ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

DYNG: Dynamic Online Growing Neural Gas for Stream Data Classification

Oliver Beyer and Philipp Cimiano

Semantic Computing Group
CITEC, Bielefeld University

Abstract. In this paper we introduce Dynamic Online Growing Neural Gas (DYNG), a novel online stream data classification approach based on Online Growing Neural Gas (OGNG). DYNG exploits labelled data during processing to adapt the network structure as well as the speed of growth of the network to the requirements of the classification task. It thus speeds up learning for new classes/labels and dampens growth of the subnetwork representing the class once the class error converges. We show that this strategy is beneficial in life-long learning settings involving non-stationary data, giving DYNG an increased performance in highly non-stationary phases compared to OGNG.

1 Introduction

Streams of data nowadays emerge in a number of application domains, in particular in those concerned with processing data originating from social media applications, sensor networks, news feeds, etc.[9]. In many scenarios, one wishes to classify the data items in these streams into a number of evolving classes. In order to scale to massive amounts of data, approaches to classify stream data need to fulfill the following requirements: i) they have to process data in one pass, ii) they should avoid the storage of data points, iii) they should be fast in their classification and iv) they should be able to work with non-stationary data, i.e. with changing data and class distributions and be able to detect new classes on the fly.

The key challenge in such life-long learning scenarios is the "stability-plasticity dilemma", which refers to the ability of acquiring new knowledge (plasticity) while retaining old memory (stability). Architectures such as *Growing Neural Gas (GNG)* [5] and *Online Growing Neural Gas (OGNG)* [1] are beneficial for such tasks. In particular, OGNG becomes interesting in the context of stream data classification problems as it satisfies the above requirements: i) OGNG sees every data-point once and updates the growing network on the fly, ii) classification is performed by comparing the new datapoint to a relatively small number of prototypes, iii) OGNG is an online approach where each seen example causes an update of the network, but examples are not explicitly stored, and iv) OGNG has mechanisms for detecting new classes and inserting new neurons.

In this paper we are concerned with exploring how to improve the classification performance of OGNG by exploiting label information. In particular we would like OGNG to behave in the following fashion: i) it directly inserts a new neuron for an unseen label, ii) it grows dynamically as the task requires

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

by inserting neurons as long as the error for a class decreases, dampening this growth once the classification error converges. Toward meeting these desiderata, in this paper, we present an extension of Online Growing Neural Gas that we call Dynamic Online Growing Neural Gas (DYNG). DYNG uses label information of the presented stimulus and tracks the classification error for each class to insert neurons as long as the classification performance for this class grows. We evaluate DYNG on the task of classifying a textual stream of data on two datasets, showing that it outperforms OGNG and that it even outperforms a SVM classifier, when both use a comparable amount of memory.

2 Dynamic Online Growing Neural Gas (DYNG)

In this section, we provide a detailed description of *Dynamic Online Growing Neural Gas (DYNG)*, which is an extension of *Online Growing Neural Gas (OGNG)* [1] and thus combines unsupervised clustering and supervised classification. The complete description of the DYNG algorithm can be found in Algorithm 1. Due to space limitations, we only discuss the main extensions to OGNG in DYNG below. In the following, $l^t(n_i)$ denotes the label of neuron $n_i \in N$ after t data points have been presented:

- *Insert a new neuron for new labels (steps 5-8):* DYNG inserts a new neuron n_{new} for a stimulus ξ with an unknown label $l^t(\xi)$.
- *Update class error (steps 10-12):* In these steps, DYNG updates the class error $classError^t(c_1)$ of the class $c_1 = l^t(n_1)$ of the winner neuron n_1 after seen t data points. We use the class error to decide if the classification performance has increased (see steps 26-32).
- *Distance-based insertion strategy (steps 15-23):* In case of a mismatch between the label of n_1 and the label of stimulus ξ, DYNG determines the nearest neuron n_{lb} with $l^t(n_{lb}) = l^t(\xi)$ and inserts a new neuron if the distance $|w_{n_{lb}} - \xi|^2 \geq |w_{n_{lb}} - \xi|^2\tau + |w_{n_1} - \xi|^2$, i.e. if n_{lb} is too far away from ξ compared to n_1. In the other case n_{lb} is adapted towards ξ. Thereby, $|w_{n_{lb}} - \xi|^2\tau + |w_{n_1} - \xi|^2$ can be seen as vigilance parameter as known from *Adaptive Resonance Theory (ART)* and thus controls the size of the additional network memory provided by DYNG.
- *Check for performance improvement (steps 26-32):* In steps 26-32, DYNG compares the class error of all classes $c_i \in C$ at iteration t to those of iteration $t - \lambda$. If the class performance has improved, we set $imp(c_i) =$ "$true$", otherwise $imp(c_i) =$ "$false$". This means that in the next $t - \lambda$ iterations the distance-based insertion strategy will only be applied to the classes $c_i \in C$ with $imp(c_i) =$ "$true$" (see step 15).

