GNG-SVM Framework – Classifying Large Datasets with Support Vector Machines Using Growing Neural Gas

Ondrej Linda, Milos Manic, *Senior Member, IEEE*

***Abstract*—Support Vector Machines (SVMs) represent a well known technique for data classification. However, the complexity of the training process makes the SVMs unsuitable for classifying large datasets. Examples of existing approaches to this problem are sampling of the input datasets or clustering of similar inputs. On the other hand, the Growing Neural Gas algorithm (GNG) is a robust tool for cluster analysis, capable of learning the topology of the data. It overcomes most of the common issues of clustering techniques such as predefined number of clusters or beforehand specified cluster radius. This paper presents a solution to the problem of classifying large datasets via learning of the data topology. The described algorithm combines the GNG algorithm with the SVM solver into a specific algorithm for classification of large datasets – the GNG-SVM framework. The input dataset is first preprocessed with the GNG algorithm. A new reduced training dataset is created from the extracted topological knowledge. Because the size of the dataset is significantly reduced, the training process of the SVM solver becomes substantially less memory demanding. The performance of the proposed GNG-SVM framework is tested on both synthetic and benchmark real world datasets.**

***Index Terms*—Data Classification, Growing Neural Gas, Large Datasets, Support Vector Machines.**

I. Introduction

DATA classification and pattern recognition have been attracting significant attention for many years [1]-[3]. Support vector machines (SVM) have proven to be very successful data classification technique [4], [5]. The successful generalization over previously unseen instances is a consequence of maximizing the width of the separation margin between different classes. To deal with linearly non-separable problems the input space is transformed into a feature space by applying the kernel trick [5].

One of the deficiencies of the standard SVM solvers are the high memory requirements of the training process. This becomes a significant problem when classifying large datasets [6], [7]. The complexity of the training process is directly dependent on the size of the dataset. Hence, there is an obvious need for a dataset size reduction.

Traditionally, the problem is solved by various data sampling or clustering techniques. Sampling techniques select a representative subset of instances from the original dataset [8], [9]. Clustering techniques group similar instances together in order to eliminate redundant information or to use the prototypes from the created clusters to represent the original data [7], [10], [11].

This paper investigates the possibility of dataset size reduction via topology learning. The topological information is of a key importance for the SVM training. It describes the density as well as the shape of the data distribution in the input space. This knowledge is used by the SVM algorithm to construct an optimal separation function. While the size of the dataset is significantly reduced, the ability of the learning algorithm to correctly distinguish between various classes is preserved.

The introduced GNG-SVM framework uses the Growing Neural Gas (GNG) algorithm to significantly reduce the size of the input dataset [12]. The GNG is a prototype based vector quantization technique, which is capable of learning the data topology in arbitrary number of dimensions. The topology of each class is learnt by the GNG algorithm. This topological knowledge is then extracted from the network into a new reduced training dataset. This new reduced dataset is an approximation to the original data distribution. Clearly, some information is lost during the reduction process, but this is a tradeoff for the significant reduction of the problem complexity. Further, the GNG algorithm is inherently robust against noise in the training data and can improve the classification performance in such cases. Consequently, the SVM solver is trained on the new reduced dataset. The performance of the proposed GNG-SVM framework is tested on a synthetically generated linearly separable and non-separable datasets, as well as on benchmark real world datasets from the UCI's Machine Learning Repository [13].

The rest of the paper is organized as follows. Section II gives a brief overview of the SVM as well as the GNG algorithm. In section III, the GNG-SVM framework is presented. Section IV discusses the observed experimental results and a final conclusion is drawn in section V.

II. Background Overview

This section presents an overview of the Support Vector Machine and the Growing Neural Gas algorithms.

A. *Support Vector Machines Overview*

The Support Vector Machines (SVMs) are common machine learning technique performing well in classification as well as in regression analyses problems [14], [15]. Its main advantage is the solid mathematical and statistical background. SVM is based on the structural risk minimization principle [4]. The constructed separation boundary maximizes the margin between the data points from different classes. In the machine learning theory this means maximizing the capability of the classifier to generalize over the input data.

