Eng Int Syst (2001) 4: 205–211
© 2001 CRL Publishing Ltd

Engineering
Intelligent Systems

Neural networks and support vector machines applied to power systems transient stability analysis

L. S. Moulin*, A. P. Alves da Silva*, M. A. El-Sharkawi [†] and R. J. Marks II[†]

**Federal Engineering School at Itajuba - EFEI - Institute of Electrical Engineering, Av. BPS, 1303, Itajuba, MG, 37500-903, Brazil*
Email: lsm,alex@iee.efei.br
[†] University of Washington, Department of Electrical Engineering, 253 EE/CSE Building - Campus Box 352500 - Seattle, WA 98195-2500, USA
Email: elsharkawi,marks@ee.washington.edu

The Neural Network (NN) approach to the Transient Stability Analysis (TSA) has been presented as a potential tool for on-line applications, but the high dimensionality of the power systems turns it necessary to implement feature extraction techniques to make the application feasible in practice. This paper presents a new learning-based nonlinear classifier, the Support Vector Machines (SVMs) NNs, showing its suitability for power system TSA. It can be seen as a different approach to cope with the problem of high dimensionality due to its fast training capability, which can be combined with existing feature extraction techniques. SVMs' theoretical motivation is conceptually explained and they are applied to the IEEE 50 Generator system TSA problem. Aspects of model adequacy, training time and classification accuracy are discussed and compared to stability classifications obtained by Multi-Layer Perceptrons (MLPs).

Keywords: Feature Extraction, Support Vector Machines, Neural Networks, Transient Stability Analysis

1. INTRODUCTION

The Transient Stability Analysis (TSA) is a crucial operation procedure to ensure secure performance of a power system experiencing a variety of disturbances and operating condition changes. The power system operates in a secure manner, from the transient stability viewpoint, when the generators maintain synchronism after the system is subjected to severe disturbances.

In the last few decades, TSA methods of practical use have been developed, and the transient stability schemes of current use are mainly based on time-domain simulations [1]. These techniques, however, require the numerical solution of a system of nonlinear equations using time-consuming numerical integrations for each contingency. With the power systems expansion and the increase in their complexity, the dynamic security analysis has become a very crucial and complex process. The current deregulation trend and the participation of many players in the power market are contributing to the decrease in the security margin [2]. This makes the security evaluation even more important, and demands the investigation of fast and accurate techniques to allow on-line TSA.

The NN approach has been introduced as an alternative solution for the analytical TSA [3,4], and has been recently studied with potential use for real-world, large-scale power systems [4–6]. In such a process, the NN-based TSA would be applied to a group of selected critical contingencies. The nonlinear input/output mapping capability of a NN can be used to produce a security index that classifies the current operating point as secure or insecure [3, 5–7]. The NN uses training data sets that are representative of different loading conditions and generation schedulings, different types of contingencies and different topology configurations.

Although successfully applied to TSA [3, 5, 8], Multi-Layer Perceptrons (MLPs) implementations require extensive training process. In general, this is the major drawback for NN applications in large power systems with hundreds (even thousands) of generators, because such a large grid will require a large number of input variables to train a NN. This can be a prohibitive task. Therefore, a feature extraction/selection method is needed to reduce the dimensionality of the NN's input space. The main objective is to use as little number of inputs as possible to reduce the NN training time, while maintaining a high degree of classification accuracy [9].

A new type of nonlinear learning based classifier has been recently introduced which has very interesting theoretical promises, the Support Vector Machines (SVMs) NNs [10]. They can map complex nonlinear input/output relationships with good accuracy and they seem to be very well suited for the TSA application [11]. SVM classifiers rely on training points located on the boundary of separation between different classes, where the stability evaluation is critical. A good theoretical development of the SVM NN, due to its foundations on the Statistical Learning Theory (SLT) [10] made it possible to devise fast training techniques even with large training sets and high input dimensions [12–14]. This feature can be exploited as an approach to address the problem of high input dimension and large training datasets in the TSA problem.

However, the SVMs capabilities cannot be explored without a good understanding of their conceptual details. In this paper, the SVM classifier is explained and analyzed in terms of advantages and disadvantages. The aim is the application to power system TSA, which is developed as follows: three feature extraction techniques are applied to the training/test transient stability data with the objective of dimensionality reduction, as presented in [9]. Stability classifications are obtained by MLP and SVM NNs, comparing the good generalization capacity of both models and exploiting SVMs' fast training. Expectations are that input feature dimensionality reduction is of lower concern for SVMs due to their fast training, but accuracy must be checked for complete and reduced features sets. Results for the IEEE 50-Generator system are presented, discussed and compared in terms of modeling characteristics, generalization performance and training time.

