Solving the Multi-Instance Problem with Neural Networks

Zhi-Hua Zhou[*] and Min-Ling Zhang

National Laboratory for Novel Software Technology, Nanjing University, Nanjing 210093, China

Abstract

Multi-instance learning was coined by Dietterich *et al*. recently in their investigation on drug activity prediction. In such a learning framework, the training examples are *bags* composed of instances, and the task is to predict the labels of unseen bags through analyzing the training bags with known labels. A bag is positive if it contains at least one positive instance, while it is negative if it contains no positive instance. However, the labels of the instances constituting the training bags are unknown. In this paper, the open problem of designing multi-instance modification for neural networks is addressed. In detail, a neural network algorithm named Backpropagation-MIP is presented, which is derived from the popular Backpropagation algorithm through employing a new error function capturing the nature of multi-instance learning, i.e. the labels of the training bags instead of that of the training instances are known. Experiments on both the real-world and artificial benchmark multi-instance data show that the performance of Backpropagation-MIP is comparable to that of some well-established multi-instance learning methods.

1. Introduction

At present, learning from examples is regarded as the most promising machine learning approach [19]. According to Maron [16], there are three frameworks for learning from examples. 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*. [9] proposed the notion of *multi-instance learning* in their investigation of drug activity prediction. In multi-instance learning, the training set is composed of many *bags* each containing many instances. 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. The labels of the training bags are known, but that of the training instances are unknown. The task is to learn something from the training set for correctly labeling unseen bags. Since such kind of problem is quite different from those addressed by supervised learning, unsupervised learning, and

[*] Corresponding author. Tel.: +86-25-359-3163; fax: +86-25-330-0710.
E-mail addresses: zhouzh@nju.edu.cn (Z.-H. Zhou).

reinforcement learning, multi-instance learning was regarded as the fourth framework for learning from examples [16].

When the notion of multi-instance learning was proposed, Dietterich *et al*. [9] indicated that a particular interesting issue in this area is to design multi-instance modifications for neural networks. In this paper, this open problem is addressed in the way that a neural network algorithm named Backpropagation-MIP, i.e. Backpropagation for Multi-Instance Problems, is proposed. As its name implied, Backpropagation-MIP is derived from the popular Backpropagation algorithm [23] through replacing its error function with a new function defined to capture the nature of multi-instance learning, that is, the labels of the training bags instead of that of the training instances are known. Experiments on the drug activity prediction data, which is the only real-world benchmark test data for multi-instance learning at present, and some artificial benchmark multi-instance data, show that the results obtained by Backpropagation-MIP are comparable to that of some well-established multi-instance methods.

The rest of this paper is organized as follows. In Section 2, the drug activity prediction problem is briefly introduced. In Section 3, previous works on multi-instance learning are reviewed. In Section 4, Backpropagation-MIP is presented. In Section 5, experiments of Backpropagation-MIP on the real-world and artificial data are reported. Finally in Section 6, the main contribution of this paper is summarized and several issues for future work are indicated.

2. Drug Activity Prediction

Most drugs are small molecules working by binding to larger protein molecules such as enzymes and cell-surface receptors. The potency of a drug is determined by the degree of binding. For molecules qualified to make a drug, one of its low-energy shapes could tightly bind to the target area. While for molecules unqualified to make a drug, none of its low-energy shapes could tightly bind to the target area.

In the middle of 1990's, Dietterich *et al*. [9] investigated the problem of drug activity prediction. The goal was to endow learning systems with the ability of predicting that whether a new molecule was qualified to make some drug, through analyzing a collection of known molecules. The main difficulty of this problem is that each molecule may have many alternative low-energy shapes, as illustrated in Fig. 1. But biochemists only know that whether a molecule is qualified to make a drug or not, instead of knowing that which of its alternative low-energy shapes responses for the qualification.

