International Journal of Neural Systems, Vol. 22, No. 5 (2012) 1250023 (8 pages)
© World Scientific Publishing Company
DOI: 10.1142/S0129065712500232

ONLINE SEMI-SUPERVISED GROWING NEURAL GAS

OLIVER BEYER* and PHILIPP CIMIANO†
Semantic Computing Group
CITEC, Bielefeld University, Bielefeld, Germany
**obeyer@cit-ec.uni-bielefeld.de*
†cimiano@cit-ec.uni-bielefeld.de

Accepted 10 August 2012
Published Online 21 September 2012

In this paper we introduce online semi-supervised growing neural gas (OSSGNG), a novel online semi-supervised classification approach based on growing neural gas (GNG). Existing semi-supervised classification approaches based on GNG require that the training data is explicitly stored as the labeling is performed *a posteriori* after the training phase. As main contribution, we present an approach that relies on online labeling and prediction functions to process labeled and unlabeled data uniformly and in an online fashion, without the need to store any of the training examples explicitly. We show that using on-the-fly labeling strategies does not significantly deteriorate the performance of classifiers based on GNG, while circumventing the need to explicitly store training examples. Armed with this result, we then present a semi-supervised extension of GNG (OSSGNG) that relies on the above mentioned online labeling functions to label unlabeled examples and incorporate them into the model on-the-fly. As an important result, we show that OSSGNG performs as good as previous semi-supervised extensions of GNG which rely on offline labeling strategies. We also show that OSSGNG compares favorably to other state-of-the-art semi-supervised learning approaches on standard benchmarking datasets.

Keywords: Topological maps; online labeling; semi-supervised learning; classification; neural networks.

1. Introduction

Approaches based on topological maps, e.g. *Self-organizing maps* (*SOMs*)[1] or *growing neural gas* (*GNG*),[2] have been successfully applied to clustering problems, allowing to represent a high-dimensional input space in a low-dimensional and interpretable feature map. GNG, for example, when used with unlabeled data, will learn "natural categories" and thus the inherent topology of the data in an incremental fashion. GNG features the advantages of unsupervised approaches that can learn categories for which no labeled data is given. Approaches such as GNG are ideal in life-long learning settings where neither the categories can be assumed to be fixed *a priori* nor labels can be assumed to be available for all categories. With appropriate extensions, topological maps such as SOMs or GNG can also be trained with labeled data and thus be used in classification tasks.[3,4] This requires appropriate labeling functions that assign labels to neurons of the network as well as prediction functions that assign labels to unseen examples. In the context of GNG, mainly offline labeling techniques have been proposed so far, i.e. the labeling is performed in batch mode after all the training data has been processed. This requires the explicit storage of training data and thus runs counter to the online nature of GNG. Training in batch mode has also been argued to be disadvantageous in some scenarios. First, in many applications we find massive streams of data that cannot be stored on standard hardware anymore (see Ref. 5). Online clustering algorithms (see Ref. 6) thus become especially relevant in the context of stream data mining as data cannot be processed in batch mode or in several passes and the model needs to be updated on-the-fly instead.

Second, it even has been shown that online learning can render the training more efficient by using the model to generate new training examples which are closer to the desired solution, thus allowing a more efficient exploration of the parameter search space (see Ref. 7). Further, there are scenarios (e.g. *Interactive Learning*) and tracking applications where batch learning is simply not suitable (see Refs. 8 and 9).

In this paper we propose an extension of the standard GNG algorithm to an online semi-supervised classifier. We propose on the one hand an extension of GNG into an online classifier by introducing a step that updates the labels of a (winner) neuron after each data point has been processed. This circumvents the need to store all the labeled training data explicitly. Second, we propose a further extension of the Online Growing Neural Gas (OGNG) classifier into a semi-supervised classifier which leverages unlabeled data by labeling unlabeled data points on-the-fly. By doing this, we extend a previous semi-supervised version of Growing Neural Gas (SSGNG) (Ref. 4) and experimentally show that our extension (OSSGNG) performs comparably while at the same time circumventing to perform the labeling in an offline phase, thus requiring to store examples while training.

