The proposed CVM based classification approach is implemented on a standard IEEE 30 bus system. The simulation results prove that the CVM classifier gives an efficient classification, enhancing its suitability for on-line security assessment.

2. THE PROPOSED METHODOLOGY

2.1 Generation of Training and Testing Data set

The process of making the dataset is an off-line process which should include data for all possible operating conditions of the power system. The data set is created for various operating conditions by varying the load between 80% to 130% of the base
case. The variation in generation is concerned to its min-max capacity. The voltage magnitude are taken between 0.90 per unit to 1.10 per unit for all test systems and line over load limit in MVA is taken as 130% of base MVA flow. For each operating condition, single line outage has been simulated and load flow solution by Newton Raphson (NR) method is obtained. For each operating condition, the corresponding pattern vectors are obtained. Each operating condition has number of operating variables called as pattern vectors. In this work, voltage magnitude V̄i, voltage angle ̂i, real power generation Pi, reactive power generation Qi, real power demand Pi, reactive power demand Qi, the real power flow in line connected between buses i and j, Qi-j the reactive power flow in line connected between buses i and j, line MVA between buses i and j have been considered. Evaluating the security constraints, each pattern is labeled as secure or insecure state.

2.2 Core Vector Machine

Recently, a number of procedures have been projected in the literature for achieving multiclass classification, but the data handling capability still needs much attention. Several real-world applications typically deal with a massive collection of data’s and hence the main issue in using SVM here is that of scalability. To overcome the problem of handling large data set, CVM has been proposed. The CVM is a technique for scaling up a two-class SVM [16] to handle large datasets. The CVM applies kernel methods to data-intensive applications involving large datasets. In CVM, the quadratic optimization problem involved in the SVM is formulated as an equivalent Minimum Enclosing Ball (MEB) problem [20]. Then, a fast approximation has been used to obtain a near-optimal solution using core sets.

2.2.1 MEB Problem

Given a set of points \( S = \{x_1, ..., x_m\} \), where each \( x_i \in \mathbb{R}^d \), the MEB of \( S \) (denoted by MEB(S)), is the smallest ball that contains all the points in the set \( S \). Let \( B(c, R) \) be the ball with centre \( c \) and radius \( R \). Given a \( \varepsilon > 0 \), a ball \( B(c, (1+\varepsilon)R) \) is a \((1+\varepsilon)\)-approximation of MEB(S) if \( R \leq \varepsilon_{\text{MEB}}(S) \) and \( S \subset B(c, (1+\varepsilon)R) \), where \( \varepsilon \) is a small positive number, and \( \varepsilon_{\text{MEB}}(S) \) is the radius of exact solution for MEB(S).

The conventional algorithms to find the exact MEBs are not efficient for the problems with \( d > 30 \). Hence, it is of practical interest to study faster approximation algorithms that return a solution within a factor of \( (1+\varepsilon) \) of the optimal value. In many shape-fitting problems, it is found that solving the problem on a subset, called the core set \( Q \subset S \) of points from \( S \), can often give an accurate and efficient approximation. As it can be seen in Figure 1, a subset \( Q \subset S \) is a core set of \( S \) (the set of squares marked in Figure 1) if an expansion by a factor \( (1+\varepsilon) \) of its MEB contains \( S \), i.e., \( S \subset B(c, (1+\varepsilon)R) \), where \( B(c, R) \) is equal to MEB(Q) [20].

The core set obtained through solving the MEB problem can be used to solve classification tasks on large datasets. The MEB problem is to find the smallest ball enclosing all training data, constrained optimization problem in equation (1). A surprising property of the CVM is that the number of iterations, and hence the size of the final core set, depends only on \( S \) but not on \( d \) (dimension of samples) or \( m \) (number of training samples). The independence of \( d \) is important in applying this algorithm to kernel methods, as the kernel-induced feature space can be infinite dimensional [15-20].

**Figure 1:** Inner circle is the MEB of a set of squares, and its \((1+\varepsilon)\) expansion (outer circle) coordinates all points and the set of squares represents each core set.