Consider a set S of N vectors $\vec{x}_i$ in m-dimensional space belonging to two disjoint classes denoted by class label l_i:

$$S = \left\{ (\vec{x}_i, l_i) \,\middle|\, \vec{x}_i \in R^m, l_i \in \{-1, 1\}, i = 1, 2...N \right\} \tag{1}$$

In case of linearly separable problem, there exists a unique separation hyper plane that maximizes the width of the separation margin. This hyper plane could be described by an m-dimensional weight vector $\vec{w}$, and a scalar bias b as:

$$\vec{w}^T \vec{x} + b = 0 \tag{2}$$

The solution to this problem is obtained by constructing a Langrangian and transforming it into a dual form as follows:

$$\max W(\vec{\alpha}) = -\sum_{i=1}^{N} \alpha_i + \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} y_i y_j \alpha_i \alpha_j \left(\vec{x}_i \vec{x}_j \right) \tag{3}$$

subject to:

$$\sum_{i=1}^{N} y_i \alpha_i = 0 \text{ and } 0 \le \alpha_i \le C, \; i = 1, 2, ...N \tag{4}$$

Here $\alpha = (\alpha_1, ..., \alpha_N)$ specifies a vector of non-negative Lagrange multipliers.

Input vectors $\vec{x}_i$ with non-zero coefficients α_i are called support vectors (SVs). The decision hyperplane is determined exclusively by the set of support vectors. In other words, the rest of the points are irrelevant for the final solution.

For instance, the α_i coefficients can be calculated by the quadratic programming technique or by the Sequential Minimal Optimization (SMO) algorithm [16], [17]. The decision surface defined by (2) is determined as:

$$\vec{w} = \sum_{\vec{x}_i \in SV_s} \alpha_i y_i \vec{x}_i \tag{5}$$

$$b = \frac{1}{N_{SV_s}} \sum_{\vec{x}_i \in SV_s} b_i \tag{6}$$

Here SV_S is the set of support vectors in dataset S, and N_{SV_S} is the number of these support vectors.

Several approaches for training the SVM on large datasets have been developed. The training task can be reduced into a set of sub-problems, which are solved separately. Chunking, decomposition or the SMO are only some of the examples of this approach [17]-[19]. Furthermore, the size of the dataset can be reduced by removing irrelevant instances. Selective sampling, active learning, or random sampling, attempt to maximize the degree of learning by trying to learn as much as possible with as small number of input instances as possible [8], [9]. Various clustering approaches can be used to either select relevant data instances close to the separation boundary or to create prototypes of data points [7], [10], [11]. On-line SVMs and incremental SVMs were proposed to deal with the case of dynamic non-stationary data [20], [21]. The model is efficiently maintained and updated as new data are being supplied.

B. *Growing Neural Gas Overview*

Inspired by the original neural gas algorithm by Martinetz and Schulten [22], Fritzke proposed the Growing Neural Gas [12]. It was developed for clustering and vector quantization. The GNG is capable of overcoming some of the limitations of standard self-organizing maps. The algorithm combines the growth mechanism of the growing cell structures (GCS) with the topology learning ability of the competitive Hebbian learning [23]. There is no fixed dimensionality of the GNG dynamic structure. Instead the GNG algorithm is able to adapt itself to the local dimensionality and density of the input data. The GCS based update strategy makes it robust against noise and overfitting.

The GNG network consists of a continually updated set of neurons. Each neuron n_i has associated a reference vector $\vec{w}_i \in R^N$, where N denotes the dimensionality of the problem. The reference vector determines the position of the corresponding neuron in the input space. Moreover, every neuron has a set of undirected edges connecting it with neighboring neurons. These edges have no weights and are only used to determine the topological neighborhood. Thus neighbors are not determined by the Euclidean distance but rather by a direct connectivity through these edges.

