Rule Learning With Negation: Issues Regarding Effectiveness

S. Chua, F. Coenen, G. Malcolm

University of Liverpool
Department of Computer Science,
Ashton Building, Ashton Street, L69 3BX Liverpool,
United Kingdom

s.chua@liverpool.ac.uk
coenen@liverpool.ac.uk
grant@liverpool.ac.uk

ABSTRACT: *An investigation of rule learning processes that allow the inclusion of negated features is described. The objective is to establish whether the use of negation in inductive rule learning systems is effective with respect to classification. This paper seeks to answer this question by considering two issues relevant to such systems; feature identification and rule refinement. Both synthetic and real datasets are used to illustrate solutions to the identified issues and to demonstrate that the use of negative features in inductive rule learning systems is indeed beneficial.*

KEYWORDS: *Inductive rule learning, Negation, Classification*

1. Introduction

Inductive Rule Learning (IRL) is a generic term used to describe machine learning techniques for the derivation of rules from data. IRL has many applications; this paper is concerned with IRL techniques to build rule-based classifiers. The advantage offered by IRL, over many other forms of machine learning techniques (such as support vector machines, neural networks and self organising maps) is that the disjunctive normal form (DNF) rules produced are expressive while at the same time being easily interpretable by humans.

In the context of classification, the derived rules are typically of the form *condition* → *conclusion*; where the *condition* (antecedent) consists of a conjunction of features, while the *conclusion* (consequent) is the resulting class label associated with the condition. For example, the rule $a \wedge b \wedge c \rightarrow x$ (where a, b and c are features that appear in a dataset, and x is a class label) is interpreted as, if a and b and c occur together in a document, then classify the document as class x. With respect to most IRL systems, rules do not normally include the negation of features. For example, $a \wedge b \wedge \neg c \rightarrow x$, which would be interpreted as, if a and b occur together in a document and c does not occur, then classify the document as class x. Intuitively, rules that include negation seem to provide a powerful mechanism for distinguishing examples for classification; the inclusion of negation should serve to improve classification accuracy. This paper seeks to establish whether the use of negation in IRL is indeed beneficial with respect to classification. When considering the effectiveness of IRL with negation, there are two significant issues that need to be considered:

a. Feature identification: The identification of appropriate features to be negated.
b. Rule refinement strategies: The strategies for learning rule with negation.

The rest of this paper is organized as follows. A brief review of relevant previous work is presented in Section 2. In Section 3, a scenario illustrating the need for rules with negation is presented. Section 4 will discuss the issues highlighted. Section 5 describes the experiments carried out to determine the effectiveness of rules with negation, as well as the results and analysis. Section 6 concludes.

2. Previous Work

Existing work on IRL for classification tends to adopt a two-stage process: rule learning, followed by rule pruning. Examples of such systems include: (i) Reduced Error Pruning (REP) (Brunk et al., 1991), which incorporates an adaptation of

decision tree pruning; (ii) Incremental Reduced Error Pruning (IREP) (Fürnkranz et al., 1994), an enhancement over REP, (iii) Repeated Incremental Pruning to Produce Error Reduction (RIPPER) (Cohen, 1995), a further enhancement over IREP, and (iv) Swap-1 (Weiss et al., 1993). All these systems use the covering algorithm for rule learning, shown in Figure 1, whereby rules are "learned" sequentially based on training examples. The examples "covered" by a learnt rule are then removed and the process is repeated until some terminating condition is met.

