Neural Networks for Multi-Instance Learning

Zhi-Hua Zhou Min-Ling Zhang

National Laboratory for Novel Software Technology
Nanjing University, Nanjing 210093, China
zhouzh@nju.edu.cn zml@ai.nju.edu.cn

Abstract

Multi-instance learning originates from the investigation on drug activity prediction, where the task is to predict whether an unseen molecule could be used to make some drug. Such a problem is difficult because a molecule may have many alternative shapes with low energy, yet only one of those shapes may be responsible for the qualification of the molecule to make the drug. Because of its unique characteristics and extensive existence, multi-instance learning is regarded as a new machine learning framework parallel to supervised learning, unsupervised learning, and reinforcement learning. In this paper, an open problem of this area is addressed. That is, a popular neural network algorithm is adapted for multi-instance learning through employing a specific error function. Experiments show that the adapted algorithm achieves good result on the drug activity prediction data.

1. Introduction

At present, roughly speaking, there are three frameworks for learning from examples [9]. That is, *supervised learning*, *unsupervised learning*, and *reinforcement learning*. Supervised learning attempts to learn a concept for correctly labeling unseen examples, where the training examples are with labels. Unsupervised learning attempts to learn the structure of the underlying sources of examples, where the training examples are with no labels. Reinforcement learning attempts to learn a mapping from states to actions, where the examples are with no labels but with delayed rewards that could be viewed as delayed labels.

Recently, Dietterich *et al*. [7] proposed the notation of *multi-instance learning*, where the training set is composed of many *bags* each containing many instances. The bags are labeled in the way that if a bag contains at least one positive instance then it is labeled as a positive bag. Otherwise it is labeled as a negative bag. However, the labels of the instances are unknown. The task is to learn some concept from the training set for correctly labeling unseen bags. Since such kinds of learning tasks extensively exist in the world but they are quite unique from those addressed by previous learning frameworks, multi-instance learning is regarded as the fourth learning framework [9].

When the notation of multi-instance learning was proposed, Dietterich *et al*. [7] indicated that a particular interesting issue in this area is how to design multi-instance modifications for neural networks. In this paper, this open problem is addressed because a neural network algorithm BP-MIP, i.e. BP for Multi-Instance Problems, is presented. As its name implied, BP-MIP is derived from the popular neural network algorithm BP [14], which is adapted for multi-instance learning through employing a specific error function. Experiments show that BP-MIP works well on the drug activity prediction data, which is the only benchmark test data for multi-instance learning at present.

The rest of this paper is organized as follows. Section 2 briefly reviews previous works on multi-instance learning. Section 3 presents BP-MIP. Section 4 reports some preliminary experimental results of BP-MIP on the drug activity prediction data. Finally, Section 5 concludes and indicates several issues for future works.

2. Previous Works Review

In the middle of 1990's, Dietterich *et al*. [7] investigated the problem of drug activity prediction. The goal is to endow learning systems with the ability of predicting that

whether a new molecule could be used to make some drug, through analyzing a collection of known molecules. The qualification of a molecule to make a drug is determined by some of its shapes with low energy. However, as shown in Figure 1, a molecule may have many alternative shapes with low energy, but at present biochemists only know that whether a known molecule is qualified to make the drug instead of knowing that which of its alternative

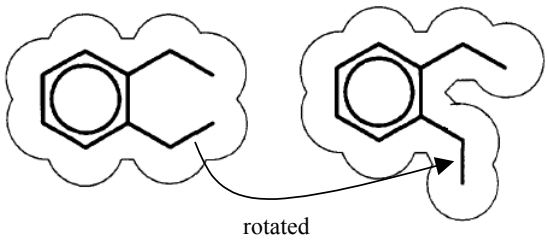

Figure 1: The shape of a molecule changes as it rotates an internal bond.

shapes responses for the qualification.

An intuitive solution is to use the supervised learning framework by regarding all the shapes of the molecules qualified to make the drug as positive examples, while regarding all the shapes of the molecules unqualified to make the drug as negative examples. However, such a method can hardly work well because there may be too many false positive examples [7].

In order to solve this problem, Dietterich *et al*. [7] regarded each molecule as a bag, and regarded the alternative shapes of the molecule with low energy as the instances in the bag, thereby initiated multi-instance learning. Then they proposed three Axis-Parallel Rectangle (abbreviated as APR) algorithms, which attempts to search for appropriate axis-parallel rectangles constructed by the conjunction of the features. Their experiments on *Musk* data, which is the only benchmark test data for multi-instance learning until now, show that the iterated-discrim APR algorithm achieves the best result, while the performance of popular supervised learning algorithms such as C4.5 decision tree and BP neural network is very poor.