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

Algorithm 1 Dynamic Online Growing Neural Gas (DYNG)

Start with two units i and j at random positions in the input space.
while $Stream.hasNext$ **do**
  $\xi := Stream.next(t)$ with $\xi \in \mathbb{R}^n$.
  Find the nearest unit n_1 and the second nearest unit n_2.
  if $l^t(\xi)$ is unknown **then**
    Add a neuron n_{new} with $w_{n_{new}} = \xi$ and $l^t(n_{new}) = l^t(\xi)$.
    Create an edge between n_{new} and n_1.
  end if
  Update the local error variable of n_1 and increment the age of all edges emanating from n_1 according to OGNG[1].
  if $l^t(n_1) \neq l^t(\xi)$ **then**
    Update the class error: $\Delta classError^t(c_1) = 1 - \frac{\Delta error(n_1)}{\max_{n \in N}(\Delta error(n))}$
  end if
  $l^t(n_1) := l^t(\xi)$ {OGNG relabel-method}
  Move n_1 and all its topological neighbors according to OGNG and connect n_1 and n_2.
  if $l^t(n_1) \neq l^t(\xi) \wedge imp(l^t(\xi)) =$ "$true$" **then**
    Find nearest unit n_{lb} with $l^t(n_{lb}) = l^t(\xi)$.
    if $|w_{n_{lb}} - \xi|^2 \geq |w_{n_{lb}} - \xi|^2 \tau + |w_{n_1} - \xi|^2$ **then**
      Add a neuron n_{new} with $w_{n_{new}} = \xi$ and $l^t(n_{new}) = l^t(\xi)$.
      Create an edge between n_{new} and n_{lb}.
    else
      Move n_{lb} towards ξ by the fraction of e_b: $\Delta w_{n_{lb}} = e_b(\xi - w_{n_{lb}})$
    end if
  end if
  Remove edges and nodes according to OGNG.
  if $t \mod \lambda = 0$ **then**
    for all classes $c_i \in C$ **do**
      if $classError^{t-\lambda}(c_i) < classError^t(c_i)$ **then**
        $imp(c_i) =$ "$true$"
      else
        $imp(c_i) =$ "$false$"
      end if
    end for
    Insert a new neuron and update the network link structure according to OGNG.
  end if
  Decrease all local error variables of all nodes n_i and all class error variables of all classes c_i by a factor β.
  $t++$.
end while

3 Experiments and results

3.1 Data sets & methodology

We evaluate our approach in a life-long learning scenario in which there is a continuous incoming stream of data points to be classified. As stream of text documents we rely on the well-known Reuters RCV1v2 [11] collection (consisting of 804.414 documents assigned to 103 different classes). We also rely on a stream of Twitter messages consisting of 82.095 data points assigned to 14.040 different classes[1]. In both cases, for each document/tweet we select the most common class label, as one document or tweet corresponds to several classes. We use TF-IDF vectors in both cases and remove terms that occur in less than 20% of the documents. Thereby, we reduce the dimension of the TF-IDF feature vector

[1]We crawled Tweets containing the hashtag "Berlin" or having the Twitter-Place-ID of Berlin from 05/03/2012-06/05/2012 using the Twitter-API. We tokenized and normalized the Twitter messages.

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

from 47.236 to 832 dimensions (ReutersRCV1v2) and from 456.837 to 2.623 dimensions (Twitter corpus). For our experiments, both datasets are ordered by their document-id/tweet-id.

We evaluate our DYNG algorithm in comparison to OGNG and to the offline classifiers 1NN and SVM as baselines[2]. DYNG and OGNG process a continuous stream of data, while seeing each data point only once. Every 1000th example they perform a prediction for the next 1000 data points. In this way we test the accuracy on unseen data coming next in the stream.