The structure of the paper is as follows. Section 2 presents the feature extraction/selection techniques as applied in the TSA. In Section 3, a summarized description of SVM classifiers is sketched with the conceptual ideas and discussions on advantages and disadvantages. In Section 4, the TSA application and results are presented. Finally, the conclusions are drawn in Section 5.

2. FEATURE EXTRACTION/ SELECTION

The feature extraction problem can be explained by assuming the classification task in 2 disjoint classes with a training set T of ordered pairs $(\boldsymbol{x}_i, y_i)$, $T = \{x_i, y_i\}_{i=1}^{N}$, where $\boldsymbol{x}_i$ is a real-valued n dimensional vector (i.e. $\boldsymbol{x}_i \in R^n$) and $y_i \in \{+1, -1\}$ is a label that represents the class to which it belongs. The feature extraction goal is to determine a transformation $\boldsymbol{f} = F(A, \boldsymbol{x})$ from the original space R^n to a subspace R^d (for dimensionality reduction, $d < n$), where A is a matrix of transformation parameters and $\boldsymbol{f} \in R^d$. If the feature extraction/selection is successful, a point in R^d can be assigned to one of the 2 classes with a minimum error. Hence, the expected number of misclassifications for a test set should be as low as possible.

A linear feature extraction is performed when $\boldsymbol{f} = A \cdot \boldsymbol{x}$, as given in the Principal Components Analysis (PCA) technique. Another case is when A is a $d \times n$ matrix made of 'zeros' and 'ones', with only a single 'one' in each row. In this case, feature extraction becomes feature selection. In other words, feature vector candidates, $\boldsymbol{f} \in R^d$, are generated by selecting d components (or input features) from the original n-dimensional input vector, $\boldsymbol{x} \in R^n$. In this case, an exhaustive search to find the best feature vector would require examining all possible d-subsets with respect to some evaluation function, $C(A)$. The number of possibilities grows exponentially with the size of d, making exhaustive search impractical. Therefore, a heuristic search method is required to search through the space of possibilities while minimizing $C(A)$.

In this paper, the feature selection techniques use a distance measure evaluation function based on the Fisher's Discriminant function (FD) [16]. It takes a particular feature vector candidate, $\boldsymbol{f} \in R^d$, made of d selected components, and determines the vector v that maximizes the Fisher's linear discriminant function:

$$FD(\nu) = \frac{\nu^t S_B \nu}{\nu^t S_w \nu} = \frac{|\tilde{m}_1 - \tilde{m}_2|}{\tilde{s}_1^2 + \tilde{s}_2^2} \qquad (1)$$

S_B is the *between-class scatter* matrix and S_w is the *within-class scatter* matrix. After the data points are projected onto ν, the mean points of each class, $\tilde{m}_1$ and $\tilde{m}_2$, and the variances, $\tilde{s}_1^2$ and $\tilde{s}_2^2$, can be calculated. By doing so, the separation of the data between the two classes can be assessed by (1), as shown in Figure 1. In Figure 1(a), a representation of the data provided by the bidimensional feature set has smaller separation with respect to the variances than in Figure 1(b), i.e. $FD(\nu)$ is greater in Figure 1(b), where the same data is represented by different features (x_1, x_3), with better classification probability.

In this section, three techniques are presented as alternatives to feature extraction applied to the TSA problem: Sequential Search, Genetic Algorithm and Principal Components Analysis.

2.1 Sequential search

This is a very simple feature selection procedure that produces subsets of input features by evaluating each one separately using the FD and ranking them in descending order. The first such d input features are taken as the d-best out of a total number of n features. Despite of being very fast and simple to implement, this technique is suboptimal and the features thus selected are not guaranteed to be the best choice for the NN classifier. The d best features are not necessarily the best d features, that is, although the features may be good individually according to the chosen criterion, they may not necessarily form the best combination. This is because the joint effect of features on the discrimination capability is not taken into

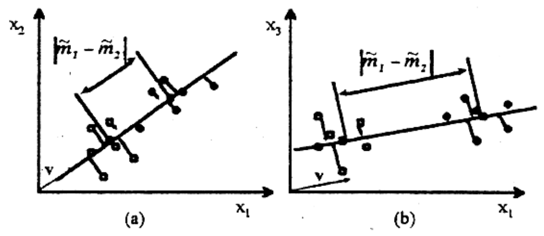

Figure 1 Data Separation as Assessed by Fisher Discriminant

account, i.e. the features are assumed independent.

