Multiple-Instance Neural Network with Strong Boundary Criteria

Pakaket Wattuya and Arnon Rungsawang
Department of Computer Engineering, Faculty of Engineering
Kasetsart University
g4565054@ku.ac.th and fenganr@ku.ac.th

Boonserm Kijsirikul
Department of Computer Engineering, Faculty of Engineering
Chulalongkorn University
boonserm.k@chula.ac.th

Abstract
The Multiple-Instance Learning (MIL) is a new learning framework proposed in 1997 and getting more attention recently in the field of machine learning. The MIL framework performs successfully on learning from ambiguous examples, whereas standard supervised learning seems to be inefficient. This paper proposes a method called the Multiple-Instance Neural Network with Strong Boundary Criteria (MINN-SBC) for solving the Multiple-Instance problem. In our proposed method, we determine the suitable boundary criteria of the neural network in order to eliminate the ambiguity of examples. We evaluate our method on the MUSK data sets benchmark and compare the method with other existing algorithms. The predictive accuracies are 91.3% and 90.2% for MUSK1 and MUSK2, respectively. The experimental results show that our proposed method is better than those of the other neural network approaches proposed for MIL.

Key-words: Multiple-Instance Learning, Neural Network, Machine Learning.

1. Introduction
Frameworks such as supervised learning, unsupervised learning and reinforcement learning have many established algorithms and theoretical tools to analyze them. However, there are many learning problems that do not fall into any of these established frameworks. Specially, situations where the examples are ambiguously labeled or cannot be simply represented as a feature vector tend to be difficult for these frameworks. A new framework, called Multiple-Instance Learning (MIL), has been proposed to deal with the situations.

The MIL framework is getting more attention recently, and has numerous interesting applications. The first application was drug activity prediction proposed by Dietterich et al. [1] in 1997. The goal is to endow learning systems with the ability of predicting whether a new molecule could be used to make some drug. In order to solve this problem, Dietterich et al. represented the target concept by an Axis-Parallel Rectangle (APR) in the n-dimensional feature space and presented three MIL algorithms for learning APR. They performed experiments on the Musk data sets [2], the only benchmark test data for MIL until now. The results 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 learning and Back-Propagation (BP) neural networks are very poor.

Following these works, there are a significant amount of research directed towards the development of multiple-instance algorithms using different learning models. Long and Tan [3] described a polynomial-time theoretical algorithm and showed that if the instances in the bags (the definition of bags will be shown later in Section 2) are independent drawn from product distribution, then the APR is PAC-learnable. Auer et al. [4] showed that if the instances in the bags are not independent then APR learning under the MIL framework is NP-hard. Moreover, they presented a theoretical algorithm that did not require product distribution. Later, the theoretical algorithm was transformed to a practical algorithm named MULTINST.

Among practical MIL algorithms, the most famous one is Diverse Density (DD) proposed by Maron and Lozano-Perez [5]. 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 [5],
stock selection [5], natural scene classification [6],
and content-based image retrieval [7].

The other popular machine learning algorithms
such as neural networks have already been adapted
for MIL. Ramon and Raedt [8] applied the simple
backpropagation neural network technique and
proposed the Multi-instance neural network. Zhou
and Zhang [9] adapted a neural network with
specific error function on MIL and presented the
BP-MIP algorithm. Andrews et al. [10] proposed
two approaches to modify and extend Support
Vector Machines (SVMs) to deal with MIL
problems named mi-SYM and MI-SFM.

In this paper, we presented a general-purpose
MIL method called, the Multi-instance Neural
Network with Strong Boundary Criteria (MINN-
SBC). Experiments show that our method yields
the good predictive accuracy compared with existing
MIL algorithms from the literature.

The organization of the paper is as follows.
Section 2 describes a framework of the multiple-
instance learning problem. In Section 3, we
present our proposed method, MINN-SBC.
Experiments are presented in Section 4. The
conclusion and future works are given in Section 5.

2. Multiple-Instance Learning

The multiple-instance learning has arisen in
complex applications of machine learning where
the learning system has partial or incomplete
knowledge about each training example. In
standard supervised learning, the learner is
provided with training examples \( x_i \in X \), also called
instances, which are represented by \( n \) real-value
feature vectors, \( V(x) \in R^n \). A training example is
given in term of \( (V(x), f(x)) \) where \( j \) is the index of
an example and the label of the example is
consistent with a target function \( f(x) \).

