Feature Subset Selection with TAR2less

Rajesh Gunnalan GUNNALAN@CSEE.WVU.EDU
Tim Menzies TIM@MENZIES.US
Kalaivani Appukutty AVANI@CSEE.WVU.EDU
Amarnath Srinivasan AMARNATH@CSEE.WVU.EDU
Lane Department of Com. Sci. and Elec. Eng., West Virginia University, Morgantown, WV, 26506 6109 USA

Ying Hu HUYING_CA@YAHOO.COM
Department of Electrical and Computer Engineering, University of British Columbia, Vancouver, Canada

Abstract

A repeated empirical result is that machine learners can learn adequate models using a small subset of the available features. Learning from such subsets can be faster, and produces simpler models. In this paper we present a new method for feature subset selection using the *TAR2 treatment learner*. TAR2 assumes *small backbones*; i.e. a small number of features will suffice for selecting preferred classes. TAR2 can be used as a pre-processor to other learners for identifying useful feature subsets. When compared to other methods described in a recent survey by Hall and Holmes (in press), TAR2 found the smallest subsets, with minimal or no change in classification accuracy.

1. Introduction

If the reader is a busy person, then he/she might not need, or be able to use, complex models. Rather, such a busy person might just want to know the *least* he/she needs to do to achieve the *most* benefits. Machine learning for busy people might not strive for (e.g.) elaborate models or (e.g.) increasing the expressive power of the language of the learnt model. Rather, a better goal might be to find the *smallest* model with the *most* impact.

Smaller models are easier and faster to read than larger models. One way to find these smaller models is to reduce the number of features in the instance set. The goal of *feature subset selection* (FSS) is to find those features that can be ignored without degrading the results of learning.

FSS helps human readers to understand a learnt model. It also can drastically reduce the search space for a learner. Numerous studies have shown that a learner can ignore many features with little or no loss in classification accuracy (e.g. Holte 1993; Kohavi & John, 1997; Hall and Holmes, in press). For example, using the "Wrapper" FSS method described in section 2, an average of 82% of the features seen in 10 domains could be ignored. Further, ignoring those features only changed classifier accuracy by an average of 5.45% (see Table 1).

Table1: Some FSS results (from Kohavi & John, 1997*)*

	Number of features										Aver -age
Before	6	36	10	6	6	13	15	8	25	180	30.5
after	2	12	2	1	1	2	2	1	3	11	3.7
%change	67	67	80	83	83	84	87	87	88	94	82
% accuracy change	7	0	0	25	6	6	5	1	0.5	4	5.45

How can ignoring features improve learning? This article offers a novel explanation based on the idea of *small backbones* (explained later). Hall & Holmes (in press) offer another, more standard explanation. They comment that including irrelevant, redundant and noisy features can slow down learning and lead to theories with poor predictive performance. Alternative explanations are specific to particular learning schemes. For examples, Kohavi & John (1997) review studies with Naïve Bayes classifiers. The accuracy of such classifiers decreases very slowly as irrelevant features are added to an instance set. However, the accuracy of the same classifiers can degrade sharply as the number of correlated features increase. Also, Witten & Frank (1999) note that effective generalization requires numerous examples. Decision tree learners recursively split instances by *ranking* features according to how much they decreases the diversity of the classes in the split sets. As learning progresses, fewer and fewer instances are available to learn the next sub-tree. If the instances contain too many features of similar *rank*, then many splits are quickly generated. Hence, instances become sparser in the sub-trees, and effective generalization becomes harder.

Standard FSS methods include *information gain ranking*, *principle component analysis, wrappers,* just to name a few. Here, we study a new FSS method based on the *TAR2 treatment learner* (Menzies et.al., 2003, in press).

Standard learners seek accurate *descriptions* of concepts whereas treatment learning seek some minimal set of *differences* between concepts. These differences are returned as a *treatment*; i.e. a conjunction of features recommending what actions to take. TAR2 inputs instances, applies a preference ordering on the classes, and outputs a treatment.

