Incremental classification of invoice documents

Hatem Hamza, Yolande Belaïd, Abdel Belaïd, Bidyut Baran Chaudhuri

Incremental classification of invoice documents

Hatem Hamza [*,**], Yolande Belaïd[**], Abdel Belaïd[**], Bidyut B. Chaudhuri [***]
[*]ITESOFT, France
[**]LORIA, University Nancy 2, France
[***]Indian Statistical Institue, Kolkata-108, India

Abstract

This paper deals with incremental classification and its particular application to invoice classification. An improved version of an already existant incremental neural network called IGNG (Incremental Growing Neural Gas) is used for this purpose . This neural network tries to cover the space of data by adding or deleting neurons as data is fed to the system. The improved version of the IGNG, called I2GNG used local thresholds in order to create or delete neurons. Applied on invoice documents represented with graphs, I2GNG shows a recognition rate of 97.63%.

1. Introduction

Incremental learning is a topic of major interest in machine learning. In [4] Giraud-Carrier defines incremental learning: 'a learning task is incremental if the training examples used to solve become available over time, usually one at a time'. Such learning is then used whenever a system cannot know in advance the whole data set it is going to process. Incremental learning is preferred to classical classification and clustering techniques whenever the number of classes can not be known in advance. Several applications using incremental learning can be found in the litterature ranging from image segmentation and classification [12] to document classification [9]. Many incremental approaches exist in the literature. Incremental K-means which is an adaptation of the well known K-means algorithm was proposed in [9] to detect novelties in online documents. This incremental K-means assigns an element to the class in which intra-class similarity is maximum. If intra-class similarity does not increase for any existing class, then this data can constitute a class of its own (or can be an outlier).

As done with classical classification techniques extended to incremental learning, neural networks have also been extended to incremental learning. For example, supervised neural networks have been proposed in [10]. Since we want to work with an unsupervised classifier, we shall not focus on such supervised algorithms. We will just present the non supervised incremental neural networks. Among the early workers first incremental neural networks were Growing Cell Structures (GCS [2]), followed by the Growing Neural Gas (GNG [3]). Then, many other variations were proposed on these two networks. The Hierarchical GNG (TreeGNG) is a network that builds classes over the classes given by the GNG. Similarly, the Hierarchical GCS (TreeGCS [8]) uses the same priniciple. IGNG is an improvement of GNG as its neuron creation process is more dependant to the variation in the data than the GNG. Other types of incremental neural networks are those which use self organizing maps. One has to make the difference between incremental neural networks which perform incremental learning (GNG, IGNG [11]) and incremental neural networks which are just incremental because they can add or remove neurons (GCS, GHSOM [1]) on a static database.

The final application of this work is to classify a database of invoices which size is increasing continuously. This database belongs to a system called CBRDIA [6]. This database has to be managed so that its access does not take too much time, and remains always accurate.

This paper is organized as follows: the second section describes IGNG and the improvements we propose, the third section presents our experiments on synthetic and real data. In the conclusion, we present some perspectives of this work.

2. Incremental Growing Neural Gas: overview and improvements

GNG [3] and IGNG[11] are incremental neural networks which can start from scratch and learn as data comes. They are dynamic networks as they add and/or remove neurons depending on the evolution of the data over the time. They have been successfully applied in incremental classification tasks (classification of images, of synthetic data) and one can expect such good results when applied in the graph classification domain. As far as we know, application of these algorithms to structured data (graphs, trees) has not been

978-1-4244-2175-6/08/$25.00 ©2008 IEEE

reported yet. Such an application requires the adaptation of the algorithm equations from the vector domain to the graph domain.