While DYNG and OGNG are online learning algorithms, not requiring the explicit storage of data, a standard SVM learns in batch mode using all examples seen and the 1NN classifier is "trained" by simply remembering all examples. Therefore, we let both of the algorithms store up to 5000 already seen training examples and retrain after every 1000th example, while performing a prediction on the same data as DYNG and OGNG. Preliminary experiments showed that for 5000 training examples the number of stored vectors by the SVM[3] (support vectors and training data) is comparable to the number of prototypes produced by DYNG. In addition, we compare to a version of DYNG without assessing the classification performance in order to evaluate the impact of this component. We refer to this version as DYNG^{-imp}.

For DYNG and OGNG, we empirically determine parameter settings on a trial-and-error basis. For the Reuters corpus, the DYNG/OGNG parameters are set as follows: insertion parameter $\lambda = 300$; maximum age $a_{max} = 100$; adaptation parameter for winner $e_b = 0.1$; adaptation parameter for neighborhood $e_n = 0.0006$; error variable decrease $\alpha = 0.5$; error variable decrease $\beta = 0.0005$ and $\tau = 0.3$ (DYNG). We used the same settings on the Twitter data except for setting $\lambda = 30$. For both DYNG and OGNG we select the *relabel-method* as labeling strategy and *single-linkage* as prediction strategy (see [1]). For the SVM we use a linear kernel function.

3.2 Results

The results of our experiments are shown in Figure 1. The two figures show the classification accuracy for four learning approaches (DYNG, OGNG, SVM, 1NN) compared to a majority baseline over the number of data points seen for the Reuters and Twitter data set. The results show that DYNG clearly outperforms OGNG on both datasets, especially in the heavily non-stationary beginning phase where many new classes are encountered. In this first phase DYNG improves the accuracy of OGNG by up to 20.28% (5.37% on average) on the Reuters data set and up to 21.4% (20.95% on average) for the Twitter data set. It is striking that for the Reuters data set DYNG starts (Iteration 0-25) with an accuracy similar to 1NN and outperforms the SVM starting from

[2]We also experimented with the Huller online SVM [4] implementation of Sebastian Nowozin (`http://www.nowozin.net/sebastian/tu-berlin-2006/huller/`), extending it to a multi-class setting by training in one-vs-all mode. The Accuracies were inexplicably low and probably due to some bug in the SVM code, so that we do not report the results here.

[3]We use libSVM in our experiments: `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

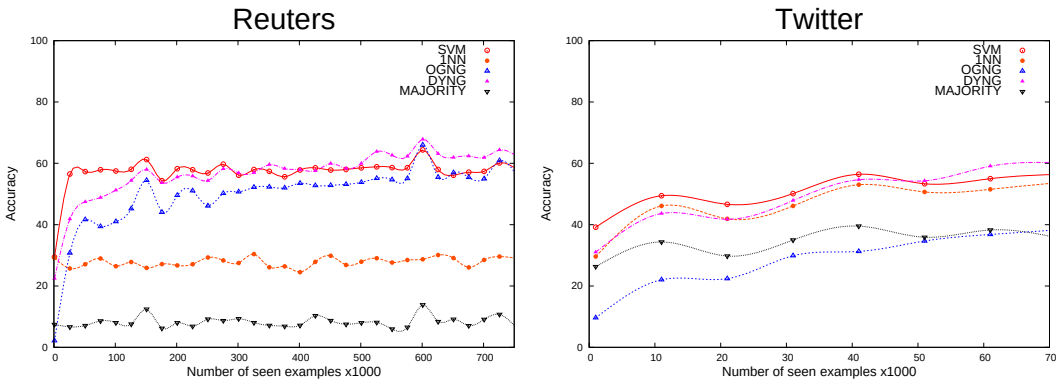

Fig. 1: Classification results of DYNG, OGNG, SVM and 1NN

iteration 450 on with an improved accuracy of up to 5.53% (3.79% on average). Thereby, DYNG stores on average around 6368 vectors (neurons) compared to 5000 vectors (training examples) of 1NN and 9599 vectors (support vectors and training examples) of the SVM. This shows the benefit of a continuous learning process.

On the Twitter data set, the DYNG, SVM and 1NN show similar results. DYNG outperforms the SVM starting from iteration 50 on with an improved accuracy of 4.04% (2.97% on average). Thereby, DYNG stores on average 9703 vectors compared to 9235 vectors of the SVM. The weak performance of OGNG shows the benefit of the neuron insertion strategies of DYNG, as OGNG is not designed to adapt to the huge amount of classes as quickly as DYNG.