Algorithm: Sequential covering. Learn a set of rules for classification.
Input:
- *D*, a data set class-labelled tuples;
- *Att_vals*, the set of all attributes and their possible values;

Output: A set of IF-THEN rules.
Method:
Rule_set = { }; //initial set of rules learned is empty
for each class *c* **do**
    repeat
        Rule = **Learn_One_Rule**(*D*, *Att_vals*, *c*);
        remove tuples covered by *Rule* from *D*;
    until terminating condition;
    Rule_set = *Rule_set* + *Rule*; //add new rule to rule set
endfor
return *Rule_set*;

Figure 1. *Basic sequential covering algorithm (Han et al., 2006)*

None of the above exemplar systems include an option to build negation into the generated rules. Examples of IRL approaches that generate rules with negation are much rarer. Wu et al. (Wu et al., 2002) and Antonie et al. (Antonie et al., 2004) considered both positive and negative Association Rules (ARs) in their work on AR mining (a classification rule of the form described in Section 1 may be considered to be a special type of AR). Negative features are also used by Zheng et al. (Zheng et al., 2003). However, their work does not involve the direct generation of rules with negation. They combined positive and negative features in their feature selection method for text classification using the Naïve Bayes classifier. Galavotti et al. (Galavotti et al., 2000) use negative evidence in a novel variant of k-NN. None of these systems can be truly described as being classification rule learning systems.

More recently, Rullo et al. (Rullo et al., 2007) have proposed a system called Olex that used both positive and negative features for rule learning. The system was directed at text classification and comprised a single stage rule learning process with no post-learning optimization (i.e. pruning). Rullo et al. proposed a paradigm of

"one positive term, more negative terms", where the positive term allows the identification of the right documents, thus, giving high recall values; while the negative terms help reduce the number of wrong classifications, thus, improving precision. The core of their method was in the selection of discriminating terms, which were selected from a reduced vocabulary to maximize the F1-measure value when using that set of terms to generate rules for classification. Each rule generated consisted of conjunctions of a single positive feature with none or more negative features. While the notion of using both positive and negative features seemed very promising, Rullo et al. also highlighted that their approach was not able to express co-occurrence based on feature dependencies (by allowing exactly one positive feature in a rule antecedent) and that this could affect the effectiveness of the text classifier. Thus, Olex is unable to generate rules of the form $a \wedge b \wedge \neg c \rightarrow x$.

It is of course possible to define features that describe the negation of features; given a feature "blue", we can define two binary-valued features: *blue* and ¬*blue*, which can then be considered by a "standard" IRL system. However, in the opinion of the authors, this is not a true IRL with negation approach. To the best knowledge of the authors, there are no reported IRL systems that incorporate the concept of negation as defined here.

3. Motivation

As noted in Section 1, rules of the form of *condition* → *conclusion* are the standard output from IRL algorithms; the *condition* part is usually a conjunction of positive features. Rules of this form are often sufficient for the classification of new and unseen data. However, there are cases where rules with negation produce a more effective rule set. This section seeks to establish that IRL with negation is necessary with respect to some data scenarios.

Assume a set of features $A = \{a, b, c, d\}$ and a set of class labels $C = \{x, y, z\}$ that can occur in a data set. Thus, we might have a data set of the form given in Table 1.

{a, b, x}
{a, c, x}
{a, d, y}
{a, d, y}

Table 1: *Example data set 1*

{a, b, x}
{a, b, c, y}
{a, c, z}

Table 2: *Example data set 2*

To apply IRL, the features must first be ordered according to which are the best discriminators, thus {d, b, c, a} (b, c and d are all excellent discriminators but d covers more records so is listed first). The strategies described in this paper (see

Section 4.2) use chi square ordering. Processing this data set in the standard IRL manner (without negative features) produces these rules: $b \rightarrow x$, $c \rightarrow x$ and $d \rightarrow y$, respectively. By introducing negation, we can get a more succinct set of rules: $a \wedge \neg d \rightarrow x$ and $d \rightarrow y$. Thus, in this case the use of negation has produced what may be argued to be a better (smaller and therefore more effective) rule set.