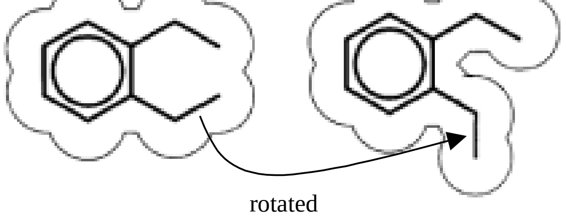

Fig. 1. The shape of a molecule changes as it rotates an internal bond

An intuitive solution is to use supervised learning algorithms by regarding all the low-energy shapes of the molecules qualified to make the drug as positive training examples, while regarding all the low-energy shapes of the molecules unqualified to make the drug as negative training examples. However, as shown by Dietterich *et al.* [9], such a method can hardly work because there may be a great many of false positive examples.

In order to solve this problem, Dietterich *et al.* [9] regarded each molecule as a bag, and regarded the alternative low-energy shapes of the molecule as the instances in the bag, thereby formulated multi-instance learning. In order to represent the shapes, the molecule was placed in a standard position and orientation and then a set of 162 rays emanating from the origin was constructed so that the molecular surface was sampled approximately uniformly, as illustrated in Fig. 2. There were also four features that represented the position of an oxygen atom on the molecular surface. Therefore each instance in the bags was represented by a 166-dimensional numerical feature vector.

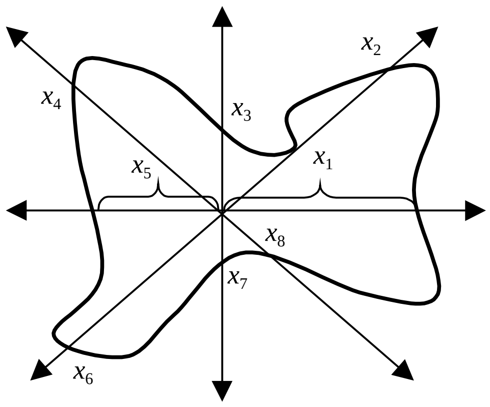

Fig. 2. The ray-based representation of the molecular shape

Based on such a representation, Dietterich *et al.* [9] 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 showed that the iterated-discrim APR algorithm achieves the best result on the *Musk* data, which is the only real-world benchmark test data for multi-instance learning until now, while the performance of popular supervised learning algorithms such as C4.5 decision tree and Backpropagation neural network is very poor. Note that Dietterich *et al.* [9] indicated that since the APR algorithms were optimized to the *Musk* data, the performance of iterated-discrim APR might be the upper bound for this data.

It should be mentioned that multi-instance problems do not emerge suddenly from drug activity prediction. In fact, they extensively exist in real-world applications [14, 24]. But unfortunately, machine learning community has not paid special attention to such kinds of problems until Dietterich *et al.* [9].

3. Previous Works Review

Long and Tan [15] initiated the investigation of the PAC-learnability of axis-parallel rectangles under the multi-instance learning framework. Resorting to P-concept [13], they showed that if the instances in the bags are independently drawn from product distribution, then the APR is PAC-learnable. They also described a polynomial-time theoretical algorithm. 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 but with smaller sample complexity than that of Long and Tan's algorithm. This theoretical algorithm was transformed to a practical algorithm named MULTINST later [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. With the help of Statistical-Query Model [12], they also presented a theoretical algorithm with smaller sample complexity than that of Auer *et al*.'s algorithm. Goldman *et al*. [10] presented an efficient on-line agnostic multi-instance learning algorithm for learning the class of constant-dimension geometric patterns, which tolerated both noise and concept shift. Later, this algorithm was extended so that it could deal with real-valued output [11].

As reviewed above, theoretical machine learning community has contributed much to multi-instance learning. But since most of their results are obtained under assumptions such as the number of instances in the bags must be a constant, which is often not the case in real problems, those results are hard to be used directly in real-world applications.

Fortunately, some practical algorithms for multi-instance learning have been presented by the applied machine learning community, the most famous one of which is the Diverse Density algorithm proposed by Maron and Lozano-Pérez [17]. 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. Thus the learning task is transformed to search for a point in the feature space with the biggest diverse density. This algorithm has been successfully applied to a series of tasks. The first is to learn a simple description of a person from a series of images [17], where the images were regarded as bags while the sub-images sampled from an image were regarded as the instances in the corresponding bag. For each image, if a specific person appeared then the corresponding bag was positively labeled. Otherwise the bag was negatively labeled. The second task is stock selection, i.e. selecting stocks perform well for fundamental reasons [17]. In this task, 100 stocks with the highest return of each month were regarded as a positive bag, while 5 stocks with the lowest return were regarded as a negative bag, where the stocks were regarded as instances. The third task is natural scene classification [18], where the images were regarded as bags, while blobs [18] sampled from an image were regarded as the instances in the corresponding bag. If the user was interested in a part of an image, such as a waterfall, then the corresponding bag was positively labeled. Otherwise the bag was negatively labeled. More recently, the Diverse Density algorithm was extended and applied to content-based image retrieval [26].

