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.

Cost-Sensitive Cascade Graph Neural Networks

Nguyen Van Tuc[2] and Ah Chung Tsoi[1] and Markus Hagenbuchner[2] *

1- Macau University of Science and Technology
Macau, China
2- University of Wollongong
NSW 2500, Australia

Abstract. This paper introduces a novel cost sensitive weighted samples approach to a cascade of Graph Neural Networks for learning from imbalanced data in the graph structured input domain. This is shown to be very effective in addressing the effects of imbalanced data distribution on learning systems. The proposed idea is based on a weighting mechanism which forces the network to encode misclassified graphs (or nodes) more strongly. We evaluate the approach through an application to the well known Web spam detection problem, and demonstrate that the generalization performance is improved as a result. Indeed the results obtained reported in this paper are the best reported so far for both datasets.

1 Introduction

A common property of challenging learning problems is the severely imbalanced nature of data. The heavily skewed distribution of one pattern class versus another pattern class in a training dataset can cause the learning algorithm to bias towards the majority class. For instance, the number of fraudulent actions in ATM cash withdrawals is far less than that of the valid ones. Thus, particular attention needs to be paid to the fraudulent transactions samples in a classification model to ensure "unbiased" classifications.

The sensitivity of learning systems to imbalanced data is a desired property. It allows such systems to be insensitive to noise under the assumption that the noisy information is a minority in the feature space. However, there are numerous situations in which one or more classes of training patterns are a minority in the feature space. It is necessary in such cases to alter the training algorithm so as to account for the imbalanced nature of pattern classes in a training dataset. A number of methods for dealing with imbalanced data have already been developed. These include: the bagging or bootstrap sampling methods [1], the boosting method [2], the non-uniform cost function method [3, 4].

This paper adopts a cost-sensitive weighted samples approach, based on an extension of the approach in [4], to deal with the data imbalance issue in a supervised learning model which is defined for the graph input domain. Traditional approaches to learning systems are limited to the encoding of feature vectors and sequences. A different approach is taken by Graph Neural Networks (GNNs) which can encode general types of labeled graphs [5]. This relatively new method has already been applied very successfully in number of classification and regression learning problems involving graphs [6]. The learning algorithm of the GNN is based on a gradient descent method not unlike the

*Supported by the Australian Research Council in the form of a Discovery Project Grant DP110103376 and Fundo para o Desenvolvimento das Ciências e da Tecnologia, Macau SAR, Project Grant number: 059/2010/A3.

one used in the training of Multilayer Perceptrons (MLPs). Recent work have shown that several GNNs organized in layers are more robust than a single GNN on graph learning problems [7, 8].

Based on the idea of non-uniformed weighting mechanisms [4] and layered GNN architectures [7], this paper proposes a novel Cost Sensitive Cascade GNN (CS-GNN), and shows that this is very effective in dealing with imbalanced data learning problems involving graphs. A biased cost matrix is constructed based on the output of each GNN layer. The cost matrix is then utilized in a consecutive GNN layer so that misclassified patterns are weighted more strongly. The proposed method is then applied to two well-known benchmark datasets which feature severely imbalanced pattern classes [9, 10]. The obtained results show convincingly the effectiveness of the proposed approach.

The rest of this paper is organized as follows: Section 2 briefly describes the GNN learning model. Section 3 offers a detailed explanation of the CS-GNN. Experimental evaluations are given in Section 4. Finally, some conclusions are drawn in Section 5.

2 Graph Neural Networks

Graphs are a data structure which model the dependencies between any pair of atomic data elements. Hence, graphs are particularly useful for learning problems which feature relationships among the atomic data elements. A very large number of practical applications can be represented as a graph. For example, document categorization problems can benefit from the document structure explicitly modelled, or in activity prediction tasks in QSAR (quantitative-structure activity relationship) with the availability of the molecular structures.

Recently a new supervised neural network known as the GNN [5] has been proposed. The GNN has been very successfully applied to a number of real world problems involving graphs [6]. The GNN can encode any type of graphs including directed, undirected, ordered, non-positional, edge-labeled, and node labeled graphs.

