Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2018) 26: 2661 – 2675

doi:10.3906/elk-1805-151

Multilabel learning for the online transient stability assessment of electric power systems

Peyman BEYRANVAND[1], **Veysel Murat İstemihan GENÇ**[2,*], **Zehra ÇATALTEPE**[1]
[1]Department of Computer Engineering, Faculty of Computer and Informatics Engineering,
İstanbul Technical University, İstanbul, Turkey
[2]Department of Electrical Engineering, Faculty of Electrical and Electronics Engineering,
İstanbul Technical University, İstanbul, Turkey

Received: 22.05.2018 • **Accepted/Published Online:** 29.07.2018 • **Final Version:** 28.09.2018

Abstract: Dynamic security assessment of a large power system operating over a wide range of conditions requires an intensive computation for evaluating the system's transient stability against a large number of contingencies. In this study, we investigate the application of multilabel learning for improving training and prediction time, along with the prediction accuracy, of neural networks for online transient stability assessment of power systems. We introduce a new multilabel learning method, which uses a contingency clustering step to learn similar contingencies together in the same multilabel multilayer perceptron. Experimental results on two different power systems demonstrate improved accuracy, as well as significant reduction in both training and testing time.

Key words: Dynamic security assessment, transient stability assessment, multilabel learning, neural networks

1. Introduction

Dynamic security assessment (DSA) of large interconnected power systems is a challenging task, often requiring an intensive computation power for both off-line and online studies. Besides being important in various steps of power system planning, a fast, accurate, and extensive DSA of possible operating points has become vital for planning operations as well as real-time operations. DSA also plays a decisive role in real-time security assessment, especially when power systems are operated under stressed conditions.

Maintaining an acceptable dynamic performance (transient stability) associated with rotor angle dynamics under credible contingencies is one of the defining criteria for the dynamic security of power systems. However, a reliable transient stability assessment (TSA) of an interconnected power system operating over a wide range of conditions against a large number of contingencies could usually be complex or computationally burdensome.

A number of different approaches have been proposed for the online TSA of power systems [1, 2]. Numerical integration algorithms for solving the differential equations associated with power system dynamics are mostly used for off-line studies and are regarded as computationally intensive for online applications [3]. Alternatively, direct methods [4] using Lyapunov functions can be applied, but they become inefficient when complex models need to be adopted. Pattern recognition methods, such as artificial neural networks (ANNs) and decision trees, can also be used to predict the security status of the system, as they establish a mapping between

*Correspondence: gencis@itu.edu.tr

the operating points (OPs) and the security of the system operating at them [5]. Most of the computation required for pattern recognition methods is done off-line for generating a proper dataset, and then training a model for online predictions. In this study, we focus on the application of a multilayer perceptron (MLP) model, a specific subclass of ANNs, which enables fast online prediction of the security status of a large power system. This approach can effectively be used to predict the security of the system either operating at a single OP for the assessment of the current status, or a large number of possible OPs for an assessment to be used in developing preventive control actions.

Contrary to the traditional single-label learning paradigm, in multilabel learning problem setup, each input instance is associated with multiple output variables [6]. Therefore, at the test time, the task is to predict all these output variables from one input instance. An approach based on multiway decision tree was used to assess power system operation security for multiple contingencies in [7]. Power system security assessment with multiclass classification was performed using multiclass support vector machine in [8].

Depending on the time frame and operating condition(s) of the system within the time frame, a large number of contingencies can be critical and therefore worth being monitored, as well as taken into consideration for the actions for maintaining the security of the system. The security assessment of the system for a large number of contingencies could also become inevitable due to the fact that any possible control action to be taken for a set of contingencies could make some other credible contingencies critical. Therefore, whether the system is considered as operating at a single OP or projected to be operating over a range of OPs within the time frame of interest, it is important to be able to assess the system's security for a batch of multiple contingencies.

In large power systems, dedicating one MLP for each credible contingency [9–12] imposes two difficulties: the size of the dataset required to properly train all the MLPs and the required time to train multiple MLPs. We propose to incorporate multilabel learning formulation into the design of the MLP-based TSA to address these difficulties. To the best of our knowledge, there is no prior work on implementation of multilabel MLPs (MLMLP) for TSA of power systems.

In the TSA problem of interest, each OP characterizes a prefault state of the power system, and the dynamic security of the system operating at the OP is to be predicted for a number of contingencies. Especially if contingencies are close to each other in the network, their security might be correlated. This fact can be utilized to learn the security assessment for multiple contingencies at the same time and hence reduce the number of MLPs required for prediction. In MLMLP [6], one MLP can be formulated such that it concurrently learns the mapping between OPs and arbitrary number of outputs.

Success of multilabel learning is affected by the degree of correlation between the labels [13]. In order to investigate the effect of label correlations on MLMLP, we first clustered contingencies using various clustering methods. We investigated the effect of using an MLMLP for each cluster instead of one MLMLP for all the labels. We measured the effect of clustering on training and prediction time, along with prediction accuracy. By employing MLMLPs and cluster of contingencies, we managed to improve the accuracy of prediction, without increasing the size of the dataset. In addition, we reduced the training and prediction time considerably.

To summarize, the novel contributions of this paper are as follows: (a) the use of multilabel MLPs for the TSA problem, (b) comparison of both training and testing time and accuracy performance of MLMLP to an MLP for each label, (c) instead of a single MLMLP for all labels or a single MLP for each label, using multiple MLMLPs on adaptively determined clusters of correlated labels.

The rest of the paper is organized as follows: In Section 2, we review the multilabel learning and clustering methods and introduce the notation used in the paper. Section 3 introduces the method we developed for the

use of MLMLPs for the TSA problem. Section 4 contains the experimental methodology and results, and the conclusions are given in Section 5.

2. Background

2.1. Multilabel learning

Let $X = \mathbb{R}^d$ denote the d-dimensional space of OPs. For the single-label learning problem setup, each input instance $x_i \in X$ is associated with a single output $y_i \in Y$, where $Y = \mathbb{R}(or\mathbb{Z})$. The dataset containing N instances is defined as $\{(x_i, y_i)|1 \leq i \leq N\}$. In the multilabel problem setup the output space is defined as $(Y^1, ..., Y^K) \in \mathbb{R}^K$, where K is the number of contingencies for which we predict the security state. An N instance dataset for the multilabel problem is defined as $\{(x_i, (y_i^1, ..., y_i^K))|1 \leq i \leq N\}$. Clearly, the single-label learning problem is a subclass of multilabel learning, where $K = 1$.

Multilabel formulation allows exploitation of the correlation (or statistical dependency) of output variables $(y_1^1, ..., y_1^K)$ to facilitate the learning process [13]. This property has been noted and utilized by researchers in domains such as web mining, information retrieval, text categorization, and bioinformatics [13].