2.2 Genetic algorithm

When the combination of features is considered, and the dimension of the original input vector, $x \in R^n$, is high, the feature selection becomes computationally demanding. The GA technique can be used to solve this problem. It is an optimization procedure to search for the best possible subset in the features vector [15].

Mimicking the natural evolution concept, the GA uses a bit string representation for each individual chromosome of a population of candidate solutions [17,18]. An individual here is a feature vector. A bit *string* of zeros and ones represents a candidate feature vector. Bits '1' represent which components of the input vector, $x \in R^n$, are selected to be part of the feature vector, and bits '0' represent which ones are eliminated. With the FD as evaluation function the number of desired features d has to be chosen beforehand, because its monotonicity property determines that larger sets of features will always have better evaluations than smaller ones. As a result, the final solution would always be the complete feature vector, without reduction. Therefore, in order to evaluate a given feature vector (selected from the original input features vector), $g \in R^k$, $k \le n$, the following fitness function is used:

$$fitness(g) = FD(g, v) + \alpha(k) \qquad (2)$$

$FD(g, v)$ is the same as in (1) and $\alpha(k)$ is a non-negative scale factor that represents a praise for candidate solutions with the desired number of features, d ($\alpha = 0$, for $k \neq d$).

2.3 Principal Component Analysis

The feature extraction performed by PCA uses the training set $T = \{x_i, y_i\}_{i=1}^N$ to find an n by n covariance matrix [19]. The input vectors, x, are linearly transformed to the features vectors, f, as

$$f = Q\,x \qquad (3)$$

where the columns of Q are the eigenvectors of the covariance matrix associated with the largest eigenvalues. The feature vector, in the sense of the minimum square error, optimally approximates the original input vector, thus minimizing the loss of information.

When PCA is used for the classification problem, the mapping transforms the original input space into a lower dimensional space based on the directions of the greatest variances. This is optimal in terms of representation when it is desired to have the lowest error when reconstructing the original data from the reduced space. However, for classification purposes, the most interesting mapping is the one for which the difference in the class means is large relative to the standard deviations, not the ones for which the standard deviations are large [16].

3. SUPPORT VECTOR MACHINES CLASSIFICATION

SVMs are nonlinear models based on theoretical results from the Statistical Learning Theory [10]. This theory formally generalizes the empirical risk minimization principle that is usually applied for NN training when the classifier is determined by minimizing the number of training errors. In NN training, a number of heuristics is traditionally used in order to avoid overfitting and to estimate a NN classifier with adequate complexity for the problem at hand.

An SVM classifier minimizes the generalization error by optimizing the relation between the number of training errors and the so-called Vapnik-Chervonenkis (VC) dimension. This is a new concept of complexity measure that can be used for different types of functions.

A formal theoretical bound exists for the generalization ability of an SVM, which depends on the number of training errors (t), the size of the training set (l), the VC dimension associated to the resulting classifier (h), and a chosen confidence measure for the bound itself (η) [20]:

$$R < \frac{t}{l} + \sqrt{\frac{h(ln(2l/h) + 1) - ln(\eta/4)}{l}} \qquad (4)$$

The risk R represents the classification error expectation over all the population of input/output pairs, even though the population is only partially known. This Risk is a measure of the actual generalization error and does not require prior knowledge of the probability distribution of the data. Statistical Learning Theory derives inequality (4) to mean that the generalization ability of an SVM is measured by an upper limit of the actual error given by the right hand side of (4), and this upper limit is valid with probability $1 - \eta$ ($0 < \eta < 1$). As h increases, the first summand of the upper bound (4) decreases and the second summand increases, so that there is a balanced compromise between the two terms (complexity and training error).

The SVMs used for two-class problems are based on linear hyperplanes to separate the data, as shown in Figure 2. The hyperplane is determined by an orthogonal vector w and a bias b, which identify the points that satisfy $w \cdot x + b = 0$. By finding a hyperplane that maximizes the margin of separation, ρ, it is intuitively expected that the classifier will have a better generalization ability (Figure 2). The hyperplane with the largest margin on the training set can be completely determined by points that are closest to the hyperplane. Two of such points are x_1 and x_2 in Figure 2(b), and they are called Support Vectors (SVs) because the hyperplane (i.e. the classifier) depends entirely on them.