Long and Tan [8] described a polynomial-time theoretical algorithm and showed that if the instances in the bags are independent drawn from product distribution, then the APR is PAC-learnable. Auer *et al*. [3] showed that if the instances in the bags are not independent then APR learning under the multi-instance learning framework is NP-hard. Moreover, they presented a theoretical algorithm that does not require product distribution. Later, the theoretical algorithm was transformed to a practical algorithm named MULTINST [2]. Blum and Kalai [5] described a reduction from the problem of PAC-learning under the multi-instance learning framework to PAC-learning with one-sided random classification noise, and presented a theoretical algorithm with smaller sample complexity than that of Auer *et al*.'s algorithm.

Among practical multi-instance learning algorithms, the most famous is Diverse Density proposed by Maron and Lozano-Pérez [10]. The diverse density of a point in the feature space is defined in the way that the more positive bags and the less negative instances near the point, the bigger the diverse density of the point. The algorithm is then search for the point with the maximal diverse density. Such an algorithm has been applied to several applications, including learning a simple description of a person from a series of images [10], stock selection [10], natural scene classification [11], and content-based image retrieval [16].

There are also many other practical algorithms. Wang and Zucker [15] extended *k*-nearest neighbor algorithm for multi-instance learning through adopting Hausdorff distance. Ruffo [13] presented a multi-instance version of C4.5 decision tree, which was named as Relic. Chevaleyre and Zucker [6] derived ID3-MI and RIPPER-MI, which are multi-instance version of decision tree algorithm ID3 and rule learning algorithm RIPPER.

Recently, some researchers begin to investigate multi-instance regression tasks with real-valued outputs. Ray and Page [12] showed that the general formulation of the multi-instance regression task is NP-hard, and proposed an EM-based multi-instance regression algorithm. Amar *et al*. [1] extended Diverse Density for multi-instance regression and designed a method for artificially generating data sets for multi-learning regression.

It is worth noting that when the term *multi-instance learning* was coined, Dietterich *et al*. [7] indicated that a particular interesting issue is how to design multi-instance modifications for decision trees, neural networks, and other popular machine learning algorithms. Multi-instance version of decision trees [6, 13], rule learning algorithms [6], and lazy learning algorithms [15], have already been presented. But designing neural networks for

multi-instance learning is still an open problem until now.

3. BP-MIP

Suppose there are N bags $\{B_1, B_2, \ldots, B_N\}$, the i-th bag contains M_i instances $\{B_{i1}, B_{i2}, \ldots, B_{iMi}\}$, each instance is a p-dimensional feature vector, e.g. the j-th instance of the i-th bag is $[B_{ij1}, B_{ij2}, \ldots, B_{ijp}]^T$ where T denotes the transpose of a vector. Suppose there is a feedforward neural network with p input units, one output unit, and one hidden layer. The activation function is *Sigmoid* function.

Since the goal of multi-instance learning is to predict the labels of unseen bags, we define the global error function at the level of bags as:

$$E = \sum_{i=1}^{N} E_i \tag{1}$$

where E_i is the error on B_i.

We assume that for each instance, if the actual output of the network is not less than 0.5, then the instance is regarded as a positive instance. Otherwise it is regarded as a negative instance. Then, E_i can be defined as:

$$E_i = \begin{cases} \min\limits_{1 \le j \le M_j} E_{ij} & if\ B_i = + \\ \max\limits_{1 \le j \le M_j} E_{ij} & f\ B_i = - \end{cases} \tag{2}$$

where "B_i = +" ("B_i = -") means B_i is a positive (negative) bag, E_{ij} is the error on B_{ij}:

$$E_{ij} = \begin{cases} 0 & if\ (B_i = +)\ and\ (0.5 \le o_{ij}) \\ 0 & if\ (B_i = -)\ and\ (o_{ij} < 0.5) \\ \frac{1}{2}(o_{ij} - 0.5)^2 & otherwise \end{cases} \tag{3}$$

where o_{ij} is the actual output of B_{ij}.

With the defined error function, the BP algorithm [4] is easy to be adapted for multi-instance learning as described as follows.