In multiple-instance learning framework, a
single training example may have \( v_i \) variant
instances denoted \( (x_{i1}, x_{i2}, ..., x_{in}) \) called “bag”.
Each of these variants will be represented by a
distinct feature vector \( V(x_{ij}) \). Therefore, a training
example (bag) can be defined as follows:

\[
\{V(x_{i1}), V(x_{i2}), ..., V(x_{in})\}, f(x_i)
\]

(1)

where each bag is associated with a label \( f(x_i) \in \{0,1\} \). The bag is labeled positive if there is at least
one of the instances in the bag is positive.
Otherwise it is labeled as a negative bag. However,
the labels of the individual instances are unknown.
The learning algorithm tries to learn a concept from

a collection of labeled bags for labeling the unseen
bags accurately. The standard supervised learning
can be viewed as a special case of multiple-
instance learning where each bag holds a single
instance. However, this problem is harder than
even noisy supervised learning since the ratio of
negative to positive instances in a positively-
labeled bag (the noise ratio) can be arbitrarily high.

In the drug activity prediction application, each
bag comprises of all possible low-energy
conformations of each molecule. In the image
retrieval application, each bag is a collection of
local image patches or image regions.

3. Methodology

In this section, we present our proposed
method, the Multi-instance Neural Network with
Strong Boundary Criteria (MINN-SBC).

3.1 Multiple-Instance Neural Network with
Strong Boundary Criteria

Given the instance space \( \alpha \), the bag space \( \beta =
2^\alpha \), the label space \( \gamma = \{0 \) (negative), \( 1 \) (positive)\}. A set of training examples can be defined as
\( T(B,L), \) where \( B = \{B_i \mid B_i \in \beta, i = 1...n\} \) is a set of
\( n \) bags and \( L = \{L_i \mid L_i \in \gamma, i = 1...n\} \) is the set of
their associated labels with \( L_i \) being the label of \( B_i \).
The \( i \)-th bag contains \( M_i \) instances \( \{B_{i1}, B_{i2}, ..., B_{imi}\} \). Each instance \( j \) is a \( p \)-dimensional feature
vector \( [B_{ij1}, B_{ij2}, ..., B_{ijp}]^T \) where \( T \) denotes the
transpose of a vector. The learning algorithm is
attempting to identify which feature vectors are
responsible for the observed classifications.

Suppose there is a three-layer feedforward
neural network. The network has an input layer of
\( p \) units depending on the number of the input
features, a hidden layer of \( h \) units, and an output
layer of one unit. The sigmoid function is used as
the activation function in the neural network.

In our proposed algorithm, each instance in a
particular bag has a label in a discrete space \( \{0 \)
(negative), \( 1 \) (positive)\}. Given the labels of all
instances in a bag, the label of each bag can be
represented by the logical OR of the labels of all
instances in the bag as follow:

\[
L_i = \lor_{j=1}^{h} l_{ij}
\]

(2)

where the label \( L_i \) is the label of bag \( B_i \) and \( l_{ip} \) is the
label of the \( j \)-th instance in \( B_i \). If \( B_i \) is predicted to be
positive if and only if at least one of the variant
instances must produce a positive result. In other
words, if none of the variant instances have
produced a positive result, \( B_i \) is labeled as negative.
Since the goal of our algorithm is to try to induce a concept that will label individual instance correctly, we define the global error function at the level of bags as follows:

\[ E = \sum_{i=1}^{N} E_i \]  

(3)

where \( E_i \) is the error on \( B_i \).

Then \( E_i \) can be defined as:

\[ E_i = \begin{cases} 
\min_{\text{all } E_{ij}} E_{ij} & \text{if } B_i = + \\
\max_{\text{all } E_{ij}} E_{ij} & \text{if } B_i = - 
\end{cases} \]  

(4)

where “\( B_i = + \)” ("\( B_i = - \”) means \( B_i \) is a positive (negative) bag.