Considering the data set given in Table 2, it is more difficult to order the features. However, features b and c can be argued to be better discriminators than a because at least, they distinguish between one and the remaining classes, thus $\{b, c, a\}$. Starting with the first record, the rule $b \rightarrow x$ will be produced, which would have to be refined to $b \wedge \neg c \rightarrow x$ to give the correct result. Moving on to the next record will give $b \rightarrow y$, and then $c \rightarrow z$. Rearranging the ordering of the data set does not avoid the need for a negated rule. This example clearly illustrates the need for IRL with negation.

4. Inductive Rule Learning with Negation

The illustration in Section 3 provides a clear motivation for IRL with negation. However, this leads to the question of which feature to add to a rule when refining a rule. If a rule with negation is to be generated, which feature should be negated? If both positive and negative features are available, is the rule better refined with a positive feature or a negative feature? This section discusses these two issues.

4.1. *Feature identification*

Using our proposed approach, rules are initiated by selecting a feature associated with a class from a chi-square ordered list of features. Thus, all rules start with a single positive feature. If a rule covers both positive and negative examples, then the rule has to be further refined in order to learn a rule that can separate the examples. Positive examples are those training set records that are classified correctly given a current rule; negative examples are those that are classified incorrectly. Using our approach, the search space can be conceptualised as containing features that belong to positive and negative examples. This paper proposes that the search space be divided into three sub-spaces that contain different kinds of feature: (i) unique positive (UP) features which are found only in positive examples, (ii) unique negative (UN) features found only in negative examples, and (iii) overlap (Ov) features that are found in both positive and negative examples. This division allows efficient and effective identification of features that can be negated. It should be noted that the UP, UN and Ov feature categories may be empty as the existence of these features is dependent upon the examples covered by a rule. Where categories contain more than one feature, the features are ordered according to the frequency with which each feature occurs in the collection of examples covered by the current rule (one count per example).

4.2. *Rule refinement strategies*

If a rule is refined with a UP or an Ov feature, then a rule with no negation is generated. If a rule is refined with a UN feature, then a rule with negation is generated. When refining a rule with a UP or UN feature, the feature with the highest document frequency (appears in the most covered examples) is selected. When refining a rule with an Ov feature, the feature with the highest frequency difference (i.e. positive frequency minus negative frequency) is selected.

Feature set for class x = *{bike, ride, harley, seat, motorcycles, honda}* Initial rule learnt = $bike \rightarrow x$ The rule covers three examples (two +ve examples and one –ve example): *{bike, ride, motorcycles, x}* *{seat, harley, bike, ride, x}* *{bike, ride, honda, y}* Identify UP, UN and Ov features UP features = *{motorcycles, seat, harley}* UN features = *{honda}* Ov features = *{ride}* Strategies for rule refinement Refine with UP feature = $bike \wedge motorcycles \rightarrow x$ Refine with UN feature = $bike \wedge \neg honda \rightarrow x$ Refine with Ov feature = $bike \wedge ride \rightarrow x$

Table 3. *Example of rule refinement with UP, UN and Ov features*

Table 3 shows an example of refining a rule with UP, UN and Ov features. The refinement process will be repeated until the stopping condition is met; either: (i) when the rule no longer covers negative examples, (ii) the rule antecedent size reaches a pre-defined threshold or (iii) there are no more features that can be added to the rule. At every round of refinement, the examples covered will change and therefore, the search space will also change.

Given the UP, UN and Ov feature categories, a number of strategies can be identified whereby these categories can be utilized. These strategies may be defined according to the order in which they are considered. The Ov category, which comprises features that occurs in both positive and negative examples, is the least likely to result in successful refinement. Thus, it is argued that this should be considered last. Thus, we have two possible strategies involving all three categories

in sequence: UP-UN-Ov (UP first if it is not empty, then UN, then Ov) and UN-UP-Ov. Alternatively, we can refine rules using only the UP or UN collection. This gives rise to two more strategies: UP and UN. Note that the UP strategy, which does not entail negation, is the bench-mark strategy (use of negation must improve on this). Note also that the UN strategy produces rules that are identical to the rule structure that Olex (Rullo et al., 2007) generates as described in Section 2.

