社団法人 情報処理学会 研究報告
IPSJ SIG Technical Report

2003－ＩＣＳ－133 (30)
2003／9／16

Distance-based Heuristics in Inductive Logic Programming for Mining from Chemical Compound Data

Cholwich NATTEE†, Sukree SINTHUPINYO†, Masayuki NUMAO†, and Takashi OKADA††

† The Institute of Scientific and Industrial Research, Osaka University
8-1 Mihogaoka, Ibaraki, Osaka, 567–0047, Japan
†† Center for Information & Media Studies, Kwansei Gakuin University
E-mail: †{cholwich,sukree,numao}@ai.sanken.osaka-u.ac.jp, ††okada@kwansei.ac.jp

Abstract We propose an approach for making FOIL better handling multiple-instance data. This learning problem arises when trying to generate hypotheses from examples in the form of positive and negative bags. Each bag contains one or more instances and a bag is labelled as positive when there is at least one positive instance, otherwise, it is labelled as negative. Several algorithms have been proposed for learning in this framework. However all of them can only handle data in the attribute-value form which has limitations in knowledge representation. Therefore, it is difficult to handle examples consisting of structures among components, such as chemical compounds data. In this paper, the Diverse Density, a measure for multiple-instance data, is applied to adapt the heuristic function in FOIL in order to improve learning accuracy in multiple-instance data. We conducts the experiments on real-world data related to chemical compound analysis in order to show the improvement.

Key words Inductive Logic Programming, Multiple-Instance Learning

距離によるヒューリスティックスを用いた 帰納論理プログラミングと化学物質データマイニング

ナッティー・チョラウィト† シンツピンヨー・スクリー† 沼尾 正行† 岡田 孝††

† 大阪大学産業科学研究所 〒567–0047 大阪府茨木市美穂ヶ丘 8–1
†† 関西学院大学情報メディア教育センター
E-mail: †{cholwich,sukree,numao}@ai.sanken.osaka-u.ac.jp, ††okada@kwansei.ac.jp

1. Introduction

Multiple-instance (MI) learning [3] is a framework extended from supervised learning in the case that training examples cannot be labelled completely. Training examples are grouped into labelled bags marked as positive if there is at least one instance known to be positive. Otherwise, they are marked as negative. The MI learning framework was originally motivated by the drug activity prediction problem which aims to determine whether aromatic drug molecules bind strongly to a target protein. As a lock and a key, the shape of molecules is the most important factor for determining this binding. The molecules can nevertheless adapt their shapes widely. Then, each shape is represented as an instance and the positive bags are the molecules with at least one shape binding well. On the other hand, the negative bags contains molecules whose shapes did not bind at all. Dietterich et al. formalised this framework and proposed the axis-parallel rectangles algorithm [3]. After this work, many approaches have been proposed for MI learning [2], [4], [7], they nevertheless aim for handling only data in the attribute-value form where an instance is represented as a fixed-length vector inheriting a limitation that complicated relations among instances become difficult to be denoted, for example, representing chemical compound structures by describing atoms and bonds among atoms.

Inductive Logic Programming (ILP) has introduced more expressive first-order representation to supervised learning. ILP has been successfully applied to many applications and the first-order logic can also be represented the MI data well. However, in order to make the ILP systems able to generate more accurate hypotheses, the distance among instances would be useful because in MI data the positive instances cannot be specified exactly, thus the distance between positive instances and negative instances plays an important role in this determination. If there is an area that many instances from various positive bags locating together and that area is far from the instances from negative bags, the target concept would come from the instances in that area.

This paper presents the extension of FOIL [5] using the Diverse Density (DD) [4], a measure for evaluating MI data. Applying this measure will make FOIL more precisely identify the positive instances and generate more suitable hypotheses from training data. In this research, we focus on applying this approach to predict the characteristics of chemical compound. Each compound (or molecule) is represented using properties of atom and bonds between atoms.

The paper is organised as follows. The next section described the background of DD and FOIL which are the bases of our approach. Then the modification of FOIL algorithm is considered. We evaluate the proposed algorithm with chemical compound data. Finally the conclusion and our research direction are given.