A curious feature of TAR2 is that, in the usual case, very small treatments are often adequate for selecting preferred classes. One possible explanation of this curious observation is the *small backbone* idea described later in this article. Whatever the reason, TAR2's treatments usually contain only a small subset of the available features; i.e. TAR2 might be useful for FSS. To test this speculation, we devised and adapted TAR2 for FSS. The adapted system was called *TAR2less*.

This paper describes an experimental evaluation of TAR2less. The results generated by TAR2less are compared to results seen in a recent state-of-the-art survey of FSS methods (Hall and Holmes, in press). For naturally occurring datasets, TAR2less out-performs the standard FSS methods.

2. Standard Methods for FSS

This section describes the standard methods for FSS used in the Hall and Holmes study. Subsequently, we describe TAR2, TAR2less, and compare the performance of TAR2less with these standard methods.

FSS techniques can be broadly classified into *wrappers* and *filters* (Kohavi & John, 1997). Wrappers select features using some target *learning algorithm* to evaluate the selected features. The filter approach selects features independent of the target learning algorithm. The filter approach is much faster when compared to the wrapper methods.

IG= Information Gain Attribute Ranking: This is a simple and fast method for feature ranking (Dumais, Platt, Heckerman & Sahami, 1998; Yang & Pedersen 1997). This method measures the entropy of the class before and after observing a feature. The difference in the entropy gives a measure of the information gained because of that attribute (Quinlan, 1993). A final comparison of this measure is used in feature selection.

RLF= Relief: Relief is an instance based learning scheme (Kira & Rendell, 1992; Kononenko, 1994). It works by randomly sampling one instance within the data. It then locates the nearest neighbors for that instance from not only the same class but the opposite class as well. The values of the nearest neighbor features are then compared to that of the sampled instance and the feature scores are maintained and updated based on this. This process is specified for some user-specified *M* number of instances. Relief can handle noisy data and other data anomalies by averaging the values for *K* nearest neighbors of the same and opposite class for each instance (Kononenko, 1994). For data sets with multiple classes, the nearest neighbors for each class that is different from the current sampled instance are selected and the contributions are determined by using the class probabilities of the class in the dataset.

PC= principle components: Principal component analysis is a statistical technique that reduces the dimensionality of the data by transforming the original feature space and extracting its eigenvectors (Hall & Holmes, 2002). The eigenvectors define a linear transformation from the original feature space to a new uncorrelated space. Eigenvectors can be ranked according to the amount of variation in the original data that they account for. Based on this the features are selected.

CFS= Correlation-based Feature Selection: CFS (Correlation-based Feature Selection) uses subsets of features (Hall, 1998; Hall, 2000). This technique relies on a heuristic *merit* calculation that assigns high scores to subsets with features that are highly correlated with the class and poorly correlated with each other. *Merit* can find the redundant features since they will be highly correlated with the other features. It can also identify ignorable features since they will be poor predictors of any class. To do this CFS informs a heuristic search for key features via a correlation matrix.

CBS= Consistency-based Subset Evaluation: CBS is really a set of methods that use class consistency as an evaluation metric. The specific CBS studied by Hall and Holmes method finds the subset of features whose values divide the data into subsets with high class consistency (Almuallim & Dietterich, 1991; Liu & Setiono, 1996).

WRP= Wrapper Subset Evaluation: Kohavi & John (1997) *wrapped* their target learner in a pre-processor that used a heuristic search to grow subsets of the available features from size 1. At each step in the growth, the target learner was called to find the accuracy of the model learned from the current subset. Subset growth was stopped when the addition of new features did not improve the accuracy. In their experiments, 83% (on average) of the features in a domain could be ignored with only a minimal loss of accuracy. The advantage of the wrapper approach is that it is simple to implement. The disadvantage of the wrapper method is that each step in the heuristic search requires another call to the target learner; i.e. it may be very slow.

Hall and Holmes conclude their study by saying that there is no single approach that works for all situations.

