Voting Based Learning Classifier System for Multi-Label Classification

Kaveh Ahmadi-Abhari
Computer Science and Engineering Department, International Campus
Shiraz University
Shiraz, Iran
sk-abhari@cse.shirazu.ac.ir

Ali Hamzeh
Computer Science and Engineering Department
Shiraz University
Shiraz, Iran
ali@cse.shirazu.ac.ir

Sattar Hashemi
Computer Science and Engineering Department
Shiraz University
Shiraz, Iran
s_hashemi@shirazu.ac.ir

ABSTRACT
Learning Classifier Systems (LCSs) are rule-based systems with a discovery mechanism to find additional meaningful rules according to the results of its previous experiments. LCSs were designed to deal with both single and multistep problems. In the first category, almost all major studies focus on the single-label classification problems. However, there are more complex problems that require multi-label classification. The aim of this study is to take advantage of the power and ability of LCSs for solving multi-label classification problems. The main idea behind this research is to guide the discovery mechanism by a prior knowledge. This prior knowledge is defined as a voting mechanism that realizes the quality of the existing rules and is used in discovering new rules. Our proposed system is called Voting Based LCS (VLCS). The experimental results show the proposed method has potential for future research and progress.

Categories and Subject Descriptors
I.2 [**Artificial Intelligence**]: Learning

General Terms
Algorithms, Theory

Keywords
LCS, Multi-label Classification, Voting

1. INTRODUCTION
Classification is one of the most important problems under investigation in machine learning. Based on the number of classes that each input instance might have, classification problems can be divided into two main categories: (i) single-label and (ii) multi-label problems. In single-label problems, only one class is associated to each instance. Most previous studies are mainly related to this category of problems. However, in many domains, each instance can belong to more than one class for example, in the text categorization and bioinformatics field [3] and as it is obvious, classification in this kind of environments is more complex than classification in single-label data.

GECCO'11, July 12–16, 2011, Dublin, Ireland.

Learning Classifier Systems (LCSs), first proposed by Holland in 1971 [8], are systems that use a knowledge base of rules and include a discovery mechanism. The discovery mechanism uses an evolutionary algorithm [2] to produce more promising rules. Many different extended version of LCS have been proposed in the literature [11] which could be divided into two main categories based on the evolutionary algorithm approach which is used: (i) Michigan and (ii) Pittsburgh. In the Michigan family, each individual in the population is one classifier, however, in the Pittsburgh family; each individual in the population is a set of classifiers.

Although a lot has been done in terms of classifications using LCSs, most of these studies have been conducted for single-label classification problems and multi-label classification is in its inception [17] [18]. The aim of this study is to propose a new efficient LCS-based approach for multi-label classification that has comparable results to other multi-label classification approaches.

The remaining of this paper is organized as follows: Section 2 of this paper gives an overview of multi-label classification. Section 3 presents the usual structure and components of the LCSs. In section 4 and 5, we describe the main idea behind Voting Based LCS and the experimental results are presented in section 6.

2. MULTI-LABEL CLASSIFICATION
The need for multi-label classification methods is increasing rapidly in modern applications [13] [19] [20]. In many classification problems, each instance is associated with a single class label. On the other hand, in some domains e.g., text categorization and bioinformatics, instances are associated with more than one class which are called multi-label classification problems. In a more formal manner, it could be said that in single-label classification problems, each instance is associated with a single label l of a set of labels L, $|L|>1$. In a multi-label classification problem instances are associated with a set of labels $Y \subseteq L$ [14].

The approaches to tackle multi-label classification problems are categorized into two major families: (i) label ranking (LR) and (ii) multi-label classification (MLC) [15]. The output of LR is expected to be a *rank* of available classes for each input instance.

In MLC, the aim is to *define* relevant classes for the input instance. Both of these tasks are important in mining multi-label data. For example in text categorization, a typical task is to *rank* the available topics for related documents based on their textual content and another task is to *define* the classes that the document belongs to. There are some methods that do both tasks simultaneously. These methods are called multi-label ranking (MLR) [1].