2. Online Growing Neural Gas (OGNG)

As already mentioned, one typical approach to turn GNG into a classifier is by extending the algorithm with appropriate labeling and prediction functions that assign labels to neurons as well as labels to unseen examples. In this section, we first present a set of offline labeling functions that have been proposed in the context of other topological map approaches, but have not been systematically investigated before. Further, we introduce a set of online labeling approaches and prediction approaches that are based on linkage strategies used in cluster analysis. We present experimental results on three datasets comparing the performance of offline and online labeling strategies, coming to the conclusion that online labeling strategies do not yield significantly worse results compared to offline labeling strategies. As using offline strategies for labeling neurons requires the storage of training examples, the application of online strategies is preferable, particularly as they produce comparable results as conveyed by our experimental results.

2.1. *Offline labeling methods*

In order to apply GNG to a classification task, we require two functions: (i) a neuron labeling function $l : N \to C$ where C is the set of class labels, and (ii) a prediction function $\text{pred} : X \to C$ where X is the input space. We analyze the following offline neuron labeling functions as proposed by Lau *et al.*[10] They are offline in the sense that they assume that the pairs (x, l_x) with $x \in X_{\text{train}} \subseteq X$ and $l_x \in C$ seen in the training phase are explicitly stored:

- *Minimal-distance method* (min-dist): According to this strategy, neuron n_i adopts the label l_x of the closest data point $x \in X_{\text{train}}$:
$$l_{\text{min-dist}}(n_i) = l_x = l\left(\arg\min_{x \in X_{\text{train}}} |n_i - x|^2\right).$$
- *Average-distance method* (avg-dist): According to this strategy, we assign to neuron n_i the label of the category c that minimizes the average distance to all data points labeled with category c:
$$l_{\text{avg-dist}}(n_i) = \arg\min_c \sum_{k=1}^{|X(c)|} \frac{|n_i - x_k|^2}{|X(c)|},$$
where $X(c) = \{x \in X_{\text{train}} \mid l_x = c\}$ is the set of all examples labeled with c.
- *Majority method* (majority): According to this strategy, we label neuron n_i with that category c having the highest overlap (in terms of data points belonging to category c) with the data points in the voronoi cell for n_i. We denote the set of data points in the voronoi cell for n_i as $v(n_i) = \{x \in X_{\text{train}} \mid \forall n_j, j \neq i : |n_j - x|^2 \geq |n_i - x|^2\}$ within the topological map. The majority strategy can be formalized as follows:
$$l_{\text{majority}}(n_i) = \arg\max_c |X(c) \cap v(n_i)|.$$

In addition to the neuron labeling strategy, we need to define prediction functions that assign labels to unseen examples. These prediction functions are inspired by linkage strategies typically used in cluster analysis[11,12]:

- *Single-linkage*: According to this prediction strategy, a new data point x_{new} is labeled with that category c of the winner neuron n which minimizes the distance to this new example:
$$\text{pred}_{\text{single}}(x_{\text{new}}) = \arg\min_c (\min_{n \in N(c)} |n - x_{\text{new}}|^2),$$

where $N(c) = \{n \in N \mid l(n) = c\}$ is the set of all neurons labeled with category c according to one of the above mentioned neuron labeling functions. According to this strategy, a data point, thus, adopts the label of the winner neuron.

- *Average-linkage:* Following this strategy, example x_{new} adopts the label of category c having the minimal average distance to the example:

$$\text{pred}_{\text{avg}}(x_{\text{new}}) = \arg\min_c \left(\sum_{k=1}^{|N(c)|} \frac{|n_k - x_{\text{new}}|^2}{|N(c)|} \right).$$

- *Complete-linkage*: According to this prediction strategy, a new data point x_{new} is labeled with that category c of the neuron n which minimizes the maximal distance to this new example:

$$\text{pred}_{\text{compl}}(x_{\text{new}}) = \arg\min_c \left(\max_{n \in N(c)} |n - x_{\text{new}}|^2 \right).$$

2.2. *Online labeling strategies for GNG*

In order to extend GNG into an online classification algorithm, we extend the basic GNG by a step in which the label of the presented stimulus is assigned on-the-fly, without the requirement of an additional labeling phase. We denote the winner neuron for data point x by $w(x)$. All prediction strategies are local in the sense that they do not consider any neighboring neurons besides the winner neuron $w(x)$. As the labeling is performed on-the-fly, the label assigned to a neuron can change over time, so that the labeling function is dependent on the number of examples the network has seen and has the following form: $l : N \times T \to C$. We will simply write $l^t(n_i)$ to denote the label assigned to neuron n_i after having seen t data points.