2. Background

2.1 Diverse Density

The Diverse Density (DD) algorithm aims to measure a point in an n-dimensional feature space to be a positive instance. The DD at point p in the feature space shows both how many *different* positive bags have an instance near p, and how *far* the negative instances are from p. Thus, the DD is high in the area where instances from various positive bags are located together. It can be calculated as

$$DD(x) = \prod_i (1 - \prod_j (1 - exp(-\|B_{ij}^+ - x\|^2))) \cdot \prod_i \prod_j (1 - exp(-\|B_{ij}^- - x\|^2)) \quad (1)$$

where x is a point in the feature space and B_{ij} represents the j^{th} instance of the i^{th} bag in training examples. For the distance, the Euclidean distance is adopted then

$$\|B_{ij} - x\|^2 = \sum_k (B_{ijk} - x_k)^2 \quad (2)$$

In the previous approaches, several searching techniques are proposed for determining the value of features or the area in the feature space that maximises DD. In this paper, the DD is however applied in the heuristic function in order to evaluate each instance from the positive bags with the value between 0 and 1.

2.2 FOIL

The learning process in FOIL starts with a training set (examples) containing all positive and negative examples, constructs a function-free Horn clause (a hypothesis) to cover some of the positive examples, and removes the covered examples from the training set. Then it continues with the search for the next clause. When the clauses covering all the positive examples have been found, they are reviewed to eliminate any redundant clauses and re-ordered so that all recursive clauses come after the non-recursive ones.

FOIL uses a heuristic function based on the information theory for assessing the usefulness of a literal. It seems to provide effective guidance for clause construction. The purpose of this heuristic function is to characterise a subset of the positive examples. From the partial developing clause $R(V_1, V_2, \ldots, V_k) \leftarrow L_1, L_2, \ldots, L_{m-1}$, the training examples covered by this clause are denoted as T_i. The information required for T_i is given by

$$I(T_i) = -\log_2(|T_i^+|/(|T_i^+| + |T_i^-|)) \quad (3)$$

If a literal L_m is selected and yields a new set T_{i+1}, then the similar formula is given as

$$I(T_{i+1}) = -\log_2(|T_{i+1}^+|/(|T_{i+1}^+| + |T_{i+1}^-|)) \quad (4)$$

From above, a heuristic used in FOIL is calculated an amount of information gained when applying a literal L_m;

$$Gain(L_i) = T_i^{++} \times (I(T_{i+1}) - I(T_i)) \quad (5)$$

T_i^{++} in this equation is the positive examples extended in T_{i+1}.

This heuristic function is used over every candidate literal and the literal with a largest value is selected. The algorithm will continue until the generated clauses cover all positive examples.

3. Our Approach

The essential difference between the MI problem and the classical classification problem is in the positive examples. In the classical problem, positive and negative examples are precisely separated, where in the MI problem, positive instances cannot be specified exactly since positive bags only contain at least one positive instance. FOIL nevertheless evaluates and selects the best literal based on a number of positive and negative instances covered and uncovered. The negative examples can exactly be obtained from negative bags but for

the positive examples, they are mixed in the positive bags together with the negative ones. Thus, if the MI data are applied to the original algorithm, it would be more difficult to get the correct concept since the positive examples contain a lot of noises. In order to handle these data, most of MI learning algorithms assume the area in the feature space where instances from different positive bags locating together as the target concept and this is formalised into the measure in DD.

The basic idea of our approach is to evaluate instances from the positive bags by using DD to show the strength of the instance to be positive using a value between 0 and 1. We then modified the heuristic function in FOIL to use the sum of DD values covered instead of the number of positive examples. For negative examples, as they are exactly labelled, we then use the number of negative examples in the same manner as the original function. Therefore, $|T_i^+|$ in formula (3) is changed to the sum of DD of positive examples, but $|T_i^-|$ still remains the same as in the original approach. The modified heurisfic function can be written as follows.

$$DD_s(T) = \sum_{T_i \in T} DD(T_i) \tag{6}$$

$$I(T_i) = -\log_2(DD_s(T_i^+)/(DD_s(T_i^+) + |T_i^-|)) \tag{7}$$

$$Gain(L_i) = DD_s(T_i^{++}) \times (I(T_{i+1}) - I(T_i)) \tag{8}$$

3.1 DD computation

In order to compute DD, the features describing each instance are necessary so that the instances can be separated. However, the first-order representation is so flexible that the feature can be described in several ways using one or more predicates. Therefore, predicates representing each instance have to be specified first.