There are several techniques to deal with these tasks. As previously proposed [15], we can group these methods into two categories: (i) problem transformation, and (ii) algorithm adaptation. In the first category, problem transformation, we transform the multi-label data into similar single-label data. A large number of learning tasks is available for processing single-label data. The drawback of these methods is that they do not consider the correlation between the classes. The second category, algorithm adaptation, is the familiar single-label learning tasks which are adapted to deal with multi-label data [5] [7] [20]. A special type of problem transformation, called label-based transformation [3], is also available. In this transformation, we have N binary classifiers each associated with one of the classes, where N is the number of class labels.

3. LEARNING CLASSIFIER SYSTEMS

Learning Classifier Systems (LCS) has been proposed to be able to deal with a wide variety of machine learning problems. LCSs are rule-based systems composed of rules generally in the form of:

IF antecedent THEN consequent

The other required parts for a LCS are an inference engine, conflict-resolution and credit-allocation system, and a discovery mechanism [12]. The inference engine is responsible for diagnosis of the matched rules for the current input. In many LCSs, the representation of the antecedent part of the rules is from the ternary alphabet {0, 1, #}. Where 1 matches with 1, 0 matches with 0 and # (don't care) match with both 0 and 1. The # acts as a wildcard [2] that allows us to have generalization. For example the rule condition 0#1 matches both input 001 and 011. We can also consider different representations for the consequent part but usually a binary representation for this part of rules is considered. Therefore, in LCSs we are faced with a population of individuals, called rules, where each of them covers a part of the problem domain to control it. Because of the existence of the "*don't care"* sign in the alphabet, there might be several rules that match with each input instance. In such cases to determine which of the matched rules should act in the system, a conflict-resolution (CR) system is necessary. Additionally, a credit-allocation (CA) system is used to determine one or more measures of utility for each rule based on previous experiences [12].

The system's accuracy can be improved by manipulating the rules based on previous experiments. The discovery component is responsible for discovering better rules and improving existing ones through an evolutionary algorithm. Evolutionary algorithms are search algorithms based on natural selection theory and genetics principles [2]. To guide the evolution, we need a measure for the classifiers, called *fitness* [9], which estimates the quality of the information the classifier carries about the problem. The better the measure, the more reliability it has in the evolutionary algorithm. An evolutionary algorithm selects parents form the population of individuals and produces a new offspring using evolutionary operators. Different selection mechanisms and operators are available [6]. One of these operators is the *mutation operator*. Mutation operators randomly change small parts of the rule to reach a neighborhood of the rule that we expect to be a more promising rule. In this operator the fitness measure is used to define the *mutation rate*. The mutation rate is the probability that the mutation operator acts on each part of the rule.

4. VOTING BASED LCS

Here we propose a new LCS which we call *Voting Based LCS (VLCS)* as a method for guiding the discovery mechanism. The main idea behind VLCS is to use a voting mechanism to realize the quality of the rules. These votes are given by the input instances to their matched rules and are used as a fitness measure in the discovery mechanism. The discovery mechanism uses this prior knowledge to have more meaningful operations. In this way, we expect more robust rules form the rule discovery operations.

The given votes should have the ability to describe the quality of the rules accurately. In other words, we require promising votes that can be used instead of the fitness measure. For this reason, we define different types for the rules such that each of these types describes the quality status the rule might have. For example, in a single-label classification problem, rule types might be correct or wrong. These types defined for the rules are employed as voting options. In the example above, each rule might receive a "correct" or "wrong" vote from each matched input instance. Thus, a rule receives a combination of "correct" and "wrong" votes from its matched input instances. This combination of votes acts as the prior knowledge that is gained from the voting mechanism of past experiences to guide the discovery mechanisms to have more control on the operators. In the next section, we introduce the idea with more detail for a multi-label classification problem.

5. VOTING BASED LCS FOR MULTI-LABEL CLASSIFICATION PROBLEM

As described above, in VLCS, to have a meaningful voting mechanism, we should determine available voting options each input instance can assign to each matched rule. We denote these voting options as different types a rule might have. In a multi-label classification problem, we define five types for the classifiers. They might be correct, wrong, subset, superset or partial-set. We describe these types as below:

- A rule is **correct** from a matched input instance's perspective if the rule predicts all expected classes of the data correctly.
- A rule is **wrong** from a matched input instance's perspective if the rule predicts none of the expected classes of the data correctly.
- A rule is **subset** from a matched input instance's perspective if the rule predicts some expected classes of the rule correctly but fails to identify other expected classes.
- A rule is **superset** from a matched input instance's perspective if the rule predicts all of the expected classes correctly and wrongly assign some other non-expected classes.