In the following we denote a graph $\mathcal{G} = (N, E)$ where N is the set of nodes and E is the set of edges. If $n \in N$ denotes a node, then $l_n, x_n, ne_n, l_{ne}, x_{ne}$ are the label, state, neighboring nodes, the labels of neighbors and the states of neighbors respectively. To compute the states, the GNN engages a *local transition function* h_w as follows:

$$x_n = \sum_{u \in ne_n} h_w(x_u, l_u, l_n) \tag{1}$$

Thus, the states can be computed as all other quantities on the right hand side of Eq 1 are given. Note that Eq. 1 is a recursive function. The GNN computes the *stable state* of x_n by recursively executing Eq. 1 and through carefully designing h_w which incorporates a contraction mapping idea to force the convergence of Eq 1 (see [6] for details). Note that x_n is a consolidation of information that a system has about a node and its neighbors in a graph. Thus, a GNN encodes graph data structures by processing each node via the recursive function shown in Eq. 1. The network output o_n can be computed as follows:

$$o_n = g_w(x_n, l_n), \tag{2}$$

where g_w is said to be an *output function*. The GNN realizes both, h_w and g_w by an MLP so that its learning algorithm can adopt the back-propagation method for computing the network weights w [6]. Assuming that the learning data is denoted as $\mathcal{L} = (\mathcal{G}, n_i, t_i), i = 1, \ldots, p$, where t_i is the desired network output of node n_i and p is the number of nodes in a given graph $\mathcal{G}$, then the learning process aims at minimizing the cost function $E = \sum_{i=1}^{p}(t_i - o(n_i))^2$.

The learning algorithm involves two steps [6]: in the feed-forward phase, all the nodes' states are calculated recursively to obtain the stable point of Eq. 1 using the Almeida-Pineda algorithm [11]. Then, the network output is computed according to Eq. 2. The cost function is evaluated using the predicted outputs and the targets for all nodes. In the back-propagation phase, the gradient $\frac{\partial E}{\partial w}$ is computed in order to update the weights w by following the well-known gradient descent approach.

A composite model that encompasses several GNNs is called a Layered GNNs architecture [7]. In this case the GNNs are stacked so that the output of one GNN is utilized to re-label the nodes of the input graphs before training the next GNN in the stack. This model was shown to outperform a single GNN [7].

3 Cost-Sensitive cascade GNNs

Cost-sensitive classification assumes different costs (or penalties) when samples are misclassified [3, 4]. For example, in a two-class classification problem, we can use $+1$ and -1 to denote the positive class and the negative class respectively, and N^+ and N^- to denote the number of positive and negative samples respectively. The data is said to be unbalanced if $N^+ \ll N^-$. Moreover, it may be known that the $+1$ samples are more important than the -1 ones. For example, in the web spam detection problem, there are only a few spam hosts and there are many normal hosts. The requirement may be to capture as many spam hosts as possible even if this should come at the cost of some misclassified normal hosts. Thus the cost w^+ associated with misclassifying $+1$ samples should be much higher than the cost w^- of incorrectly classifying the -1 samples, and no cost is associated with correctly classified samples. Our novel algorithm for finding new weights in the context of cascade GNNs is shown in Algorithm 1.

In the algorithm, we optimize the threshold of a F_1 measure to balance the precision (P) and recall (R) values of the retrieval performance. The F_1 measure is defined as the harmonic means of P and R. The optimization of the threshold T could be formulated as a support vector machine formulation [12]. However in our implementation, we used an initial condition of weighing the ratio of the weights associated with the positive samples and the negative samples by the inverse ratio of the number of positive samples N^+ to the negative samples N^-, and then use a greedy algorithm to find the approximate threshold T which maximizes the F_1 measure. The CS-GNN algorithm uses a graph $\mathcal{G}_0$ as input. A GNN_0 is trained on $\mathcal{G}_0$ and an output GNN_0 is obtained. The output is used to compute a cost-matrix of all the nodes and to re-label the nodes to obtain a modified graph $\mathcal{G}_1$, which is then used to train a GNN_1, and so on. Thus, the cost-matrix is constructed for each GNN layer. The node with the corresponding higher cost value will be associated with the error at the output layer of the GNN. The approach forces the GNNs to gradually improve on the residual classification errors. The algorithm terminates when no further improvement can be obtained. Hence, the approach optimizes the classification rate which results in a general improvement of the classification performance.