- *Relabeling method* (relabel): According to this very simple strategy, the winner neuron $w(x)$ adopts the label of x:

$$l^t_{\text{relabel}}(n_i) = l_x, \quad \text{where } n_i = w(x)\,.$$

- *Frequency-based method* (freq): In this labeling method we realize a "memory" for each neuron. We assume that each neuron stores information about how often a data point of a certain category has been assigned to n_i after t examples have been presented to the network. This frequency $\text{freq}_t(c, n_i)$ is updated on-the-fly and does not require the storage of training examples and thus represents a very restricted form of memory. According to this strategy, a neuron is labeled by the category which maximizes this frequency, i.e.

$$l^t_{\text{freq}}(n_i) = \arg\max_c \text{freq}_t(c, n_i).$$

- *Limited-distance method* (limit): According to this strategy, we also implement a simple memory that stores the distance of the data point that was closest to the neuron in question. We denote this data point as $\min_t(n_i)$ and the corresponding distance as $\theta_t(n_i) = |\min_t(n_i) - n_i|^2$. The winner neuron $w(x)$ adopts the category label l_x of the data point x if the distance between them is lower than $\theta_t(w(x))$. Only in case of a smaller distance, $\theta_t(n_i)$ will be updated with the new distance.

$$l^t_{\text{limit}}(n_i) = \begin{cases} l_x, & \text{if } |n_i - x|^2 \le \theta_t(n_i) \\ l^{t-1}_{\text{limit}}(n_i), & \text{else} \end{cases}.$$

2.3. *Experiments and results*

We compare and evaluate the above mentioned labeling strategies (online versus offline in particular) on three classification data sets: (i) an artificial data set generated following a 2D Gaussian mixture distribution with six classes located at $[0, 6], [-2, 2], [2, 2], [0, -6], [-2, -2], [2, -2]$ and standard deviation of 1, (ii) the ORL face database[13] and (iii) the image segmentation data set of the UCI machine learning database.[14]

In order to compare the different labeling strategies with each other, we set the parameters for GNG as follows: insertion parameter $\lambda = 300$; maximum age $a_{\max} = 120$; adaptation parameter for winner $e_b = 0.2$; adaptation parameter for neighborhood $e_n = 0.006$; error variable decrease $\alpha = 0.5$; error variable decrease $\beta = 0.995$. These parameters have been empirically determined on a trial and error basis. A different choice of parameter might lead to very different results. In our case the algorithm stops when a network size of 100 neurons is reached.

For our experiments we randomly sampled 10 training/test sets consisting of 4 labeled examples per category, averaging the accuracy over all 10 test folds. Table 1 shows the classification accuracy for various configurations of labeling methods (min-dist, avg-dist, majority, relabel, freq, limit) and prediction strategies (*single-linkage*, *average-linkage*, *complete-linkage*), averaged over the three different data sets.

Table 1. Classification accuracy for the offline and online labeling strategies combined with prediction strategies averaged over the three data sets (ART, ORL, SEG) trained with four labeled data points of each category (best averaged results are marked).

	Offline labeling				Online labeling			
	Min-dist method	Avg-dist method	Majority method	*Average*	Relabel method	Freq method	Limit method	*Average*
Single-linkage	81.93	78.92	83.39	***81.41***	83.25	83.25	83.39	***83.30***
Average-linkage	77.15	76.35	79.46	*77.65*	80.46	81.12	81.13	*80.90*
Complete-linkage	69.72	67.56	69.92	*69.07*	69.92	69.79	69.93	*69.88*
Average	*76.27*	*74.28*	***77.59***		*77.88*	*78.05*	***78.15***	

We evaluated the accuracy of each labeling method combined with three prediction strategies (rows of the tables).