Therefore, in their simplest form, SVMs learn linear

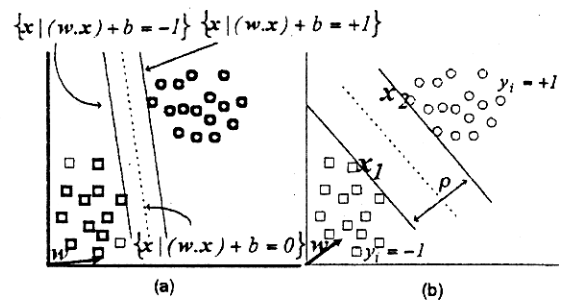

Figure 2 Maximum Margin Classifier

decision rules

$$f(x) = sign(w^t x + b) \tag{5}$$

so that (w, b) are determined as to correctly classify the training examples and maximize ρ.

For linearly separable data, as shown in Figure 2, a linear classifier can be found such that the first summand of bound (4) is zero. It is always possible to scale w and b so that

$$w^t x + b = \pm 1 \tag{6}$$

for the SVs, with

$$w^t x + b > +1 \text{ and } w^t x + b < -1 \tag{7}$$

for non-SVs. Using the SVs x_1 and x_2 of Figure 2 and Equation (6), the margin can be calculated as

$$\rho = \frac{w^t}{\|w\|}(x_2 - x_1) = \frac{2}{\|w\|} \tag{8}$$

For linearly separable data the VC dimension of SVM classifiers can be assessed by [10]

$$h < min\left\{n, \frac{4D^2}{\rho^2}\right\} + 1 = min\,\{n, D^2\ \|w\|^2\} + 1 \tag{9}$$

where n is the dimension of the training vectors and D is the minimum radius of a ball which contains the training points. Therefore the risk (4) can be decreased by decreasing the complexity of the SVM, that is, by increasing the margin of separation ρ, which is equivalent to decreasing $\|w\|$.

As practical problems are not likely to be separable by a linear classifier, the linear SVM can be extended to a nonlinear version by mapping the training data to an expanded feature space with a nonlinear transformation:

$$\Phi(x) = (\phi_1(x), \ldots, \phi_m(x)) \in R^m \tag{10}$$

where $m > n$. Then, the maximum margin classifier of the data on the new space can be determined. With this procedure, the data which are non-separable in the original space may become separable in the expanded feature space. The next step is to estimate the SVM by minimizing

$$V(w) = 1/2\ w^t w \tag{11.1}$$

subject to the condition that all training patterns are correctly classified, that is,

$$y_i(w^t\Phi(x_i) + b) \geq 1, \quad i = 1,\ldots, l \tag{11.2}$$

However, depending on the type of nonlinear mapping (10), the training points may happen to be not linearly separable, even in the feature space. That means, it will be impossible to find an SVM classifier that fulfills all the conditions (11.2). Therefore, instead of solving (11), a new cost function is used to minimize (4) [10]:

$$V(w, \varepsilon) = \frac{1}{2} w^t w + C \sum_{i=1}^{l} \varepsilon_i \tag{12}$$

where l *slack* variables ε_i are introduced to allow for training errors, that is, training patterns for which $y_i(w^t\Phi(x_i) + b) \geq 1 - \varepsilon_i$ and $\varepsilon_i > 1$. By minimizing the first summand of (12), the complexity of the SVM is decreased and by minimizing the second summand of (12), the number of training errors is decreased. C is a positive penalty constant that must be chosen to act as a tradeoff between the two terms.

The minimization of the cost function (12) leads to the SVM training as a quadratic optimization problem with unique solution. In practice, the nonlinear mapping (10) is indirectly obtained by the so called Mercer Kernel Functions, which correspond to inner products of data vectors in the feature space, $K(a, b) = \Phi(a)\ \Phi(b), a, b \in R^n$ [14]. In order for this equivalence to be valid, a Kernel function must satisfy some requirements called Mercer Conditions. These conditions have limited the number of Kernel Functions applied in practice so far, and the most commonly used are the Gaussian RBF Kernel

$$K(a, b) = e^{-\frac{\|a-b\|^2}{\sigma^2}} \tag{13}$$

and the Polynomial Kernel

$$K(a, b) = (a^t b + 1)^p \tag{14}$$

where the parameters σ and p in (13) and (14) must be preset. Details on the solution of (11) and the final SVM architecture are shown in the Appendix.