Every iteration an input signal ξ is drawn from the input probability distribution $P(\xi)$. Using the Euclidean distance, the two nearest neurons for the input signal are determined. If not yet existing, a new edge is created between them. The local error counter $error_a$ of the nearest neuron n_a is updated according to the following rule:

$$error_a = error_a + \left(\vec{w}_a - \xi\right)^2 \quad (7)$$

The reference neighborhood vectors of neuron n_a and its direct topological neighbors are updated as follows:

$$\vec{w}_a = \vec{w}_a + e_b\left(\xi - \vec{w}_a\right) \quad (8)$$

$$\vec{w}_j = \vec{w}_j + e_n\left(\xi - \vec{w}_j\right), \; \forall n_j \in \Gamma\left(n_a\right) \quad (9)$$

$$e_b, e_n \in \left[0, 1\right], \quad e_b > e_n \quad (10)$$

Here e_b and e_n are the coefficients for updating weights and symbol Γ denotes the set of neurons directly connected through an edge to the given neuron.

The algorithm starts with two randomly initialized neurons. Every λ iterations, a new neuron is added between the neuron with the highest accumulated local error and its neighbor with the highest local error. Therefore, new neurons are inserted in areas that do not approximate the training data well.

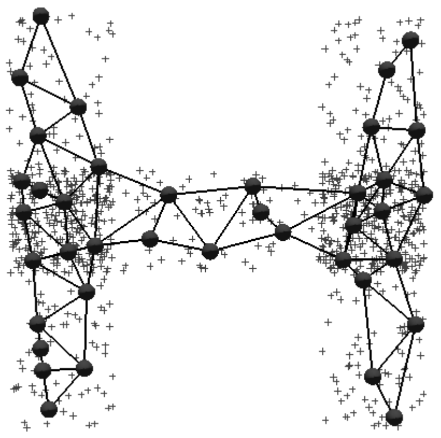


Fig. 1. The results of approximating 2D distribution with the GNG structure.

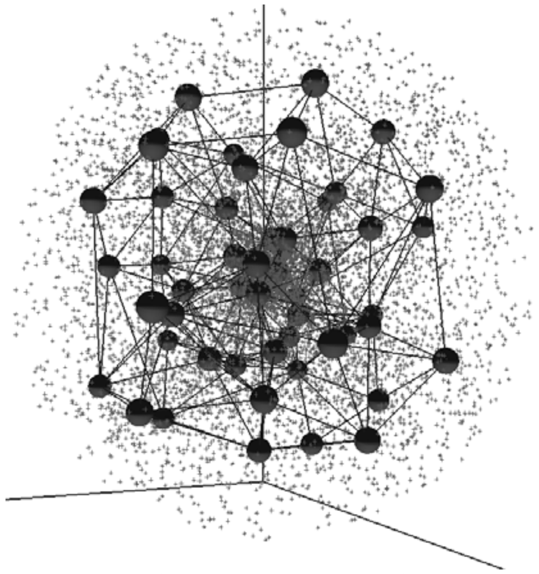


Fig. 2. Visualization of the topology learning of a 3D spherical data distribution. The sphere, composed of 10.000 points, is approximated by a GNG structure with 50 neurons.

In order to maintain only the valid edges, an aging mechanism is implemented. Every time a new edge is created its age is set to 0. When the winning neuron n_a is located, the age of all its edges is increased by 1. Every iteration, the age of each edge is compared to the established maximum age a_{max}. All edges older then a_{max} are removed as well as any neurons having no edges.

The GNG algorithm is superior to other clustering techniques in several ways. It does not require an upfront specification of the number of clusters as it is the case with the K-Means or the C-Means algorithms [3], [24]. The network continuously grows inside the training data until a specified convergence criterion is reached. Also no cluster radius has to be fixed like in the case of the nearest-neighbor clustering or the subtractive clustering [3], [11]. The pattern is assigned to the nearest neuron; hence the radius is continuously adjusted for each neuron as the size of the network grows. The iterative nature of the GNG algorithm makes it suitable for the task of learning the topology of large datasets.

Fig 1 and Fig. 2 show examples of the topology learned by the GNG algorithm.