TAR2less= Our method: We have explored FSS using the TAR2 weighted class learner. TAR2less ran TAR2 many times, each time giving each successive class the highest weight. Each single run of TAR2 found features that were most selected for one class. Over all the runs, TAR2less found the union of all the features that most selected for every class..

3. TAR2

The FSS method discussed here is based on the TAR2 treatment learner. This section is our standard description of TAR2 (Menzies & Hu, 2002).

TAR2 outputs a rule of the form:

$$\textit{If Feature}_1 = \textit{range}_1 \wedge \textit{Feature}_2 = \textit{range}_2 \wedge ..$$
$$\textit{then good= more} \wedge \textit{bad=less}$$

where *good* and *bad* are sets of classes that the learner likes and dislikes respectively; and *more* and *less* are the frequency of these classes, compared against the current situation, which we call the baseline.

Formally, TAR2 is a *weighted-class minimal contrast association rule learner* that utilizes *confidence-based* pruning. These terms are explained below.

Association rule learning: Classifiers like C4.5 (Quinlan, 1993) and CART (Breiman, Friedman, Olshen, & Stone, 1998) learn rules with a single feature pair on the right-hand side; e.g. *class*= *Z*. Association rule learners like APRIORI (Agrawal & Srikant, 1994) and TAR2 generate rules containing multiple feature pairs on both the left-hand side and the right-hand-side of the rules.

General association rule learners like APRIORI input a set of *D* transactions of items *I* and return associations between items of the form *LHS* ➔ *RHS* where $LHS \subseteq I$ and $RHS \subseteq I$ and $LHS \cap RHS = \varnothing$. Specialized association rule learners like CBA (Liu, Hsu & Ma, 1998) and TAR2 imposes restrictions on the right-hand-side. Specifically, TAR2 restricts the right-hand-side features to just those class features containing criteria assessment. These right-hand-sides show a prediction of the change in the class distribution if the constraint in the left-hand-side were applied.

Weighted learning: Standard classifier algorithms such as C4.5 and CART have no concept of class weighting. That is, these systems have no notion of a good or bad class. Such learners therefore can't filter their learnt theories to emphasize the location the good classes or bad classes. Association rule learners such MINWAL (Cai, Fu, Cheng & Kwong , 1998), TARZAN (Menzies and Sinsel, 2000) and TAR2 explore weighted learning in which some classes are given a higher priority weighting than others. Such weights can focus the learning onto issues that are of particular interest to some audiences.

Contrast sets: Instead of finding rules that describe the current situation, association rule learners like STUCCO (Bay & Pazzani, 1999) finds rules that differ meaningfully in their distribution across groups. TAR2's variant on the STUCCO strategy is to combine contrast sets with weighted classes with minimality. That is, TAR2 treatments can be viewed as the smallest possible contrast sets that distinguish situations with numerous highly-weighted classes from situations that contain more lowly-weighted classes.

Confidence-based pruning: In the terminology of APRIORI, the association *X* ➔ *Y* has *support s* if *s*% of the *D* transactions contains $X \wedge Y$; i.e. $s = |X \wedge Y| / |D|$ |(where $|X \wedge Y|$ denotes the number of examples containing both *X* and *Y*). The *confidence c* of an association rule is the percent of transactions containing *X* which also contain *Y*; i.e. $c= |X \wedge Y| / |X|$

Many association rule learners use *support-based pruning* i.e. when searching for rules with high confidence, sets of items I_i,... I_k are only be examined only if all its subsets are above some minimum support value. Support-based pruning is impossible in weighted association rule learning since with weighted items, it is not always true that subsets of *interesting* items (i.e. where the weights are high) are also interesting (Cai, Fu, Cheng, Kwong, 1998). Another reason to reject support-based pruning is that it can force the learner to only miss features that apply to a small, but interesting subset of the examples (Wang, He, Cheung, Chin, 2001)

Without support-based pruning, association rule learners rely on confidence-based pruning to reject all rules that fall below a minimal threshold of adequate confidence. TAR2 uses confidence based pruning.