Wang and Zucker [25] extended *k*-nearest neighbor algorithm for multi-instance learning through adopting Hausdorff distance. Two algorithms, i.e. Bayesian-*k*NN and Citation-*k*NN, were presented. Bayesian-*k*NN labels a bag through analyzing its neighboring bags with Bayes theory. Citation-*k*NN borrows the notion of *citation* of scientific references, which labels a bag through analyzing not only its neighboring bags but also the bags that regard the concerned bag as a neighbor. Ruffo [22] presented a decision tree algorithm named Relic, which is the multi-instance version of C4.5. Later, 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, where the key is a multiple-instance entropy and a multiple-instance coverage function respectively.

It is worth mentioning that Chevaleyre and Zucker [6] indicated that some tasks, such natural scene classification, are quite different from drug activity prediction in nature. This is because that in drug activity prediction, the instances of a bag are the alternative descriptions of the bag, which cannot appear simultaneously. While in natural scene classification, the instances of a bag are the descriptions of different parts of the bag, which should appear simultaneously. In order to distinguish those two situations, Chevaleyre and Zucker [6] coined the term *multi-part learning* for cases where the instances are partial descriptions of the bags, keeping the term *multi-instance learning* for cases where the instances are alternative descriptions of the bags. However, they indicated that multi-part problems could be solved with multi-instance learning algorithms [6].

In the early years of the research of multi-instance learning, most works are on multi-instance classification with discrete-valued outputs. Recently, multi-instance regression with real-valued outputs begins to attract the attention of some researchers. Ray and Page [20] 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 the Diverse Density algorithm for multi-instance regression. Moreover, they designed some method for artificially generating multi-regression data. Their data sets are available from http://www.cs.wustl.edu/~sg/multi-inst-data/.

Multi-instance learning has even attracted the attention of the Inductive Logic Programming community. De Raedt [8] showed that multi-instance problems could be regarded as a bias on inductive logic programming. He also suggested that the multi-instance paradigm could be the key between the propositional and relational representations, being more expressive than the former, and much easier to learn than the latter. Zucker and Ganascia [28, 29] presented REPEAT, an ILP system based on an ingenious bias which firstly reformulates the relational examples in a multi-instance database, and then induces the final hypothesis with a multi-instance learner.

It is worth noting that when Dietterich *et al.* [9] coined the term *multi-instance learning*, they indicated that a particular interesting issue in this area is to design multi-instance modifications for decision trees, neural networks, and other popular machine learning algorithms. During recent years, multi-instance version of decision trees [6, 22], rule learning algorithms [6], and lazy learning algorithms [25], have already been presented. However, designing neural networks for multi-instance learning is still an open problem until now.

4. Backpropagation-MIP

Suppose the training set is composed of N bags, i.e. $\{B_1, B_2, \ldots, B_N\}$, the i-th bag is composed of M_i instances, i.e. $\{B_{i1}, B_{i2}, \ldots, B_{iM_i}\}$, each instance is a p-dimensional feature vector, e.g. the j-th instance of the i-th bag is $[B_{ij1}, B_{ij2}, \ldots, B_{ijp}]^T$. The desired output of a positive training bag is 1, while that of a negative training bag is 0.

Now suppose a neural network with p input units and one output unit is used to learn from the training set. Since the goal of multi-instance learning is to predict the labels of unseen bags, the global error of the network on the training set could be defined as:

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

where E_i is the error of the network on B_i, which could be defined as:

$$E_i = \frac{1}{2}\left(\max_{1 \le j \le M_j} o_{ij} - d_i\right)^2 \tag{2}$$

where o_{ij} is the actual output of the network on B_{ij}, d_i is the desired output of B_i.