III. GNG-SVM Framework

The presented GNG-SVM framework constitutes a hybrid data classification algorithm specifically developed for classifying large datasets. The binary decision problem with two disjoint classes is described by (1). Generally, a machine learning problem can have an arbitrary number of classes. Hence, (1) can be generalized for a learning problem with C classes as follows:

$$S = \left\{ \left(\vec{x}_i, l_i\right) \middle| \, \vec{x}_i \in R^m, \, l_i \in \left\{1, \ldots C\right\}, \, i = 1, 2 \ldots N \right\} \quad (11)$$

Here m denotes the dimensionality of the problem, N is the number of training instances and $\vec{x}_i$ and l_i are the input vector and class label respectively.

Detailed description of multi-class SVM techniques can be found in [25]. For instance, the multi-class solver can be a combination of multiple binary SVM classifiers or it can be transformed into a complex optimization problem.

The SVM training process is directly dependent on the size of the input dataset. This makes the SVM unsuitable for handling large datasets. Therefore, dataset size reduction is necessary. However, this reduction has to preserve the relevant information, which is crucial for a successful training.

The presented GNG-SVM framework consists of two phases. In the first phase the topology of the data is extracted by training multiple instances of the GNG algorithm. The original dataset is reduced to this extracted topological information. In the second phase the SVM solver is trained using the new reduced dataset.

The GNG-SVM framework will be presented on a multiclass classification problem (11). For an input dataset S with C classes the algorithm goes as follows:

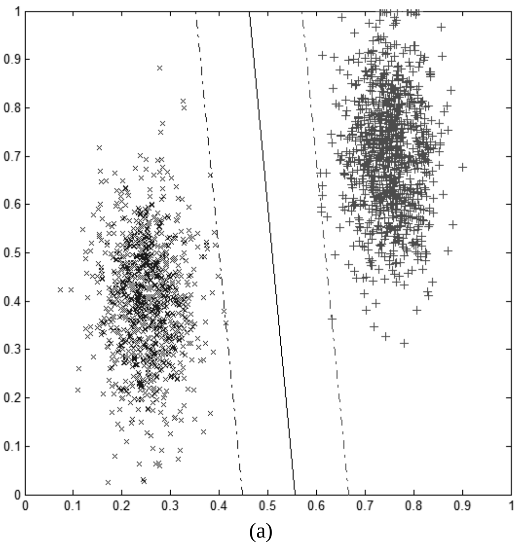
Step 1.1: Divide the training dataset S into C subsets S_i, each containing instances of one class.

$$\{S_1,\ S_2,...,S_C\} \tag{12}$$

Step 1.2: Construct a set of C instances of the growing neural gas algorithm GNG_i.

$$\{GNG_1, GNG_2,..., GNG_C\} \tag{13}$$

Step 1.3: Assign each subset S_i to its instance of the growing neural gas GNG_i.



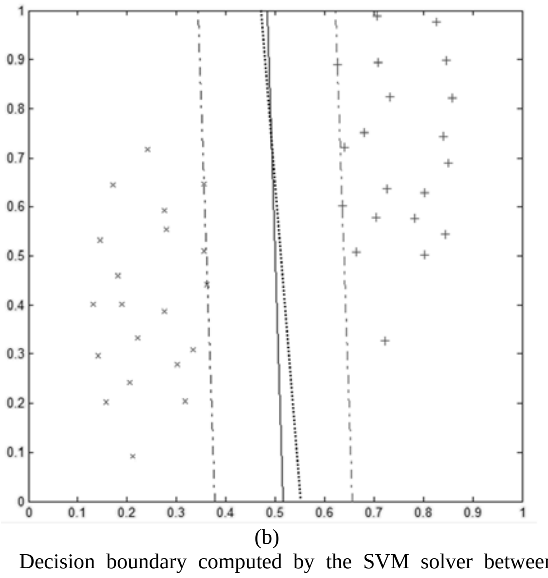


Fig. 3. Decision boundary computed by the SVM solver between two classes using the original dataset (a) and using the reduced dataset (b).

Step 1.4: Train each instance GNG_i on the training set S_i until a specified convergence criteria is met (e.g. certain number of iteration, certain number of neurons in the network).