When refining rules using UP or UN, only one type of feature is used for the refinement. In contrast, the sequence combinations of UP-UN-Ov and UN-UP-Ov allow the use of UP, UN and Ov features when the preceding feature category in the sequence does not exist. A more flexible proposed strategy is UP-or-UN. The mechanism for this is to refine a rule by generating two versions and selecting the better version; one version is refined by UP and another version is refined by UN. The rule with the higher Laplace estimation accuracy is selected as the better rule.

5. Experimental Evaluation

This section describes the experimental setup used to investigate the proposed use of feature sub-spaces (UP, UN and Ov) and the five different rule refinement strategies devised. The results and analysis of each experiment are also discussed. Three different categories of data set were used for the experimental evaluation: (i) a collection of synthetic data sets covering all possible combination of a given set of features and classes, (ii) text mining data sets extracted from the well known 20 Newsgroups collection, and (iii) a selection of data sets taken from the UCI repository (Blake et al. 1998). In all cases, single-labelled (as opposed to multi-labelled) classification was conducted.

5.1 Synthetic Datasets

The synthetic data sets were constructed by considering every combination of a set of features A = *{a, b, c}* and a set of class labels C = *{x, y, z}*. Given that |A| = 3, there are 2^3-1 = 7 possible feature combinations. It was assumed that each record could contain only a single class label. Thus, there were 7*3 = 21 variations per record. Each data set was assumed to comprise 3 records, thus overall 21^3 = 9261 data sets were generated covering all possible record permutations (including data sets containing contradictions). The five strategies described in Section 4.2 were applied to the data sets. The results are shown in Table 4. The rows in Table 4 indicate the number of synthetic data sets where the generated classifier accurately covered all 3 records (100% accuracy), only 2 records (67% accuracy) and only 1 record (33% accuracy) respectively. Comparing the results using the UP and UN strategies in Table 4 provides further evidence for the need for IRL with negation.

Using the UN strategy, many more 100% accurate classifiers are generated than using the UP strategy. Using the UP-UN-Ov and UN-UP-Ov strategies allows the inclusion of all feature types which enhances the result even further. Inspection of the 2,436 cases where 100% accuracy was not obtained using both the UP-UN-Ov and UN-UP-Ov strategies, indicates that these mostly include contradictions which can never be entirely satisfactorily resolved. Use of the UP-or-UN strategy produces identical results to when the UN strategy is used; suggesting that at every round of refinement in the UP-or-UN strategy, the rule refined by UN is a better rule that is selected. The reason that the results for the UP-UN-Ov and UN-UP-Ov strategies, and for the UN and UP-or-UN strategies are identical is also due to the small size of the individual data sets used in the experiment, where the number of features is small. In general, it can be observed that strategies involving the generation of rules with negation produce better results than strategies without the use of negation.

	Rule Refinement Strategy				
Accuracy	UP	UN	UP-UN-Ov	UN-UP-Ov	UP-or-UN
100%	4,503	6,717	6,825	6,825	6,717
67%	3,324	2,352	2,316	2,316	2,352
33%	1,434	192	120	120	192
Total	9,261	9,261	9,261	9,261	9,261

Table 4. *Results for synthetic data sets*

5.2 Text Mining Datasets

For the text mining experiment, the 20 Newsgroups data set[1] was used in the context of binary classification. The 20 Newsgroups dataset is a collection of news items comprising 19,997 documents and 20 classes. The dataset was split into two parts: 20 Newsgroups A (20NGA) comprising 10,000 documents and the first 10 classes, and 20 Newsgroups B (20NGB) comprising 9,997 documents and the remaining 10 classes. Stop words removal was applied; followed by feature selection, based on the chi-square metric, where the top 1,000 features in each class was selected to be used in the text representation vector. Chi-square was chosen as the feature selection method due to its reported success in the literature (Yang et al., 1997; Debole et al., 2003; Zheng et al., 2003). The 1,000 features threshold was chosen to ensure a sufficiently large collection of features for each class is obtained. A rule size threshold of five was imposed on rule learning to generate rules that

