Improved Multilabel Classification with Neural Networks

Rafał Grodzicki[1], Jacek Mańdziuk[1], and Lipo Wang[2]

[1] Faculty of Mathematics and Information Science
Warsaw University of Technology
Plac Politechniki 1, 00-661 Warszawa, Poland
{R.Grodzicki,J.Mandziuk}@mini.pw.edu.pl
[2] School of Electrical and Electronic Engineering
Nanyang Technological University
Block S1, Nanyang Avenue, Singapore 639798
elpwang@ntu.edu.sg

Abstract. This paper considers the multilabel classification problem, which is a generalization of traditional two-class or multi-class classification problem. In multilabel classification a set of labels (categories) is given and each training instance is associated with a subset of this label-set. The task is to output the appropriate subset of labels (generally of unknown size) for a given, unknown testing instance. Some improvements to the existing neural network multilabel classification algorithm, named BP-MLL, are proposed here. The modifications concern the form of the global error function used in BP-MLL. The modified classification system is tested in the domain of functional genomics, on the yeast genome data set. Experimental results show that proposed modifications visibly improve the performance of the neural network based multilabel classifier. The results are statistically significant.

Keywords: multilabel, learning system, neural network, backpropagation, bioinformatics, functional genomics.

1 Introduction

Multilabel classification is a generalization of traditional two-class or multi-class classification. In both cases a finite set of labels (categories) is given, but unlike in the latter case, where the task is to associate each problem instance with one category, the multilabel classification associates each instance with a subset of the set of labels. In other words, a multilabel classifier transforms the domain of instances X to the power set of the set of labels 2^Y:

$$h: X \rightarrow 2^Y \tag{1}$$

where $X \subseteq R^d$ denotes the set of instances and $Y = \{0, 1, \ldots, Q-1\}$ represents the set of possible labels. In many practical situations the mulitilabel classification problem is converted to the problem of defining a function $f: X \times Y \rightarrow R$ such that for any $x_p \in X$

$$f(x_p, y_1) > f(x_p, y_2) \tag{2}$$

G. Rudolph et al. (Eds.): PPSN X, LNCS 5199, pp. 409–416, 2008.

for all $y_1 \in Y_p$ and $y_2 \notin Y_p$. In other words instead of defining the classification of the form (1) it is sufficient to find function $f(.,.)$ which provided higher outputs for the elements belonging to Y_p than for those not belonging to Y_p, where (x_p, Y_p) is a training (testing) instance.

Many real-world problems can be modeled by multilabel classification systems. The most popular application domains include text categorization [6], [7] and bioinformatics [3], in particular a functional genomics area. This latter problem is considered in this paper in order to verify the efficacy of proposed modifications.

Our work is inspired by Min-Ling Zhang and Zhi-Hua Zhou's paper [1], where a neural network based method is proposed as an approach to multilabel classification problem in the domains of functional genomics and text categorization. Up to our knowledge, the paper [1] is the first attempt to apply neural networks to multilabel classification. Previous approaches include using decision trees [4], [11] and kernel methods [3], [8]. In the case of single-label classification problem, current studies are based on kernel methods. However, experimental results presented in [1] indicate that in the case of multilabel classification, neural network based method [1] outperforms kernel method proposed in [3]. So we decided to consider neural network approach.

The paper is organized as follows: in the next section a brief description of the neural network based multilabel classifier introduced in [1] is presented together with some modifications to the global error function proposed in this paper. The most popular performance measures that are applied in this paper are also introduced. Experimental results in the domain of functional genomics are presented and discussed in Section 3. The last section is devoted to conclusions and possible future work.

2 Neural Networks in Multilabel Classification

The simplest approach to solve the multilabel classification problem is its decomposition into multiple set of classification problems – one for each label. This solution, however, has a significant disadvantage – it does not take into account dependencies between different categories. Hence a different approach need to be employed. One of the candidate algorithms is the well-known BackPropagation (BP) learning method [9], [10] which, after appropriate adaptation to the multilabel classification case, can be used to solve the problem. This idea was exploited in [1], where the algorithm named BP-MLL (Backpropagation for Multilabel Learning) was developed and experimentally verified.