According to Table 1, there is no offline labeling method which significantly outperforms the others. Comparing the accuracy results averaged over all prediction strategies, the *majority method* is the most effective labeling method as it provides the highest accuracy with 77.59%, followed by the *min-dist method* with 76.27% and the *avg-dist method* with 74.28%. Concerning the prediction strategies, the *single-linkage prediction* strategy shows best results averaged over all methods with 81.41%, followed by the *average-linkage prediction* strategy with an accuracy of 77.65%. The *complete-linkage* yielded the worst results with an averaged accuracy of 69.07%. According to Table 1, all three online labeling strategies are almost equal in their classification performance. The *limit method* performs slightly better compared to the other two methods and achieves an accuracy of 78.15%, followed by the *freq method* with an accuracy of 78.05% and the *relabel method* with an accuracy of 77.88%. As with the offline labeling strategies, it is also the case here that the *single-linkage prediction* is the best choice with an accuracy of 83.30%, followed by the *average-linkage prediction* with an accuracy of 80.90% and the *complete-linkage prediction* with an accuracy of 69.88%. Comparing the averaged accuracy of all labeling methods of Table 1, the results show that there is no significant difference between them in terms of accuracy. The online labeling methods even provide a slightly higher accuracy. Strategies relying on some sort of memory (e.g. storing the frequency of seen labels as in the *freq method*), do not perform significantly better than a simple memory-free method (*relabel method*) performing decisions on the basis of new data points only. This shows that the implementation of a label memory does not enhance the classifiers' performance.

Overall, the results of our experiments show that using online labeling strategies does not significantly deteriorate the performance of a classifier based on GNG in comparison to using offline labeling strategies.

2.4. *Online semi-supervised growing neural gas* (*OSSGNG*)

In order to extend GNG into a semi-supervised classifier, we add two steps (steps 4 and 5) to the original GNG algorithm, as shown in Fig. 1. In step (4), in case x is an unlabeled example, a label for x is predicted according to the chosen prediction strategy. The prediction strategy we use is the *single-linkage* prediction from Sec. 3.1. In step (5), the label of the presented stimulus is assigned to the winner neuron in each iteration of GNG. The label assignment is performed by an online labeling function, which in our case is the *limit* method described in Sec. 3.2.

In contrast to previous semi-supervised extensions of GNG (i.e. the approach by Zaki *et al.*[4]), no separate steps are thus required. In the approach of Zaki *et al.* — denoted as SSGNG in the following — two steps are iterated: In a first step, the network is trained using labeled examples only. Then, labels are assigned to neurons using an offline labeling approach. Finally, in a second step, unlabeled examples are classified into the network and labeled appropriately. These two steps are then iterated until the labeling converges, similar to the *expectation-maximization* (*EM*) approach.

In contrast to SSGNG, OSSGNG processes labeled and unlabeled training examples uniformly

Online Semi-supervised Growing Neural Gas

```
1. Start with two units $i$ and $j$ at random positions $w_i, w_j$ in the input space.
2. Present an input vector $x \in R^n$ from the input data.
3. Find the nearest unit $n_1$ (winner) and the second nearest unit $n_2$ (second winner).
4. If the label of $x$ is missing, assign a label to $x$ according to the selected prediction strategy.
5. Assign the label of $x$ to $n_1$ according to the present labeling strategy.
6. Increment the age of all edges emanating from $n_1$.
7. Update the local error variable by adding the squared distance between $w_{n_1}$ and $x$:
       $\Delta\text{error}(n_1) = |w_{n_1} - x|^2.$
8. Move $n_1$ and all its topological neighbors (i.e. all the nodes connected to $n_1$ by an edge) towards $x$ by fractions of $e_b$ and $e_n$ of the distance:
       $\Delta w_{n_1} = e_b(x - w_{n_1}),$
       $\Delta w_n = e_n(x - w_n),$
   for all direct neighbors $n$ of $n_1$.
9. If $n_1$ and $n_2$ are connected by an edge, set the age of the edge to 0 (refresh). If there is no such edge, create one.
10. Remove edges having an age greater than $a_{\max}$. If this results in nodes having no emanating edges, remove them as well.
11. If the number of input vectors presented or generated so far is an integer or multiple of a parameter $\lambda$, insert a new node $r$ as follows:
    • Determine the unit $q$ with the largest error.
    • Among the neighbors of $q$, find node $f$ with the largest error.
    • Insert a new node $r$ halfway between $q$ and $f$ as follows:
          $w_r = \frac{w_q + w_f}{2}.$
    • Create edges between $r$ and $q$, and $r$ and $f$. Remove the edge between $q$ and $f$.
    • Decrease the local error variable of $q$ and $f$ by multiplying them with a constant $\alpha$. Set the error $r$ with the new error variable of $q$.
12. Decrease all local error variables of all nodes $i$ by a factor $\beta$.
13. If the stopping criterion is not met, go back to step (2). (For our experiments, the stopping criterion has been set to be the maximum network size.)
```

Fig. 1. GNG algorithm with extension for online semi-supervised learning.