Confidence-based pruning: TAR2 seeks the features that "nudge" a system away from undesired classes and towards desired classes. TAR2's score for each range is the *confidence1* measure. This value is high if a range occurs frequently in desired situation and infrequently in undesired situations. That is, if we were to impose this range as a constraint, then it would tend to "nudge" the system into better behavior.

To find confidence1, we assume that we can access some numeric value assigned to each class. The class with the highest value is the *best* class. The *lesser* classes are the set of all classes, less the *best* class. To compute confidence1, TAR2 sums the difference in the frequencies of feature/range pairs seen in the *best* and all the *lesser* classes (weighted by the difference in the value between the *lesser* and the *best* class). This weighed sum is normalized by the total frequency count of the feature in all classes.

TAR2 prunes all feature/range pairs with a confidence1 value below some user-specified threshold *min-confidence*. From the remaining feature/ranges, a set of candidate treatments are generated by extracting all combinations of size *N*. Each candidate selects some subset of the training instances (i.e. all instances that are not inconsistent with the candidate). Each such subset awards a score to the candidate that created it: the larger the frequency of the preferred classes, the higher the score. The highly-score candidates are then assessed via an N-way cross validation.

Theoretically, TAR2 is impractically slow. Recall that treatments are generated by exploring all combinations of size *N* of the available feature/range pairs. If *X* pairs are found above *min-confidence*, then the number of such pairs to be searched is:

$$\binom{X}{N=1}+\binom{X}{N=2}+\ldots+\binom{X}{N=X-1}+\binom{X}{N=X}\approx 2^X$$

(This calculation is actually an over-estimate since it ignores the *value exclusion* property; i.e. different values of the same feature such as *F1=a* and *F1=b* can never be contained in the same treatment. Nevertheless, it gives a sense of how large this search can grow.)

These theoretical concerns have yet to be realized in practice. In studies with dozens of domains, TAR2 usually found effective treatments from N<*4* features and never with features *N>8* (Menzies & Hu, 2002). The success of TAR2 is not too surprising. Holte's 1R study (Holte, 1993) and numerous FSS experiments (Kohavi & John, 1997; Hall and Holmes, in press) all concur that effective models can be learnt using only a small *N* subset of the available features. Nevertheless, we'd prefer some better reason, grounded in a general theory, for trusting TAR2. "Small backbones" are such a reason.

4. Small Backbones

Defining small backbones, and understanding their implications, requires a little background. We begin by noting that learners try to *summarize theories* which, in the usual case, they can't access. Instead, learners usually work from a *log of instances of the behavior* of that theory.

Logically, theories can be viewed as a set of *constraints*. Each instance used by a learner is a set of feature/value pairs showing one *solution* to the constraints of a theory. In the case of weighted class learning, some oracle adds a class symbol to each solution. This class symbol shows how much the oracle approves of that solution.

A recent empirical observation from the constraint satisfaction community is that, for *solvable constraint satisfaction problems* (CSPs), there exists a set of critical feature/value pairings called the *backbone*. This backbone holds the pairings that always appear in the best solutions. The effort required to solve a CSP is a function of the backbone size: the larger the backbone, the greater the effort (Parkes, 1997).

Our explanation for the success of TAR2 is that most instances contain small backbones. There are several reasons for believing that this might be so. Firstly, if we assume that the instances used in machine learning come only from solvable problems, then we could also assume that machine learners usually contains small backbone instances. Secondly, Menzies & Singh (2001) offer an analytical argument that, for under-constrained problems, it is thousands to millions of times more likely that problems in the under-constrained zone contain very small backbones (which they call "funnels").

Note that, in terms of understanding a theory, the features outside the backbone are less important that the features in the backbone. These less important features might therefore be ignorable by a learner. In instances with small backbones, many variables could be ignored. That is, small backbones explain not only the success of TAR2 but also explain the repeated success of FSS.

