Automatic Case Generation for Pattern Classification

Zahra A Shah, M.M.Awais & Shafay Shamail
Lahore University of Management Sciences, Lahore, Pakistan
{zahraas, awais & sshamail}@lums.edu.pk
http://cs.lums.edu.pk

Abstract. This paper shows how automatic cases can be generated using statistical learning methods such as support vector machines. This work also introduces new conflict resolution and revision strategies for enhancing the performance of the Case Based Reasoning system. The input to this system is a set of training samples out of which automatic cases are generated using support vector machines. These cases are subsequently used to classify unknown patterns and if a case is not found, it is added to the existing set of cases for future reference. In order to test the effectiveness of the proposed system, this algorithm was tested on a benchmark dataset and it was observed that it compares well with similar studies conducted.

Keywords: Case-based reasoning, pattern classification, support vector machine, fuzzy-logic.

1 Introduction

Case Based Reasoning (CBR) is a relatively new paradigm developed to solve problems by using the important properties of *retrieve, reuse, revise and retain*. These properties of CBR are very powerful and intuitive as human beings also use them for problem solving. Normally there are two parts to a CBR system i.e. a *Case Based Reasoner* and a *Case-Base*. The Case-Base contains history of similar past cases and the Case Based Reasoner retrieves cases from the case-base and tries to find an appropriate solution for an unknown problem presented to it [1].

CBR covers a wide range of methods for generating, organizing, retrieving and utilizing the knowledge retained in past cases. Some methods can be purely automatic or may interact heavily with the user for support and guidance. On the other hand, some methods assume a large amount of widely distributed cases in their case-base while others have a limited set of typical cases. These methods can be roughly categorized as: *exemplar-based reasoning, instance-based reasoning, memory-based reasoning, case-based reasoning and analogy-based reasoning* [2]. The domain of CBR is versatile and has found many applications in various fields such as medicine, engineering, law, finance and manufacturing just to name a few. CBR has also been successfully applied to solve pattern classification problems encountered frequently in the aforementioned fields. The following literature review will give an overview of some important work done in the context of CBR pattern classifiers.

The authors in [3] have proposed an improved approach to diagnose lung disease which is an amalgamation of two concepts namely: CBR and Rule-Based Reasoning (RBR). Their technique assigns an equal weight to the results obtained by the CBR and the RBR systems and calculates the combining value for classification purposes. If the disease diagnosed by both of these systems is different, then it is up to the physician to come to a conclusion based on experience. They have tested their approach on a hypothetical dataset, which proves the feasibility of the proposed approach.

The work in [4] presents a system for pattern classification of Respiratory Sinus Arrhythmia classified with CBR and RBR using physiological time series analysis of heart and respiratory measurements. The classification process is divided into two stages i.e. analysis of respiration measurements and heart measurements. From these findings, cases are generated for the case-base. This system was tested on 100 pre-recorded measurements from a normal population of 17-year olds, and the percentage classification on the first attempt was reported to be 73.4%.

In [5], the authors propose a fast approach which builds a hybrid CBR classifier by incorporating the feature reduction and case selection processes. For feature reduction, they have modified the fast rough-set based approach. For case generation, they have developed four algorithms which use the concepts of large coverage, similar coverage and k-Nearest Neighbour (k-NN). They have also compared their technique with Kernel Principal Component Analysis (KPCA) and Support Vector Machine (SVM). They have tested their approach on four datasets i.e. *House-votes-84*, *Mushroom*, *Multiple features* and *Text*

documents taken from [8]. Their results show that the rough-set based approach outperforms KPCA, SVM and a combination of both.

In [6], the authors propose a case generation technique which uses a fuzzy representation of rough sets. Fuzzy rules are generated using rough-set theory, which are then mapped to generate cases. The fuzzy membership function for each case is also stored in the case-base, which are used at the time of case retrieval as a similarity measure. This technique has been tested on three real-life data sets: *Forest Covertype*, *Multiple features* and *Iris* taken from [8], and compared with other standard techniques used for case generation.