in every iteration step, with the exception that a label is only predicted for an unlabeled example. This means that OSSGNG is able to solve a classification task after each iteration step. The main advantage of the OSSGNG lies in its ability to train in an online fashion without the need of storing training examples explicitly. Further, the OSSGNG algorithm still provides the ability of GNG to perform a clustering on an unlabeled training set. This means that clusters can be formed without the knowledge about categories. The main disadvantage of the SSGNG is the fact that labels are assigned to each neuron *a posteriori* after the end of the training phase. Thus, the approach is not able to process a continuous stream of labeled and unlabeled training examples. Furthermore, labeled and unlabeled examples are processed in different phases and therefore need to be stored until the SSGNG training ends. Another disadvantage of SSGNG is that a minimal set of labeled examples for each class is crucial for the training.

2.5. *Experiments and results*

We evaluate the OSSGNG algorithm with respect to six datasets (g241c, g241d, Digit1, USPS, COIL, BCI) that have been proposed as benchmarks for semi-supervised classification (Ref. 15). We use SSGNG as baseline for our approach and evaluate the classification accuracy using 12-fold cross-validation on all six datasets.

As we want both GNG variations, SSGNG and OSSGNG, to be comparable, we chose a fixed set of parameters that was proposed by Zaki *et al.*[4] for SSGNG and used these throughout our experiments. The parameters are thus set as follows: insertion parameter $\lambda = 300$; maximum age $a_{\max} = 100$; adaptation parameter for winner $e_b = 0.2$; adaptation parameter for neighborhood $e_n = 0.006$; error variable decrease $\alpha = 0.5$; error variable decrease $\beta = 0.995$. The algorithm stops when a network size of 200 neurons is reached.

Our experiments are carried out using a 12-fold cross validation with 100 labeled examples per fold, respectively. This setup corresponds to the setup used in Chapelle *et al.*[16] where a number of state-of-the-art SSL algorithms were compared. We evaluate the accuracy of all compared algorithms on test, as shown in Table 2. Each row in the table represents the accuracy on test averaged over the 12 folds. The best accuracy is marked in each row. We also compare to a 1-*nearest neighbor* $(1 - NN)$ classifier and a *support vector machine* (*SVM*) classifier with a linear kernel in order to provide an overall baseline for all SSL approaches. Both classifiers were only trained with the labeled data points of our data sets.

According to Table 2, OSSGNG clearly outperforms SSGNG on three out of six datasets (g241c,

Table 2. Classification results (accuracy) of a 12-fold cross-validation for OGNG, SSGNG and OSSGNG performed on the six datasets (g241c, g24d, Digit1, USPS, COIL, BCI).

	OGNG (labeled data)	SSGNG	OSSGNG
g241c	46.43	49.64	**52.98**
g241d	54.98	44.03	**62.66**
Digit1	**97.81**	96.20	96.77
USPS	**94.35**	92.58	93.07
COIL	77.11	75.49	**81.45**
BCI	77.33	**80.51**	79.47
Average	*74.67*	*73.08*	***77.73***

g241d and COIL) while having a comparable performance on the datasets Digit1 and USPS. On average, OSSGNG has also a higher accuracy (77.73%) compared to SSGNG (73.08%). While we cannot claim that the differences are significant, it is valid to claim that our online version performs as good as our baseline (SSGNG) while circumventing the need to explicitly store training examples and to perform several passes over the data until reaching convergence.

The results in Table 2 show that extending OGNG with a semi-supervised component (OSSGNG) can improve its classification performance by up to 7.68% (dataset g241d). There are only two datasets (Digit1 and USPS) for which OSSGNG yields worse results compared to OGNG, albeit these differences are clearly minor. Interestingly, these are also the datasets for which a semi-supervised SVM classifier (TVSM) performs worse than a standard SVM (see Table 3).

We additionally compared our results to the results of standard semi-supervised classification algorithms published by Chapelle *et al.*,[16] namely *transductive SVM* (*TSVM*)[17] (using a linear kernel), *cluster-kernel*,[24] *data-dependancy regularization*[18] and *low-density separation* (*LDS*).[16] We did not reimplement these algorithms, but compared our results to the published results using the same data under same conditions. The results are summarized in Table 3. On two datasets (g241c, g241d), OSSGNG performs definitely worse compared to the other semi-supervised learning approaches. On the other four datasets Digit1, USPS and COIL, BCI, the performance of OSSGNG is better than the one of a standard SVM and comparable to those of other state-of-the-art semi-supervised learning approaches. On one dataset, i.e. BCI, it is even the case that OSSGNG outperforms all other approaches by far. These results clearly license the conclusion that OSSGNG can compete with other semi-supervised learning approaches.