In this research, the distance between predicates is calculated from the difference between each parameter in the predicates, then, these difference values are combined to the distance by using the Euclidean distance. For example, in the chemical compound data, we treat each atom in a molecule as an instance. The atom may be defined as *atm(compoundid, atomid, elementtype, atomtype, charge)*. The distance between two atoms can be computed by using the difference between parameters. However, a parameter may be discrete or continuous value. In case of continuous value, the difference is computed by substraction.

$$\Delta p = |p_1 - p_2| \tag{9}$$

In case of discrete value, the difference value will be 0 if they are the same value. Otherwise, it will bé 1.

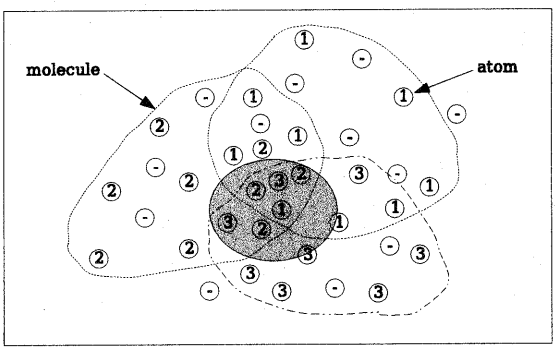

図 1 An example of problem domain for MI data (a molecule represents a bag and an atom is an instance in a bag.)

$$\Delta p = \begin{cases} 0 & \text{if } p_1 = p_2, \\ 1 & \text{otherwise.} \end{cases} \tag{10}$$

Then, the distance between two predicates can be calculated in the same way as formula 2 as

$$\|P_1 - P_2\|^2 = \sum_{p_{1i} \in P_1, p_{2i} \in P_2} (\Delta p)^2 \tag{11}$$

For example, the distance between *atm(m1, a1_1, c, 20, 0.1)* and *atm(m1, a1_2, o, 15, 0.2)* will be calculated from the difference between 'c' and 'o', '20' and '15' (these values are treated as discrete because it is the atom type), and '0.1' and '0.2' that is $1^2 + 1^2 + 0.1^2 = 1.01$. Figure 1 shows an example of problem domain for MI data that a molecule represents a bag and an atom is an instance in a bag. The DD approach tries to evaluate the area that instances from various positive bags locating together and is far from negative instances. From the figure, the shaded area has high DD value. For the chemical compound prediction, this area shows the characteristics of atoms that are common among the positive molecules.

3.2 The Algorithm

From the proposed approach, we examined the heuristic calculation in order to suit the MI data. We then considered modifying the algorithm.

Figure 2 shows the main algorithm used in the proposed system. This algorithm starts by initialising the set *Theory* to null, and the set *Remaining* to the set of positive examples. The algorithm loops to find rules and add each rule found to *Theory* until all positive examples are covered. It can be seen that this main algorithm is the same as FOIL. We modified the heuristic calculation which is in subroutine *FindBestRule*.

Subroutine *FindBestRule* is shown in figure 3. As explained above, the DD is applied for calculating a heuristic function. Another problem can nevertheless be considered

- $Theory \leftarrow \emptyset$
- $Remaining \leftarrow Positive(Examples)$
- While not $StopCriterion(Examples, Remaining)$
  - $Rule \leftarrow$ **FindBestRule**$(Examples, Remaining)$
  - $Theory \leftarrow Theory \cup Rule$
  - $Covered \leftarrow Cover(Remaining, Rule)$
  - $Remaining \leftarrow Remaining - Covered$

図 2 The main algorithm.

in this learning approach. When using the DD in counting the number of positive examples covered, there are many cases that the heuristic value may not increase during the searching process (the information gained equals to 0) because there are usually few true positive instances in one positive bag; hence, most of instances from positive bags have the DD value close to 0. This situation makes it difficult to find the best rules using only the hill-climbing approach as in FOIL since there are various candidates with the same heuristic value, aimlessly selecting the candidate may lead to the wrong direction. In order to avoid this problem, the beam search is applied to the proposed system so that the algorithm maintains a set of good candidates instead of selecting of the best candidate at that time. This searching method makes the algorithm able to backtrack to the right direction and finally get to the goal.