Step 1.5: Extract the topological knowledge from each converged instance of the GNG algorithm GNG_i. The topological knowledge is contained in the set of reference vectors W_i.

$$\{W_1, W_2,..., W_C\} \tag{14}$$

Step 1.6: Extend each set of reference vectors W_i into a set W_i^* by assigning class label l_i of class i to each reference vector $\vec{w}_j$ in the class.

$$W_i^* = \{(\vec{w}_j, l_i)\}_{j=1,2,...,N_i} \tag{15}$$

Here N_i denotes the number of reference vectors in set W_i.

Step 1.7: Create a new reduced input dataset S^* by joining all labeled sets of reference vectors W_i^*.

$$S^* = \bigcup_{i=1}^{C} W_i^* \tag{16}$$

Step 2.1: Train the SVM solver on the new reduced training dataset S^*.

Fig. 3a shows the separation margin computed by the SVM solver on the original training dataset consisting of two classes. The separation hyper plane is marked with a straight line, while the boundaries of the margin are drawn with dashed-and-dotted lines. For a comparison, Fig. 3b illustrates the separation margin constructed by the SVM solver on the new reduced training dataset. The original dataset was reduced into 20 neurons per class. The dotted line in Fig. 3b shows the original separation hyper plane from Fig. 3a. This comparison demonstrates that while the size of the training dataset was significantly reduced, the constructed separation margin was altered only slightly.

IV. Experimental Results

The implementation of the GNG-SVM framework consists of two parts. The input dataset is first processed with a GNG algorithm implemented in C++ programming language. Consequently the new reduced input dataset is supplied to the SMO SVM solver in Weka [3]. In this section, the experimental results are presented and analyzed. First the performance of the GNG-SVM framework is evaluated on both synthetic and real world classification problems. Additionally the tradeoff between the training time and the performance of the algorithm is investigated.

The parameters of the GNG algorithm were set in all cases according to Table I, following the original implementation in [8]. These parameters' values yield satisfactory behavior of the GNG algorithm. The SMO

TABLE I
SUMMARY OF PARAMETERS VALUES FOR THE GNG ALGORITHM

λ	100
ε_b	0.2
ε_n	0.006
α	0.5
$a_{\max}$	200

SVM solver in Weka was used with polynomial kernels and the complexity parameter set to 1.

A. Synthetic Dataset

Firstly, the GNG-SVM framework was evaluated on two synthetic dataset. Both generated datasets consisted of data points in 2D space divided into two classes. Each class had 50,000 instances and was generated using several Gaussian distributions. In the first dataset the classes were linearly separable, while the second dataset contained two linearly non-separable classes with overlap between their boundaries.

The number of neurons in the network was used as the convergence criterion. The training of the GNG algorithm terminated upon reaching 50 neurons in the network.

The datasets were 10 times split randomly into training and testing sets each containing about 50% of the data instances. After learning the topology of the training set with the GNG algorithm, the SVM algorithm was trained on the new reduced dataset, containing 50 input instances for each class. The trained SVM was tested on the testing dataset consisting of 50,000 data instances. Table II and III summarize the obtained results averaged over 10 runs.

The testing on the linearly separable synthetic dataset proved the correctness of the proposed algorithm. The classification accuracy of the GNG-SVM framework was nearly perfect. In the case of the linearly non-separable dataset, the overall percentage of correctly classified instances dropped by about 4%. However, the size of the original input dataset was reduced by two orders of magnitude (500 times).

B. Real World Data Set

Secondly, the GNG-SVM framework was tested on benchmark real world datasets with higher dimensionality. Two datasets were chosen from the UCI's Machine Learning Repository [22]. The Magic gamma telescope dataset consists of 10-dimensional instances of two classes, with 12,332 and 6,688 examples respectively. The Shuttle dataset consists of 43,500 9-dimensional instances. Originally, 7 different classes were included in this dataset, 80% of which belonged to a single class. For the purpose of the GNG-SVM framework testing, the 6 less abundant classes were grouped into a single class.

Among the other available benchmark datasets, these two were chosen for the following reasons. They are typical examples of datasets that could significantly benefit from using the presented GNG-SVM framework. Both are large datasets with higher dimensionality and real valued attributes. On the other hand, they still have reasonable size for the training with a classical SVM algorithm. This is necessary for the performance comparison between the training of the SVM solver using the whole dataset and the performance of the GNG-SVM framework.