5. Using TAR2 for FSS

TAR2 relies on small backbones. Many features within instances with small backbones are ignorable. Ignoring features is a synonym for FSS. This line of reasoning suggests that TAR2 could be a useful FSS device. This section describes a test of that speculation.

5.1. TAR2less

Figure 1 shows *TAR2less*: our adaptation of TAR2 to FSS. TAR2less executes as follows:

- For various target learners:
  - Initialize the *SELECTED* features to nil.
  - For each class in turn, declare it to be TAR2's "best" class. Then enter the following loop:
    - Set treatment size *N* to 1
    - Find the "best" treatment of size *N* via TAR2.
    - If the score of the best treatment is no better than that of the best treatment of size *N-1,* then…
      - Add the features seen in the best treatment to *SELECTED.*
    - Else, *N*++ and loop.
  - Collect the average accuracy seen in a 10-way cross validation of the target learner using
    - just the features seen in *SELECTED*
    - all features

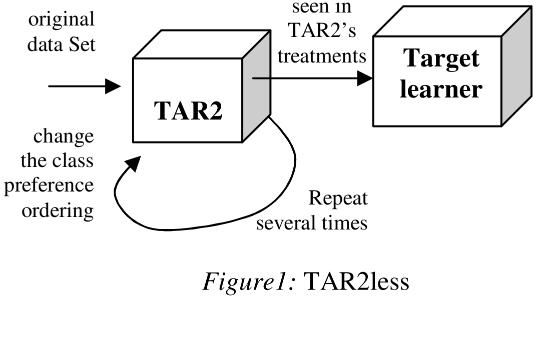

original data Set
TAR2
union of all features seen in TAR2's treatments
Target learner
change the class preference ordering
Repeat several times

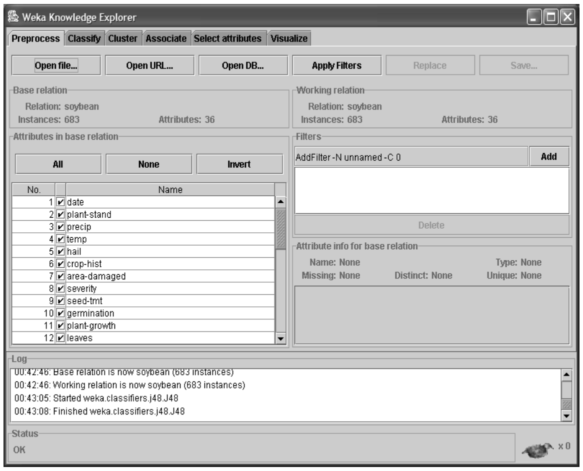
Figure1: TAR2less

5.2. Support Tools

To test TAR2less, we compared it to the results seen in a recent state-of-the-art survey of FSS methods (Hall and Holmes, in press). Our study used the same target learners as used in that survey: a decision tree learner and a Naïve Bayes classifier. These two learners were deliberately selected by Hall and Holmes to assess the utility of FSS on radically different learning schemes:

- Recall from the introduction that decision tree learners recursively split instances by *ranking* feature ranges according to how much they decreases the diversity of the classes in the split sets. Hall & Holmes used C4.5, a decision tree that uses an entropy measure to rank feature ranges.
- Naïve Bayes classifiers work in a very different manner. Statistics are collected on the distribution of feature ranges in different classes. Those statistics are used to estimates the probability that some new combination of features belongs to a certain class.

Like Hall and Holmes**,** we used the implementation of C4.5 and Naïve Bayes classifier found in WEKA: the Waikato Environment for Knowledge Analysis (Witten & Frank, 1999). The WEKA is a free, JAVA-based, open source, GUI tool that provides a rich variety of machine learners, preprocessing tools, and visualization tools. Figure 2 shows the WEKA user interface.