Multilabel learning approaches can be categorized into problem transformation methods and algorithm adaptation methods [13]. The former category of approaches transforms the multilabel learning problem into simpler learning algorithms, transforming it into binary relevance task [14], binary classification task [15], label ranking task [16], and multiclass classification task. The second category of approaches that we employed in this study adapt the popular learning algorithms for the multilabel task. Multilabel KNN [17], multilabel decision trees [18], and rank-SVM [19] are examples of such approaches. In this study, we adopted the MLMLP algorithms proposed in [6] for predicting power system security.

An MLP is a general function estimator with the form $\hat{y} = f(W, x)$, where x is the vector of inputs, $\hat{y}$ is the prediction that the MLP produces for input instances, and W represents the parameters of the MLP that should be adapted to the dataset to minimize the discrepancy between y and $\hat{y}$. We used a single-hidden-layer MLP estimator in this study, so the parameters of the MLP function can be shown as $W = (W^{Input}, W^{Hidden})$, where W^{Input} is the matrix of weights connecting the input layer and hidden layer, and W^{Hidden} is the matrix of weights connecting the hidden layer and output layer. Figure 1 shows the diagram of an MLMLP. Note that when the output layer predicts only one of the y^i's, the MLP and the learning problem will have a single label.

The critical step in learning multiple labels concurrently using an MLMLP is to correctly formulate a loss function to include the prediction error of multiple labels. In training MLPs, the weights of the network, W, are modified by back-propagation method [20] so that the loss function is minimized. Therefore, when the loss function includes errors of multiple labels, the weights of the MLMLP will adapt to minimize the prediction error of all the labels. We propose to use a loss function of the form,

$$E = \sum_{l=1}^{K} E_k, \tag{1}$$

where E_k is the prediction error for y^k. E_k can be defined as cross-entropy error for classification task and square error for regression task. Since we are dealing with the security classification task, we use cross-entropy error,

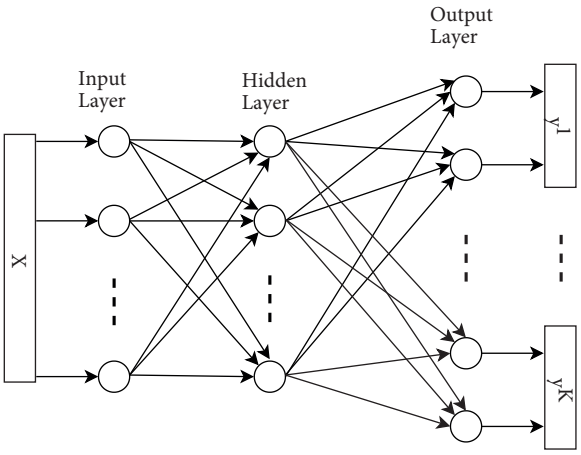

Figure 1. Multilabel MLP architecture.

$$E_k = -\frac{1}{N}\sum_{i=1}^{N} \hat{y}_i^k \ln y_i^k + (1 - y_i^k) \ln(1 - \hat{y}_i^k). \tag{2}$$

We expect the information gathered from all outputs in the loss function to help with the generalization of the learned mapping on unseen OPs at the test time, because having multiple related output variables to predict may help to reveal the underlying structure of the data and the input-output relationship faster. This can be investigated by comparing the correlations between labels as well as the prediction errors of contingencies of a single-label MLP with the errors of the approaches using MLMLP.

In addition, when one MLP is used for each contingency, the input to hidden unit weights W^{Input} will need to be learned for each label separately. On the other hand, in the MLMLP setting, especially if labels are correlated, parts of the learned W^{Input} could be shared between labels, reducing the training and testing time of the MLMLP.

2.2. Clustering

The improvement of prediction accuracy when using a single MLMLP for all contingencies, instead of one MLP per contingency, raises the question of whether there is an intermediate number of MLMLPs, more than one and less than the number of contingencies, which yields the best prediction accuracy. To the best of our knowledge, there is no prior investigation of this hypothesis. Therefore, here we employed clustering methods to form subsets of contingencies, and then we trained an MLMLP for each subset. In other words, we determined the most similar contingencies based on a similarity measure, and used them as outputs of one MLMLP. We experimented with different similarity measures and investigated the quality of prediction using the resulting clusters.

Clustering algorithms aim to form coherent groups of input patterns. In this study, we used agglomerative hierarchical clustering [21] method to form subsets of similar contingencies. Hierarchical clustering groups data points that are most similar to each other in an incremental manner. Initially, we assign each contingency to a separate cluster, thus, we have as many clusters as the number of labels. Then, incrementally, each pair of clusters that are most similar are merged into a single cluster. If the cluster has more than one contingency, the average similarity (average linkage) between corresponding two cluster members is used as the similarity

measure [22]. The merge operation continues until a stopping criterion, such as a certain number of clusters, or a level of average within cluster similarity measure is met. Here, we stop when the desired number of clusters have been formed.

The similarity measure used impacts the quality of the clusters obtained. Here, we utilized two widely used similarity measures: symmetric uncertainty (SU) and Pearson correlation coefficient (PCC). Pearson correlation coefficient is a measure of the strength of linear correlation of two variables [23]. Let $C^k = (y_1^k, ..., y_N^k)$ represent the column vector of all observed values of contingency k in the dataset. Let $y^k = \frac{1}{N}\sum_{i=1}^{N} y_i^k$ denote the average of all observed values for contingency k. For two different contingencies k_1 and k_2, and a dataset of size N, the PCC between k_1 and k_2 is defined as

$$PCC(C^{k_1}, C^{k_2}) = \frac{\sum_{i=1}^{N}(y_i^{k_1} - \bar{y}^{k_1})(y_i^{k_2} - \bar{y}^{k_2})}{\sqrt{\sum_{i=1}^{N}(y_i^{k_1} - \bar{y}^{k_1})^2}\sqrt{\sum_{i=1}^{N}(y_i^{k_2} - \bar{y}^{k_2})^2}}. \tag{3}$$

PCC takes values in the range $[-1, 1]$. Although it can capture the linear relationships between variables, it cannot capture the nonlinear dependencies between them.

Unlike PCC, symmetric uncertainty can capture the nonlinear correlations between two variables [24] and therefore could be a more useful measure of similarity when nonlinear dependencies exist in the dataset. Symmetric uncertainty between two contingencies k_1 and k_2 is defined as

$$SU(C^{k_1}, C^{k_2}) = 2\frac{MI(C^{k_1}, C^{k_2})}{H(C^{k_1}) + H(C^{k_2})}, \tag{4}$$

where mutual information, MI, is defined as $MI = H(C^{k_1}) + H(C^{k_2}) - H(C^{k_1}, C^{k_2})$, and $H(.)$ is the entropy function [24]. The use of marginal and joint probability distributions of variables, as opposed to just the divergence from the mean as in the case of the PCC, allows MI to capture nonlinear relationships between variables. In Eq. (4), normalization by the entropy of both variables allows the comparison of similarities between different pairs of contingencies.