2.1 A Brief Description of BP-MLL

BP-MLL is applied in [1] to a multilayer perceptron with sigmoidal neurons with one hidden layer and additional biases from the input and hidden layer. The size of the input layer is equal to the instance domain dimension (plus a bias neuron). The size of the output layer equals the number of labels (i.e. Q). Training is based on the classical BP algorithm, but in order to address the dependencies between labels, the new global error function of the following form is proposed:

$$E_1 = \sum_{p=1}^{m} \frac{\sum_{(r,s)\in Y_p \times \overline{Y}_p} e^{-\left(c_r^p - c_s^p\right)}}{\left|Y_p\right|\left|\overline{Y}_p\right|}$$
$$h(x_p) = \left\{q \in Y : c_q(x_p) > t(x_p)\right\}, \quad c_q(x_p) = c_q^p \tag{3}$$

where m is the number of learning pairs, $Y_p \subseteq Y = \{0, 1, \ldots, Q\text{-}1\}$ is the set of labels associated with the *p-th* training instance, c_q^p (named *rank value*) is the actual output value of the *q-th* output neuron (corresponding to the *q-th* label), $\overline{Y_p}$ denotes the complementary set of Y_p (i.e. $\overline{Y_p} = Y \setminus Y_p$) and $h(x_p)$ denotes the set of labels attached to x_p by the network. Minimizing (3) tends to get higher output values by neurons corresponding to the labels belonging to Y_p than those not belonging to Y_p.

The next step to achieve multilabel classifier is determining the set of labels belonging to the input instance. This information can be retrieved from the neural network output values (rank values) by means of the threshold function which depends on the input vector. If the output neuron value is higher than the threshold value, then corresponding label belongs to the input instance. Otherwise, the label does not belong to the instance. More detailed description of BP-MLL can be found in [1].

2.2 Error Function Modifications

In this paper we propose some improvements of the error function used in [1]. The first introduced modification is integration of the threshold value into the error function used in BP-MLL. It results in the following form of the error function:

$$E_2 = \sum_{p=1}^{m} \frac{\sum_{(r,s)\in Y_p \times \overline{Y}_p} e^{-\left(c_r^p - c_s^p\right)} + \sum_{r\in Y_p} e^{-\left(c_r^p - c_Q^p\right)} + \sum_{s\in \overline{Y}_p} e^{-\left(c_Q^p - c_s^p\right)}}{\left|Y_p\right|\left|\overline{Y}_p\right| + \left|Y_p\right| + \left|\overline{Y}_p\right|}$$
$$h(x_p) = \left\{q \in Y : c_q(x_p) > c_Q(x_p)\right\}, \quad c_q(x_p) = c_q^p \tag{4}$$

The last output neuron's value (c_Q^p) is interpreted as the threshold. The meaning of the remaining output neurons is the same as in case of using the BP-MLL method. Proposed solution allows to determine the threshold value by adaptation during neural network learning. Hence, unlike in the method described in [1], additional step devoted to definition of the threshold function is not required.

The above error function (4) can be further generalized (and the whole process becomes more autonomous) by introducing independent thresholds for different labels:

$$E_3 = \sum_{p=1}^{m} \frac{\sum_{(r,s)\in Y_p \times \overline{Y}_p} e^{-\left(c_{2r}^p - c_{2s}^p\right)} + \sum_{r\in Y_p} e^{-\left(c_{2r}^p - c_{2r+1}^p\right)} + \sum_{s\in \overline{Y}_p} e^{-\left(c_{2s+1}^p - c_{2s}^p\right)}}{\left|Y_p\right|\left|\overline{Y}_p\right| + \left|Y_p\right| + \left|\overline{Y}_p\right|}$$
$$h(x_p) = \left\{q \in Y : c_{2q}(x_p) > c_{2q+1}(x_p)\right\}, \quad c_q(x_p) = c_q^p \tag{5}$$

In the case of equation (5) two output neurons (indexed by *2q* and *2q+1*) per each category *q* are considered. The first one of them (number *2q*) represents the output of the respective category (label) like in (3), and the other one (number *2q+1*) defines the respective threshold value for the *q-th* category.