Algorithm 1 CS-GNNs

```
Input:
$k = 1$, $GNN_k$ denotes the $k$-the GNN trained on the $\mathcal{G}_k$-the graph.
$N$ nodes of the original graph $\mathcal{G}_1$. Nodes may be labeled by feature vectors.
Costs of nodes $w_i, i = 1, 2...N$.
Training target of node $n_i$: $Targ_i = 1$ if $n_i \in$ +1 class, $Targ_i = 0$ if $n_i \in$ -1 class.
$PC$, a constant weighting the positive class.
$NC$, a constant weighting the negative class.

Train $GNN_1$ on $\mathcal{G}_1$ to obtain prediction values $Pred_i \in \mathcal{R}, i = 1, 2...N$
repeat
  Build a new graph $\mathcal{G}_{k+1}$ with nodes labeled by $Pred$
  Compute threshold $T$ that maximize $F_1$ on the training dataset.
  for $i = 1$ to $N$ do
    if $(Targ_i = 1)$ AND $(Pred_i < T)$ then
      $w_i = |Targ_i - Pred_i| \times PC$
    else
      if $(Targ_i = 0)$ AND $(Pred_i > T)$ then
        $w_i = |Targ_i - Pred_i| \times NC$
      end if
    else
      $w_i = 1$
    end if
  end for
  Train $GNN_{k+1}$ with the cost-error driven approach represented in error function
  $E = \sum_{i=1}^{N}[(Targ_i - Pred_i) \times w_i]^2$
  Update prediction values $Pred$
until $K$ specified maximum training time is reached or no improvement is observed on training dataset.
```

4 Experimental Evaluation

Web spam detection problems are known to be one of the most challenging real-world machine learning problems. This paper uses the UK2006 [9] and UK2007 [10] web spam datasets to evaluate the proposed algorithm. Table 1 presents the information on the two datasets. The number of spam hosts account for only 25.8% and 0.5% of all labelled hosts in the UK2006 and UK2007 datasets respectively. It is known that the classification of a host depends on the content of the host or on the hyperlink structure or both. The dataset provides a 275-dimensional feature vector describing each host. There are 96 content-based, 41 raw link-based and 138 transformed link-based features [9, 10], as well as the hyperlinks between the hosts. We use all available information during the experiments.

Table 1: Detailed information of the two web spam detection datasets

Properties	UK2006	UK2007
Number of hosts	11,402	114,529
Labeled hosts	7,473	6,479
Training set	5,622	4,275
Test set	1,851	2,204
Number of hyperlinks	730,774	1,885,820
Average links per node/host	64	16

The key evaluation method used is the Area under the receiver operation (ROC) curve (AUC). The AUC reflects the probability that a learning model assesses a ran-

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.

Table 2: The best training and corresponding testing AUC performance on the UK2006 and UK2007 datasets.

Features	**Models**	**UK2006**		**UK2007**	
		TrainAUC	TestAUC	TrainAUC	TestAUC
L_1 Regular	**CS-GNNs**	**0.975**	**0.956**	**0.821**	**0.798**
(100 features)	Layered GNNs	0.972	0.951	0.819	0.787
	GNN	0.970	0.948	0.816	0.795
	MLPs	0.971	0.911	0.789	0.758
L_1 feature+PM-GraphSOM output	**CS-GNNs**	**0.982**	**0.965**	**0.922**	**0.853**
(102 features)	Layered GNNs	0.974	0.956	0.900	0.846
	GNN	0.973	0.951	0.899	0.847
	MLPs	0.969	0.913	0.855	0.792
Content+RawLink	CS-GNNs	0.969	0.940	0.801	0.789
(137 features)	Layered GNNs	0.959	0.934	0.768	0.769
	GNN	0.956	0.925	0.763	0.779
	MLPs	0.965	0.879	0.722	0.743
Content+TranLink	CS-GNNs	0.981	0.945	0.742	0.763
(234 features)	Layered GNNs	0.976	0.940	0.736	0.740
	GNN	0.971	0.936	0.734	0.738
	MLPs	0.979	0.900	0.752	0.736

domly chosen positive sample higher than a randomly selected negative one.

Intuitively, not all the features are useful in the classification task, as a result, we applied the L_1 regularization method which is considered a continuous function approach for feature selection [13]. We apply this method in a cross validation approach to reduce the 275 features to ≈ 100 features.