In this testing the convergence criterion was set to 100 neurons in the network. The original datasets were 10 times split randomly in half (training and testing datasets). The results, averaged over 10 runs, are presented in Table IV and V.

The 10-dimensional Magic dataset proved to be a relatively difficult problem for the SVM classifier. This is shown in Table IV by the relatively lower classification accuracy (79.09%), when training with the whole original training dataset. On the other hand, the 9-dimensional Shuttle dataset proved to be a relatively easier task for the SVM solver. This is shown in Table V by the relatively higher classification accuracy (96.57%), when training on the whole original training dataset. However, in both cases,

TABLE II
PERFORMANCE OF THE GNG-SVM FRAMEWORK ON LINEARLY SEPARABLE SYNTHETIC DATASET

Dataset preprocessing	No	GNG
Correctly Classified Instances	99.99%	99.94%
Class 1 True Positive Rate	99.99%	99.98%
Class 2 True Positive Rate	99.98%	99.92%

TABLE III
PERFORMANCE OF THE GNG-SVM FRAMEWORK ON LINEARLY NON-SEPARABLE SYNTHETIC DATASET

Dataset preprocessing	No	GNG
Correctly Classified Instances	97.59%	93.52%
Class 1 True Positive Rate	98.21%	88.69%
Class 2 True Positive Rate	96.91%	98.35%

TABLE IV
PERFORMANCE OF THE GNG-SVM FRAMEWORK ON THE MAGIC DATASET

Dataset preprocessing	No	GNG
Correctly Classified Instances	79.09%	79.01%
Class 1 True Positive Rate	89.78%	90.18%
Class 2 True Positive Rate	59.51%	58.54%

TABLE V
PERFORMANCE OF THE GNG-SVM FRAMEWORK ON THE SHUTTLE DATASET

Dataset Preprocessing	No	GNG
Correctly Classified Instances	96.57%	91.34%
Class 1 True Positive Rate	98.42%	99.18%
Class 2 True Positive Rate	89.73%	62.79%

the significant reduction of the size of the training datasets (from the 9,510 and 21,750 training instances respectively to 200 instanced in the reduced data set in both cases), resulted in a slight decrease of performance.

The percentage of correctly classified instances dropped by less than 1% in the case of the Magic dataset and by about 5% in case of the Shuttle dataset. This can be considered as an acceptable tradeoff for the significant reduction of dataset size. However, Table V also shows that the true positive rate of the less abundant class 2 in the Shuttle dataset decreased by almost 27%, which is very significant. Analyses of possible causes revealed that this class was created by combining 6 classes in the original Shuttle dataset. Thus, most likely this class consists of multiple clusters in different areas of the attribute space. This complex topology in 9-dimensional space would require higher number of neurons. The following test demonstrates that the performance improves as more neurons are used.

C. *Training Time vs. Performance Tradeoff*

When using the number of neurons in the network as the convergence criterion, there are two boundary cases for the GNG-SVM framework. If only a single neuron is allowed in the network, the whole algorithm reduces to computing the center of gravity (COG) of each class in the training set. The SVM separation hyper-plane is then constructed using only the COGs, which would lead to a poor performance in case of clusters with complex shapes. In the opposite case, when the number of neurons allowed in the network is equal to the number of training patterns, the GNG network only stores the position of all data points and no reduction is achieved. The performance in this case would be identical as if training on the SVM solver on the whole original training dataset.

The tradeoff between the training time and the performance of the GNG-SVM framework was investigated. The classification performance on the Shuttle dataset was measured for different values of the convergence criterion for the GNG algorithm. It was expected that as the number of neurons in the network increases the more topological knowledge is extracted and the better should be the classification performance of the SVM solver.

Fig. 4 and Fig. 5 show the percentages of correctly classified instances and class 1 and class 2 true positive rates as functions of the number of neurons for the GNG training convergence criterion.