Generally, a neural network with one output unit regularly sets a boundary criterion to 0.5 for binary-class classification. The drawback of this criterion is when the lower boundary of a positive class and the upper boundary of a negative class are close, every example having an actual output near this boundary (0.5) may be ambiguous.

For MIL problem, the basic idea for using a neural network as classifier is that “when a positive instance of a positive bag is found, the rest of the instances are disregarded and then the weight adaptation will be ignored for that iteration”. In the case that when the actual output of an instance in a positive bag is close to 0.5 (e.g. equal to 0.51), the weight will be ignored for this iteration. In this situation, the distance between actual output and target is still large. Thus, the distance must be decreased by adjusting the weight. For our proposed method, the boundary criteria are therefore set to 0.8 and 0.2 for the lower boundary of a positive class and the upper boundary of a negative class, respectively.

Then, \( E_{ij} \) is the error on \( B_{ij} \) can be defined as:

\[ E_{ij} = \begin{cases} 
0 & \text{if } (B_i = +) \text{ and } (0.8 \leq o_{ij}) \\
0 & \text{if } (B_i = -) \text{ and } (o_{ij} \leq 0.2) \\
\frac{1}{2}(o_{ij} - L_i)^2 & \text{otherwise}
\end{cases} \]  

(5)

where \( o_{ij} \) is the actual output of \( B_{ij} \) and \( L_i \) is the target of \( B_i \).

The error backpropagation (BP) algorithm, based on the stochastic gradient descent method, is adapted for training our method. The detail is described as follows.

To train the MINN-SBC, the training bags are fed into the network one by one. When the instance \( B_{ij} \) is fed, \( E_{ij} \) is computed according to Eq.(5). For a positive bag \( B_i \), if \( E_{ij} \) is zero then all the rest instance of \( B_i \) are not be fed into the MINN-SBC in this epoch, and the weights in the MINN-SBC are not changed for \( B_i \). Otherwise \( E_i \) is computed according to Eq.(4) 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 [11]. Then, \( B_{ij+1} \) is fed into the MINN-SBC 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.

To test the performance of a properly trained MINN-SBC, the feature vector of a testing bag is presented to the MINN-SBC and the set of actual outputs of all instances are generated by using the weights received from the training process. The maximum actual output is selected. If the maximum actual output is more than or equal to 0.5, then the bag is predicted as a positive bag. Otherwise it is predicted as a negative bag as follow:

\[ p_i = \begin{cases} 
+ & \text{if } \max_{\text{all } o_{ij}} o_{ij} \geq 0.5 \\
- & \text{otherwise}
\end{cases} \]  

(6)

where \( p_i \) is the predicted answer of \( B_i \).

4. Experiments

4.1 The MUSK data sets

The MUSK data sets are the benchmark data sets used in virtually all previous approaches and have been described in detail in the landmark paper [1]. There are two datasets, MUSK1 and MUSK2, publicly available at UCI Machine Learning Repository [2]. Both data sets contain descriptions of molecules using multiple low-energy conformations. Each conformation is represented by a 166-dimensional feature vector derived from surface properties. MUSK1 contains a set of 92 molecules of which 47 are judged by human experts to be musks (positive class) and the remaining 45 molecules are judged to be non-musks (negative class), while MUSK2 consists of a set of 102 molecules of 39 musks and 63 non-musks as shown in Table 1. MUSK1 contains on average approximately 6 conformations per molecule, while MUSK2 has on average more than 60 conformations in each bag.

<table>
<thead>
<tr>
<th>Data Set</th>
<th>Musks</th>
<th>Non-Musks</th>
<th>Total</th>
<th>Conformations</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>47</td>
<td>45</td>
<td>92</td>
<td>476</td>
</tr>
<tr>
<td>2</td>
<td>39</td>
<td>63</td>
<td>102</td>
<td>6598</td>
</tr>
</tbody>
</table>