Eq.(2) means that if at least one instance of a positive training bag is perfectly predicted as positive, or all the instances of a negative bag are perfectly predicted as negative, then the error on the concerned bag is zero and the weights of the network will not be updated. Otherwise the weights will be updated according to the error on the instance whose corresponding actual output is the maximal among all the instances in the bag. Note that such an instance is the most easy to be predicted as positive for a positive bag, while is the most difficult to be predicted as negative for a negative bag. It seems that this sets a low burden on producing a positive output but a strong burden on producing a negative output. However, as Amar *et al.* [1] indicated, the value of a bag is fully determined by its instance with the maximal output, nevertheless how many real positive or negative instances in the bag. Therefore, in fact the burden on producing a positive or negative output is not unbalanced.

Thus, the global error of the network on the training set could be written as:

$$E = \sum_{i=1}^{N} E_i = \sum_{i=1}^{N} \frac{1}{2}\left(\max_{1 \le j \le M_j} o_{ij} - d_i\right)^2 \tag{3}$$

For comparison, if the Backpropagation algorithm is directly used then the global error should be:

$$E = \sum_{i=1}^{N} E_i = \sum_{i=1}^{N}\sum_{j=1}^{M_i} E_{ij} = \sum_{i=1}^{N}\sum_{j=1}^{M_i} \frac{1}{2}\left(o_{ij} - d_i\right)^2 \tag{4}$$

Note that the error function shown as Eq.(4) is defined at the level of training instances, while the one shown as Eq.(3) is defined at the level of training bags. The main difference in their appearances is that Eq.(3) replaces the second summation term of Eq.(4) with some terms incorporating the characteristics of multi-instance learning, i.e. a positive bag contains at least one positive instance while a negative bag contains no positive instance. It is also worth noting that the gradient direction derived from Eq.(3) and that derived from Eq.(4) are quite different. In other words, the gradient defined at the level of training bags and that defined at the level of training instances are different. We believe that this explains Dietterich *et al.*'s observation that Backpropagation could not work well on multi-instance problem [9].

With the error function shown as Eq.(3), the training process of Backpropagation-MIP could be easily derived because it is almost identical to the Backpropagation algorithm except that in the latter the weights are updated for each training instance, but in the former the weights are updated for each training bag. In detail, in each training epoch of Backpropagation-MIP, the training bags are fed to the network one by one. For bag B_i, if it has been correctly predicted then no weight in the network is changed. Otherwise the weights are modified according to the error on the instance whose corresponding actual output is the maximal in B_i. After that, $B_{i,j+1}$ is fed to the network and the training process is iterated until the global error E decreases to a threshold or the number of training epochs increases to a threshold.

In short, Backpropagation modifies networks according to training instances, while Backpropagation-MIP Modifies networks according to training bags. In this way, Backpropagation-MIP captures the nature of multi-instance learning where the labels of the training bags instead of that of the training instances are known.

Note that in training a Backpropagation-MIP network, the desired outputs of the bags, i.e. 1 for positive while 0 for negative, can be replaced with 0.9 for positive while 0.1 for negative, which is a trick for speeding up the training process [21]. When a trained Backpropagation-MIP network is used in prediction, a bag is positively labeled if and only if the output of the network on at least one of its instances is not less than 0.5.

5. Experiments

5.1 *Musk* data sets

The *Musk* data is the only real-world benchmark test data for multi-instance learning at present. The data is generated by Dietterich *et al.* in the way described in Section 2. There are two data sets, both of which are publicly available from UCI Machine Learning Repository [4]. Information of those two data sets is summarized in Table 1. Note that there are some irrelevant features and all the instances are unique [9].

Table 1. *Musk* data (72 molecules are shared in both data sets)

data set	dim.	bags			instances	instances per bag		
		total	musk	non-musk		min	max	ave.
Musk1	166	92	47	45	476	2	40	5.17
Musk2	166	102	39	63	6,598	1	1,044	64.69

Feedforward neural networks with one output unit, one hidden layer, and 166 input units each corresponds to a dimension of the 166-dimensional feature vectors, are trained with the Backpropagation-MIP algorithm. The activation functions of the functional units are *Sigmoid*. The learning rate is set to 0.05. The number of hidden units varies from 20 to 100 with an interval of 20, while the number of training epochs varies from 50 to 1,000 with an interval of 50.