In summary, some nonlinear mapping (10) can be indirectly defined by a Kernel Function (i.e. there is no need for specifying (10)), for example (13) or (14). The parameters σ and p affect how sparse and easily separable the data are in feature space, and consequently, affect the complexity of the resulting SVM classifier and the number of training errors. The parameter C also affects the model complexity. Currently, there are no clues on how to set C, how to choose the best Kernel Function (the nonlinear map Φ) and how to set the Kernel parameters. In practice, a range of values has to be tried for C and for the Kernel parameters, and then the performance of the SVM classifier is assessed on each of these values.

An example of the calculation of the bounds (4) and (9) is shown in Figure 3, where training and test sets obtained from transient stability simulations are used. An SVM with Gaussian RBF Kernel and a very large value of C has been used so that it could achieve zero training error. For each

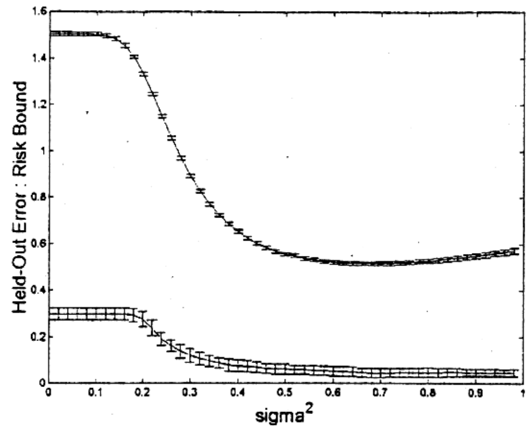

Figure 3 SVM Risk Bound Evaluation

value of σ^2 varying from 0 to 1, the training set has been randomly split in 10 subsets of the same size. One subset has been held-out for testing and the others have been used for SVM training. The procedure is repeated for each subset so that 10 independent values of classification error could be obtained. The mean and the standard deviation of these test errors have been calculated and shown as points of the lower curve in Figure 3. The upper curve is the risk bound (4) as a function of (9). In this example, it can be seen that although the risk bound have loose values, it predicts well the behavior of the SVM classification error, in relative terms, since the curves are strongly correlated.

In practice, inequality (9), as an estimation of h, is only valid when the data are separable (when no training errors are obtained). For values of C and of the Kernel parameters that result in an SVM with training errors, the estimation of a reliable risk bound (4) becomes practically impossible.

However, by following the principles outlined in this section, minimizing (12) and using an independent test set to evaluate the SVM model, it is expected that a classifier with good generalization and fast training time can be found. With all these aspects in mind, a better idea about advantages and disadvantages of SVMs can be acquired from their application on the TSA problem.

4. TRANSIENT STABILITY ANALYSIS TESTS AND RESULTS

This section explains how the feature extraction techniques, connected with MLP and SVM NNs' training, are actually applied to obtain power system transient stability evaluations.

The IEEE 50-Generator system has been used [21] to generate training and test examples. Different operating conditions have been created by changing the generation and load patterns of the system. Each case has been validated by a load flow execution. For each operating condition, the same contingency has been simulated in the time domain using the ETMSP software [22] and the corresponding critical clearing time (CCT) has been determined. The complete input features set is composed of the active and reactive powers of each generator and the total active and reactive loads of the system at the moment of the fault, with a total of 102 inputs and 1 output indicating the security class.

The feature extraction techniques presented in Section 2 have been run on the training set to reduce the input space dimension. Besides the complete set of features, reduced sets have been obtained with d = 50, 30, 20 and 10.

Multi-Layer Perceptrons have been trained with the Levenberg-Marquardt backpropagation training algorithm to give security estimations based on binary outputs corresponding to the stable/unstable classes. The classification of the system as stable/unstable is determined based on a given security threshold, which represents the realized clearing time of the contingency. If a given sample output, i.e. the simulated CCT, is above the threshold, the input state is considered stable, otherwise it is unstable.

SVMs have also been trained on the examples with binary outputs to indicate the stable and unstable classes. A Gaussian RBF Kernel has been used and the values of σ^2 and C have been sought in a heuristic manner. The software SVM^{light} has been used [13].