2.1. Incremental Growing Neural Gas Classifier

The general idea of the IGNG is the following:

- the network starts with one neuron (it can also start from more neurons).
- For every new data E, the nearest neurons are searched in the network. If the distance between E and these neurons is below a threshold S, then E belongs to the class of its nearest neuron. Otherwise, a new neuron is created at E.
- This neuron remains in an embryon state (not participating in the classification process, but participating though in the learning process) till its age (number of times it is excited by close data) becomes bigger than a threshold a_{neuron}.
- Similarly to the GNG, if a E is close to the network neurons, the two nearest neurons are updated and the edges between them are also updated.

This neural network IGNG suffers however from the choice of the threshold S. In the original paper [11], Prudent and al. proposed to initialize S at the standard deviation of the whole database on which classification is done. This is in our opinion contrary to the principles of incremental learning as we do not know $a\ priori$ which kind data is going to come later.

2.2. Improved IGNG: I2GNG

The first point on which we worked was to try to be freed from the choice of the threshold S. The first constraint is that the only information available at a time T is the information about the already processed data. Moreover, we cannot use the whole previous data to determine the class of the new data. The solution is to use some local information related to each neuron. Let:

- N be the number of IGNG neurons at T.
- E be an entry.
- m_i be the average distance between every element in a class i and its representative neuron n_i, σ_i the standard deviation of these distances.

It is logical to say that E belongs to a class i if $d(E, n_i) < m_i$, where n_i is the nearest neuron to E. In order to be more flexible, we propose that the threshold S becomes: $S = m_i + \alpha.\sigma_i$. By using this new threshold, the only class we are dealing with is the nearest one. Moreover, by taking into account the mean and standard deviation of this class, we are using intrinsic parameters related to this class, not to the whole data. Two cases typically occur:

- the new data is close enough to the nearest class (meaning $d(E, n_i) < m_i + \alpha.\sigma_i$), this data will belong to the class i and the neuron n_i is updated.
- the new data is too far from its nearest class. In this case, a new neuron is created (embryon neuron), and becomes effective in classification only if its age exceeds a_{neuron}.

Compared to the IGNG, I2GNG can capture better the variation of data. Two general examples can prove it:

- If the IGNG threshold S is chosen too big, two different classes can be regrouped with the same neuron. This is the case of figure 1.

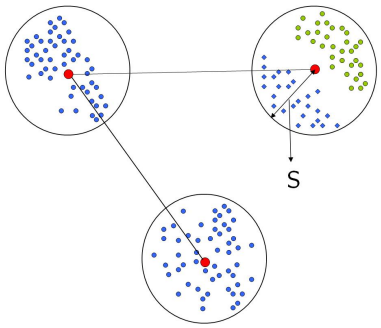

Figure 1. Problem with a big threshold

- If the IGNG threshold is chosen too small, then if a class is homogeneous but is too large, then it will be divided into many other classes. This is the case of the biggest class in figure 2.

Using a local threshold is then the solution to avoid such configurations.

2.3. Adaptation to the case of graph classification

We used edit distance to compare graphs. Many other distances exist and can be used [7]. As presented before, the IGNG was first created to deal with numerical data represented with vectors. This paper proposes to deal with structured data. The adaptation we propose in the following can be done for GNG, IGNG and I2GNG.

In [5], Bunke proposed a method of adapting Self Organizing Maps (SOM) to graph classification. He applied it