1. http://people.csail.mit.edu/jrennie/20Newsgroups/

were not overly specific. Post-processing of the generated rule set was conducted by removing rules with coverage lower than a pre-defined threshold of 1.5% of the documents in the class (i.e. 15 documents with respect to the 20 Newsgroups), and a Laplace estimation rule accuracy value lower than 60%. Average ten-fold cross validation accuracy and F1-measure results across all classes in each fold using the different refinement strategies are presented in Table 5 (best results are highlighted in **bold** font).

From Table 5, it is noted that the UN strategy has the best results for accuracy in both 20NGA and 20NGB. In terms of the F1-measure, the UN strategy has the highest value in 20NGB while the UP-or-UN strategy did best in 20NGA. The UP and UP-UN-Ov strategies recorded the same results, suggesting that at every round of rule refinement, UP features exist and therefore, only rules without negation are generated. The UN-UP-Ov strategy did not improve on the UN strategy. This hinted that using the UN strategy may be sufficient in learning an effective rule set. The UP-or-UN strategy obtained a slightly higher F1-measure than the UN strategy although its accuracy is slightly lower. Overall, the results indicate sound support for the use of negation in IRL.

Datasets	Rule refinement with									
	UP		UN		UP-UN-Ov		UN-UP-Ov		UP-or-UN	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
20NGA	92.6	62.7	**93.1**	63.0	92.6	62.7	92.6	61.3	92.4	**63.7**
20NGB	93.4	66.6	**94.0**	**70.8**	93.4	66.6	93.7	67.3	93.2	68.0

Table 5. Results for 20 Newsgroups datasets

5.3 UCI Datasets

Further binary classification experiments were conducted using data sets selected from the UCI repository (Blake et al. 1998), namely: Anneal, Breast Cancer, Iris, Pima Indians and Wine. The datasets were first normalised and discretized using the LUCS-KDD normalisation software[2]. Again, a rule size threshold of five was imposed on rule learning. Post-processing of the generated classification rules was conducted by removing rules with a Laplace estimation rule accuracy value lower than 60%. Average accuracy and F1-measure value using ten-fold cross validation across all classes in each fold with the different rule refinement strategies are presented in Table 6 (again, best results are highlighted in **bold** font).

2. http://www.csc.liv.ac.uk/~frans/KDD/Software/LUCS-KDD-DN_ARM/lucs-kdd_DN.html

From Table 6, it can be observed that results are mixed. The first observation that can be made is that there are notable differences in the results obtained for UP-UN-Ov and UN-UP-Ov with that of UP and UN respectively, indicating that with respect to some of the generated rules there are no UPs and/or UNs. The best overall accuracy recorded for the Anneal data set was using the UP-UN-Ov strategy, while the highest overall F1-measure was obtained using the UN strategy. In the Breast Cancer data set, the UP-UN-Ov and UN-UP-Ov strategies produce the highest accuracy and F1-measure. It is also worth noting that in this case UP-UN-Ov and UN-UP-Ov significantly out-performed the other strategies. The UP-or-UN strategy produced the best accuracy and F1-measure for the Iris data set; and the UN strategy recorded the best accuracy and F1-measure for the Pima data set. The only data set where the UP strategy recorded the best accuracy and F1-measure was the Wine data set. It can also be observed that using the UP-UN-Ov strategy always improves on the UP strategy except in the Wine data set. Overall, the results indicate that strategies that allow the generation of rules with negation generally perform better than strategies that generate rules without negation.