2.2.2. Considering the MEB Problem as a QP Problem

Let \( \varphi \) be the feature map corresponding to kernel \( k \). Given a set of q-mapped points \( S = \{\varphi(x_1), ..., \varphi(x_m)\} \), its MEB, denoted \( B(c^*, R^*) \) with centre \( c^* \) and radius \( R^* \), is the smallest ball that encloses all these points [23]:

\[
\min_{c, R} R^2, \quad \forall i \in \{1, \ldots, m\} \quad \|c - \varphi(x_i)\|^2 \leq R^2 \quad \forall i.
\]

\[
\max_{\alpha_i} \sum_{i=1}^{m} \alpha_i k(x_i, x_j) - \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j k(x_i, x_j)
\]

subject to \( \alpha_i \geq 0 \) and \( \sum_{i=1}^{m} \alpha_i = 1 \), where \( \alpha_i \) (i = l ... m) is the Lagrange multiplier.

Equation (2) can be rewritten in matrix form, as follows:

\[
\alpha^T \text{diag}(k) - \alpha^T k \alpha
\]

\[
\alpha \geq 0, \quad \alpha^T 1 = 1,
\]

where \( \alpha = [\alpha_1, \ldots, \alpha_m]^T, 0 = [0, \ldots, 0]^T, 1 = [1, \ldots, 1]^T \) and \( k_{x,y} = [k(x,y)]^T \) is the kernel matrix. As it can be seen, Eq. (3) is a QP problem when \( k \) satisfies \( k(x_i, x_j) = k \) (and \( k \) is a constant). Using this point, Eq. (3) can be rewritten as follows:

\[
\max_{\alpha} -\alpha^T k \alpha
\]

\[
\alpha \geq 0, \quad \alpha^T 1 = 1
\]

The optimal \( \alpha \) can be obtained and determined using Eq. (4). Also, for \( c \) and \( R \),

\[
C = \sum_{i=1}^{m} \alpha_i \varphi(x_i)
\]

\[
R = \sqrt{\alpha^T \text{diag}(k) - \alpha^T k \alpha} = \sqrt{k - \alpha^T k \alpha}
\]

Therefore, any QP of this form can be regarded as an MEB problem and vice versa. Here core set, the ball’s centre, and radius at the \( \varepsilon \) iteration are denoted by \( S_c, c, R \) respectively. Moreover \( \varphi(x) \) denotes the feature map corresponding to the transformed kernel \( k \). Similar to SVM solvers the CVM algorithm requires the input of the termination parameter \( \varepsilon \). In particular, it can be shown that when \( \varepsilon = 0 \), CVM outputs the exact clarification of the kernel problem. When \( \varepsilon > 0 \), it becomes an \((1+\varepsilon)^2\) approximation algorithm.

2.3 CVM Algorithm

From the above points, the CVM algorithm can be summarized as follows [25]:
Step 1: Initialize parameters, $S_0 = \{\varphi(z_0)\}$, $c_0 = \varphi(z_0)$, and $R_0 = 0$.
Step 2: Terminate if no $\varphi(z)$ falls outside $B$ ($c_0$, $(1 + \epsilon)R_0$); otherwise, let $\varphi(z_0)$ be such a point.
Then, the set $S_{t+1}$ is equal to $S_t \cup \{\varphi(z_0)\}$.
Step 3: Find $\text{MEB}(S_{t+1})$ and set $C_{t+1} = \text{MEB}(S_{t+1})$.
Step 4: Increment $t$ by 1 and go back to Step 2.

2.4 Feature Selection

Using CVM to perform pattern classification, variables must be presented to the CVM for training. Suppose a pattern $X$, consists of $n$ variables. As the size of power system increases, the number $n$ of variables for a pattern would be very large. It is not undesirable nor is it necessary to use all the available variables to obtain the pattern discrimination function. There is a need to determine a relatively small number or subset of variables which will be distinctive for each of the classes of pattern. The process of dimensionality reduction is referred to as feature selection. The objective of feature selection is to enhance biased information of the measurement data and to reduce the dimensions of training pattern for easy implementation. Thus, a large scale power system can be handled with relatively small artificial neural networks.