Leave-one-out test is performed on both data sets. In detail, one bag is used to test while the others are used to train a neural network. Such a process is repeated in the way that each bag in the data set has been used as the test bag once. In other words, the number of neural networks trained for each kind of configuration, i.e. the number of hidden units and training epochs, equals to the number of bags in the data set. The final result for each kind of configuration is the average result of the neural networks trained for the configuration.

The curves of the predictive accuracy on *Musk1* and *Musk2* changes as the number of training epoch increasing are depicted in Fig. 3 and Fig. 4 respectively.

Fig. 3 and Fig. 4 show that the number of hidden units does not significantly affect the predictive accuracy of the Backpropagation-MIP networks, while the number of training epoch significantly affects the predictive accuracy. In fact, there is an obvious trend that as the number of training epoch increasing, the predictive accuracy of the networks also increases. Due to the limitation of computational costs, at present we have only

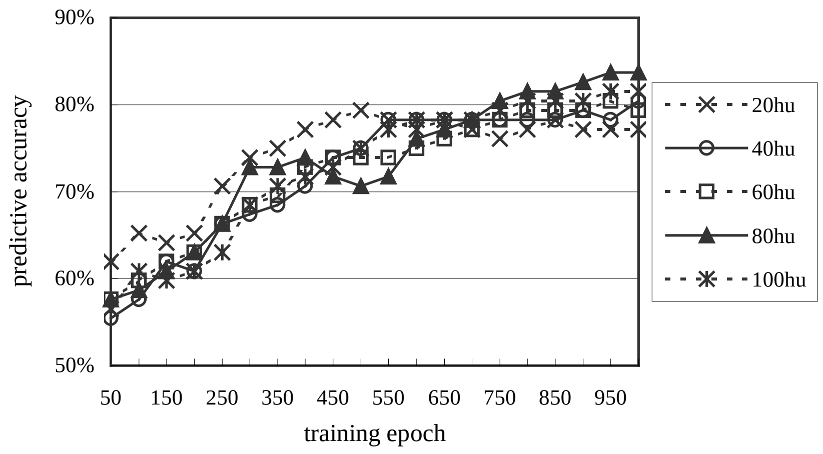

Fig. 3. The predictive accuracy on *Musk1* changes as the number of training epoch increasing

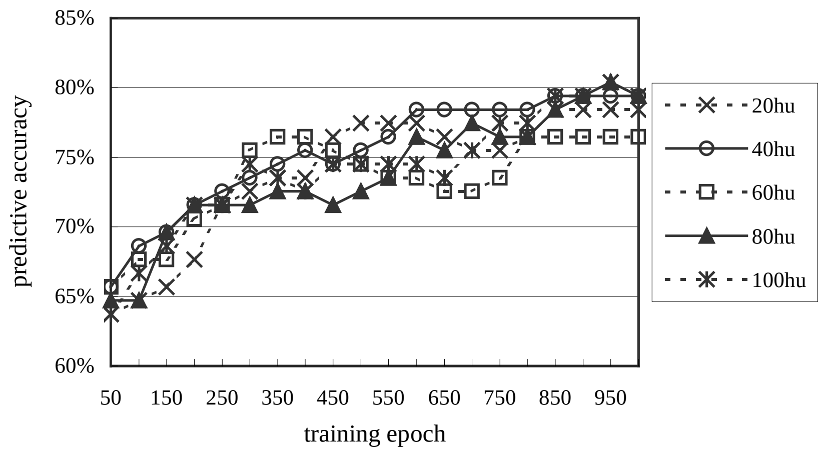

Fig. 4. The predictive accuracy on *Musk2* changes as the number of training epoch increasing

trained the networks to 1,000 epochs. But we believe that if the networks were trained with more epochs, the predictive accuracy could be further improved.