In [7], trained SVM has been used to generate fuzzy rules. These rules are then used for pattern classification purposes. Just like other pattern classifiers, this technique is not revisable (i.e. an unknown pattern if not classified correctly will be discarded). Also, the authors of [7] did not explicitly mention the conflict resolution strategy used to resolve conflicts in the rule-base obtained. Thus the use of SVM for generating fuzzy rules can be modified in the following ways:

1. The non-revisable nature of pattern classifiers can be changed by exploiting the revision property of CBR systems.
2. The algorithm in [7] can be modified by incorporating an improved conflict resolution strategy that would reduce the classification error of the system considerably.

In light of the above two suggestions, we have incorporated the modifications and tested the new algorithm on a benchmark dataset of liver disorders [8].

For the sake of completeness, we will briefly explain the system developed in [7] before moving on to explain our suggested modifications. The authors have developed a technique to extract fuzzy rules from trained SVM. They train a SVM to obtain support vectors. These support vectors are then projected in the coordinate axes to obtain membership degrees of each attribute of the support vectors. The next step is to construct a given number of fuzzy-sets for each attribute of the support vectors. Finally, fuzzy rules are generated by using the support vectors and the given fuzzy-sets. This technique is illustrated in the Fig 1:

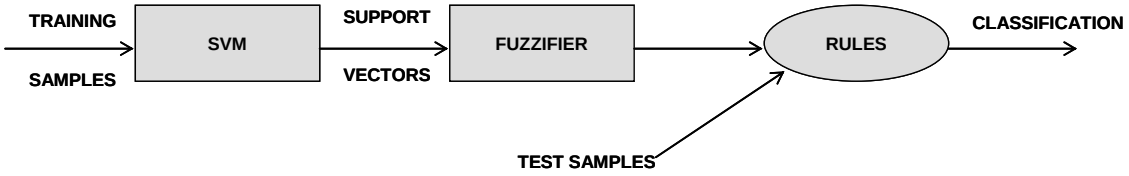

Fig. 1. Extraction of fuzzy rules from SVM

For evaluating this technique, the authors use a benchmark dataset of liver disorders taken from [8]. This dataset contains 345 patterns, broken down into 50-50 training-test sets. The SVM was trained using the Radial Basis Function (RBF) as kernel and with parameters C = 100 and $\sigma^2 = 0.5$. For generating fuzzy rules, they have used 3-fuzzy sets, 5-fuzzy sets and 7-fuzzy sets. From their analysis, they have concluded that by using the abovementioned experimental setup and 3-fuzzy sets, they achieve the best classification error of 46.8%.

As mentioned before, the work presented in this paper differs from the previously explained strategy of [7] in at least two dimensions. These differences are explained thoroughly in the next section.

2 Generating Automatic cases for pattern classification

The following modifications are suggested in the technique suggested by [7]:

1. Conflict resolution has been incorporated.
2. Test the system, *revise* and *retain* new cases in the case-base.

After integrating the proposed modifications, the automatic case generating algorithm is summarized below in Fig 2:

```
STEP 1:
        data = getTrainingData(file);
        kernel = getKernel('RBF',C,sigma);
        sv = trainSVM(data, kernel);
STEP 2:
        proj = projectAttributes(sv);
        [fuzzyCases, numCases] = generateFuzzyCases(proj);
STEP 3:
        for all the fuzzy cases obtained
            case = getCases(antecedent); //get all the cases with same antecedent
            if size(case) == 2 AND different consequents
              discard both the cases
            if size(case) == 2 AND same consequents
              pick any one case
            else
                 pick the case label where frequency of consequent is higher
            end
            add case to the case-base
        end
STEP 4:
        for all the patterns in the test set
            newCase = getCase (unKnownPattern);
            if newCase == NULL
               add case to case-base
            else
                assign label of that case
            end
        end
```

Fig. 2. Proposed Algorithm

As we are extending a technique already proposed, the first two steps of our algorithm remain the same. Once the support vectors are fuzzified using 3-fuzzy sets, we get our case-base. At this point a lot of conflicting cases are obtained which if not resolved will confuse the classifier. Conflicts are resolved by using step 3 of our algorithm. After the application of this step, we obtain a conflict-free case-base. This case-base is now used to classify unknown patterns of the test set. If an unknown pattern conforms to a case in the case-base, it is assigned the class of the case. Otherwise if the pattern does not conform to any case in the case-base, we know that an example is encountered for which we have no history. We therefore, add this new case to the case-base which is then used for classifying future unknown samples. After making these required modifications in the technique, we have reduced the classification error to approx 36% for a 50-50 break-up with 3-fuzzy sets, which is drastic improvement over 46.8% error mentioned in [7]. The complete procedure is illustrated in Fig 3:

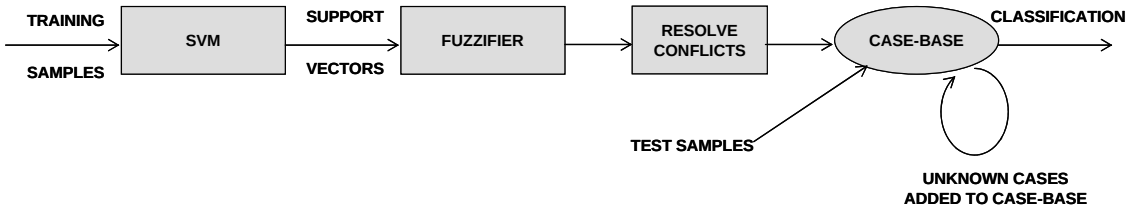

Fig. 3. Proposed modifications

We have conducted more experiments on the same dataset and performed a detailed analysis, which will be explained shortly.

3 Experimental Results

The proposed algorithm has been tested on a medical dataset of liver disorders. The results of the proposed algorithm have been compared with the technique mentioned in [7]. Our automatic case generation module has been implemented in MATLAB by using the MATLAB SVM toolbox [9]. For evaluating this system, a benchmark classification dataset (Bupa Liver Disorders [8]) was used. It consists of 345 samples and contains information about people who are either suffering from liver disease or are not suffering from liver disease. The following table summarizes the results obtained for this dataset given how the dataset was broken into training and test sets:

Table 1. Experimental Results

Sr #	Experiment	# of Support Vectors	# of Non-Conflicting Cases	# of Unknown Cases	Classification error (%)	# of Cases after revision	New Classification error (%)
1	50-50(selected in-order)	115	33	48	56.6	64	36
2	50-50(selected randomly)	112	21	12	37.5	52	33.5
3	70-30(selected randomly)	161	31	7	37	45	32
4	80-20(selected randomly	177	34	5	36	44	31.5

The results are summarized in the table above and are explained below:

In *Experiment 1,* we divided the dataset (50% - training, 50% - testing) in the given order. For this, we obtained 115 cases (support vectors), which was reduced to 33 after conflict resolution. When this case-base was used to identify unknown cases, 56.6% classification error was obtained and 48 un-identified cases were reported. These cases (after conflict resolution) were added to the case-base. After revision our case-base had 64 cases and the classification error thus obtained was reduced to 36%, which is a good improvement as compared to 46.8% as reported in [7]. Although randomizing a dataset is a key step before training any classifier, the result obtained clearly shows a marked improvement after revision is incorporated in the system.

In *Experiment 2*, we divided the dataset (50% - training, 50% - testing) by random selection. It was observed that the classification error was reduced to 37%, which was further reduced to 33.5% after revision. Further experiments (Experiment 3 & 4) show that as we increase the number of samples in the training set, the classification error for the test set decreases, which is further decreased when revision is done.

These results are also presented in Fig 4. The graph shows the best possible training-test set break-up after which the performance of the system becomes constant. The most appropriate break-up of training and test sets is 70-30, which gives us a classification error of 32% after revision and updating of the case-base.

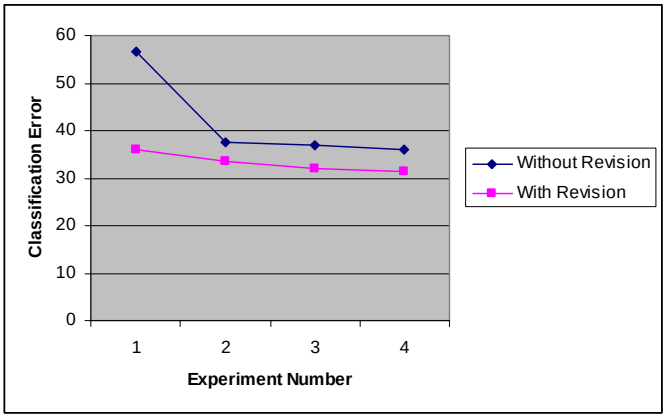

Fig. 3. Best cut for training-test sets

From all the experiments stated above, we can conclude that our technique is a good improvement over the rule extraction technique mentioned in [7].