The experiment revealed the benefits of allowing more time for the learning of the topological information. Fig. 4 shows that the overall percentage of correctly classified instances as well as the class 1 positive rate stays nearly the same, while the size of the network is increasing. However, because of the disproportional size of each class in the dataset (20% and 80% respectively), the default classification rate on the Shuttle dataset is 80%. In such cases the classification rate achieved on the less abundant class is a more relevant measure than the overall classification accuracy. As shown in Fig. 5, the class 2 true

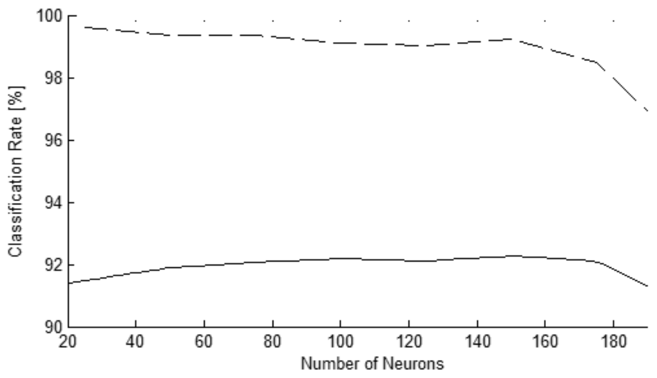


Fig. 4. The percentage of correctly classified instanced (solid line) and the class 1 true positive rate (dashed line) as a function of the number of neurons in the GNG network.

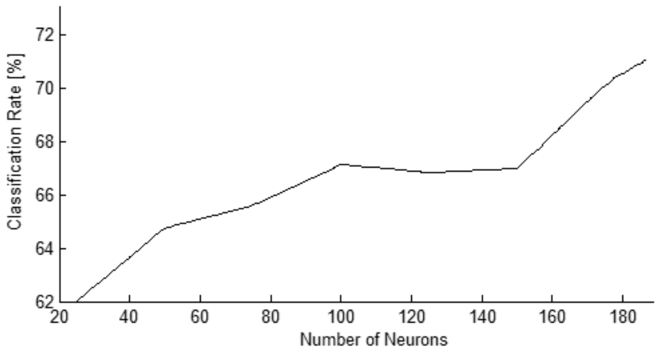


Fig. 5. The class 2 true positive rate as a function of the number of neurons in the GNG network.

positive rate is continually raising as the number of neurons in the network increases. In other words, the more time is allocated for the training of the GNG algorithm, the better will be the performance of the SVM solver.

V. Conclusion

The GNG-SVM framework was presented in this paper. It combined the Growing Neural Gas algorithm with the Support Vector Machine into a specific algorithm for classifying large datasets. The input data set was first preprocessed with the GNG algorithm. A new reduced training dataset was created from the extracted topological knowledge. Because the size of the dataset was significantly reduced, the training process of the SVM solver became substantially less memory demanding.

The performance of the GNG-SVM framework was tested on synthetic as well as on real world datasets. Testing on the linearly separable and non-separable synthetic datasets proved that the presented algorithm can significantly reduce the size of the input dataset, while preserving important topological information. The classification accuracy of the GNG-SVM framework on the Magic dataset stayed nearly the same compared to the SVM training on the whole original dataset. In case of the Shuttle dataset, the less abundant class 2 experienced a significant decrease of performance, which was attributed to the complexity of the problem. This showed that the GNG-SVM framework is suitable for a certain type of problems where classes form more compact clusters in the attribute space.

Further, the tradeoff between the training time and the classification accuracy of the GNG-SVM framework was investigated. It was demonstrated that the more time is allocated for the GNG training phase; the better the performance of the SVM solver will be.

ACKNOWLEDGMENT

The authors would like to thank the University of Idaho Nuclear Engineering Program for providing support for this project.

REFERENCES

[1] T. Mitchell, *Machine Learning.* McGraw-Hill, 1997.

[2] Ch. M. Bishop, *Pattern Recognition and Machine Learning.* Springer-Verlag, New York, 2006.

[3] I. H. Witten, E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques (Second Edition).* Morgan Kaufmann, San Francisco, 2005.

[4] V. N. Vapnik, *Statistical Learning Theory.* John Wiley and Sons, 1998.

[5] C. J. Burges, "A tutorial on support vector machines for pattern recognition," *Data Mining and Knowledge Discovery*, vol 2., no. 2, 1998.