In Fig.3 and Fig.4, the best performance of Backpropagation-MIP were 83.7% on *Musk1* and 80.4% on *Musk2*, both were obtained by the networks with 80 hidden units and 950 training epochs. Table 2 compares the results with those reported in the literatures. Note that no standard deviation is shown in Table 2 because most of the results are obtained using leave-one-out test and there is no variation for this test method although usually the difference between the leave-one-out and the 10-fold is not significant [25].

Table 2 shows that, Backpropagation-MIP is significantly better than MULTINST [2], and is comparable to Relic [22] on *Musk1*; it is significantly better than RIPPER-MI [6] and GFS elim-count APR [9], and is comparable to GFS elim-kde APR [9] on *Musk2*.

Note that although Backpropagation-MIP is only better than some instead of all the well-established multi-instance learning methods on predictive accuracy, it has its own advantages. For example, Dietterich *et al.*'s APR algorithms [9] were specially designed for the *Musk* data, while Backpropagation-MIP is a general algorithm so that its applicability is better than that of the APR algorithms. Wang and Zucker's extended *k*NN

Table 2. Comparison of the predictive accuracy on the *Musk* data

algorithm	*Musk1* %correct	algorithm	*Musk2* %correct
iterated-discrim APR [9]	92.4	iterated-discrim APR [9]	89.2
Citation-*k*NN [25]	92.4	Relic [22]	87.3
GFS elim-kde APR [9]	91.3	Citation-*k*NN [25]	86.3
GFS elim-count APR [9]	90.2	MULTINST [2]	84.0
Bayesian-*k*NN [25]	90.2	Diverse Density [17]	82.5
Diverse Density [17]	88.9	Bayesian-*k*NN [25]	82.4
RIPPER-MI [6]	88.0	**Backpropagation-MIP**	**80.4**
Backpropagation-MIP	**83.7**	GFS elim-kde APR [9]	80.4
Relic [22]	83.7	RIPPER-MI [6]	77.0
MULTINST [2]	76.7	GFS elim-count APR [9]	75.5
Backpropagation [9]	75.0	Backpropagation [9]	67.7
C4.5 [9]	68.5	C4.5 [9]	58.8

algorithms [25] are lazy learning methods requiring much time cost for iterative processes on prediction, while the time cost of Backpropagation-MIP on prediction is trivial. Maron and Lozano-Pérez's Diverse Density algorithm [17] employed some feature selection mechanism, while Backpropagation-MIP has not performed feature selection and we believe that its performance could be further improved if appropriate feature selection mechanism is used. More important, Backpropagation-MIP is easy to be adapted for multi-instance regression problems.

Table 2 also indicates that the performance of all the multi-instance learning methods is better than that of Backpropagation and C4.5, which is especially obvious on *Musk2* that is more difficult to learn than *Musk1*. This observation supports Dietterich *et al.*'s claim [9] that traditional supervised learning methods can hardly work well on multi-instance problems because they have not incorporated the characteristics of multi-instance learning.

5.2 Artificial data sets

Prior research performed under multi-instance learning is for classification. However, binding affinity between molecules and receptors is quantitative, borne out in quantities such as the energy released by the molecule-receptor pair upon binding and hence a real-valued label of binding strength is preferable. Fortunately, Amar *et al.* [1] presented a method for creating artificial multi-instance data. This method generates an artificial receptor at first. Then, artificial molecules with several instances per bag are generated, with each feature value considered as the distance from the origin to the molecular surface when all molecules are in the same orientation. Each feature has a *scale factor* to represent its importance in the binding process. The binding energies between the artificial molecules and receptor are calculated based on the *Lennard-Jones potential* for intermolecular interactions.

The artificial data sets are named as LJ-*r.f.s* where *r* is the number of relevant features, *f* is the number of features, and *s* is the number of different scale factors used for the relevant features. To partially mimic the *Musk* data, some data sets only use labels that are not near 1/2 (indicated by the 'S' suffix) and all scale factors for the relevant features are randomly selected between [0.9, 1]. Note that these data sets are mainly designed for multi-instance regression, but they can also be used for multi-instance classification through rounding the real-valued label to 0 or 1.