Finally, in the error function comparisons between all the rank values of categories belonging to Y_p ($\overline{Y_p}$ resp.) and their respective threshold values can be taken into account, which leads to the following equations of the error function (6) and (7):

$$E_4 = \sum_{p=1}^{m} \frac{\sum_{(r,s)\in Y_p\times\overline{Y}_p} e^{-\left(c^p_{2r}-c^p_{2s}\right)} + \sum_{r\in Y_p}\sum_{t\in Y_p} e^{-\left(c^p_{2r}-c^p_{2t+1}\right)} + \sum_{s\in\overline{Y}_p}\sum_{t\in\overline{Y}_p} e^{-\left(c^p_{2t+1}-c^p_{2s}\right)}}{\left|Y_p\right|\left|\overline{Y}_p\right| + \left|Y_p\right|^2 + \left|\overline{Y}_p\right|^2}$$
$$h\left(x_p\right) = \left\{q\in Y : c_{2q}\left(x_p\right) > c_{2q+1}\left(x_p\right)\right\}, \quad c_q\left(x_p\right) = c^p_q \tag{6}$$

$$E_5 = \sum_{p=1}^{m} \frac{\sum_{(r,s)\in Y_p\times\overline{Y}_p} \left(e^{-\left(c^p_{2r}-c^p_{2s}\right)} + e^{-\left(c^p_{2s+1}-c^p_{2r+1}\right)}\right) + \sum_{r\in Y_p}\sum_{t\in Y_p} e^{-\left(c^p_{2r}-c^p_{2t+1}\right)} + \sum_{s\in\overline{Y}_p}\sum_{t\in\overline{Y}_p} e^{-\left(c^p_{2t+1}-c^p_{2s}\right)}}{2\left|Y_p\right|\left|\overline{Y}_p\right| + \left|Y_p\right|^2 + \left|\overline{Y}_p\right|^2}$$
$$h\left(x_p\right) = \left\{q\in Y : c_{2q}\left(x_p\right) > c_{2q+1}\left(x_p\right)\right\}, \quad c_q\left(x_p\right) = c^p_q \tag{7}$$

Note that (6) and (7) differ from (5), where each rank value is compared only with one threshold assigned to it. In (6) and (7) each rank value of category belonging to Y_p ($\overline{Y_p}$ resp.) is compared with each threshold value of category belonging to Y_p ($\overline{Y_p}$ resp.) Moreover, (7) extends (6) by also considering differences between threshold values (cf. the first terms in the numerators of both equations). In effect minimization of (7) results in lower threshold values corresponding to labels belonging to Y_p , for $p = 1, \ldots, m,$ than to those not belonging to this set.

2.3 Evaluation Metrics

Before presentation of experimental results let us briefly introduce the most popular error measures used in multilabel classification domain [1], [3]. The three of them, namely the *Hamming loss*, the *one-error* and the *ranking loss* are considered in this paper.

The *Hamming loss* measure (8) indicates the frequency (with respect to the size of the testing set *K*) of incorrect classification (the instance is classified as associated with particular label when it is actually not the case or *vice-versa* the instance is not classified as associated with this label in case it should be).

$$hloss(h) = \frac{1}{K}\sum_{p=1}^{K}\frac{1}{Q}\left|h\left(x_p\right)\Delta Y_p\right|,$$
$$h\left(x_p\right)\Delta Y_p = \left(h\left(x_p\right)\cup Y_p\right)\setminus\left(h\left(x_p\right)\cap Y_p\right) \tag{8}$$

The *one-error* measure (9) points out how often the label with the highest rank value (the top-one) does not belong to Y_p. Function *f* in (9) is defined by equation (2).

$$one-error(f)=\frac{1}{K}\sum_{p=1}^{K}\left[\left(\arg\max_{y\in Y} f(x_p,y)\right)\notin Y_p\right] \tag{9}$$

The third error measure, the *ranking loss* (10) indicates how often the label belonging to Y_p has got lower or equal rank value than the one not belonging to Y_p (which is not the expected outcome).

$$rloss(f)=\frac{1}{K}\sum_{p=1}^{K}\frac{1}{\left|Y_p\right|\left|\overline{Y}_p\right|}\left|\left\{(y_1,y_2)\in Y_p\times\overline{Y}_p : f(x_p,y_1)\le f(x_p,y_2)\right\}\right| \tag{10}$$

Ranking loss and *one-error* denote the function f defined in (2) and *Hamming loss* denotes the function h defined in (1). Both *Hamming loss* and *ranking loss* consider the frequencies of incorrect output values (in the case of *Hamming loss* incorrect label assignment and in the case of *ranking loss* incorrect order of rank values). Both measures well address multilabel classification characteristics. *One-error* takes into account only the label with the highest rank value and ignores other labels. *One-error* does not measure general performance of multilabel classifier but is specialized for penalizing classifiers which frequently give the highest rank value to the labels not belonging to Y_p.