In each training epoch, the training bags are fed to the network one by one. When the instance B_{ij} is fed, E_{ij} is computed according to Eq.(3). For a positive bag B_i, if E_{ij} is zero then all the rest instances of B_i are not be fed to the network in this epoch, and the weights in the network are not changed for B_i. Otherwise E_i is computed according to Eq.(2) after all the instances of B_i are fed, and the weights in the network are modified according to the weight-updated rule of BP [14]. Then, $B_{i,j+1}$ is fed to the network and the training process is repeated until the global error E decreases to some pre-set threshold or the number of epochs increases to some pre-set threshold.

4. Experiments

We have tested BP-MIP on the *Musk* data, which is the only benchmark test data for multi-instance learning at present. There are two data sets, i.e. *Musk1* and *Musk2*, both are publicly available at UCI Machine Learning Repository [4]. Here we used *Musk1*, which contains 47 positive bags, i.e. musk molecules, and 45 negative bags, i.e. non-molecules. The number of instances contained in each bag varies from 2 to 40. In average each bag contains 5.17 instances. Each instance corresponds to a shape with low energy of the molecule, which is described by a 166-dimensional feature vector. All the features are numeric.

The BP-MIP network we used contains 166 input units each corresponds to a dimension of the 166-dimensional feature vectors, one output units which outputs [0.5, 1] for positive while [0, 0.5] for negative, one hidden layer with 100 hidden units. The learning rate is set to 0.01. Note that both the number of hidden units and the learning rate have not been finely tuned.

10-fold cross validation is performed on the data set. In detail, the original data set is partitioned into ten roughly equal-sized subsets with roughly same proportion of positive/negative bags as that of the original data set. Then, in each fold, the union of nine subsets is regarded as the training set while the remaining subset is regarded as the test set. Such a process is repeated in ten times so that each subset has been used as test set for one fold. In each fold, five BP-MIP networks are trained where the best predictive accuracy is recorded as the result of that fold. The final result is the average result of those ten folds.

The predictive accuracy curve of BP-MIP is shown in Figure 2, where the horizontal axis is the number of training epochs.

Figure 2 shows that the best performance of BP-MIP is 83.8%, which roughly appears after 350 training epochs. Table 1 shows the comparison of the result with those reported in the literatures, where the value follows "±" is the standard deviation.

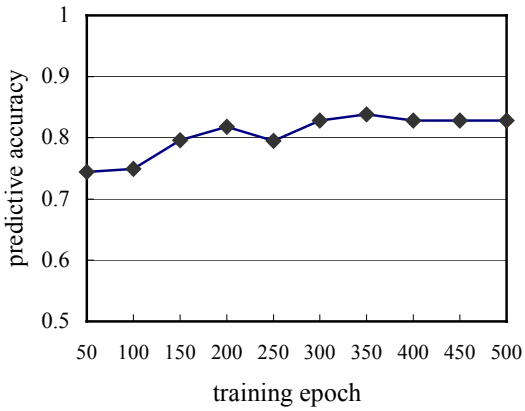

Figure 2: The predictive accuracy curve of BP-MIP.

Table 1 shows that, on the *Musk* data, BP-MIP is significantly better than Auer's MULTINST algorithm [2], and is comparable to Ruffo's Relic algorithm [13]. But its performance is not so good as Dietterich *et al*.'s iterated-discrim APR algorithm [7] and Maron and Lozano-Pérez's Diverse Density algorithm [10].

As Dietterich *et al*. indicated [7], the performance of iterated-discrim APR might be the upper bound of learning algorithms on the *Musk* data because this algorithm was optimized toward this data. So, although the predictive accuracy of BP-MIP is not so good as that of iterated-discrim APR on the *Musk* data, its applicability is better because it is a general algorithm that has not been optimized toward any data. As for Diverse Density, feature selection is incorporated in this algorithm [10]. We believe that the performance of BP-MIP may be improved if some appropriate feature selection mechanism is employed.

Moreover, Table 1 shows that the performance of BP-MIP is significantly better than that of BP and C4.5. This observation supports the claim that supervised learning methods can hardly work well on multi-instance problems because they have not consider the unique characteristics of multi-instance learning [7].

Table 1: Comparison of the predictive accuracy on the *Musk1* data set

algorithm	%correct
iterated-discrim APR [7]	92.4
Diverse Density [10]	88.9
BP-MIP	83.8 ± 9.2
Relic [13]	83.7
MULTINST [2]	76.7 ± 4.3
BP [7]	75.0
C4.5 [7]	68.5

5. Conclusion

In this paper, an open problem of multi-instance learning, that is, designing the multi-instance version of neural network algorithms, is addressed. Through devising a specific error function incorporates the characteristics of multi-instance problems, BP-MIP is proposed, which is observed to work well on the benchmark test data.