FindBestRule(Examples, Remaining)

- Initialise *Beam* with an empty rule, R as

  $$R(V_1, V_2, V_3, \ldots, V_k) \leftarrow$$

- $R \leftarrow BestClauseInBeam(Beam)$
- While $Cover(R, Negative(Examples))$
  - $Candidates \leftarrow SelectCandidate(Examples, R)$
  - For each C in $Candidates$
    - GenerateTuple$(Examples, Tuples)$
    - If C contains new relation Then re-calculate DD.
    - Calculate *heuristic value* for *Tuples* and attach to C.
  - $UpdateBeam(Candidates, Beam)$.
  - $R \leftarrow BestClauseInBeam(Beam)$

図 3 The algorithm for finding the best literals

4. Experiments and Discussion

We conduct experiments on datasets related to chemical structures and activity. The objective of these dataset is to predict characteristics or properties of the chemical molecules which consist of several atoms and bond between atoms. Therefore, the first-order logic would be more suitable for representing this kind of data since it is able to denote relations among atoms comprehensibly. This learning problem can also be considered as multiple-instance problem because each molecule may consist of a lot of atoms but only some connected atoms may effect on the characteristics or properties of the molecule. Therefore, we treat a molecule as a bag that consists of instances as atoms.

Approach	Accuracy
Proposed method	0.82
Progol	0.76 [1]
FOIL	0.61 [1]

表 1 Accuracy on the mutagenesis dataset comparing to the other ILP systems.

4.1 Mutagenesis Prediction Problem

The problem aims to discover rules for testing mutagenicity in nitroaromatic compounds which are often known to be carcinogenic and also cause damage to DNA. These compounds occur in automobile exhaust fumes and are also common intermediates used in chemical industry. The training examples are represented in form of atom and bond structure. 230 compounds were obtained from the standard molecular modelling package QUANTA [6].

- *bond(compound, atom1, atom2, bondtype)*, showing that there is a bond of *bondtype* betweem the atom *atom1* and *atom2* in the *compound*.
- *atom(compound, atom, element, atomtype, charge)*, showing that in the *compound* there is the *atom* that has element *element* of *atomtype* and partial charge *charge*

We conduct the experiment on this dataset and evaluate the results with 10-fold cross validation comparing to the existing ILP systems (FOIL and Progol). Table 1 shows the experimental results on this dataset. Examples of rules generated from the proposed system is shown in figure 4.

(1) `active(A) :- atm(A,B,C,D,E), D=95.`
The molecule that consists of an atom whose type is 95.

(2) `active(A) :- atm(A,B,C,D,E), D=27, E<0.`
The molecule that consists of an atom whose type is 27 and charge is less than 0.

(3)
```
active(A) :- atm(A,B,C,D,E), E>=0.816, E<0.823,
             atm(A,F,G,H,I), I>=0.817.
```
The molecule that has two atoms.
One has charge value between 0.816 and 0.823.
Another one has charge value greater than 0.817.

(4)
```
active(A) :- atm(A,B,C,D,E), D=27,
             atm(A,F,G,H,I), H=27,
             bond(A,B,F,J).
```
The molecule that has two atoms.
Both atoms are the same type which is 27 and there is a bond between them.

図 4 Examples of rules generated from the proposed system on the mutagenesis dataset.

4.2 Dopamine Antagonists Activity

This is another dataset that we conducted the experiment

(1) `active(A) :- atm(A,B,C,X,Y), C=h, X<6.`
The molecule that consists of the hydrogen atom whose position in the X axis is less than 6.

(2) `active(A) :- atm(A,B,C,X,Y), C=cl, Y>=5, X<3.`
The molecule that consists of the chroline atom whose position in the X axis is less than 3 and the position in the Y axis is greater than or equal to 5.