An analysis of the neurons introduced by DYNG compared to DYNG^{-imp} on the Reuters dataset reveals that DYNG introduces on average 26.39% less neurons for the Reuters dataset and on average 33.05% less neurons for the Twitter dataset compared to DYNG^{-imp}, while having a similar classification performance. Having less neurons is a benefit as it reduces the network's complexity and speeds up the classification, so that we conclude that monitoring the classification performance on the fly is indeed a crucial element of DYNG.

4 Related Work

In recent years, there have been a number of GNG-based algorithms that incorporate label information, such as *Incremental GNG (iGNG)* [7], *Enhanced Self-organized Incremental Neural Network (ESOINN)* [8], *Online Growing Neural Gas (OGNG)* [1], *Online Semi-supervised Growing Neural Gas (OSSGNG)* [2] and *Semi-supervised Growing Neural Gas (SSGNG)* [12]. However, none of these approaches uses the label information in order to influence the clustering itself. Similar to *Supervised Growing Neural Gas (SGNG)* [10], DYNG incorporates the labels to adapt the neuron insertion strategy of GNG. In contrast to DYNG, SGNG does not provide a method to dampen the growth of the network, as it

ESANN 2013 proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning. Bruges (Belgium), 24-26 April 2013, i6doc.com publ., ISBN 978-2-87419-081-0.
Available from http://www.i6doc.com/en/livre/?GCOI=28001100131010.

simply stores the best network configuration according to an evaluation step, which considers the compactness, the impurity and the cluster separation of the network. This approach requires SGNG to store the training data explicitly, thus rendering SGNG unsuitable for life-long learning scenarios in which a non-ending stream of data needs to be processed. Other prototype-based online approaches such as *Category Learning Vector Quantization (cLVQ)* [6] and the online LVQ algorithm proposed by Bharitkar et al.[3] assume a small training set in order to perform an offline initialization or optimization of the network.

5 Conclusion

We have presented a new algorithm as an extension of OGNG, which incorporates the knowledge about labels in order to adapt its insertion strategy for neurons. We have successfully applied the algorithm on two stream data sets in a life-long learning scenario and shown that it improves upon an existing online classification algorithm based on GNG, i.e. OGNG, particularly outperforming it in highly non-stationary phases and even outperforming an SVM with a comparable amount of memory storage. Monitoring the change in classification error, while not affecting classification performance substantially, has turned out to contribute to minimize the number of neurons required, thus simplifying the network and fostering faster classification.

References

[1] Beyer O, Cimiano P. Online Labelling Strategies for Growing Neural Gas. *Proceedings of the International Conference on Intelligent Data Engineering and Automated Learning (IDEAL).*; 2011:76–83.

[2] Beyer O, Cimiano P. Online Semi-supervised Growing Neural Gas. *International journal of neural systems.* 2012;22(5).

[3] Bharitkar S, Filev D. An online learning vector quantization algorithm. *Int. Symposium on Signal Processing and its Applications* 2001; 394-397.

[4] Bordes A, Bottou L. The Huller: a simple and efficient online SVM. Machine Learning *Proc. of the European Conference on Machine Learning* 2005;505-512

[5] Fritzke B. A growing neural gas network learns topologies. *Advances in neural information processing systems.* 1995:625-632.

[6] Kirstein S, Wersing H, Gross H, Körner E. A vector quantization approach for life-long learning of categories. *Advances in Neural Information Processing Systems* 2009; 803-810.

[7] Prudent Y, Ennaji a. An incremental growing neural gas learns topologies. *Proc. of. IEEE International Joint Conference on Neural Networks.* 2005;(1):1211-1216.

[8] Furao S, Ogura T, Hasegawa O. An enhanced self-organizing incremental neural network for online unsupervised learning. *Neural networks* 2007;20(8):893-903.

[9] Gaber, M. M., Zaslavsky, A., Krishnaswamy, S. Mining data streams: a review. *ACM Sigmod Record.* 34(2), 18-26.

[10] Jirayusakul A, Auwatanamongkol S. A supervised growing neural gas algorithm for cluster analysis. *International Journal of Hybrid Intelligent Systems.* 2007;4(2):129-141.

[11] Lewis, D. D.; Yang, Y.; Rose, T.; and Li, F. RCV1: A New Benchmark Collection for Text Categorization Research. *Journal of Machine Learning Research.*2004;5:361–397

[12] Zaki SM, Yin H. A Semi-Supervised Learning Algorithm for Growing Neural Gas in Face Recognition. *Journal of Mathematical Modelling and Algorithms.* 2008;7(4):425-435.