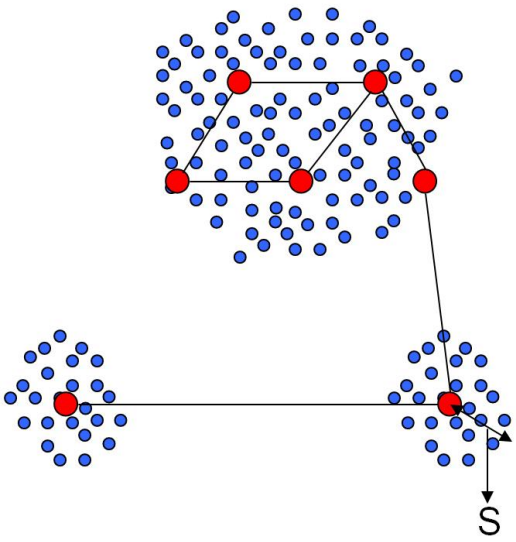

Figure 2. Problem with a small threshold

to classify handwritten digits. This method is based on the computation of the edit distance between graphs, and every formula in the SOM algorithm was then adapted based on the edit path between any two graphs. Here is a simple explanation of the idea: let G1 and G2 be two graphs. Let d(G1,G2) be the edit distance between G1 and G2. This distance corresponds to the cost of some additions, deletions or substitutions of nodes or/and edges which transform G1 into G2.

$$d(G1, G2) = \sum cost(edition)$$

In the vector domain, when the distance between a vector X and vector Y is d(X,Y), it is easy to transform X by $\epsilon(X, Y)$, ϵ being a real number. The same operation in the graph domain means that G1 has to be modified by $\beta = \epsilon \cdot d(G1, G2)$ (equivalent to $Neuron = \epsilon_b.(Neuron_{nearest} - entry)$). Modifying G1 by β means that we have to apply only β edit operations on G1. As we already know the edit path that allowed us to compute the distance between G1 and G2, then β corresponds just to a part of this edit path. In this way modifying G1 becomes an easy task as we just have to find the edit operations in which cost approaches β as much as possible. More elaborate details can be found in [5].

Adapting the I2GNG formulae using the principles cited above allows us to classify graphs or trees using I2GNG.

2.4. Adaptation of the I2GNG to the case of invoice classification

Incremental invoice classification belongs to the learning part of the system CBRDIA. Thus, we have to integrate some heuristics and information related to the invoice domain before starting the use of the improved IGNG. First of all, I2GNG in CBRDIA is not initialized with random classes. In order to enhance the efficiency of the classification, the starting classes should be as different as possible. Moreover, one can even start with more than only two classes (as done in the classical GNG) if this information is available.

In invoice processing systems, some very rare cases of invoices can be processed from time to time. Theses cases have also to be learnt. If they are similar to invoices existing already in the database, then they are just added to the class of these invoices. Otherwise, these rare cases can form new neurons. These neurons can remain always as embryo neurons as they are very rare, but they can also become mature and participate in the classification process. One solution to avoid having them in the classification process is to not consider them at all during the classification if we know in advance that these rare cases will not occur at all. This step of deleting temporarily one or more nodes in the I2GNG can be done by a user (contrarily to the automatic deletion that happens in GNG and IGNG).

3. Experiments

3.1. Tests on the MNIST database

Another experiment was performed on the MNIST database. In this experiment, the learning examples were given to the I2GNG progressively, and tests were performed after each 10000 images. The distance used to compute the similarity between each pair of images is the Euclidian distance. The results are shown in table 1.

samples	recognition
10000	88.45%
20000	91.02%
30000	92.58%
40000	93.66%
50000	94.06%
60000	94.29%

Table 1. I2GNG results on the MNIST database.

The obtained results are far from the best results obtained on the MNIST data. However, one should notice that these results were obtained after one single pass of the data. The closest work to ours on the MNIST is a work done by Wilder using K nearest neighbours classifier, after preprocessing the database images. The error rate of this work reached 1.22%.

With similar pre-processing steps, we can expect identical accuracy with the I2GNG.

3.2. Experiments on document classification

Our experiments are performed on a dataset of real documents (invoices) taken from a real invoice processing chain. Every invoice is modeled with its graph and then given to the I2GNG. The graphs consist in a set of keywords which are the graphs nodes, whereas the edges represent the spatial relationships between these keywords. The keywords are extracted from the documents via matching with a predefined dictionary of words.

The dataset is divided in two parts: a learning set (324 documents) and a testing set (169 documents). 8 classes of invoices are used for this purpose. We chose this strategy of I2GNG evaluation as the learning procedure helps in knowing about the incremental capabilities of the I2GNG applied to graphs, whereas the testing phase helps knowing about its classification properties.