Figure 4 presents Receiver Operating Characteristic (ROC) curves of the SVM performance on the test, set after it has been trained with the complete set of 102 input features. The false dismissal rate on the x-axis is the ratio of test points that have been incorrectly classified as stable. The detection rate on the y-axis is the ratio of stable test points that have been correctly classified. Several SVMs have been trained with increasing values of σ^2 and the corresponding values of the detection and false dismissal rates are shown in Figure 4. These are two conflicting values that increase together for a fixed value of C and increasing σ^2. Values of C = 0.1, 1, 10, 100, 1000 and 10000 have been tried. The solid line in Figure 4 corresponds to the SVM with C = 1. The dashed line corresponds to the SVM with C = 10, 100, 1000 and 10000, which show the same results. The curve for the SVM with C = 0.1 is not shown in Figure 4 due to the difference in scales. It presents much higher values of false dismissal rates, going as far as 0.08 and lower values of detection rate than the curves shown in Figure 4. In a cross-validation procedure, ROC curves like these can be drawn for performances of held-out test errors, and values of C and σ^2 can be identified according to a desired value of false dismissal rate to choose the best SVM, which is something critical in the TSA application. For any classifier it is desirable that the detection rate is maximized and the false dismissal

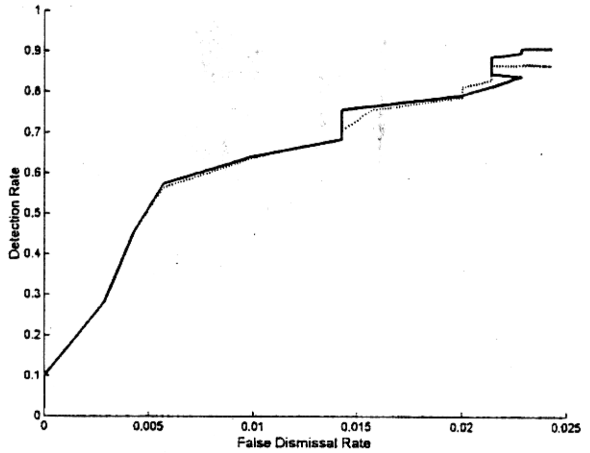

Figure 4 SVM ROC Curve

rate is minimized for complete or reduced features sets.

An MLP has also been trained on the complete set of 102 features, but an ROC curve like the one of Figure 4 cannot be drawn to show the performance on the test set. The number of factors affecting its performance is large and interelated.

The SVM training time with 1400 training examples ranged from 27 seconds to 63 seconds in a Pentium 233 MHz, depending on σ^2. The value of C has not affected the training time significantly. The MLP training time with 1400 training examples and 5 neurons in the hidden layer was 40min42sec.

Next, SVM and MLP classifiers have been trained on reduced features sets. For the MLP, a cross-validation training has been performed until the test error started increasing or stopped decreasing.

The results of transient stability classifications using both models of NNs are shown in Table 1. It shows specifications of the input features, the detection rate, false dismissal rate, error rate (ratio of test points which have been incorrectly classified), training time and number of SVs. For SVMs trained with different features sets, ROC curves have been drawn for different values of C. Then, for the values of C that resulted in the best ROC curves, fixed values of false dismissal rates have been set: 0.01, 0.02 and 0.03. The corresponding values of σ^2 resulted in the models SVM-0.01, SVM-0.02 and SVM-0.03, shown in Table 1. These three models have shown the best results with $C = 1$. SVM-102 is the SVM classifier with 102 inputs and the lowest error rate. For the MLPs, MLP-102 is the model with 102 inputs, MLP-1 is the model with the lowest error rate and MLP-2 is the model with the smallest training time.

Table 1 shows that the adequacy of feature extraction techniques depends on the classifier, as expected. SVM-0.01 has performed better with the 50 features obtained by the d-best technique. SVM-0.02 and SVM-0.03 performed better with the 30 best features. MLP-1 has had the lowest error rate with the 50 best features, and MLP-2 has had the smallest training time with 10 features. Among the three feature extraction techniques, the GA has resulted in the 10 feaures with the best performance.

In this application, MLPs and SVMs have had the best performances with $d = 30$ and $d = 50$. With this reduction rate, the sequential search feature extraction performs better than the others. It is very difficult for genetic algorithms to explore the large number of features combinations when the reduction is from 102 features to 30 or 50. However, when the reduction is from 102 to 10 or 20, the number of feature combinations is smaller so that the GA feature extraction provides better performance classifiers. For larger power systems, the final choice will depend on the trade-off between classification accuracy and training time.

MLP-102 and SVM-102 have almost the same performance, but the second model's training time is extraordinarily lower. MLP-1, with 50 input features, has slightly better performance than SVM-0.03, with 30 features, but the second model's training time is also much lower. It could be noticed that the MLP training time has dramatically increased with the number of input features and the number of hidden neurons.