In this paper, four artificial data sets, i.e. LJ-160.166.1, LJ-160.166.1-S, LJ-80.166.1, and LJ-80.166.1-S, are used. Each data set contains 92 bags. Feedforward neural networks with one output unit, one hidden layer with 80 units, and 166 input units each corresponds to a dimension of the 166-dimensional feature vectors, are trained with the Backpropagation-MIP algorithm. The activation functions of the functional units are *Sigmoid*. The learning rate is set to 0.05. The number of training epochs varies from 50 to 500 with an interval of 50.

Leave-one-out test is performed on these data sets. The curves of the predictive accuracy changes as the number of training epoch increasing are depicted in Fig. 5.

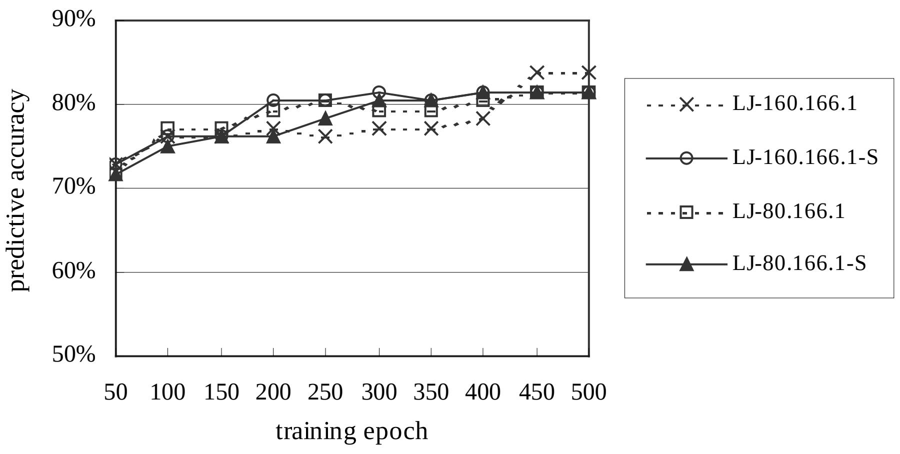

Fig. 5. The predictive accuracy on artificial data sets changes as the number of training epoch increasing

Fig. 5 shows an obvious trend that as the number of training epoch increasing, the predictive accuracy of the networks also increases. The best performance of Backpropagation-MIP in Fig.5 is compared with those reported in the literature [1], as shown in Table 3 where both the predictive error and squared loss are reported.

Table 3. Comparison of the predictive error and squared loss on the artificial data sets

data	Backpropagation-MIP		Diverse Density		Citation-*k*NN	
	%err	loss	%err	loss	%err	loss
LJ-160.166.1	16.3	0.0398	23.9	0.0852	4.3	0.0014
LJ-160.166.1-S	18.5	0.0731	0.0	0.0052	0.0	0.0022
LJ-80.166.1	18.5	0.0487	N/A	N/A	8.6	0.0109
LJ-80.166.1-S	18.5	0.0752	53.3	0.116	0.0	0.0025

Table 3 shows that, although the performance of Backpropagation-MIP is worse than that of Citation-kNN [25], it is significantly better than that of the most famous multi-instance learning algorithm, i.e. Diverse Density [17], on LJ-160.166.1 and LJ-80.166.1-S. This not only shows the effectiveness of Backpropagation-MIP in multi-instance regression, but also suggests that the rank of Backpropagation-MIP among multi-instance learning methods may be better than that was shown in Table 2.

6. Conclusion and Future Work

In this paper, a neural network algorithm named Backpropagation-MIP, which is the multi-instance version of Backpropagation, is proposed. Through employing a new error function, Backpropagation-MIP captures the nature of multi-instance learning, i.e. the labels of the training bags instead of that of the training instances are known. Experiments on both the real-world and artificial benchmark multi-instance data show that Backpropagation-MIP achieves performance comparable to some well-established multi-instance learning methods. This addresses the open problem raised by Dietterich *et al*. [9], that is, designing multi-instance modifications for neural networks.

Maron [16] indicated that from the aspect of the ambiguity in the training sets, multi-instance learning should be placed in somewhere between *supervised learning*, whose training sets are with no ambiguity, and *unsupervised learning*, whose training sets are with maximal ambiguity. Since some popular supervised learning algorithms could be adapted for multi-instance learning, such as some researchers' works on decision trees [6, 22], rule learning algorithms [6], lazy learning algorithms [25], and our work on neural networks in this paper, we believe that multi-instance learning should be placed more close to *supervised learning* than to *unsupervised learning*.