Base relation
Relation: soybean
Instances: 683 Attributes: 36

Working relation
Relation: soybean
Instances: 683 Attributes: 36

Attributes in base relation

No.		Name
1	☑	date
2	☑	plant-stand
3	☑	precip
4	☑	temp
5	☑	hail
6	☑	crop-hist
7	☑	area-damaged
8	☑	severity
9	☑	seed-tmt
10	☑	germination
11	☑	plant-growth
12	☑	leaves

Filters
AddFilter -N unnamed -C 0

Attribute info for base relation
Name: None Type: None
Missing: None Distinct: None Unique: None

Log
```
00:42:46: Base relation is now soybean (683 instances)
00:42:46: Working relation is now soybean (683 instances)
00:43:05: Started weka.classifiers.j48.J48
00:43:08: Finished weka.classifiers.j48.J48
```

Status
OK

Figure 2. WEKA tool

5.3. Datasets

Our experiments were run on 10 of the datasets analyzed by Hall and Holmes. These datasets, described in Table 2, originally come from the UCI (University of California at Irvine) repository. These datasets had a wide range of nominal and numeric features. The size of these datasets varied from a few hundred to a few thousand instances.

Table 2. Data sets used

DATASET	INSTANCES	NUMERIC	NOMINAL	CLASSES
anneal	898	6	32	5
breast-c	286	0	9	2
credit-g	1000	7	13	2
diabetes	768	8	0	2
horsecolic	368	7	15	2
ionosphere	351	34	0	2
lymph	148	3	15	4
segment	2310	19	0	7
soybean	683	0	35	19
vote	435	0	16	2

5.4. Results

Table 3 show the average accuracies seen in 10-way cross validation by the two target learners. The colored cells in Table 3 show where the classification accuracy changed by more than 1% (gray indicates an improvement and black indicates a decrease).

The features rejected by TAR2less changed classification accuracy very little~~.~~. The largest difference seen in Table 3 was the 4.8% loss seen for "vote". This large difference was not the usual case. On average, the classification accuracies changed by less than 1%. More specifically:

- a 0.97% average decrease for C4.5
- a 0.87% average increase for Naïve Bayes

Tables 4 and 5 compare the size of the features selected by the FSS methods described in section 2 and TAR2less. The TAR2less results come from our work. The results from the other FSS methods come from Hall & Holmes (in press). The two tables show results from our two target learners: a decision tree learner and a Naïve Bayes classifier. In these tables, gray cells denote the smallest subset of features found by any method.

Table 3. Accuracy of C4.5 and Naïve Bayes before and after TAR2less (averages over a 10-way cross validation). Colored cells indicate where accuracy changes greater 1%: black denotes accuracy decrease and gray denotes accuracy increase.

Data Sets	C4.5		Naïve Bayes	
	Original	TAR2	Original	TAR2
anneal	98.2	98.2	86.6	84.3
breast-c	75.2	75.2	74.1	75.2
credit-g	73.9	72.3	75.9	74.3
diabetes	74.5	72.8	76.0	74.6
horsecolic	85.3	81.5	78.8	79.6
ionosphere	88.6	87.8	82.9	87.5
lymph	76.4	74.3	81.8	77.7
segment	97.1	96.6	79.8	86.3
soybean	92.4	93.0	92.7	93.0
vote	95.9	96.1	90.1	94.9

Table 4. Number of features selected by TAR2less and six other FSS methods using C4.5 as the target learner. Gray cells denote the smallest subsets found by any method.

Data-Set	Original	IG	CFS	CBS	RLF	WRP	PC	TAR2LESS
Anneal	38	17	21	15	20	18	36	7
breast-c	9	4	4	7	7	4	4	2
credit-g	20	8	7	8	9	8	4	5
Diabetes	8	33	3	4	4	4	6	1
Horse colic	22	4	4	2	3	5	3	2
Iono-sphere	34	12	7	9	9	7	10	2
lymph	18	6.8	5.3	4	4	6	9	3
Seg ment	19	16	12	9	13	9	16	4
Soy bean	35	19	24	35	32	19	30	16
vote	16	12	10	6	11	9	11	6

Table 5. Number of features selected TAR2less and six other FSS methods using Naïve Bayes as the target learner. Gray cells denote the smallest subset found by any method.