2 different series of tests were performed. The first one studies the influence of the threshold age of neurons (above which neurons become mature) with $\alpha = 2.5$. The second one studies the influence of α with $a_{neuron} = 10$. The results are shown in table 2.

a_{neuron}	neurons	rec	α	neurons	rec
10	14	99.40%	0.5	10	98.22%
20	18	97.63%	1	15	98.22%
30	18	97.63%	1.5	12	98.81%
40	16	98.22%	2	14	98.81%
50	16	98.22%	2.5	12	99.40%
60	16	98.22%	3	18	97.63%

Table 2. Influence of α and a_{neuron}

We can notice from these tables that the number of neurons is always greater than the number of classes (8). This is due to the variations that are present in these classes. Representing one class with several neurons is not a problem so long as every neuron represents homogeneous data. In the training process, we tag manually the obtained neurons (by giving them the name of the class they represent).

The obtained results mean that the I2GNG is working well. As shown in table 2, the bigger α is, the more neurons we obtain. This can be explained as the following:

- when α is big, the threshold $(m+\alpha.\sigma)$ for each neuron is also big. A new class is created if and only if it is outside the 'range' of an existing neuron.

- when neurons are quite far one from the other (because of a big α, their ages increase quickly, and they become mature quickly too. On the other side, when α is small, neurons are very close to each other. One class can be represented by a large number of neurons, a few of which become mature because of the high competition among classes.

4. Conclusion

We presented in this paper two improvements of the incremental growing neural gas. These modification are related to the threshold of creation of a new neuron, and to its application on graphs of documents. Some work still needs to be done. First, we need to test this approach on a bigger set of documents. Moreover, we need to apply it on other types of data to test its versality. Some theoretical points need also to be studied. For example, the choice of the maximum age of the edges, as well as the age of maturation of a neuron need to be divided automatically from the data itself.

References

[1] M. Dittenbach, D. Merkl, and A. Rauber. Hierarchical clustering of document archives with the growing hierarchical self-organizing map. In *ICANN*, pages 500–508, 2001.

[2] B. Fritzke. Growing cell structures–a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9):1441–1460, 1994.

[3] B. Fritzke. A growing neural gas network learns topologies. In *NIPS*, pages 625–632, 1994.

[4] C. G. Giraud-Carrier. A note on the utility of incremental learning. *AI Commun.*, 13(4):215–224, 2000.

[5] S. Günter and H. Bunke. Self-organizing map for clustering in the graph domain. *Pattern Recognition Letters*, 23(4):405–417, 2002.

[6] H. Hamza, Y. Belaïd, and A. Belaïd. Case-based reasoning for invoice analysis and recognition. In *ICCBR*, pages 404–418, 2007.

[7] H.Bunke, P. Foggia, C. Guidobaldi, and M. Vento. Graph clustering using the weighted minimum common supergraph. In *GbRPR*, pages 235–246, 2003.

[8] V. J. Hodge and J. Austin. Hierarchical growing cell structures: Treegcs. *Knowledge and Data Engineering*, 13(2):207–218, 2001.

[9] S. Khy, Y. Ishikawa, and H. Kitagawa. Novelty-based incremental document clustering for on-line documents. In *ICDE Workshops*, page 40, 2006.

[10] R. Polikar, L. Upda, S. S. Upda, and V. Honavar. Learn++: an incremental learning algorithm for supervised neural networks. *IEEE Transactions on Systems, Man, and Cybernetics, Part C*, 31(4):497–508, 2001.

[11] Y. Prudent and A. Ennaji. A new learning algorithm for incremental self-organizing maps. In *ESANN*, pages 7–12, 2005.

[12] A. Vailaya and A. K. Jain. Incremental learning for bayesian classification of images. In *ICIP (2)*, pages 585–589, 1999.