As described in Section 4, there are two *Musk* data sets, i.e. *Musk1* and *Musk2*. Due to the time limitation, at present we have only get results of BP-MIP on *Musk1*. Obtain the results of BP-MIP on *Musk2* is a work we hope to do in the near future.

The experimental results reported in this paper are preliminary also because that we have not finely tuned the architecture and parameters of BP-MIP. Exploring better configurations of BP-MIP is an issue for our future works.

Moreover, there may be other ways to adapt popular neural network algorithms such as BP for multi-instance problems. Design more effective modification than BP-MIP is another issue for our future works.

Acknowledgements

This research is supported in part by the National Natural Science Foundation of China under Grant Number 60105004 and the Innovative Investigator Project of the Natural Science Foundation of Jiangsu Province under Grant Number BK2001406.

References

[1] R. A. Amar, D. R. Dooly, S. A. Goldman, and Q. Zhang. "Multiple-instance learning of real-valued data," in: *Proceedings of the 18th International Conference on Machine Learning*, Williamstown, MA, pp.3-10, 2001.

[2] P. Auer. "On learning from multi-instance examples: empirical evaluation of a theoretical approach," in *Proceedings of the 14th International Conference on Machine Learning*, Nashville, TN, pp.21-29, 1997.

[3] P. Auer, P. M. Long, and A. Srinivasan. "Approximating hyper-rectangles: learning and pseudo-random sets," *Journal of Computer and System Sciences*, vol.57, no.3, pp.376-388, 1998.

[4] C. Blake, E. Keogh, and C. J. Merz. "UCI repository of machine learning databases" [http://www.ics.uci.edu/~mlearn/MLRepository.html], Department of Information and Computer Science, University of California, Irvine, CA, 1998.

[5] A. Blum and A. Kalai. "A note on learning from multiple-instance examples," *Machine Learning*, vol.30, no.1, pp.23-29, 1998.

[6] Y. Chevaleyre and J.-D. Zucker. "Solving multiple-instance and multiple-part learning problems with decision trees and decision rules. Application to the mutagenesis Problem," in *Lecture Notes in Artificial Intelligence 2056*, E. Stroulia and S. Matwin, Eds. Berlin: Springer, pp.204-214, 2001.

[7] T. G. Dietterich, R. H. Lathrop, and T. Lozano-Pérez. "Solving the multiple-instance problem with axis-parallel rectangles," *Artificial Intelligence*, vol.89, no.1-2, pp.31-71, 1997.

[8] P. M. Long and L. Tan. "PAC learning axis-aligned rectangles with respect to product distributions from multiple-instance examples," *Machine Learning*, vol.30, no.1, pp.7-21, 1998.

[9] O. Maron. "Learning from ambiguity," PhD dissertation, Department of Electrical Engineering and Computer Science, MIT, Jun. 1998.

[10] O. Maron and T. Lozano-Pérez. "A framework for multiple-instance learning," in: *Advances in Neural Information Processing Systems 10*, M. I. Jordan, M. J. Kearns, and S. A. Solla, Eds. Cambridge, MA: MIT Press, pp.570-576, 1998.

[11] O. Maron and A. L. Ratan. "Multiple-instance learning for natural scene classification," in *Proceedings of the 15th International Conference on Machine Learning*, Madison, WI, pp.341-349, 1998.

[12] S. Ray and D. Page. "Multiple instance regression," in *Proceedings of the 18th International Conference on Machine Learning*, Williamstown, MA, 2001.

[13] G. Ruffo. "Learning single and multiple instance decision tree for computer security applications," PhD dissertation, Department of Computer Science, University of Turin, Torino, Italy, 2000.

[14] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning internal representations by error propagation," in *Parallel Distributed Processing: explorations in the microstructure of cognition*, vol.1, D. E. Rumelhart and J. L. McClelland, Eds. Cambridge, MA: MIT Press, pp.318-362, 1986.

[15] J. Wang and J.-D. Zucker. "Solving the multiple-instance problem: a lazy learning approach," in *Proceedings of the 17th International Conference on Machine Learning*, San Francisco, CA, pp.1119-1125, 2000.

[16] C. Yang and T. Lozano-Pérez. "Image database retrieval with multiple-instance learning techniques," in *Proceedings of the 16th International Conference on Data Engineering*, San Diego, CA, pp.233-243, 2000.