When σ^2 approaches the value of best generalization for the SVM classifier, the number of SVs decreases. It could be noticed that the SVM training time depends on the number of input features, on the value of σ^2 and on the number of SVs of the resulting classifier. From Table 1, SVM-0.03 is the SVM model with the lowest test classification error, lowest training time and lowest number of SVs. SVM-0.02 and SVM-0.03 have the same number of inputs but different training times because different values of σ^2 have been used.

5. CONCLUSIONS

This paper shows that the SVMs are a new NN model that fits the TSA application. It provides a different strategy to tackle the curse of dimensionality, regarding computational effort, because of very low training times compared to MLPs.

In this application, the reduction on the number of features from 102 to 30 and 50 resulted in classifiers with the best performances. The accompanying reduction in the sparsity of the data has turned the training process into an easier task. On the other hand, larger training sets could be used for SVMs to improve the performance, while the training time would not be considerably increased.

The SVM model allows a good understanding of its theoretical details, as shown in Section 3, and this can be used to identify the important parameters for the classifier. In particular, improved search strategies for σ^2 and C (in the case of RBF Kernel), and different types of Kernel functions should be tried. Future research will focus on aditional feature extraction techniques for MLP and SVM NNs, which consider nonlinear relations included in the data, feature extraction techniques customized for SVMs, and will explore different Kernel functions for SVM training, applied to TSA of large power systems.

APPENDIX

The computation of the decision boundary of an SVM, $f(x) = \text{sign}(w^t\Phi(x) + b)$, for the non-separable case consists in solving the following optimization problem [10]:

Table 1 IEEE 50-Generator System Transient Stability Classifications

	SVM-102	SVM-0.01	SVM-0.02	SVM-0.03	MLP-102	MLP-1	MLP-2
Input Features	102	50-BEST	30-BEST	30-BEST	102	50-BEST	10-GA
Detection Rate	0.92	0.74	0.85	0.95	0.91	0.95	0.92
False Dismissal Rate	0.026	0.01	0.02	0.03	0.021	0.023	0.024
Error Rate	0.047	0.08	0.06	0.043	0.047	0.036	0.046
Training Time	37 seconds	57 seconds	34 seconds	12 seconds	40min42sec	11min24sec	1min42sec
No. of SVs	433	1309	1065	298	–	–	–

$$\begin{aligned} &\textit{minimize: } V(w, \varepsilon) = \frac{1}{2} w^t w + C \sum_{i=1}^{l} \varepsilon_i \\ &\textit{subject to: } y_i(w^t\Phi(x_i) + b) \geq 1 - \varepsilon_i, \quad i = 1, \ldots, l \qquad (15) \\ &\qquad \varepsilon_i > 0, \quad i = 1, \ldots, l \end{aligned}$$

Instead of solving (15) directly, it is much easier to solve the dual problem (16), in terms of the Lagrange multipliers, α_i [10]:

$$\begin{aligned} \textit{minimize: } W(\alpha) &= -\sum_{i=1}^{l} \alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j \Phi(x_i)^t\Phi(x_j) \\ &= -\sum_{i=1}^{l} \alpha_i + \frac{1}{2}\sum_{i=1}^{l}\sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j K(x_i, x_j) \\ \textit{subject to: } &\sum_{i=1}^{l} y_i\alpha_i = 0 \text{ and } 0 \leq \alpha_i \leq C, \quad i = 1, \ldots, l \quad (16) \end{aligned}$$

which is a quadratic optimization problem. From the solution, α_i, $i = 1, \ldots, l$ of (16) the decision rule $f(x)$ can be computed as [10]

$$\begin{aligned} f(x) = w^t\Phi(x) + b &= \sum_{i=1}^{l} \alpha_i y_i \Phi(x_i)^t\Phi(x) + b \\ &= \sum_{i=1}^{l} \alpha_i y_i K(x_i, x) + b \end{aligned} \qquad (17)$$

The training points with $\alpha_i > 0$ are the SVs, and (17) depends entirely on them. The threshold b can be calculated using (6), which is valid for any SV:

$$b = y_{SV} - w^t\Phi(x_{SV}) \qquad (18)$$

An SVM can be represented as in Figure 5, where the number of units $K(x, x_i)$ is determined by the number of SVs.

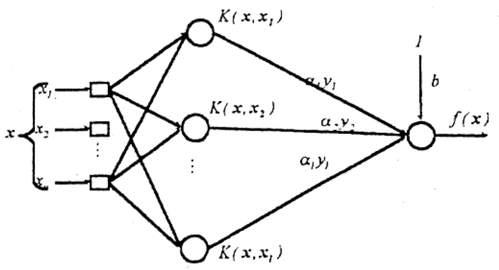

Figure 5 SVM NN Architecture

ACKNOWLEDGEMENTS

This work has been supported by the Brazilian National Research Council (CNPq).