- A rule is **partial-set** from a matched input instance's perspective if the rule predicts some of the expected classes correctly, but fails to identify all expected classes and also wrongly assigns some other non-expected classes.

For example in a multi-label classification problem, as mentioned, we have rules in the form below where "/" delimit consequent part.

Antecedent / Consequent

###1 / 110

0011 / 001

The antecedent part of the rule matches with the feature vector of the input instances. The consequent part of the rule is the output, which are the classes predicted by the rule for each matched input instance. In this representation, we consider one bit for each class in the consequent part, where the value 1 in the bit indicates existence of the respective class. For example, the consequent 011 means the first class is not relevant to the input, but the second and third classes are assigned to the input instance. If a rule predicts classes of an input instance as expected the rule receives the vote "correct". The rule receives the vote "wrong" if it predicts none of the expected classes correctly, receives the vote "subset" if it predicts some expected classes correctly, receives the vote "superset" if predicts all the expected classes and some additional non-expected classes, and receives the vote "partial-set" if predicts some of the expected classes and some additional non-expected classes as described above. Table 1 presents an example of a rule that might get different votes from matched input instances.

Table 1: Example of how votes are given to a rule from input instances

Input Instance	Expected output	Selected Rule	Rule Type
0001	1, 2	###1 / 110	Correct
0101	1, 2, 3	###1 / 110	Subset
0111	1	###1 / 110	Superset
1111	1,3	###1 / 110	Partial-set
0011	3	###1 / 110	Wrong

The next step is to use this stored information about the rules as prior knowledge in the discovery mechanism. Consider a rule that all matched input instances vote it as a superset rule. With this information, we can infer that this rule is covering an appropriate area of the problem but it predicts greater number of classes for the matched input instance. With this knowledge we understand that the number of the classes the rule predicts should be subtracted.

In the discovery mechanism an evolutionary algorithm with four mutation operators is defined. Two of them act in the antecedent part of the rule, and the other two act in the consequent part. One of the operators that acts in the antecedent part, named MA-G, works to generalize the rule by flipping the 0 or 1 bits to # and another, named MA-S, specializes the rule by flipping # bits to 1 or 0. The other two operators that act in the consequent part are MC-S that subtract the number of predicted classes by flipping 1 bits to 0; and another, named MC-A, adds more classes to predicted classes by flipping 0 bits to 1.

The votes each rule has received guide which mutation operator should act. For example, if the rule has received votes as being a superset, the discovery mechanism is guided to use the MC-S operator to subtract some classes. Table 2 presents some of the guidance the discovery mechanism gets by the given votes. Note that in cases in which more than one mutation operator is activated, all of the operators act simultaneously to produce a new offspring. The probability of activation of each of them could depend on the combination of votes the rule has received.

Table 2: Mutation operators chosen to act based on rule votes

Rule Received Votes	Activated Mutation Operator
Correct	MA-G
Subset	MC-A
Superset	MC-S
Partial-Set	MC-A, MC-S
Wrong	MC-A, MC-S
Correct, Subset	MA-S
Correct, Superset	MA-G
Correct, Partial-Set	MA-S
Correct, Wrong	MA-S
Wrong, Subset	MA-S, MC-A
Wrong, Partial	MA-S
Correct, Subset, Wrong	MA-S, MA-G

The mutation operator performs bit flipping using a probability, which is the mutation rate. To define the mutation rate, we use the strength of the rule. The strength of a rule is the amount of reward we predict the system to receive if the rule acts [9]. Here, we define the strength of a rule as the mean of the rewards the rule gets over time. The mutation rate has reverse association with the strength of a rule: the more the strength, the less the mutation rate.