The process of finding a reduced feature set can be concluded in the following stages; first the features are selected from pattern vector based on maximization of a criterion function. The $F$-value defined by equation (9) is used as the criterion function for selection of a variable as feature

$$F = \frac{[m_0 - m_i]}{\sigma_i^2}$$

Where, $m_0$ - Mean of the variable in the secure class, $m_i$ - Mean of the variable in the insecure class, $\sigma_i^2$ - Variance of the variable in the secure class, $\sigma_i^2$ - Variance of the variable in the insecure class.

2.5 The Process of Feature Selection

The top $m$ variables which have the most information about class separability and less correlation than those not selected will be chosen as features of a pattern. Selecting a suitable numbers $d$ of features is a tradeoff between classification accuracy and CVM design.

The process of feature selection is described as follows:
Step 1: Calculate index $F$, of every variable ($i = 1, 2, \ldots n$);
Step 2: Rank these variables in a descending order according to their index $F$;
Step 3: Go to the first ranked variable;
Step 4: Calculate the correlation coefficients of all lower-ranked variables with respect to this variable;
Step 5: Eliminate all lower-ranked variables whose correlation coefficients are greater than the threshold value;
Step 6: Go to the next-highest-ranked variable and go to Step 4, until all $n$ variables are either ranked or discarded;
Step 7: A set of $m$ ranked variables are selected as the key features;
Step 8: Apart from the $m$ key features, other variables which are ranked in a descending order according to their index $F$ are added after the key features;
Step 9: Get the optimal set of feature set.

The optimal set of feature set serves as an input database for designing the CVM classifier. The overall performance of any security classifier is scaled by using the following measures.

Accuracy: It defines the quality or state of being correctly classified.
Misclassification rate: It defines the numbers of samples wrongly classified.
False alarm: Is one in which the assessment says that the system is insecure while in reality no system constraints have been violated.
False dismissals: Is one in which the assessment says that, a critically insecure case was judged to be secure.

In power system security, the false alarms are not of much hurt, but false dismissals may lead to atrocious collapse. Having chosen the data set, the next step is to decide the number of features from the data set, before crafting the CVM classifier.

3. SIMULATION RESULTS AND DISCUSSION

The design of CVM based classifier model for static security assessment is implemented and tested in a standard IEEE 30 bus test system and the effectiveness of the proposed classifier has been demonstrated by using Tanagra software and comparing its results with SVM. The data set required for training and testing phases are obtained by off-line simulation performed using MATPOWER Toolbox with MATLAB. This data set is obtained by varying the generation and load from 80% to 130% of their base case value with generation variation restricted to their minimum and maximum limits.

The standard IEEE 30 bus sample system has 6 generators, 30 buses, 41 lines and 6 condensers. One at a time, outage studies are performed and form the set of disturbances to be utilized for steady state security in the Power system. The patterns or variables are generated through the load flow results. The generated variable set consists of 24 numbers of voltage magnitude variables ($V_i$), 29 numbers of voltage angle ($\delta_i$), 6 numbers of real power generation variables ($P_{gi}$), 6 numbers of reactive power generation variables ($Q_{gi}$), 20 numbers of real power demand variables ($P_{Di}$), 20 numbers of reactive power demand variables ($Q_{Di}$), 41 numbers of active real power flow variables ($P_{i-j}$), 41 numbers of reactive power flow variables ($Q_{i-j}$), and 40 numbers of line MVA variables ($S_{i-j}$). The unimportant variables at certain buses such as zero load, zero generation and constant values are neglected and finally 227 patterns are considered for classification process. All feasible 227 patterns are subjected to static security check with voltage limit and line flow limit.

The results of data generated for training, testing of CVM classifier and feature extraction are shown in Table 1. For a possible 904 operating scenarios, 753 operating scenarios are found to be secure and the remaining 151 cases are found to be insecure. The training and testing samples are split at random by the ratio of 90% (809cases) for training phase and 10% (95 cases) for testing phase.