2.6. *Discussion*

Our experiments show the benefit of the semi-supervised OGNG (OSSGNG). It clearly outperforms SSGNG and improves the classification performance of OGNG in four out of six datasets. The two datasets (Digit1, USPS) in which the semi-supervised approaches (OSSGNG, TSVM) yield worse results compared to their original algorithms (OGNG, SVM) seem to be very easy to classify as every compared algorithm achieves an accuracy over 90%. The results of the 1-NN approach also license this observation as it performs much better on those datasets than on the others. It seems that in these cases semi-supervised learning cannot improve the classification performance further.

Table 3. Classification results (accuracy) of a 12-fold cross-validation for our baseline (1-NN, SVM), OGNG, different standard SSL approaches and OSSGNG performed on the six datasets (g241c, g24d, Digit1, USPS, COIL, BCI).

	1-NN	SVM (labeled data only)	OGNG	TSVM	Cluster-Kernel	Data-Dep. Reg.	LDS	OSSGNG
g241c	59.72	76.89	46.43	81.54	**86.51**	79.69	81.96	52.98
g241d	62.51	75.36	54.98	77.58	**95.05**	67.18	76.26	62.66
Digit1	93.88	94.47	**97.81**	93.49	96.21	97.56	96.54	96.77
USPS	92.36	90.25	94.35	90.23	90.68	94.90	**95.04**	93.07
COIL	76.73	77.07	77.11	74.20	78.01	**88.54**	86.28	81.45
BCI	55.17	65.69	77.33	66.75	64.83	52.53	56.03	**79.47**
Average	*73.40*	*79.96*	*74.67*	*80.63*	***85.23***	*80.07*	*82.02*	*77.73*

For four out of six datasets, OSSGNG achieves better results compared to a standard SVM (with a linear kernel) while also being comparable to other standard SSL algorithms. It is striking that OSSGNG outperforms all other approaches by far on the BCI dataset. This dataset is characterized by the availability of only few data points (400 in total) as well as by low-dimensional feature vectors (117 dimensions). OSSGNG thus seems to generalize better on low numbers of examples.

3. Related Work

To our knowledge, there has been no systematic investigation and comparison of different labeling strategies (offline versus online in particular) for GNG. This is a gap that we have intended to fill. The question of how GNG can be extended to an online classification algorithm has also not been addressed previously. In most cases, offline strategies have been considered that perform the labeling after the training phase has ended and the network has stabilized to some extent as in the WEBSOM[3,19] and LabelSOM[20] approaches. In both of these approaches, the label assignment is essentially determined by the distance of the labeled training data point to the neurons of the already trained network. Such offline labeling strategies run counter to the online nature of GNG, whose interesting properties are that the network grows over time and only neurons, but no explicit examples, need to be stored in the network. Our results indeed license the claim that extending a clustering algorithm (based on GNG) with online labeling strategies does not yield a worse classification performance compared to using offline labeling functions. Another closely related approach was proposed by Shen *et al.*[21] They presented a *self-organizing incremental neural network* (*SOINN*) which provides a growing structure and is capable of processing labeled and unlabeled data. During the learning process, nodes are successively inserted into the network, separated into subclusters and merged into bigger clusters. In contrast, our approach relies on simpler yet effective machinery that does not require any additional heuristics. In recent years, there has been substantial work in the area of semi-supervised learning, both in the context of classification and clustering tasks. Those approaches have been successfully applied to a number of applications such as *text classification*,[17] *face recognition*[4] and *medical diagnosis*.[22,23]

4. Conclusion

We have presented an extension of GNG to an online semi-supervised classifier which relies on online labeling strategies to assign labels to neurons and label unlabeled data points on-the-fly. We have shown in particular that using online labeling strategies yields comparable results to using offline labeling strategies. As using offline strategies for labeling neurons requires the storage of training examples, the application of online strategies is preferable, particularly as they produce comparable results as conveyed by our experimental results. Further, we have shown that the semi-supervised extension to GNG compares favorably to other state-of-the-art semi-supervised learning approaches.