An advantage of Backpropagation-MIP is that it is a general algorithm that has not been optimized toward any data, which means that it could be easily applied to applications such as content-based image retrieval. Find more real-world multi-instance problems and apply Backpropagation-MIP to them is another interesting issue for future works. Note that when applying Backpropagation-MIP to real-world problems, it is not very feasible to experimentally choose appropriate configurations, i.e. the number of hidden units and training epochs, to achieve the best performance. Therefore an important work is to develop some mechanisms for estimating the appropriate configuration of Backpropagation-MIP for concrete problems.

Moreover, since many algorithms working well on multi-instance problems, such as iterated-discrim APR [9] and Diverse Density [17], have inbuilt feature selection schemes, it will be interesting to explore that whether the performance of Backpropagation-MIP could be significantly improved with the help of feature selection.

Furthermore, recent research has shown that neural network ensemble could significantly improve the generalization ability of neural network based learning systems, which has become a hot topic in both machine learning and neural network communities [27]. So, it is interesting to see that whether better results could be obtained with ensembles of Backpropagation-MIP networks.

Acknowledgements

The comments and suggestions from the anonymous reviewers greatly improved this paper. This research is supported 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] Y. Chevaleyre and J.-D. Zucker. "A framework for learning rules from multiple instance data," in *Lecture Notes in Artificial Intelligence 2167*, L. De Raedt and P. Flach, Eds. Berlin: Springer, pp.49-60, 2001.

[8] L. De Raedt. "Attribute-value learning versus inductive logic programming: the missing links," in *Lecture Notes in Artificial Intelligence 1446*, D. Page, Ed. Berlin: Springer, pp.1-8, 1998.

[9] 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.

[10] S. A. Goldman, S. S. Kwek, and S. D. Scott. "Agnostic learning of geometric patterns," *Journal of Computer and System Sciences*, vol.62, no.1, pp.123-151, 2001.

[11] S. A. Goldman and S. D. Scott. "Multiple-instance learning of real-valued geometric patterns," Technical Report UNL-CSE-99-006, Department of Computer Science and Engineering, University of Nebraska, Lincon, NE, 2000.

[12] M. Kearns. "Efficient noise-tolerant learning from statistical queries," in *Proceedings of the 25th Annual ACM Symposium on Theory of Computing*, San Diego, CA, pp.392-401, 1993.

[13] M. J. Kearns and R. E. Schapire. "Efficient distribution-free learning of probabilistic concepts," *Journal of Computer and System Sciences*, vol.48, no.3, pp.464-497, 1994.

[14] R. Lindsay, B. Buchanan, E. Feigenbaum E, and J. Lederberg. *Applications of Artificial Intelligence to Organic Chemistry: The DENDRAL Project*, New York: McGraw-Hill, 1980.

[15] 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.

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

[17] 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.

[18] 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.

[19] T. M. Mitchell, *Machine Learning*, New York: McGraw-Hill, 1997.

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

[21] R. D. Reed and R. J. Marks II, *Neural Smithing*, Cambridge, MA: MIT Press, 1999.

[22] 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.

[23] 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.

[24] M. Sebag and C. Rouveirol. "Tractable induction and classification in first order logic," in *Proceedings of the 15th International Joint Conference on Artificial Intelligence*, Nagoya, Japan, pp.888-893, 1997.

[25] 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.

[26] 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.

[27] Z.-H. Zhou, J. Wu, and W. Tang. "Ensembling neural networks: many could be better than all, " *Artificial Intelligence*, vol.137, no.1-2, pp.239-263, 2002.

[28] J.-D. Zucker and J.-G. Ganascia. "Changes of representation for efficient learning in structural domains," in *Proceedings of the 13th International Conference on Machine Learning*, Bary, Italy, pp.543-551, 1996.

[29] J.-D. Zucker and J.-G. Ganascia. "Learning structurally indeterminate clauses," in *Lecture Notes in Artificial Intelligence 1446*, D. Page, Ed. Berlin: Springer, pp.235-244, 1998.