3 Application to Functional Genomics

Our modifications to the BP-MLL method were tested against real-world multilabel classification problem in the domain of functional genomics [1], [3], [4], [5]. The goal is to determine (various) functions of genes based on biological data such as gene expression levels [5] (from DNA micro arrays), sequence data (sequences of nucleotides or amino acids) or phylogenetic profiles [5].

3.1 Yeast Genome Data Set and Learning Parameters

In particular in our experiments a data set [2] dealing with yeast genome was considered. This set was also used by other researchers, e.g. [1], [3]. It contains 2417 genes associated with functional classes. Every gene is described by 103-dimensional vector consisting of the information about phylogenetic profile and gene expression levels. This vector forms the neural network input. Each input vector is associated with a subset of the set of 14 possible functional classes. In average, each example (gene) is associated with 4.24 ± 1.57 labels.

During training process the learning rate was set to 0.05. In order to avoid overfitting, similarly to [1] a weight decay (equal to 0.5) was introduced. Due to the relatively small size of the data set, tenfold cross-validation was applied. Training was performed separately on each of the five multilabel classifiers – one for each global error function (3) – (7).

The size of a hidden layer was equal to 20 and 40 neurons, respectively for the classifiers with error functions (3), (4) and (5), (6), (7). The number of training epochs was equal to 100. In the case of (3) the threshold function (t) was set to be the zero constant function.

3.2 Experimental Results

Five experiments, each based on tenfold cross validation, were performed for each classifier. This resulted in 50 evaluations of each of the tested classifiers for each of the three error measures (Hamming loss, one-error and ranking loss). Table 1 presents means and standard deviations of those evaluations. Results of statistical tests (t-test at 5 percent significance level) are shown in Table 2.

Table 1. Means and standard deviations of considered multilabel classifiers evaluations

Error function	Hamming loss		One-error		Ranking loss	
	Mean	Std. Dev.	Mean	Std. Dev.	Mean	Std. Dev.
(3)	0,2754	0,0188	0,2324	0,0292	0,1729	0,0150
(4)	0,2005	0,0071	0,2311	0,0255	0,1705	0,0102
(5)	0,2023	0,0094	0,2351	0,0215	0,1721	0,0107
(6)	0,1988	0,0094	0,2252	0,0230	0,1659	0,0120
(7)	0,1987	0,0089	0,2247	0,0242	0,1657	0,0117

Table 2. Statistical tests results (t-test at 5 percent significance level). The results below the 5 percent level are presented in boldface.

Test	Hamming loss p-value	One-error p-value	Ranking loss p-value
(3) vs. (4)	**0**	0,8202	0,3569
(3) vs. (5)	**0**	0,5950	0,7786
(3) vs. (6)	**0**	0,1778	**0,0116**
(3) vs. (7)	**0**	0,1578	**0,0089**
(4) vs. (5)	0,2787	0,4010	0,4335
(4) vs. (6)	0,2981	0,2289	**0,0410**
(4) vs. (7)	0,2594	0,2026	**0,0310**

The results of experiments presented in both tables allow to make some performance comparisons between various neural network multilabel classifiers taken into account. Considering the Hamming loss shows that all modified classifiers (i.e. (4), (5), (6) and (7)) are significantly better than the original one (3). Moreover, there are no statistically significant differences in the results accomplished by classifier (4) vs. (5), (6) or (7). One-error performance measure does not permit to make any conclusions about potential differences between multilabel classifiers in question. They are statistically comparable. This can be caused by the characteristics of one-error measure which only considers some details of classifier and does not address multilabel classifier performance in general. Finally, classifiers (6) and (7) outperform (3) and (4), with statistical significance when ranking loss measure is considered.