3. Methodology

The method of multilabel learning for online TSA, which is the main contribution of this paper, can be suggested as an integral part of an inclusive methodology covering the tasks of both security assessment and enhancement for transient stability, see Figure 2. In this methodology, the power system is monitored through the measurements and a state estimator to characterize the current steady state of the system evolving in time, t. The security of the current OP is to be predicted by means of a number of neural networks (ANNs in general, and MLPs and MLMLPs in particular). When the time for periodical update of the ANNs is reached, e.g., $T = 1$ hour, or if an unexpected change, e.g., tripping of a critical line or generator occurs in the system, the whole process of building or updating the TSA tool is performed.

In order to perform ANN training and testing, a dataset covering all the OPs that are realizable in the next period of time, T, needs to be created. A contingency scan together with a series of time-domain simulations are performed to identify the critical contingencies and the security of the system operating at the OPs against these contingencies. Thus, a dataset is formed to train and test the ANNs. Followed by the generation of the dataset, the procedure of building or updating the ANNs, which are multilabel multilayer

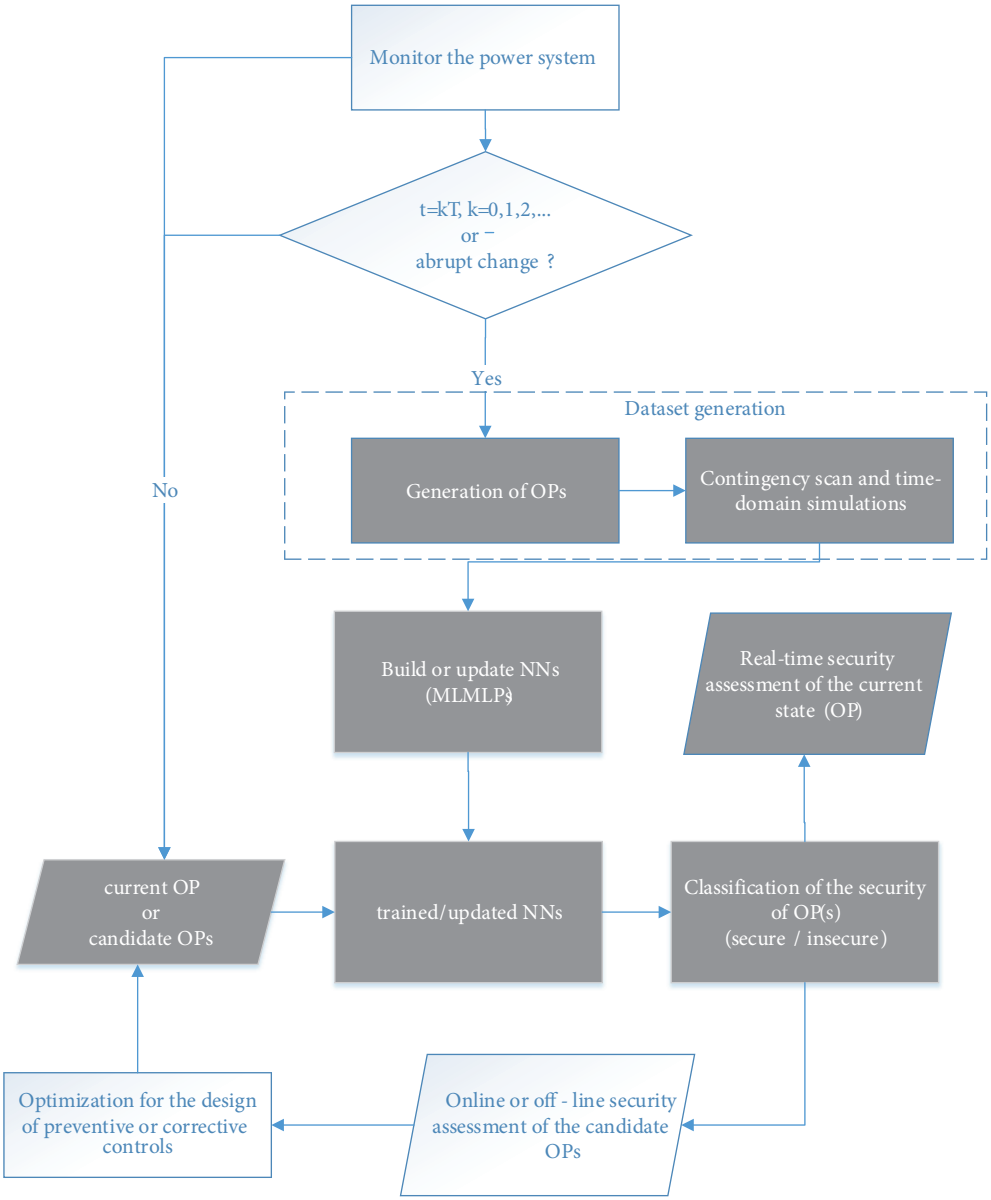

Figure 2. A general framework for TSA and enhancement, including Multilabel MLPs (processes studied in this paper are shaded).

perceptrons, is performed. The ANNs are updated periodically if the previously built ones perform poorly for the recently generated dataset, or if some other new contingencies become critical.

Once the neural networks are built, they can be used to predict the security of an operating point against any critical contingency. The operating point at which the system's security is predicted can either represent the current state of the system or a candidate solution of an optimization algorithm. Thus, the classification of the security of the current OP can be done for a real-time TSA, while the same ANNs can also be used for predicting the security of the candidate solutions of the optimization algorithms in which a fast online or off-line TSA for a large number of OPs is required.

The steps of the experiment pipeline include generating the dataset, clustering the contingencies into a predefined number of clusters, l_{max}, training the predictor, and finally evaluating the prediction performance. Each step is described in the following subsections.

3.1. TSA dataset generation

An important step in data-driven TSA is to collect enough instances in a dataset so that the general behavior of the power system is properly represented. The generated dataset should include the whole space of OPs at which the system could be operated in the time frame of interest for TSA. Considering the framework in Figure 2, if the security assessment of the continuously changing OP is to be performed, then the dataset could involve only the projected conditions that are shaped by optimal power flows. If, however, the security assessment of the candidate OPs that are produced by some optimization algorithms for preventive or corrective controls is required, then the dataset must also cover all other feasible OPs that are not optimal.

In this study, for a given power system, we created a large set of steady-state OPs, using power flow solutions, for different settings of generation schedule and distributions of load demand. Thus, the generated dataset covers all the realizable OPs that can be suggested by a preventive control, such as generation rescheduling or load curtailment. A contingency scan was performed to identify the critical contingencies resulting in transient instabilities when the system is operated over the range of OPs considered in the dataset. Time-domain simulations were carried out under credible contingencies to distinguish the critical ones, against which the system is insecure while it is operated at any of the OPs, and to determine the security of the system operating at all OPs against any of the critical contingencies.