Acknowledgments

This work has been funded by the Excellence Initiative of the Deutsche Forschungsgemeinschaft (DFG). Thanks to Barbara Hammer for comments and feedback on a first draft of this paper.

References

1. T. Kohonen, Self-organized formation of topologically correct feature maps, *Biol. Cybern.* **43**(1) (1982) 59–69.
2. B. Fritzke, A growing neural gas network learns topologies, in *Proc. Int. Conf. Advances in Neural Information Processing Systems* (*NIPS*) (1995), pp. 625–632.
3. T. Kohonen, S. Kaski and K. Lagus, Self organization of a massive document collection, *IEEE Trans. Neural Netw.* **11**(3) (2000) 574–585.
4. S. M. Zaki and H. Yin, A semi-supervised learning algorithm for growing neural gas in face recognition, *J. Math. Model. Algorithms* **7**(4) (2008) 425–435.
5. M. M. Gaber, A. Zaslavsky and S. Krishnaswamy, Mining data streams: A review, *ACM Sigmod Record* **34**(2) (2005) 18–26.
6. W. Barbakh and C. Fyfe, Online clustering algorithms, *Int. J. Neural Syst.* **18**(3) (2008) 185–194.
7. M. Rolf, J. J. Steil and M. Gienger, Online Goal Babbling for rapid bootstrapping of inverse models in high dimensions, in *IEEE Int. Conf. Development and Learning* (*ICDL*), 2011, pp. 1–8.
8. J. Steil, H. Ritter and E. Körner, Online learning of objects in a biologically motivated visual architecture, *Int. J. Neural Syst.* **17**(4) (2007) 219–230.

9. D. P. Mandic, P. Vayanos, M. Chen and S. L. Goh, Online detection of the modality of complex valued real world signals, *Int. J. Neural Syst.* **18**(2) (2008) 67–74.
10. K. Lau, H. Yin and S. Hubbard, Kernel self-organising maps for classification, *Neurocomputing* **69**(16–18) (2006) 2033–2040.
11. M. R. Anderberg, *Cluster Analysis for Applications* (Academic Press, New York, 1973).
12. S. C. Johnson, Hierarchical clustering schemes, *Psychometrika* **32**(3) (1967) 241–254.
13. F. S. Samaria, Parametrization of a stochastic model for human face identification, in *Proc. of the IEEE Workshop on Applications of Computer Vision*, 1994, pp. 138–142.
14. C. L. Blake and C. J. Merz, UCI repository of machine learning databases (1998).
15. O. Chapelle, B. Schölkopf and A. Zien, *Semi-Supervised Learning*, Vol. 2 (MIT Press, 2006).
16. O. Chapelle and A. Zien, Semi-supervised classification by low density separation. in *Proc. 10th Int. Workshop on Artificial Intelligence and Statistics*, 2005, pp. 57–64.
17. T. Joachims, Transductive Inference for text classification using support vector machines, in *Proc. 16th International Conference on Machine Learning* (*ICML*), 1999, pp. 200–209.
18. A. Corduneanu and T. Jaakkola, Data dependent regularization. *Semi-supervised Learning* **2006** (2006) 163–182.
19. K. Lagus and S. Kaski, Keyword selection method for characterizing text document maps, in *Proc. of Int. Conf. Artificial Neural Networks* (*ICANN*), 1999, pp. 371–376.
20. A. Rauber, LabelSOM: On the labeling of self-organizing maps, in *Proc. Int. Joint Conf. Neural Networks* (*IJCNN*), 1999, pp. 3527–3532.
21. F. Shen, H. Yu, K. Sakurai and O. Hasegawa, An incremental online semi-supervised active learning algorithm based on self-organizing incremental neural network, in *Neural Computing and Applications* (Springer, 2010).
22. R. Cruz-Barbosa and A. Vellido, Semi-supervised analysis of human brain tumour from partially labeled MRS information using manifold learning models, *Int. J. Neural Syst.* **21**(1) (2011) 17–29.
23. A. Fornells, J. M. Martorell, E. Golobardes, J. M. Garrell and X. Vilas, Management of relations between cases and patterns from SOM for helping experts in breast cancer diagnosis, *Int. J. Neural Syst.* **18**(1) (2008) 33–43.
24. J. Wenston, C. S. Leslie, E. Le D. Zhou, A. Elisseef and W. S. Noble, Semi-supervised protein classification using cluster kernels, *Bioinformatics* **21**(15) (2003) 3241–3247.