(3) `active(A) :- atm(A,B,C,X,Y), C=s, Y>=12, X>=6, bond(A,G,B,H), H=4.`
The molecule that consists of the sulfur atom whose position in the X axis is greater than or equal to 6 and the position in the Y axis is greater than or equal to 12. There is also a bond of type 4 from this atom.

図 5 Example of rules generated on the dopamine antagonist analysis.

on. We used the MDDR database of MDL Inc. This dataset contains 1,364 molecules that describe dopamine antagonist activity with atoms and bonds structure in the similar manner to the mutagenesis dataset in the previous experiment. Dopamine is a neurotransmitter in the brain that neural signals are transmitted via the interaction between dopamine and proteins known as dopamine receptors. An antagonists are a chemical compound that binds to a receptor, but does not function as a neurotransmitter. It blocks the function of the dopamine molecule. Antagonists for these receptors might be useful for developing schizophrenia drugs. There are four antagonist activities (D1, D2, D3, and D4). In this dataset, each atom is represented by its type and position in 2-dimensional area when the molecular structure is plotted.

We conducted the empirical experiment on this dataset in order to generate hypotheses for D1 activity. Therefore, D1 compounds are used as positive examples, other compounds are used as negative. The example of rules for D1 activity are shown in figure 5.

4.3 Discussion

From the experiments, we found that the proposed method generates more accurate rules when comparing to Progol and FOIL. Example of rules in figure 4 also shows the benefit of the proposed method which produces hypotheses in the first-order represenation, for instance, rule (3) and (4) consist of two atoms or a bond between atoms. These kinds of rule cannot be represented using the propositional logic. Moreover, when considering the knowledge discovery, only properties of one atom may not be good enough for describing the characteristic of molecule. Therefore, in this classification the first-order logic would be more suitable than the propositional logic. We will also try to improve the heuristic function or the search technique in order to generate hypotheses that incorporate a group of atoms and bonds between atoms.

5. Conclusions

We have presented the extension of FOIL for better handling multiple-instance data by using Diverse Density to evaluate tuples from positive bags. This evaluation is similar to setting the instances with different sets of feature which is actually the benefit of using the first-order representation. The experimental results show that our approach learns from the real-world problem better than Progol and FOIL.

For the future work, the scaling factor of the feature should be considered in the heuristic value calculation so that the system can produce more suitable heuristics from training data. Since the proposed approach works only in the top-down ILP system such as FOIL, it would be better to adopt this approach in the other kind of ILP system such as the one with bottom-up approach. Moreover, extending the MI learning problem to ILP would bring the possibility to various applications. We also plan to evaluate the proposed system to the other applications.

文　献

[1] Y. Chevaleyre, N. Bredeche, and J.-D. Zucker. Learning rules from multiple instance data : Issues and algorithms. In *Proceedings of the 9th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU02)*, 2002.

[2] Yann Chevaleyre and Jean-Daniel Zucker. A framework for learning rules from multiple instance data. In *Proceedings of the 12th European Conference on Machine Learning*, pages 49–60, Freiburg, Germany, September 2001.

[3] Thomas G. Dietterich, Richard H. Lathrop, and Tomás Lazano-Pérez. Solving the multiple instance problem with axis-parallel rectangles. *Artificial Intelligence*, 89(1-2):31–71, 1997.

[4] Oded Maron and Tomás Lazano-Pérez. A framework for multiple-instance learning. *Neural Information Processing Systems 10*, 1998. Available at ftp://ftp.ai.mit.edu/pub/users/oded/papers/NIPS97.ps.Z.

[5] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5(3):239–266, 1990.

[6] A. Srinivasan, S. Muggleton, R.D. King, and M.J.E. Sternberg. Mutagenesis: ILP experiments in a non-determinate biological domain. In S. Wrobel, editor, *Proceedings of the 4th International Workshop on Inductive Logic Programming*, volume 237, pages 217–232. Gesellschaft für Mathematik und Datenverarbeitung MBH, 1994.

[7] Q. Zhang and S. A. Goldman. Em-dd: An improved multiple-instance learning technique. In T. G. Dietterich, S. Becker, and Z. Ghahramani, editors, *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002. MIT Press.