To calculate the reward of one rule, we use an alteration of the reward estimate mechanism which is previously proposed for multi-label classification rules [18]. We define the reward estimate measure as below:

$$R = 1 - \frac{\left| C_{rule} \Delta C_{expected} \right|}{\left| C_{rule} \cup C_{expected} \right|} \qquad \text{Equation (1)}$$

Where C_{rule} is the set of classes predicted by the rule and $C_{expected}$ is the set of expected classes the input instance has and Δ is the symmetric difference operator. The symmetric difference of two sets is the number of elements which are in either of the sets and not in both of them. The symmetric difference between two sets A and B can be expressed with the XOR operation as below:

$$A \Delta B = \left\{ x : \left(x \in A \right) \oplus \left(x \in B \right) \right\} \qquad \text{Equation (2)}$$

Using this metric, the subset rule receives a reward that is not as much as the reward that the correct rule receives because of the expected classes it fails to predict. In the same way, the superset rule receives a reward that is not as much as the reward that the correct or subset rules receive because of the non-expected classes that is wrongly predicted by the rule, and the partial-set rule receives a reward based on the number of expected classes correctly predicted by the rule and the number of non-expected classes wrongly predicted by the rule. Table 3 presents an example of how the reward estimate measure works. This is what the credit-allocation task is responsible to do. This measure is also used by the conflict-resolution system to determine which of the matched rules should act.

Table 3: Example of how the credit-allocation system rewards the rules

Input Instance	Expected output	Selected Rule	Reward
0001	1, 2	###1 / 110	1
0101	1, 2, 3	###1 / 110	0.66
0111	1	###1 / 110	0.50
1111	1,3	###1 / 110	0.33
0011	3	###1 / 110	0

6. THE EXPERIMENTAL RESULTS

In this section, we present an experiment on the described VLCS for multi-label classification on a binary dataset in the bioinformatics domain. The creators of this dataset [4] have originated it from the Uniprot dataset [16] which is one of the largest protein datasets. In the dataset used in this experiment, each instance represents a protein, and the absence or presence of each of the 152 PROSITE patterns [10] is shown as a binary attribute. This dataset has 135 instances in which each instance could have one or both of the available class labels, Anti-oncogene and Apoptosis. For the VLCS, we use a 5-fold cross validation in which the training part is used to evaluate the rules using the voting mechanism described above. The credit-allocation system is also used to assign the related reward to each rule. We use a fixed size population which has 500 members which initially are the most general possible rules. In each generation, each rule is voted by its matched instances, and for each matched instance a reward is assigned to the rule by the credit-allocation system; afterwards the discovery mechanism system uses the defined mutation operators to discover new rules using the generated knowledge as explained. The new generated rules would also be similarly evaluated. The combination of the best rules among the parents and the off-springs make the next generation. We stop the training phase if the mean strength of the rules decreases in 5 consecutive generations. It should be noted that the probability of flipping the # bits to 1 or 0 in the MA-S operator comes from the proportion of 0 and 1 in the dataset in this experiment.

The results on the test part of the data are then compared with another multi-label classification method, ML-KNN [20]. In ML-KNN, we set the number of neighbors to 3. Table 4 presents the mean and the standard deviation of 5 experiments for both methods with the same cross validation sets.

Table 4: Comparison of predictive accuracy (%) in the test set for both algorithms on 5 experiments

Method	Accuracy
ML-KNN	0.912 ± 2.4
VLCS	0.895 ± 3.2

7. CONCLUSION AND FUTURE WORK

Because of the ability of LCSs in adaptation, LCSs are widely applied to different machine learning problems. Classification is one these areas in which the results of using LCSs have been promising. The main focus of this study was to employ the power of LCSs in multi-label classification problems that are more complex classification problems. To achieve this goal a new discovery mechanism for LCSs is presented that uses the prior knowledge gained from past experiences in the discovery mechanism to guide the evolutionary operators. The discovery mechanism defined in this paper uses a voting mechanism according to how correct the rule is to refine the rules. We call the LCS that works with this discovery mechanism Voting Based LCS.

The primary conclusion of this study is that guiding the discovery mechanism with a prior knowledge, such that is used in VLCS, can help us solve applicable problems. In the multi-label classification problem that was investigated in this paper on one binary dataset, the result of the VLCS method was comparable to the ML-KNN method which shows the potential of this method.

Future works involves a representation for dealing with numeric and nominal datasets and thus expand the application of this method on more complex datasets. Future studies on scalability and stability of the system is necessary. Additional studies on system performance in dealing with imbalanced data and noise is also required. There is also potential for improving evolutionary operators, guiding mechanism and rule refinement of this method.