Datasets	Rule refinement with									
	UP		UN		UP-UN-Ov		UN-UP-Ov		UP-or-UN	
	Acc	F1	Acc	F1	Acc	F1	Acc	F1	Acc	F1
Anneal	96.7	64.8	97.0	**66.7**	**97.6**	65.6	96.4	64.8	97.5	64.4
Breast	78.8	83.0	77.2	83.0	**92.6**	**92.3**	**92.6**	**92.3**	85.5	87.1
Iris	90.2	85.1	96.7	94.8	95.1	91.7	95.3	92.5	**96.9**	**95.0**
Pima	51.4	34.6	**73.3**	**66.7**	70.7	60.5	72.1	64.3	66.1	52.2
Wine	**91.0**	**86.1**	87.6	77.4	89.5	83.6	90.6	84.6	89.7	85.1

Table 6. Results for UCI datasets

6. Conclusion

This paper has sought to establish whether IRL with negation is effective or not with respect to the classification problem. This entails two issues: (i) the mechanism for identifying features to be negated and (ii) the strategies for deciding whether to add a positive or a negative feature. The paper proposes a solution to the first by dividing the search space, with respect to a current rule, into three sub-spaces designated as UP, UN and Ov. Five strategies for refining rules are considered, including a bench mark strategy (UP) that does not generate negated rules. The

reported experimental results indicate that the use of negation in IRL is indeed beneficial. For future work, the authors intend to conduct further experiments and investigate alternative strategies. This includes the comparison of different feature selection methods with respect to IRL with negation.

7. References

Antonie, M-L., Zaïane, O. R.: "An associative classifier based on positive and negative rules". In: Proceedings of *the 9th ACM SIGMOD Workshop on Research Issues in Data Mining and Knowledge Discovery*, pp. 64--69 (2004)

Blake, C. L., Merz, C. J. UCI Repository of machine learning databases. http://www.ics.uci.edu/~mlearn/MLRepository.html, Irvine, CA: University of California, Department of Information and Computer Science (1998)

Brunk, C., Pazzani, M.: "Noise-tolerant relational concept learning algorithms". In: Proceedings of *the 8th International Workshop on Machine Learning*. Morgan Kaufmann, New York (1991)

Cohen, W.: "Fast effective rule induction". In: Proceedings of *the 12th International Conference on Machine Learning (ICML)*, pp. 115--123. Morgan Kaufmann (1995)

Debole, F., Sebastiani, F.: "Supervised term weighting for automated text categorization". In: Proceedings of *the 18th ACM Symposium on Applied Computing*, pp. 784--788. (2003)

Fürnkranz, J., Widmer, G.: "Incremental reduced error pruning". In: Proceedings of *the 11th International Conference on Machine Learning (ICML)*. Morgan Kaufmann (1994)

Galavotti, L., Sebastiani, F., Simi, M.: "Experiments on the use of feature selection and negative evidence in automated text categorization". In: Proceedings of *the 4th European Conference on Research and Advanced Technology for Digital Libraries*, pp. 59--68 (2000)

Han, J., Kamber, M.: *Data Mining: Concepts and Techniques*. Morgan Kaufmann (2006)

Rullo, P., Cumbo, C., Policicchio, V. L.: "Learning rules with negation for text categorization". In: Proceedings *of the 2007 ACM Symposium on Applied Computing*, pp. 409--416. ACM (2007)

Weiss, S. M., Indurkhya, N.: "Optimized rule induction". *IEEE Expert: Intelligent Systems and Their Applications*. 8(6), 61--69 (1993)

Wu, Z., Zhang, C., Zhang, S.: "Mining both positive and negative association rules". In: Proceedings of *the 19th International Conference on Machine Learning*, pp. 658--665 (2002)

Yang, Y., Pedersen, J.: "A comparative study on feature selection in text categorization". In: Proceedings of *the 14th International Conference on Machine Learning (ICML)*, pp. 413--420. (1997)

Zheng, Z., Srihari, R.: "Optimally combining positive and negative features for text categorization". In: Proceedings of *the International Conference on Machine Learning (ICML), Workshop on Learning from Imbalanced Datasets II*. (2003)