Table 3 presents comparison of BP-MLL with other approaches to multilabel classification (decision tree based method ADTBOOST.MH [11] and kernel method RANK-SVM [3]) on the Yeast Genome Data Set considered in this paper. This

Table 3. Mean values and standard deviations of considered multilabel classifiers evaluations for three approaches to multilabel classification (BP-MLL, ADTBOOST.MH and RANK-SVM)

Error function	Hamming loss		One-error		Ranking loss	
	Mean	Std.Dev.	Mean	Std.Dev.	Mean	Std.Dev.
BP-MLL	0,206	0,011	0,233	0,034	0,171	0,015
ADTBOOST.MH	0,207	0,010	0,244	0,035	-	-
RANK-SVM	0,207	0,013	0,243	0,039	0,195	0,021

comparison was made by authors of [1] and shows that BP-MLL outperforms both ADTBOOST.MH and RANK-SVM.

4 Conclusions and Future Work

Multilabel classification problem generalizes traditional two-class or multi-class classification since each instance in the training/testing set is associated with several (usually more than one) classes (labels). The problem is not easy to solve also because the size of the label-set associate with particular unseen instance is generally unknown. Various approaches to tackle this problem were presented in the literature, but – up to our knowledge – there has been only one attempt to apply a neural network for solving this task [1]. In this paper a few modifications of the global error function proposed in [1] are presented and experimentally evaluated. Generally, all of them improve performance of the multilabel neural classifier. The improvement – in case of the two most elaborate functions, i.e. (6) and (7) is noticeable and statistically significant. Overall, including the threshold values into the error function and considering differences between the rank values and the thresholds proved to be a promising direction for improvement of the multilabel classifier performance.

Currently, we are focused on performing more tests (especially with other sizes of hidden layer) and on other data sets in order to further verify the efficacy of proposed modifications.

References

1. Zhang, M.L., Zhou, Z.H.: Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization. IEEE Transactions on Knowledge and Data Engineering 18(10), 1338–1351 (2006)
2. LIBSUM Data: Multilabel Classification, `http://www.csie.ntu.tw/~cjlin/libsumtools/datasets/multilabel.html#yeast`
3. Elissef, A., Weston, J.: A Kernel Method for Multi-Labelled Classification. In: Dietterich, T.G., Becker, S., Ghahramani, Z. (eds.) Advances in Neural Information Processing Systems, vol. 14, pp. 681–687 (2002)
4. Clare, A., King, R.D.: Knowledge Discovery in Multi-Label Phenotype Data. In: Siebes, A., De Raedt, L. (eds.) PKDD 2001. LNCS (LNAI), vol. 2168, pp. 42–53. Springer, Heidelberg (2001)

5. Pavlidis, P., Weston, J., Cai, J., Grundy, W.N.: Combining Microarray Expression Data and Phylogenetic Profiles to Learn Functional Categories using Support Vector Machines. In: 5th Annual International Conference Computational Molecular Biology (RECOMB 2001), pp. 242–248 (2001)
6. McCallum, A.: Multi-Label Text Classification with a Mixture Model Trained by EM. In: Working Notes Am. Assoc. Artificial Intelligence Workshop Text Learning (AAAI 1999) (1999)
7. Schapire, R.E., Singer, Y.: BoosTexter: A Boosting-Based System for Text Categorization. Machine Learning 39(2/3), 135–168 (2000)
8. Kazawa, H., Izumitani, T., Taira, H., Maeda, E.: Maximal Margin Labeling for Multi-Topic Text Categorization. In: Saul, L.K., Weiss, Y., Bottou, L. (eds.) Advances in Neural Information Processing Systems, vol. 17, pp. 649–656 (2005)
9. Werbos, P.J.: Beyond Regression: New Tools for Prediction and Anlysis in the Behavioral Sciences. PhD thesis, Harvard University (1974)
10. Rumelhart, D.E., Hinton, G.E., Williams, R.J.: Learning Internal Representations by Error Propagation. In: Rumelhart, D.E., McClelland, J.L. (eds.) Parallel Distributed Processing: Explorations in the Microstructure of Cognition, vol. 1, pp. 318–362 (1986)
11. Comite, F.D., Gilleron, R., Tommasi, M.: Learning Multi-Label Alternating Decision Tree from Texts and Data. In: Perner, P., Rosenfeld, A. (eds.) MLDM 2003. LNCS, vol. 2734, pp. 35–49. Springer, Heidelberg (2003)