The problem of TSA can be formulated as either a regression or a classification problem. Here, we opt for the latter formulation. In the classification task, each OP included in the dataset is associated with the system's security against all of the critical contingencies.

The security of the system operating at an OP against a critical contingency is translated into a *{secure,insecure}* set as the following angle stability index η is calculated,

$$\eta = \frac{360 - \delta_{max}}{360 + \delta_{max}} \times 100, \tag{5}$$

where δ_{max} is the maximum postcontingency separation angle, in degrees, of any two generators. We label the corresponding OP as secure if and only if η is positive and as insecure otherwise. Although in this paper we work on a classification problem, we also computed and stored the critical clearing time values in order to use them for label similarity computation, as explained in Section 3.2.

Each operating point i can be represented by an input vector x_i which contains the steady state values from the power system. On the other hand, the output vector y_i represents the contingencies whose security status is calculated.

3.2. Contingency (label) clustering

We used agglomerative clustering to group similar contingencies. We used three different methods for cluster assignment.The first cluster assignment was done by using Pearson correlation coefficient (PCC) as the similarity measure. As mentioned before, the dataset contains the continuous values for contingencies, which we map to *{secure,insecure}* set. However, for calculating the PCC values, we used the continuous critical clearing time values.

The second clustering assignment was performed using the symmetric uncertainty (SU) similarity. We calculated the similarity of contingencies using symmetric uncertainty and performed cluster assignment for each contingency and cluster size.

The final cluster assignment was done randomly. We randomly assigned cluster labels to each instance of dataset such that the number of contingencies in each cluster stays roughly the same. The random cluster assignment serves the purpose of a control experiment. By comparing the final results of different clustering assignment with random assignment, we can isolate accidental improvements and make sure that the performance improvements are, in fact, the result of proper cluster assignment.

The result of the clustering step is a cluster label for each contingency. We define each cluster as $b_l = \{y^k | k \leq K\}$ for $l \leq l_{max}$. Note that any contingency can only be a member of a single cluster. In other words, b_l's are mutually exclusive, $\bigcup_{l=1}^{l_{max}} b_l = (Y^1, ..., Y^K)$ and $\bigcap_{l=1}^{l_{max}} b_l = \emptyset$.

Our purpose was to compare the performance of multilabel learning with that of the single-label learning. First, we chose l_{max} equal to the number of contingencies, where each cluster contains one contingency, and created the single-label learning setup. Then, we decreased l_{max} until $l_{max} = 1$, which corresponds to multilabel learning on all the labels.

3.3. Training the MLMLP

The general architecture of the MLMLP that we implemented in our experiments is presented in Figure 1. We performed experiments for different values of l_{max}. Concretely, at each step of experiment, we trained l_{max} for different neural networks and for each of them the set of output variables to predict is given by b_l. If b_l contains one contingency, the neural network is a simple MLP, and for b_l containing more than one contingency, it is an MLMLP. Note that all of the neural networks, either MLP or MLMLP, use the whole set of N instances in the dataset for training, predicting only the subset of contingencies that is specified by b_l.

We limited the structure of the neural network to have only one hidden layer and used the loss specified in Eq. (1) augmented with regularization to train the network. Using regularization is a common practice to avoid overfitting in neural networks. The resulting loss function can be expressed as

$$loss = E + \lambda ||W||_2^2, \tag{6}$$

where λ is the regularization parameter. For training the neural network, we applied Nesterov's accelerated gradient method [25] to adapt weights of the neural network. The procedure of training includes calculating the prediction of the network output $\hat{y}$ for all instances, calculating the prediction loss using Eq. (6), calculating the gradient loss with respect to weights W by back-propagation, and finally updating the weights using the calculated gradient and Nesterov's accelerated gradient method.

4. Simulations and results

4.1. Test systems

In order to demonstrate the effectiveness of the proposed methodology, two different test systems operating at stressed conditions leading to a large number of critical contingencies are chosen: (a) the U.S. Northeastern and Ontario 16-generator-68-bus system [26], (b) the WSCC 37–generator-127-bus system [27], see Figures 3 and 4.

For the two test systems, first, we started with generating the datasets covering a range of 1600 operating points at which the systems could be operated, as the feasible power (real and reactive) injections by the

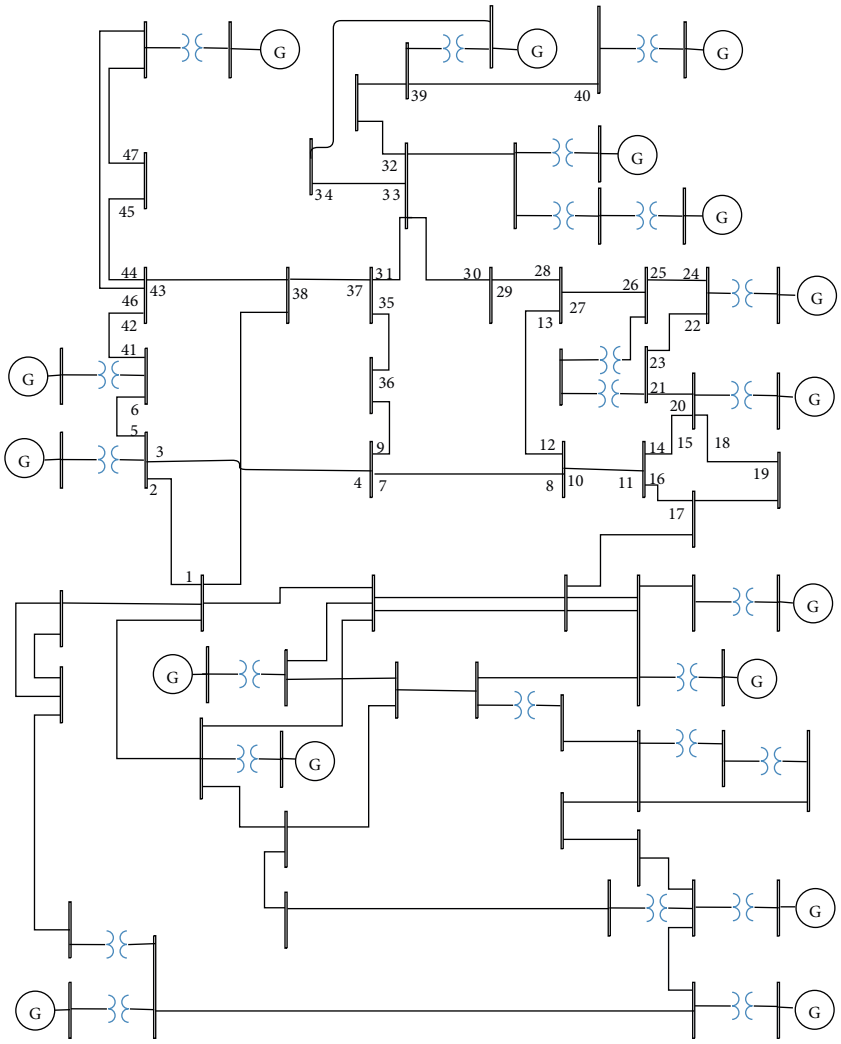

Figure 3. U.S. Northeastern and Ontario 16-generator-68-bus system.