Table 1. The musk data sets.
Table 2. Comparison of the predictive accuracy on the MUSK data sets 1 and 2.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>MUSK data set 1 (MUSK1)</th>
<th>MUSK data set 2 (MUSK2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Iterated-discrim APR [1]</td>
<td>92.4</td>
<td>90.2</td>
</tr>
<tr>
<td>MINN-SBC</td>
<td><strong>91.3</strong></td>
<td></td>
</tr>
<tr>
<td>Diverse Density [5]</td>
<td>88.9</td>
<td>84.3</td>
</tr>
<tr>
<td>Multi-instance neural network [8]</td>
<td>88.0</td>
<td>84.0</td>
</tr>
<tr>
<td>mi-SVM [10]</td>
<td>87.4</td>
<td>83.6</td>
</tr>
<tr>
<td>BP-MIP [9]</td>
<td>83.8</td>
<td>82.5</td>
</tr>
<tr>
<td>MI-SVM [10]</td>
<td>77.9</td>
<td>82.0</td>
</tr>
<tr>
<td>MULTINST [4]</td>
<td>76.7</td>
<td></td>
</tr>
</tbody>
</table>

Table 3. Comparison of the predictive accuracy detail on the MUSK data sets 1 and 2.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>MUSK data set 1 (MUSK1)</th>
<th>MUSK data set 2 (MUSK2)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>true positives</td>
<td>false negatives</td>
</tr>
<tr>
<td>MINN-SBC</td>
<td>47</td>
<td>0</td>
</tr>
<tr>
<td>iterated-discrim APR [1]</td>
<td>42</td>
<td>5</td>
</tr>
</tbody>
</table>

4.2 Experimental Setting

We have performed experiments on MUSK data sets to evaluate the proposed algorithm compared with 7 existing algorithms, i.e., iterated-discrim APR [1], Diverse Density [5], Multi-instance neural network [8], mi-SVM [10], BP-MIP [9], MI-SVM [10] and MULTINST [4]. Among the above seven algorithms, the Multi-instance neural network and the BP-MIP algorithms are neural network approaches.

Our MINN-SBC contains 166 input units, one hidden layer with 100 hidden units, and one output unit. Ten-fold cross-validation is performed on each data set. The original data are partitioned into ten disjoint sets of roughly equal size with roughly same proportion of the positive/negative bags as that of the original data set. Ten training sets are then constructed by overlapping the ten disjoint sets and dropping out a different one systematically. The different sets, which are dropped, are used as test sets.

For each fold, we trained the MINN-SBC with learning rate 0.0001 and momentum 0.97. The training process is repeated until the global error $E$ decreases to 0.01 or the number of epochs increases to 500.

4.3 Experimental results and Discussions

The experimental results are summarized in Table 2. For both MUSK1 and MUSK2 data sets, MINN-SBC achieved competitive accuracy. MINN-SBC gave the second best accuracy in MUSK1, and provided the best accuracy in MUSK2. The results show that MINN-SBC was comparable to iterated-discrim APR which was optimized toward MUSK data sets [5], and performed better than the other algorithms.

From the above experimental results, we may make a number of qualitative conclusions based on the quantitative results:

(i) MINN-SBC achieves a good result on MUSK1. MUSK2 is harder than MUSK1 because it contains examples with more than 1000 instances, and the ratio of negative to positive instances in a positively-labeled bag (the noise ratio) is pretty high. Therefore, the predictive accuracy of MINN-SBC decreases in MUSK2.

(ii) The results show that MINN-SBC clearly improves on the BP-MIP and the Multi-instance neural network for both data sets. The reason seems to be that we use the strong boundary criteria of the example classes. In order to eliminate the ambiguity for both positive and negative examples, we set the boundary thresholds to 0.8 for the lower boundary of a positive class and 0.2 for the upper boundary of a negative class instead of 0.5 for both boundaries.

(iii) For MUSK1, there are no false negative errors (as shown in Table 3), which is important for the drug discovery application since the final hypothesis would be used to filter potential drugs and a false negative error means that a potential
good drug molecule would not be tested and thus it is good to minimize such errors [12].

5. Conclusion and Future works

In this paper, the Multiple-Instance Neural Network with Strong Boundary Criteria (MINN-SBC) has been proposed for solving the multiple-instance learning (MIL) problem. We have performed experiments on MUSK data sets using ten-fold cross-validation. The results are compared with previous works. The accuracies are 91.3% and 90.2% for MUSK1 and MUSK2, respectively and better than those of the other neural network approaches [8,9] proposed for MIL. In the future, we plan to apply our algorithm to other MIL applications.

6. References