Data-Set	Original	IG	CFS	CBS	RLF	WRP	PC	TAR2LESS
Anneal	38	10	4	5	39	7	25	7
breast-c	9	4	7	6	5	3	3	2
credit-g	20	13	14	14	20	12	11	5
Diabetes	8	3	4	4	6	3	4	1
Horse colic	22	9	4	4	23	6	6	2
Iono-sphere	34	8	8	11	18	13	12	2
lymph	18	17	13	14	15	2	13	3
Seg ment	19	11	11	5	15	8	9	4
Soy bean	35	31	31	33	36	26	21	16
Vote	16	1	2	3	15	1	3	6

Note that, in Table 4 and Table 5, The subsets found by TAR2less were smaller than any other method here in 17 of the 20 experiments. For decision tree target learners, TAR2less found the smallest subset in 9 of the 10 experiments. For Naïve Bayes target learners, TAR2less found the smallest subsets in 8 of the 10 experiments.

In summary, TAR2less was the best overall FSS method studied here; i.e. it found the smallest feature subsets and those subsets resulted in minimal or no change in classification accuracy.

Hall & Holmes do not offer runtimes for their FSS methods. Hence, we can't compare the runtimes of TAR2less with the other FSS methods shown in Tables 2,3 and 4. However, we have some evidence that TAR2 will be a much faster than some FSS methods. Kohavi & John (1997) report that their Wrapper method can take up to hundreds or thousands of seconds to terminate. TAR2less runtime for any of the domains shown in Tables 2,3 or 4 is much faster: i.e. less than ten seconds in most cases.

8. Conclusion

We have explored FSS using the TAR2 weighted class learner. TAR2less has been compared here with a recent state-of-the-art survey in feature subset selection. In that comparison, TAR2less almost always selected a smaller set of features than other FSS methods. Also, measured in terms of averages over a 10-way cross validation, the impact on accuracy was minimal. Further, we believe that TAR2less runs faster than other FSS methods, but the evidence for this last conclusion is very limited since run times for other FSS methods except for Wrapper were not available. The significance of our results is that unlike prior results in FSS our approach is a all-round performer both in terms of accuracy and number of features selected.

We claim that small backbones explain the success of our method. Due to the backbone, certain feature/range frequencies will be unusually high amongst instances from the preferred classes. Such feature ranges can be detected using TAR2's confidence1 measure. Further, due to small backbones, the total number of these critical features will be very small. Hence, a practical method of selecting the most effective features is to explore all subsets (of size, small *N*) of feature ranges with very high confidence1 values.

Kohavi & John have reported FSS studies on artificially generated data sets such as Monk1. We did not explore such data sets since our goal in this paper was to show how real world data sets have certain characteristics (i.e. small backbones) that simplify the learning process. Future work would also involve trying our approach on artificially generated datasets with backbones of varying sizes and more features.

Acknowledgements

The editorial assistance of Lisa Montgomery was both timely and useful and we are most grateful. This research was conducted at the University of British Columbia and the West Virginia University under NASA contract NCC2-0979 and NCC5-685. The work was partially sponsored by the NASA Office of Safety and Mission Assurance under the Software Assurance Research Program led by the NASA IV\&V Facility. Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States Government.

References

Agrawal, R & Srikant, R., (1994) . *Fast algorithms for mining association rules.*The 20th International Conference on Very Large Databases, 1994.

Almuallim, H., & Dietterich, T.G., (1991) *Learning with many irrelevant features,* The Ninth National Conference on Artificial Intelligence.(pp. 547-552), AAAI Press.

Bay, S.B & Pazzani, M.J., (1999*). Detecting change in categorical data: Mining contrast sets.* The Fifth International Conference on Knowledge Discovery and Data Mining.

Breiman, L, Friedman, J.H, Olshen, R.A & Stone, C.J.,(1984) *Classification and regression trees.* Technical report, Wadsworth International, Monterey, CA.