generators and loads are randomly changed and their respective power flow solutions are obtained. A contingency scan was also performed for each system operating over the range of OPs considered. The credible contingencies include all the possible three-phase bus faults cleared by tripping of a connected line or a transformer in the network. Among them, we distinguished the critical contingencies resulting in unstable rotor angle deviations when the systems are operated over the range of OPs projected. For the 68-bus test system, 48 different contingencies were found critical, whereas 58 contingencies were found critical for the 127-bus system.

Thus, the datasets that are composed of the OPs and the systems' security against each of the critical contingency were generated for the two test systems. They were obtained by first solving the power flow solution for each OP, and then performing time-domain simulation for each contingency to determine the security of the system using a power systems simulation software, DSATools [28]. The instances in the dataset cover a range of OPs, each of which is represented by the prefault bus voltages (magnitudes and angles), power (real and reactive) injections, and power (real and reactive) flows through the branches of the network.

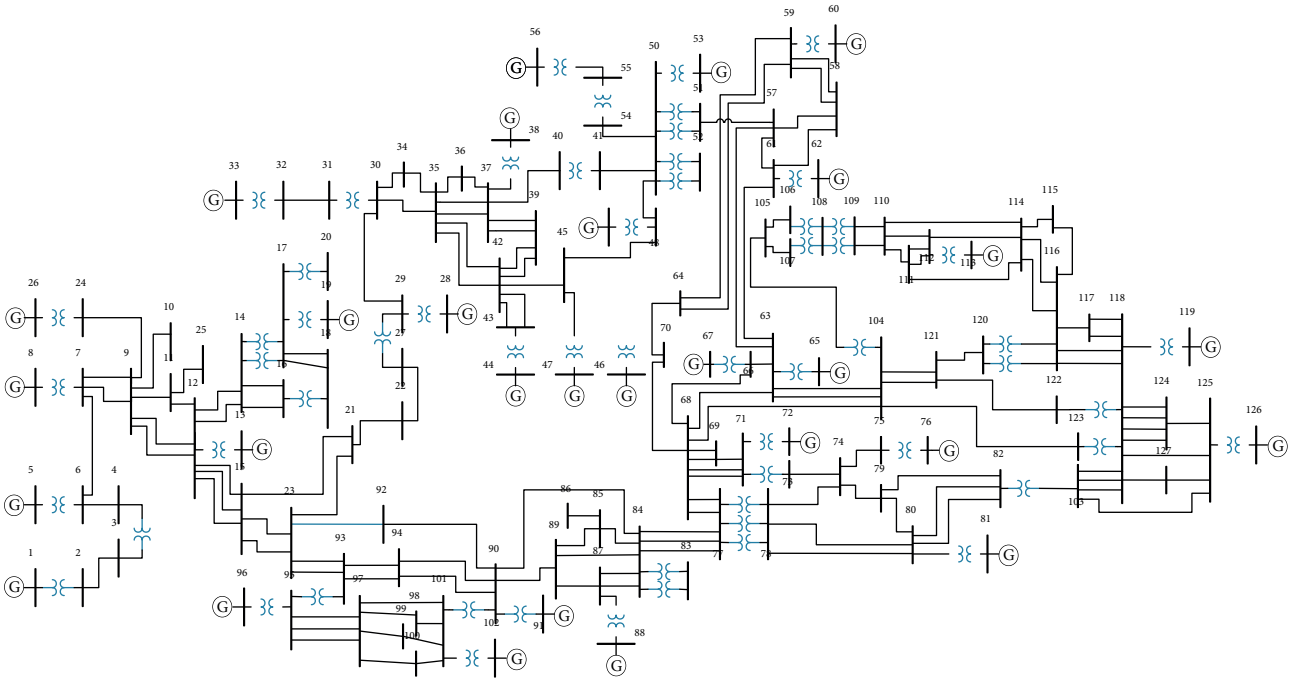

Figure 4. WSCC 37–generator-127-bus system.

4.2. Training and evaluation

The datasets we generated, Dataset-A including 4800 instances and Dataset-B including 1600 instances, are based on the 68-bus system and the 127-bus system, respectively. 20 percent of the instances was reserved for final performance evaluation and was used in any step of training. We used the remaining 80 percent of the data for training and 5-fold cross-validation. In order to avoid overfitting while training the neural network, we used early stopping on cross-validation error. We repeated the experiments for 1, 5, 10, 20, 30, and 40 clusters/MLPs.

We considered three main performance measures to compare multilabel and single-label learning paradigms, i.e., prediction error, training time, and testing time. Training/testing error can be calculated as $\frac{t_c}{N_c}$, where t_c is the number of incorrectly predicted instances and N_c is the number of all instances being considered. In a multilabel setting, the error for each label is computed separately, and then the average is calculated and reported as the training/test error. Training and testing time, on the other hand, are recorded directly.

In order to set hyperparameters, regularization parameter λ and the number of hidden units of the neural network, we used grid search and 5-fold cross-validation. We randomly split the training samples into 5 folds of the same size. Afterwards, we used 4 subsets to train the neural network, either MLMLP or MLP, on different values of λ and number of hidden units. We repeated the same process for each fold and finally chose the combination of the λ and number of hidden units that yielded the lowest average error across all 5 folds. The final reported performance of the predictor was computed by using the whole training set for training the neural network, with the chosen hyperparameters, and measuring the prediction on the hold-out test set.

Figure 5 shows the test results for both datasets. In these figures, the dashed line shows the average error achieved by training only one MLMLP for predicting all contingencies concurrently. The solid line shows

the average error when training an MLP for each contingency. The horizontal axis shows the number of clusters/MLPs used.