3.1 Performance evaluation on feature selection for SSA

Latest efforts have uncovered that feature subset selection can have an optimistic shape on the performance of machine learning algorithms. A few algorithms can be slowed or their performance adversely affected by too much data some of which may be irrelevant or redundant to the learning task. Feature subset selection, then, is a way for enhancing the performance of learning algorithms, reducing the hypothesis search space, and, in some cases, reducing the storage requirement. This paper describes a
feature subset selector that uses a correlation based heuristic method to determine the goodness of feature subsets and evaluates its effectiveness with CVM classifier.

An optimal set of patterns selected by using feature selection process is shown in Table 2. The effectiveness of the dimensionality reduction is determined with a threshold value of 0.8 and the highly correlated variables are discarded from the total pattern variables. i.e. Good feature subsets contain features highly correlated (predictive of) with the class, yet uncorrelated with (not predictive of) each other.

Table 1: Data set for Training and Testing Phases

<table>
<thead>
<tr>
<th>Scenarios</th>
<th>Overall</th>
<th>Training</th>
<th>Testing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Total no of cases</td>
<td>904</td>
<td>899</td>
<td>95</td>
</tr>
<tr>
<td>Secure cases</td>
<td>755</td>
<td>720</td>
<td>33</td>
</tr>
<tr>
<td>Insecure cases</td>
<td>151</td>
<td>89</td>
<td>62</td>
</tr>
</tbody>
</table>

Table 2: Feature Selection Process

<table>
<thead>
<tr>
<th>Case study</th>
<th>Standard IEEE30 Bus system</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of pattern variables</td>
<td>227</td>
</tr>
<tr>
<td>No. of features selected</td>
<td>106</td>
</tr>
<tr>
<td>Dimensionality Reduction</td>
<td>46.6 %</td>
</tr>
</tbody>
</table>

3.2 Performance evaluation of CVM classifier for SSA

Based on the relevant definitions [2], the following formulations are obtained to compute the performance of CVM classifier.

Classification Accuracy (%) = \( \frac{\text{Number of correct samples}}{\text{Total samples}} \times 100 \)

Misclassification (%) = \( \frac{\text{Number of false samples}}{\text{Total samples}} \times 100 \)

False Alarm (%) = \( \frac{\text{Number of false alarms}}{\text{Total true secure states}} \times 100 \)

False Dismissal (%) = \( \frac{\text{Number of false dismissals}}{\text{Total true insecure states}} \times 100 \)

3.3 Evaluation without Feature selection

The proper selection of optimal values for CVM performance parameters decides the higher value of classification accuracy and minimal error rate. In this research paper, the following parameters listed below have been utilized for static security assessment using CVM classifier.

The optimal values of CVM performance parameters are provided in Table 3. The optimum value of ‘gamma’ is selected as 0.5. This value of ‘gamma’ forces to move the support vectors within their boundaries resulting in higher classification accuracy. The tolerance of termination criteria ‘epsilon’ is optimized as 0.0001.

Results shown in Table 4 prove that obtaining the useful support vectors from the whole dataset is the fundamental part of this evaluation process. The number of support vectors for secure and insecure cases are presented. Moreover, support vectors are calculated based on the rule of inverse square distance kernel function that defines the boundaries between secure and insecure classes. The performance of CVM classifier with 227 attributes are shown in Table 5. Based on this, the system is assessed for static security with 93.6 % (89/95) of testing accuracy.

The performance of CVM classifier with 106 attributes, after feature reduction with 95.8 (91/95) of testing accuracy is shown in Table 6. From the above simulation results, the classification accuracy of CVM classifier with feature selection is 95.8 % as compared with accuracy of 93.6 % without feature selection. It is clearly evident that the performance of the CVM classifier is improved with selection of good feature set and elimination of surplus data from the overall data set.