[6] C.-C. Chang, C.-J. Lin, "Training nu-support vector classifiers: Theory and algorithms," *Neural Computation*, 13:2119-2147, 2001.

[7] H. Yu, J. Yang, J. Han, "Classifying large data sets using SVM with hierarchical clusters," In *Proceedings of the SIGKDD 2003*, Washington, DC, 2003, pp 306-315.

[8] J. L. Balczar, Y. Dai, O. Watanabe. "A random sampling technique for training support vector machines," In *Proc 13th Int. Conf. Algorithmic Learning Theory*, Washington, DC, 2001.

[9] S. Tong, D. Koller, "Support vector machine active learning with applications to text classification," In *Proc. 17th Int. Conf. Machine Learning*, Stanford, CA, 2000.

[10] B. Jin, Y.-Q. Zhang, "Classifying Very Large Data Sets with Minimum Enclosing all Based Support Vector Machine," In *Proc. IEEE Int. Conf. on Fuzzy Systems*, Vancouver, Canada, 2006.

[11] S.-W. Xiong, X.-X. Niu, H.-B. Liu, "Support vector machines based on substractive clustering," In *Proc 14th Int. Conf. Machine Learning and Cybernetics*, Guangzhou, China, 2005.

[12] B. Fritzke, "A growing neural gas network learns topologies," In G. Tesauro, D. D. Touretzky, & T. K. Leen (Eds.), *Advances in neural information processing systems*, MIT Press, Cambridge, 1995, pp. 625-632.

[13] A. Asuncion, D. J. Newman, UCI Machine Learning Repository [http://www.ics.uci.edu/~mlearn/MLRepository.html]. Irvine, CA: University of California, School of Information and Computer Science.

[14] T. Joachims, "Text categorization with support vector machines," *Proc. 10th European Conference on Machine Learning*, Chemnitz, Germany, 1998.

[15] A. Smola, B. Sch, "A tutorial on support vector regression," Technical report, 1998.

[16] L. Kaufman, "Solving the quadratic programming problem arising in support vector classification," In B. Scholkopf, C. J. C. Burges and A. J. Smola, (Eds.), *Advances in Kernel Methods-Support Vector Learning*, MIT Press, 1998, pp. 147-168.

[17] J. C. Platt, "Fast training of SVMs using sequential minimal optimization," In B. Scholkopf, C. J. C. Burges, and A. J. Smola, (Eds.), *Advances in Kernel Methods-Support Vector Learning*, MIT Press, 1998, pp. 185-208.

[18] R. Collobert, S. bengio, "SVMTorch: Support vector machines for large-scale regression problems," *Journal of Machine Learning Research*, 2001, pp. 143-160.

[19] T. Joachims, "Making large-scale support vector machine learning practical," In B. Scholkopf, C. J. C. Burges, and A. J. Smola, (Eds.), *Advances in Kernel Methods-Support Vector Learning*, MIT Press, 1998.

[20] J. Kivinen, A. J. Smola, R. C. Williamson, "Online learning with kernels," In *Proc. Advances in Neural Information Processing Systems*, Cambridge, MA, 2002.

[21] N. Syed, H. Liu, K. Sung, "Incremental learning with support vector machines," In *Proc. The Workshop on Support Vector Machines at the International Joint Conference on Artificial Inteligence*, Stockholm, Sweden, 1999.

[22] T. M. Martinetz, K. J. Schulten, "A neural gas network learns topologies ," In T. Kohonen, K. Makisara, O. Simula, J. Kangas, (Eds.), *Artificial Neural Networks*, North-Holland, Amsterdam, 1991, pp. 397-402.

[23] B. Fritzke, "Growing cells structures – a self-organizing network for unsupervised and supervised learning," *Neural Networks*, 1993, pp 1441-1460.

[24] J. V. de Oliviera, W. Pedrycz (Editors), *Advances in Fuzzy Clustering and its Applications*, Wiley, June, 2007.

[25] C.-W. Hsu, C.-J. Lin, "A comparison of methods for multi-class support vector machines," *In IEEE Transaction on Neural Networks*, vol. 13, 1998, pp. 415-425.