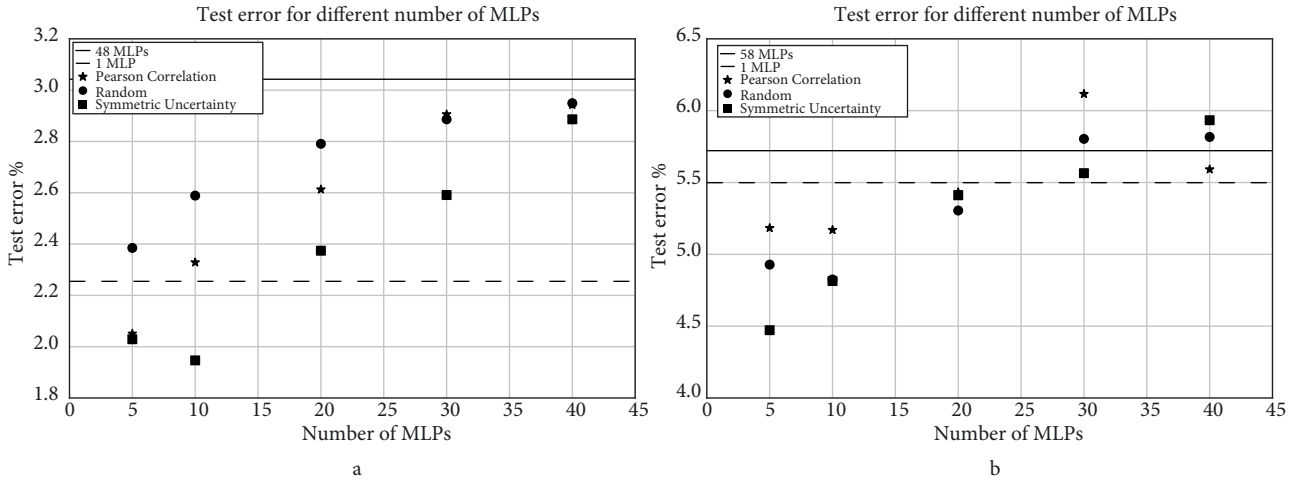

Figure 5. Test errors for Dataset-A(left) and Dataset-B(right).

The first observation for both datasets is that, using one MLMLP for prediction, the average error has reduced across all contingencies. The advantage of the MLMLP approach has been more significant for Dataset-A.

In Figure 5, the test errors are reduced in general, as smaller number of clusters (and hence more labels together in each MLMLP) are used. Clustering contingencies and then using MLMLPs for these clusters results in smaller test error than using a single MLMLP for both datasets. Using 5 or 10 MLMLPs for Dataset-A and using 5 to 20 MLMLPs for Dataset-B is better than using a single MLMLP, or as many MLPs as the number of contingencies. In both cases, as long as the number of clusters is small enough, even random clustering of contingencies results in better performance than MLPs. Among the clustering methods, SU is better than PCC for small number of clusters. Based on the average test error over all contingencies, we can conclude that any sort of MLMLP, as long as the number of contingency clusters is within a certain range, will result in an improved test error.

Figures 6 and 7 show the individual contingencies' error analysis for random and SU-based clustering for Datasets A and B. In both cases of clustering, the best error observed for each contingency tends to appear in smaller number of clusters, and mostly, as expected, on the exact number of clusters that yields the best average error for the dataset. When the best cluster numbers for random and SU-based contingency clustering are considered, in general, SU-based clustering of contingencies results in smaller test error for smaller number of clusters. SU can identify nonlinear dependencies and bring similar contingencies together, hence they can learn from each other's labels when they are included in the same MLMLP.

The training and testing time is one of the most important factors in online TSA. In addition to prediction error improvement, using multilabel learning also improves the time for training and testing. Figure 8 compares test and training time for multilabel and single-label learning cases for Dataset-A. Each point on the figure shows one of the 5 experiments performed on the dataset. By decreasing the number of clusters, training time was reduced by 9 and 8 folds for Datasets A and B, respectively. Similarly, the testing time was also reduced by approximately 30 times.

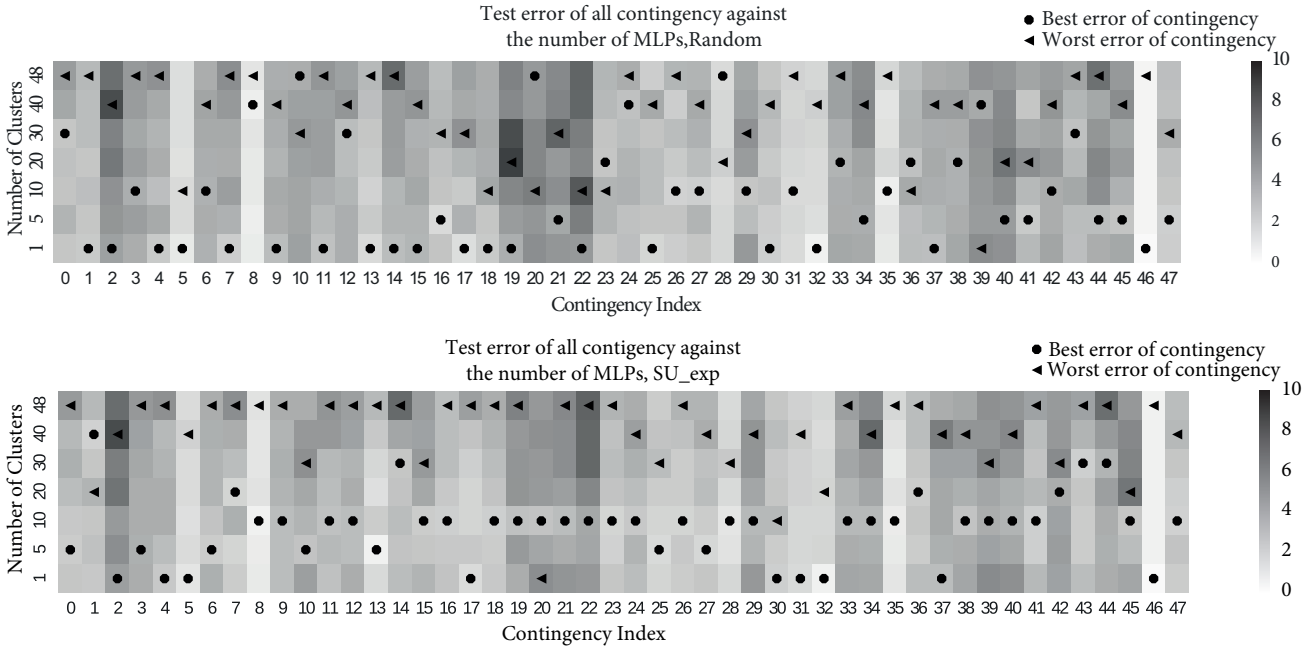

Figure 6. The number of the best (circle) and the worst (triangle) clusters for Dataset-A.