We conducted three sets of experiments by using three different sets of input features, namely Content + Raw link based features, Content + Transformed link based features, and L_1 features and the results are as shown in Table 2. In Algorithm 1, the parameters PC and NC are set to 10 and 3 respectively, and spam nodes are referred to as +1 samples, and normal hosts are referred to as -1 samples. It is observed that the results show that the CS-GNN has better generalization performance than those of MLP, or a single stage GNN, using the cost sensitive weighting as proposed in this paper. However, when we compare the results with those of the winning entries respectively in the UK2006 and the UK2007 competition, it is noted that our proposed algorithm is competitive on the UK2006 dataset – the winning entry [14] obtained AUC = 0.956, using some additional knowledge about the given set of hosts –, but it falls well short of the winning entry of the UK2007 dataset – the winning entry [10] by Geng et al used a balancing approach with C4.5 base learner, obtained AUC = 0.848.

We surmised that this might be due to the long term dependency issue [15] in the training algorithm. Consequently, we deploy a learning algorithm involving an unsupervised learning frontend – PM-GraphSOM [16] followed by GNN [8] and then extended it to be followed by GNNs as proposed in this paper and re-run the L_1 regularization [13] to select the number of features to be deployed [1]. It is observed in this case, together with the proposed algorithm, the CS-GNNs obtains the best AUC result (AUC = 0.853, so far. Thus, a side benefit of this work is to show that deploying GNN-type al-

[1]The PM-GraphSOM is trained with L_1 feature and graph topology.

gorithm on the UK2006 dataset, the effect of long term dependency is less pronounced than that of the UK2007 dataset; an observation which was never made previously.

5 Conclusion

In this paper, we have introduced a novel learning architecture that relies on the ideas of cost-sensitive learning together with a cascade style arrangement of GNNs. The model is capable of dealing with imbalanced data as it applies a non-uniform distribution on the weight matrix. The samples of important misclassified classes are weighted more strongly. The effectiveness of this approach has been demonstrated on two challenging benchmark datasets, viz., the web spam detection problems. For future research, we believe that a boosting approach to GNNs could further enhance the GNN's abilities on learning problems. Applications of the proposed algorithm to other real-world problems can also be considered.

References

[1] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.

[2] R. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 1990.

[3] H. Schwenk and Y. Bengio, "Boosting neural networks." *Neural Computation*, vol. 12, no. 8, pp. 1869–1887, 2000.

[4] M. Kukar and I. Kononenko, "Cost-sensitive learning with neural networks," in *Proceedings of the 13th European conference on artificial intelligence (ECAI-98)*, 1998, pp. 445–449.

[5] F. Scarselli, M. Gori, A. Tsoi, M. Hagenbuchner, and G. Monfardini, "Computational capabilities of graph neural networks," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 81–102, January 2009.

[6] ——, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, no. 1, pp. 61–80, January 2009.

[7] N. Bandinelli, M. Bianchini, and F. Scarselli, "Learning long-term dependencies using layered graph neural networks," in *The 2010 International Joint Conference on Neural Networks*, 2010, pp. 1–8.

[8] L. D. Noi, M. Hagenbuchner, F. Scarselli, and A. C. Tsoi, "Web spam detection by probability mapping graphsoms and graph neural networks," in *The 2010 International Conference on Artificial Neural Networks-ICANN*. Springer, 2010, pp. 372–381.

[9] "Web spam challenge," http://webspam.lip6.fr/, 2007.

[10] L. Denoyer, "The 2008 web spam challenge," http://webspam.lip6.fr/wiki/-pmwiki.php?n=Main.PhaseIIIResults, 2008.

[11] L. Almeida, "A learning rule for asynchronous perceptrons with feedback in a combinatorial environment," in *IEEE International Conference on Neural Networks*, M. Caudill and C. Butler, Eds., vol. 2. San Diego, 1987: IEEE, New York, 1987, pp. 609–618.

[12] D. R. Musicant, V. Kumar, and O. A., "Optimizing f-measure with support vector machines," in *Proceedings, Florida Artificial Intelligence Research Society Conference. St. Augustine, Florida*, May 2003, pp. 356–360.

[13] T. Hastie, R. Tibshirani, and J. H. Friedman, *The Elements of Statistical Learning*. Springer, August 2001.

[14] G. Cormack, "Content-based web spam detection," in *Proceedings of the 3rd International Workshop on Adversarial Information Retrieval on the Web (AIRWeb)*, 2007.

[15] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Transactions on Neural Networks*, vol. 5, no. 2, pp. 157–166, March 1994.

[16] M. Hagenbuchner, S. Zhang, A. Tsoi, and A. Sperduti, "Projection of undirected and nonpositional graphs using self organizing maps," in *European symposium on Artificial Neural Networks*, Bruges, Belgium, April 2009, pp. 22–24.