8. REFERENCES

[1] Brinker, K., Fürnkranz, J. and Hüllermeier, E. A Unified Model for Multilabel Classification and Ranking. In *Proceedings of the 17th European Conference on Artificial Intelligence (ECAI'06)*. IOS Press, 2006, 489-493.

[2] Bull, L. Learning Classifier Systems: A Brief Introduction. *Applications of Learning Classifier Systems*, Springer, 2004, 3-14.

[3] de Carvalho, A. and Freitas, A. A Tutorial on Multi-label Classification Techniques. *Foundations of Computational Intelligence*, Springer, Berlin / Heidelberg, 2009, 177-195.

[4] Chan, A. and Freitas, A. A New Ant Colony Algorithm for Multi-Label Classification with Applications in Bioinformatics. In *Proceedings of the 8th Annual Conference in Genetic and Evolutionary Computation (GECCO'06)*, ACM Press, 2006, 27-34.

[5] Clare, A. and King, R.D. Knowledge Discovery in Multi-label Phenotype Data. In *Proceedings of the 5th European Conference on Principles of Data Mining and Knowledge Discovery (PKDD'01)*, Springer, 2001, 42-53.

[6] Eiben, A.E. and Smith, J.E. *Introduction to Evolutionary Computing.* Springer, 2007.

[7] Elisseeff, A. and Weston, J. A Kernel Method for Multi-Labelled Classification. *Advances in Neural Information Processing Systems*, 2001, 681-687.

[8] Holland, J.H. *Adaptation*. Progress in Theoretical Biology, New York, 1976.

[9] Holmes, J.H., Lanzi, P.L., Stolzmann, W. and Wilson, S.W. Learning classifier systems: New models, successful applications. *Information Processing Letters*, 2002, 23-30.

[10] Prosite, *http://ca.expasy.org/prosite/* (visited 2011)

[11] Sigaud, O. and Wilson, S. Learning Classifier Systems: A Survey. *Soft Computing - A Fusion of Foundations, Methodologies and Applications*, Springer, 2007, 1065-1078.

[12] Smith, R.E. and Goldberg, D.E. Reinforcement Learning with Classifier Systems: Adaptive Default Hierarchy Formation. *Applied Artificial Intelligence*, Taylor & Francis, Inc., 1992, 79-102.

[13] Trohidis, K., Tsoumakas, G., Kalliris, G. and Vlahavas, I. Multilabel Classification of Music into Emotions. In *Proceedings of the 9th International Conference on Music Information Retrieval (ISMIR'08)*, 2008.

[14] Tsoumakas, G. and Katakis, I. Multi Label Classification: An Overview. *International Journal of Data Warehouse and Mining*, Idea Group Publishing, 2007, 3, 1-13.

[15] Tsoumakas, G., Katakis, I. and Vlahavas, I. Mining Multi-label Data. *Data Mining and Knowledge Discovery Handbook*, Springer, 2009.

[16] Uniprot database, *http://www.uniprot.org* (visited 2011)

[17] Vallim, R. M. M., Duque, T. S., Goldberg, D. E. and Carvalho, A. C. The multi-label OCS with a genetic algorithm for rule discovery: implementation and first results. In *Proceedings of the 11th Annual conference on Genetic and evolutionary computation (GECCO'09)*, ACM, 2009, 1323-1330.

[18] Vallim, R.M.M., Goldberg, D.E., Llorà, X., Duque, T.S. and Carvalho, A.C. A New Approach for Multi-Label Classification Based on Default Hierarchies and Organizational Learning. In *Proceedings of the 10th Annual Conference in Genetic and Evolutionary Computation (GECCO'08)*, ACM, 2008, 2017-2022.

[19] Yang, S., Kim, S.K. and Ro, Y.M. Semantic Home Photo Categorization. *IEEE Transactions on Circuits and Systems for Video Technology, Circuits and Systems for Video Technology*, 2007, 324-335.

[20] Zhang, M.L. & Zhou, Z.H. ML-kNN: A Lazy Learning Approach to Multi-Label Learning. *Pattern Recognition*, Elsevier Science Inc., 2007, 2038-2048.