Cai, C.H, Fu, A.W.C, Cheng, C.H & Kwong, W.W.,(1998) *Mining association rules with weighted items*. In Proceedings of International Database Engineering and Applications Symposium (IDEAS 98).

Dumais, S., & Platt, J., Heckerman, D., & Sahami. M., (1998). *Inductive learning algorithms and representations for text categorization*, The International Conference on Information and Knowledge Management (pp. 148-155).

Hall, M. A., (1998). *Correlation-based feature selection for machine learning,* Ph.D. thesis, Department of Computer Science, University of Waikato, Hamilton, New Zealand.

Hall, M., (2000). *Correlation-based feature selection for discrete and numeric class machine learning,* The 17th International Conference on Machine Learning (ICML2000).

Hall, M. and Holmes, G. (In press). *Benchmarking attribute selection techniques for discrete class data mining*. IEEE Transactions on Knowledge and Data engineering.

Holte, R.C. (1993) *Very Simple Classification Rules Perform Well on Most Commonly Used Datasets*, Machine Learning, vol. 3, pp. 63-91.

Kononenko, I., (1994). *Estimating attributes: Analysis and extensions of relief*, The Seventh European Conference on Machine Learning. (pp. 171-182), Springer-Verlag.

Hall, M.A., & Holmes, G., (2002). *Benchmarking Attribute Selection Techniques for Discrete Class Data Mining.* IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, VOL. XX, NO. Y, MONTH 2002

Kira, K & Rendell,L., (1992). *A practical approach to feature selection*, The Ninth International Conference on Machine Learning. (pp. 249-256), Morgan Kaufmann.

Kohavi, R & John, G.H., (1997). *Wrappers for feature subset selection*. Artificial Intelligence, 97(1-2):273-324.

Liu, H and Setiono, R., (1996)., *A probabilistic approach to feature selection: A filter solution*, The 13th International Conference on Machine Learning.(pp. 319-327), Morgan Kaufmann.

Liu B. and Hsu W. and Ma Y. (1998) *Integrating classification and association rule mining, KDD*, pages 80—86, Sept, 1998, Available from http://citeseer.nj.nec.com/liu98integrating.html

Menzies, T & Sinsel E., (2000). *Practical large scale what-if queries: Case studies with software risk assessment.* In*Proceedings ASE 2000*.

Menzies T. & Singh H (2001) *Many Maybes Mean (Mostly) the Same Thing*, 2nd International Workshop on Soft Computing applied to Software Engineering (Netherlands), February, 2001, available from tim.menzies.com/pdf/00maybe.pdf.

Menzies T. & Hu Y. (2002) *Just Enough Learning (of Association Rules): The TAR2 "Treatment" Learner",* WVU CSEE tech report, Available from http://tim.menzies.com/pdf/02tar2.pdf.

Menzies T., Chiang E., Feather M., Hu Y. and J.D. Kiper J.D. (in press), *Condensing uncertainty via Incremental Treatment Learning,* Annals of Software Engineering, Available from tim.menzies.com/pdf/02itar2.pdf

Parkes, A.J. (1997) *Clustering at the Phase Transition*, AAAI/IAAI, pages 340-345, 1997, available from citeseer.nj.nec.com/parkes97clustering.html

Quinlan, J. R., (1993). *C4.5: Programs for Machine Learning*, Morgan Kaufmann, San Mateo, CA.

Wang K. and He Y. and Cheung D. and Chin F (2001) *Mining confident rules without support requirement,* 10th ACM International Conference on Information and Knowledge Management (CIKM 2001), Atlanta. Available from www.cs.sfu.ca/~wangk/publications.html.

Witten I.H. & Frank E. (1999) *Data Mining,* Morgan Kaufmann, 1999

Yang.Y & Pedersen,J.O., (1997). *A comparative study on feature selection in text categorization* International Conference on Machine Learning(pp. 412-420).