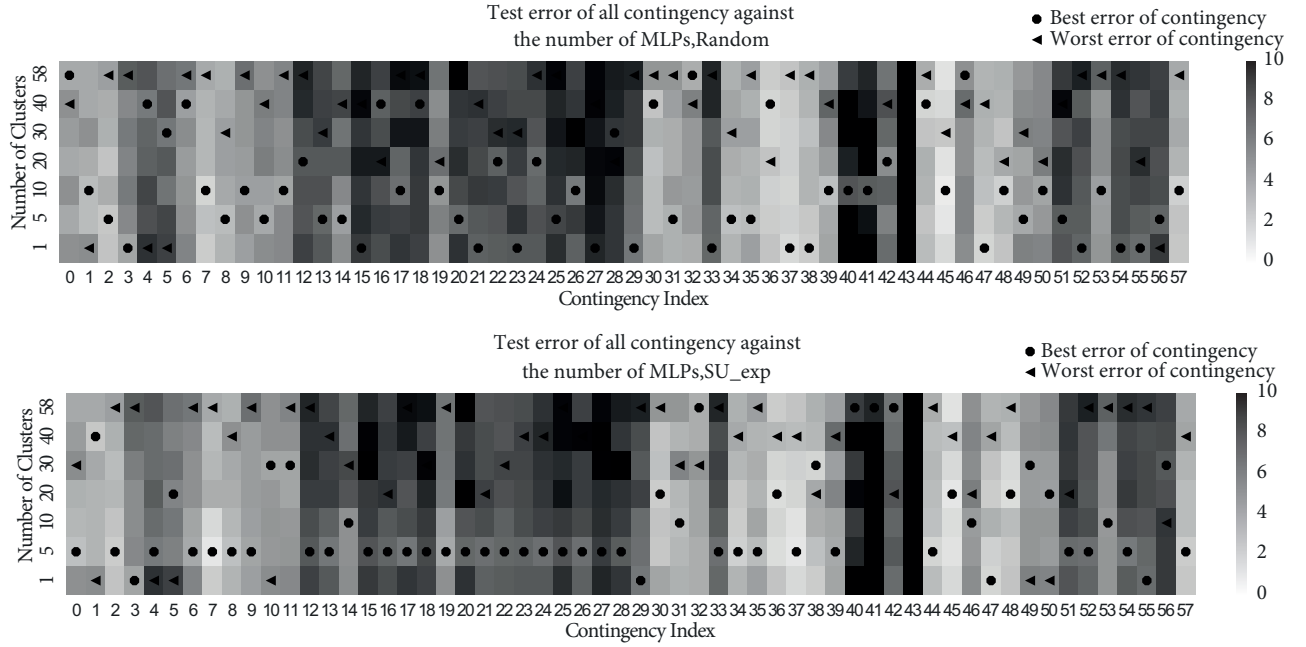

Figure 7. The number of the best (circle) and the worst (triangle) clusters for Dataset-B.

The results of our experiments showed that incorporating multilabel learning paradigm for online TSA leads to both improvement of prediction error and reduction of the time needed for training and testing. Our experiments also showed that grouping contingencies by means of contingency clustering, and then training an MLMLP on each contingency cluster results in even better test errors than using one MLMLP for all contingencies.

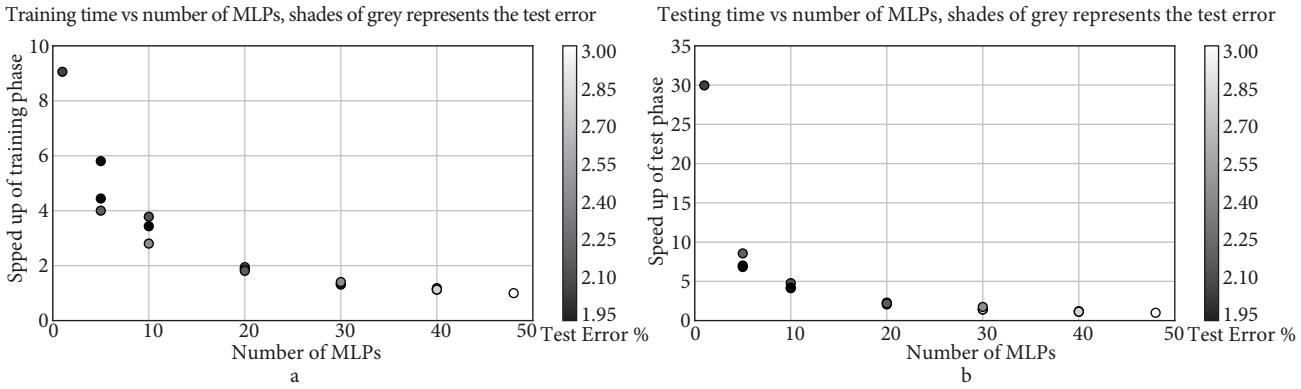

Figure 8. Training and testing time for Dataset-A.

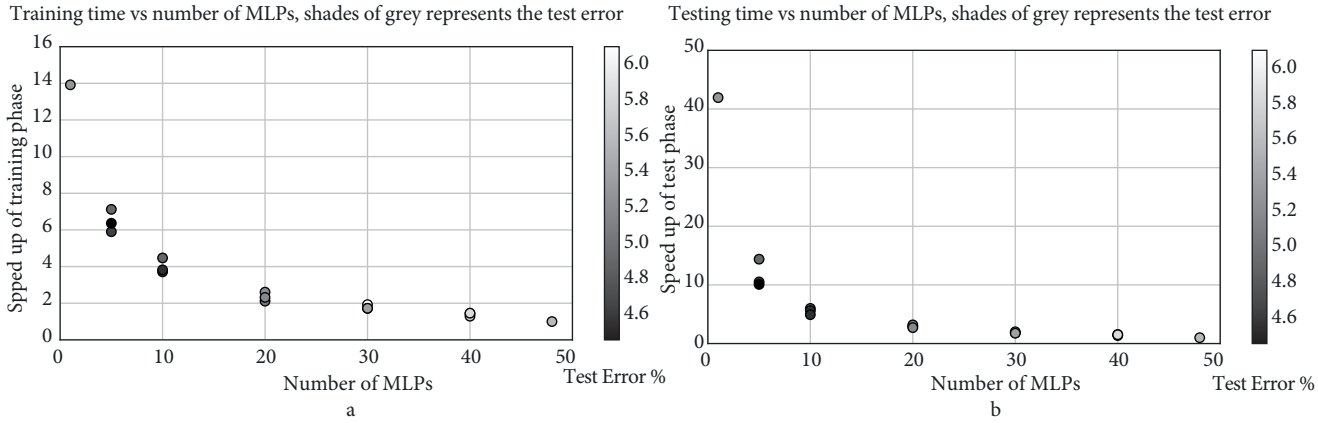

Figure 9. Training and testing time for Dataset-B.