Table 3: Parameters of CVM classifier

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Optimal Values</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel type</td>
<td>Inverse Square distance</td>
</tr>
<tr>
<td>Degree of kernel</td>
<td>1.00</td>
</tr>
<tr>
<td>Gamma</td>
<td>0.5</td>
</tr>
<tr>
<td>Tolerance of termination</td>
<td>0.0001</td>
</tr>
<tr>
<td>C (Complexity Cost)</td>
<td>1</td>
</tr>
<tr>
<td>Use shrinking heuristics</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 4: Classifier characteristics during evaluation

<table>
<thead>
<tr>
<th>Characteristic</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of classes</td>
<td>2</td>
</tr>
<tr>
<td>Number of support vectors</td>
<td>658</td>
</tr>
<tr>
<td>Number of support vectors for secure</td>
<td>569</td>
</tr>
<tr>
<td>Number of support vectors for insecure</td>
<td>89</td>
</tr>
</tbody>
</table>

Table 5: Classification of Static Security using CVM classifier without FS

<table>
<thead>
<tr>
<th>Performance Evaluation</th>
<th>Without Feature selection (227 patterns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>Testing</td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>99.8 (807/809)</td>
</tr>
<tr>
<td>Misclassification (%)</td>
<td>0.2 (2/809)</td>
</tr>
<tr>
<td>False alarm (%)</td>
<td>0.13 (1/720)</td>
</tr>
<tr>
<td>False Dismissal (%)</td>
<td>2.2 (2/89)</td>
</tr>
</tbody>
</table>

Table 6: Classification of Static Security using CVM classifier with FS

<table>
<thead>
<tr>
<th>Performance Evaluation</th>
<th>With Feature selection (106 patterns)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Training</td>
<td>Testing</td>
</tr>
<tr>
<td>Accuracy (%)</td>
<td>99.8 (807/809)</td>
</tr>
<tr>
<td>Misclassification (%)</td>
<td>0.2 (2/809)</td>
</tr>
<tr>
<td>False alarm (%)</td>
<td>0 (0/720)</td>
</tr>
<tr>
<td>False Dismissal (%)</td>
<td>1.12 (1/89)</td>
</tr>
</tbody>
</table>

Table 7: Comparative Results of Static Security classification

<table>
<thead>
<tr>
<th>Performance</th>
<th>CVM classifier</th>
<th>SVM classifier</th>
<th>ANN</th>
</tr>
</thead>
<tbody>
<tr>
<td>Accuracy (%)</td>
<td>95.8</td>
<td>95.2</td>
<td>82.6</td>
</tr>
<tr>
<td>Misclassification (%)</td>
<td>4.2</td>
<td>4.76</td>
<td>13.29</td>
</tr>
<tr>
<td>False alarm (%)</td>
<td>3.03</td>
<td>4.16</td>
<td>8.79</td>
</tr>
<tr>
<td>False Dismissal (%)</td>
<td>4.83</td>
<td>5.5</td>
<td>12.86</td>
</tr>
</tbody>
</table>

3.4 Evaluation of CVM Classifier with Feature Selection

The comparative study of CVM classifier with SVM and ANN for static security assessment is shown in Table 7. Results prove that CVM is the finest classifier within a preselected contingent environment. In addition, CVM-based security assessment algorithm produces enough support vectors. Therefore, it is faster than the existing methods. Simulation results show that the proposed CVM-based security assessment has small training time and smaller vector dimension compared with SVM and Neural Network.

4. CONCLUSION

The proposed CVM is an absolutely novel method adopted for constructing an optimal hyper plane via different kernels that separate the two classes with optimal margin in classification problems. The proposed correlation based feature selection algorithm is an efficient method to deal with the problem of high dimensionality in the design of machine learning classifiers. The experiment proves that the classification accuracy of core vector machine is equivalent to that of standard support vector machine. However, in the training of large scale data, core vector machine possesses fast training speed and takes up less space compared with standard support vector machine. Therefore, the overall performance of core vector machine is
superior to that of standard support vector machine and it is applicable to the learning of large scale power system models in online and real time applications. The proposed model holds the promise as quick classifier for static security assessment of large scale power systems.

5. ACKNOWLEDGEMENT

The authors wish to thank the authorities of Annamalai University, Annamalai nagar, Tamilnadu, India For the facilities provided to prepare this paper.

REFERENCES