REFERENCES

1 **Kundur, P** *Power System Stability and Control*, McGraw-Hill (1994)

2 **Mansour, Y** Competition and System Stability – The Reward and the Penalty, *Proceedings of the IEEE*, Vol. 882, (Feb. 2000), 228–234

3 **Sobajic, D J and Pao, Y-H** Artificial neural-net based dynamic security assessment for electric power systems, *IEEE Trans. On Power Systems*, Vol. 4, No. 1 (February 1989) 220–228

4 **Kumar, A B R, Ipakchi, A, Brandwajn, V, El-Sharkawi, M and Cauley, G** Neural networks for dynamic security assessment of large-scale power systems: requirements overview. *Proceedings of the First International Forum on Applications of Neural Networks to Power Systems* (1991) 65–71

5 **Mansour, Y, Chang, A Y, Tamby, J, Vaahedi, E and El-Sharkawi, M A** Large scale dynamic security screening and ranking using neural networks, *IEEE Trans. On Power Systems*, Vol. 12, No. 2 (May 1997) 954–960

6 **Mansour, Y, Vaahedi, E and El-Sharkawi, M A** Dynamic security contingency screening and ranking using neural networks, *IEEE Transactions on Neural Networks*, Vol. 8, No. 4 (July 1997) 942–950

7 **Park, Y M, Kim, G-W, Choi, H-S and Lee, K Y** A new algorithm for kohonen layer learning with application to power system stability analysis, *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. 27, No. 6, (December 1997), 1030–1033

8 **Demaree, K, Athay, T, Cheung, K W, Mansour, Y, Vaahedi, E, Chang, A Y, Corns, B R and Garrett, B W** An on-line dynamic security analysis system implementation, *IEEE Transactions on Power Systems*, Vol. 9, No. 4 (November 1994) 1716–1722

9 **Moulin, L S, El-Sharkawi, M A, Marks II, R J and Alves da Silva, A P** Automatic feature extraction for neural network based power systems dynamic security evaluation. *Proc. Of the IEEE ISAP2001 International Conference on Intelligent Systems Applied to Power Systems*, Budapest, Hungary (2001) Paper No. 21

10 **Vapnik, V** *Statistical Learning Theory*, John Wiley & Sons, New York (1998)

11 **Gavoyiannis, A E, Vogiatzis, D G and Hatziargyriou, N D** Dynamic security classification using support vector machines. *Proceedings of the IEEE ISAP2001 International Conference on Intelligent Systems Applied to Power Systems*, Hungary, Budapest (2001) Paper No. 22

12 **Osuna, E, Freund, R and Girosi, F** Training support vector machines: an application to face detection. *Proceedings of CVPR'97*, Puerto Rico (1997)

13 **Joachims, T** *Making Large-Scale SVM Learning Practical, in Advances in Kernel Methods – Support Vector Learning*, Scholkopf, B, Burges, J C C and Smola, A J (editors), MIT Press, Cambridge, USA (1998)

14 **Cristianini N and Shawe-Taylor J** *An Introduction to Support Vector Machines and Other Kernel-Based Learning Methods*, Cambridge University Press (2000)

15 **Brill, F Z, Brown, D E and Martin, W N** Fast genetic selection of features for neural network classifiers, *IEEE Transactions On Neural Networks*, Vol. 3, No. 2 (1992) 324–328

16 **Duda, R O and Hart, P E** *Pattern Classification and Scene Analysis*, John Wiley and Sons (1973)

17 **Holland, J H** *Adaptation in Natural and Artificial Systems*, University of Michigan Press (1975)

18 **Goldberg, D E** *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley (1989)

19 **Haykin, S** *Neural Networks – A Comprehensive Foundation. 2nd Edition*, Prentice Hall (1998)

20 **Vapnik, V** *The Nature of Statistical Learning Theory*, Springer-Verlag, New York (1995)

21 **Vittal, V (Chairman)** Transient stability test systems for direct stability methods, *IEEE Transactions on Power Systems*, Vol. 7, No. 1 (1992) 37–43

22 ETMSP – Extended Transient Midterm Stability Program User's Manual, Vol. 1-6, Version 3.1, EPRI TR-102004-V2R1, Final Report (1994)