4 Future Directions

This work is an initial attempt to generate automatic cases for pattern classification by using SVM's and fuzzy-logic. We have proposed and tested two important enhancements to the work done by [7]. From the results obtained and critical analysis, we suggest the following ways in which this work can be further enhanced:

1. We intend to test and obtain results for kernels other than the Radial Basis function. Some of the kernels that can be used are: polynomial, Gaussian radial basis and sigmoid. The motivation behind experimenting with these parameters is just an attempt to try and reduce the classification error of the system further than it already is. Similar experiments can be conducted while selecting membership functions to obtain membership degrees of the attributes and by increasing or decreasing the number of fuzzy-sets.
2. This work was tested on a medical database of liver disorders. This was done so that we could compare our proposed technique with the one suggested by [7]. We now intend to test this system on other kinds of datasets e.g. image, voice data etc. so that we can infer if this method generalizes well or not.
3. Right now, the parameter selection for the SVM is completely manual i.e. training parameters for RBF are selected by trial and error. We intend to make this process completely automatic. For this purpose, we intend to explore techniques like v-fold cross validation, k-means and EM-Algorithm.
4. This work is limited to binary-class support vector machines. Due to this reason, this classification module is limited and can be used for classifying binary-class datasets only. There are two main categories by which multi-class classifiers can be implemented using SVM's i.e. multi-class SVM's implemented by combining various binary-class SVM's and by solving the multi-class SVM problem in one step, which is a complex optimization problem [10, 11]. We intend to extend our technique for multi-class SVM's so as to remove the limitation of bi-class classification.

5 Conclusion

This research is an initial attempt to generate automatic cases for pattern classification using support vector machines and fuzzy-logic and is an improvement over the work done by [7]. We have made the

existing method better by incorporating conflict resolution and revision, which is a key property of CBR systems. The results of this study are promising and suggest that an average classification error of 32% can be achieved by appropriately adjusting the training parameters. The study also shows a good improvement of 31.6% in the classification error.

References

1. Sankar K. Pal, Tharam S. Dillon and Daniel S. Yeun, Soft Computing in Case Based Reasoning, Springer-Verlag London UK (2000).
2. A. Aamodt, E. Plaza (1994), Case-Based Reasoning: Foundational Issues, Methodological Variations, and System Approaches. AI Communications. IOS Press, Vol. 7: 1, pp. 39-59.
3. Nguyen Hoang Phuong, Prasad N.R., Dang Huu Hung, Drake, J.T., Approach to combining case based reasoning with rule based reasoning for lung disease diagnosis, IFSA World Congress and 20th NAFIPS International Conference, 2001, page(s): 883-888.
4. Markus Nilsson, Peter Funk. A case-based classification of respiratory sinus arrhythmia, Advances in Case-Based Reasoning, ECCBR'04, September 2004: 673-685.
5. Yan Li, Simon C.K. Shiu, Sankar K. Pal, Combining Feature Reduction and Case Selection in Building CBR Classifiers, IEEE Transactions on Knowledge and Data Engineering, March 2006 pp. 415-429.
6. Sankar K. Pal, Pabitra Mitra, Case Generation Using Rough Sets with Fuzzy Representation, IEEE Transactions on Knowledge and Data Engineering, March 2004, pp. 292-300.
7. Adriana da Costa F. Chaves, Marley Maria B. R. Vellasco, Ricardo Tanscheit, Fuzzy Rule Extraction from Support Vector Machines, Hybrid Intelligent Systems, 2005 Page(s):6 pp.
8. Machine Learning Database at University of California Irvine: ftp://ftp.ics.uci.edu/pub/machine-learining-databases/
9. MATLAB Support vector machine toolbox: http://theoval.sys.uea.ac.uk/~gcc/svm/toolbox/
10. C.-W. Hsu and C.-J. Lin, A comparison of methods for multi-class support vector machines, *IEEE Transactions on Neural Networks*, 13(2002), 415-425.
11. Kai-Bo Duan, S. Sathiya Keerthi, Which Is the Best Multiclass SVM Method? An Empirical Study, Multiple Classifier Systems 2005: 278-285.