5. Conclusion and future work

For the online TSA using pattern recognition methods, both prediction success and the time needed to train and test the predictors can become limiting factors, especially when the task gets complicated by a large number of contingencies. In this study, we developed a method that uses multilabel artificial neural networks (MLMLP) for predicting concurrently the system security against multiple contingencies and demonstrated its success in the TSA of two test systems. Using only one MLMLP for all contingencies, we achieved better performance in terms of test error and training and test time, compared to a set of single-label MLPs, each of which is assigned to only one contingency. We conclude that multilabel learning paradigm can leverage the underlying structure and dependencies of contingencies to simplify the learning process. This observation provides multitude of opportunities for online security assessment of power systems. We also performed additional analysis using contingency clustering to investigate the possibility of further performance improvement. Based on their similarity according to Pearson correlation coefficient and symmetric uncertainty, we grouped contingencies into clusters. We found out that instead of using a single MLMLP on all contingencies, using an MLMLP for contingencies in the same cluster, the test errors could be further decreased. Therefore, in order to achieve better performances in online or off-line TSA under multiple contingencies, the application of MLMLPs using clustered contingencies is promising. Further work could concentrate on the use of similarity measures other than symmetric uncertainty and Pearson correlation coefficient to compute the similarities between contingencies.

Using different contingency clustering methods and training MLMLPs on these clusters and then using classifier combination methods such as voting or stacking to decide on the final contingency label prediction is another research area. In our work, we have only used a single-layer MLMLP, whereas the use of multilabel deep neural networks [29] for DSA is another research direction.

Acknowledgment

This work was supported by The Scientific and Technological Research Council of Turkey (TÜBİTAK) grant number 114E157.

References

[1] Sauer PW, Tomsovic KL, Vittal V. Dynamic security assessment. In: Grigsby LL, editor. Power System Stability and Control. Boca Raton, FL, USA: CRC Press, 2007. pp. 1-10.

[2] Kezunovic M, Meliopoulos S, Venkatasubramanian V, Vittal V. Application of Time-Synchronized Measurements in Power System Transmission Networks. Cham, Switzerland: Springer, 2014.

[3] Kundur P, Balu NJ, Lauby MG. Power System Stability and Control. New York, NY, USA: McGraw-Hill, 1994.

[4] Fouad AA, Vittal V. Power System Transient Stability Analysis Using the Transient Energy Function Method. Englewood Cliffs, NJ, USA: Prentice-Hall, 1992.

[5] Wehenkel LA. Automatic Learning Techniques in Power Systems. Norwell, MA, USA: Kluwer Academic Publishers, 1998.

[6] Zhang ML, Zhou ZH. Multilabel neural networks with applications to functional genomics and text categorization. IEEE T Knowl Data En 2006; 18: 1338-1351.

[7] Oliveira WD, Vieira JP, Bezerra UH, Martins DA, Rodrigues BDG. Power system security assessment for multiple contingencies using multiway decision tree. Electr Pow Syst Res 2017; 148: 264-272.

[8] Kalyani S, Swarup KS. Classification and assessment of power system security using multiclass SVM. IEEE T Syst Man Cy C 2011; 41: 753-758.

[9] Voumvoulakis EM, Gavoyiannis AE, Hatziargyriou ND. Application of machine learning on power system dynamic security assessment. In: Intelligent Systems Applications to Power Systems (ISAP); 5–8 November 2007; Toki Messe, Niigata, Japan: IEEE. pp. 1-6.

[10] Hoballah A, Erlich I. Real-time operation of deregulated electricity market: an integrated approach to dynamic stability assurance. In: Energy Conference and Exhibition (EnergyCon); 18–22 December 2010; Manama, Bahrain: IEEE. pp. 262-267.

[11] Kucuktezcan CF, Genc VMI. A comparison between ANN based methods of critical clearing time estimation. In: International Conference on Electrical and Electronics Engineering (ELECO); 28–30 November 2013; Bursa, Turkey: IEEE. pp. 132-136.

[12] Swarup KS, Reddy KP. Neural network based pattern recognition for power system security assessment. In: International Conference on Intelligent Sensing and Information Processing; 4-7 January 2005; Chennai, India: IEEE. pp. 234-239.

[13] Zhang ML, Zhou ZH. A review on multi-label learning algorithms. IEEE T Knowl Data En 2014; 26: 1819-1837.

[14] Boutell MR, Luo J, Shen X, Brown CM. Learning multi-label scene classification. Pattern Recogn 2004; 37: 1757-1771.

[15] Read J, Pfahringer B, Holmes G, Frank E. Classifier chains for multi-label classification. Mach Learn 2011; 85: 333-359.

[16] Gaber MM, Zaslavsky A, Krishnaswamy S. A survey of classification methods in data streams. In: Aggarwal CC, editor. Data Streams. Boston, MA, USA: Springer, 2007. pp. 39-59.

[17] Zhang ML, Zhou ZH. ML-KNN: a lazy learning approach to multi-label learning. Pattern Recogn 2007; 40: 2038-2048.

[18] Clare A, King RD. Knowledge discovery in multi-label phenotype data. In: European Conference on Principles of Data Mining and Knowledge Discovery; 3–7 September 2001; Freiburg, Germany: Springer-Verlag. pp. 42-53.

[19] Elisseeff A, Weston J. A kernel method for multi-labelled classification. In: Conference on Neural Information Processing Systems (NIPS); 3–8 December 2001; Vancouver, British Columbia, Canada: MIT Press. pp. 681-687.

[20] Rumelhart DE, Hinton GE, Williams RJ. Learning representations by back-propagating errors. Nature 1986; 323: 533-536.

[21] Jain AK, Murty MN, Flynn PJ. Data clustering: a review. ACM Comput Surv 1999; 31: 264-323.

[22] Zambelli AE. A data-driven approach to estimating the number of clusters in hierarchical clustering. International Society for Computational Biology Community Journal 2016; 5: 1-13.

[23] Kojadinovic I. On the use of mutual information in data analysis: an overview. In: International Symposium on Applied Stochastic Models and Data Analysis (ASMDA); 17–20 May 2005; Brest, France: ENST Bretagne. pp. 738-747.

[24] Witten IH, Frank E, Hall MA, Pal CJ. Data Mining: Practical Machine Learning Tools and Techniques. Cambridge, MA, USA: Morgan Kaufmann, 2017.

[25] Sutskever I, Martens J, Dahl G, Hinton G. On the importance of initialization and momentum in deep learning. In: International Conference on Machine Learning; 16–21 June 2013; Atlanta, GA, USA: PMLR. pp. 1139-1147.

[26] Rogers G. Power System Oscillations. New York, NY, USA: Springer Science & Business Media, 2012.

[27] Fan D. Synchronized measurements and applications during power system dynamics. PhD, Virginia Tech, USA, 2008.

[28] DSATools^{TM}. Powertech Labs Inc., Surrey, British Columbia, Canada.

[29] Bengio Y. Learning deep architectures for AI. Foundations and Trends in Machine Learning 2009; 2: 1-127.