in: *Proceedings of the IJCAI-99 Worskhop: Automating the Construction of Case Based Reasoners,*
S.S. Anand, A. Aamodt, D.W. Aha (eds.), pp. 28–33.

Genetic Algorithms for Feature Selection and Weighting[1]

Jacek Jarmulak and Susan Craw
School of Computer and Mathematical Sciences, The Robert Gordon University
St Andrew Street, Aberdeen, AB25 1HG, Scotland, UK
{jacek|smc}@scms.rgu.ac.uk

Abstract

Automated techniques to optimise the retrieval of relevant cases in a CBR system are desirable as a way to reduce the expensive knowledge acquisition phase. This paper concentrates on feature selection methods that assist in indexing the case-base, and feature weighting methods that improve the similarity-based selection of relevant cases. Two main types of method are presented: filter methods use no feedback from the learning algorithm that will be applied; wrapper methods incorporate feedback and hence take account of learning bias. Wrapper methods based on Genetic Algorithms have been found to deliver the best results with a tablet design application, but these generic methods are flexible about the criterion to be optimised, and should be applicable to a wide variety of problems.

1 Introduction

The majority of CBR systems rely on a good case-base organisation, an effective index and a (possibly knowledge intensive) similarity matching to select cases, that can then be used to solve a problem, see Figure 1. The purpose of the index is twofold; firstly, it reduces the number of the cases that have to be matched, thus speeding up the retrieval, and secondly it partitions the case-base, thereby hopefully reducing the chance of retrieving wrong cases. (However, an incorrect index may also prevent the right cases from being retrieved.)

Many CBR tools provide standard means of constructing indexes. Isoft's ReCall is typical in using a C4.5 [Quinlan 1993] generated decision tree, constructed from the cases in the case-base, as the index. However, induction algorithms like C4.5 apply a greedy selection approach and so the features used by the index are not always the optimal ones. This is a particular problem when the cases contain many features irrelevant to the problem solving [John, Kohavi & Pfleger 1994], as is often true

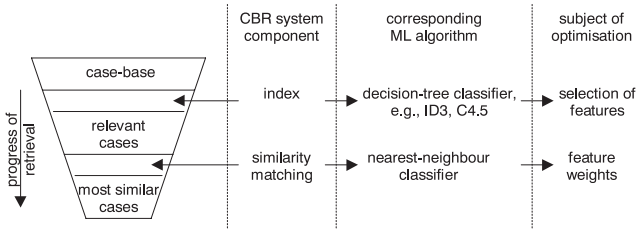

Figure 1: Components of a CBR retrieval stage and possible ways of their optimization.

when the cases had been derived from a database. Almuallim & Dietterich [1992] show empirically that ID3 (C4.5's predecessor) does not perform well if presented with a large number of irrelevant features. Therefore, the selection of relevant features, prior to using induction, can be expected to lead to a better index. Although relevant features could be determined by a domain expert, applying an automated algorithm for relevant-feature selection has many advantages.

The cases identified by the index are next ranked according to their similarity to the new problem. The simplest similarity metric is Euclidean distance between normalised feature vectors. However, a "useful" (from the point of view of solving a problem) similarity should take account of the relative importances of various features. Certainly in a situation where many features are irrelevant to the problem to be solved, a simple similarity measure is insufficient. This problem can be partially solved by identifying and removing irrelevant features as before. However, a more flexible method assigns weights to the features to indicate their relative importance to the problem solving. Although the selection of the relevant features can usually be done quite accurately by an expert, feature weighting can only be done approximately by an expert, often by categorising the relevance as one from a small set of possible degrees of relevance. Therefore, applying an automated algorithm to find feature weights is attractive.

Section 2 reviews feature selection and weighting methods. Our tablet formulation problem domain is introduced in Section 3. Sections 4 & 5 present methods to select relevant features for a case-base index and to pro-

[1] This work is supported by EPSRC grant GR/L98015 awarded to Susan Craw.

pose feature weights for a similarity measure. Results for tablet formulation are outlined in Section 6. Section 7 summarises our findings and future work.

2 Related Work

Algorithms for feature selection or feature weighting can be classified into two main categories depending on whether the method uses feedback from the subsequent performance of the machine learning algorithm. John et al. [1994] describe these two approaches in the context of feature selection, but we shall also apply these approaches to feature weighting.

A *filter* method is a no-feedback, pre-selection method that is independent of the later Machine Learning (ML) algorithm to be applied, see Figure 2. This diagram refers to feature selection for an induction algorithm but a similar approach applies to feature weighting for a Nearest Neighbour (NN) algorithm where a set of proposed feature weights is passed to the NN algorithm. The data (e.g. from a case-base) is first analysed, for instance using statistical techniques, to determine which features describing the data records (cases) are relevant for the given task, say prediction. Afterwards, the relevant features are used in training an induction algorithm, such as a decision tree. No feedback from the subsequent performance of the induction algorithm is used, except perhaps to determine whether and by how much the performance has improved compared to a system without feature pre-selection.

In contrast, a *wrapper* method is a feedback method that incorporates the ML algorithm in the feature selection process, see Figure 3. Again, the approach can easily be applied to feature weighting, although here we describe it in terms of feature selection for induction algorithms. The optimal feature selection is determined by a search in the space of possible selections. This means that selections of features are generated and then have to be evaluated. The evaluation is done by running the induction algorithm through the train and test phases (though sometimes the train phase is sufficient to give an approximate evaluation) using each selection of features, and providing feedback on its learning performance. Usually an exhaustive search is too expensive, so non-exhaustive search techniques like hill-climbing, random

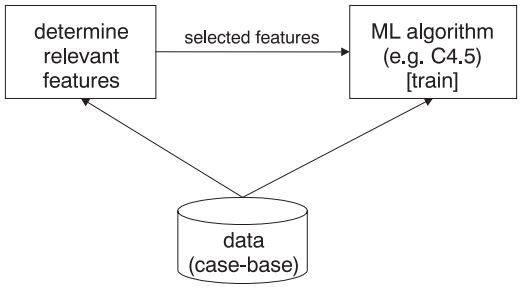

Figure 2: Filter method for feature selection.

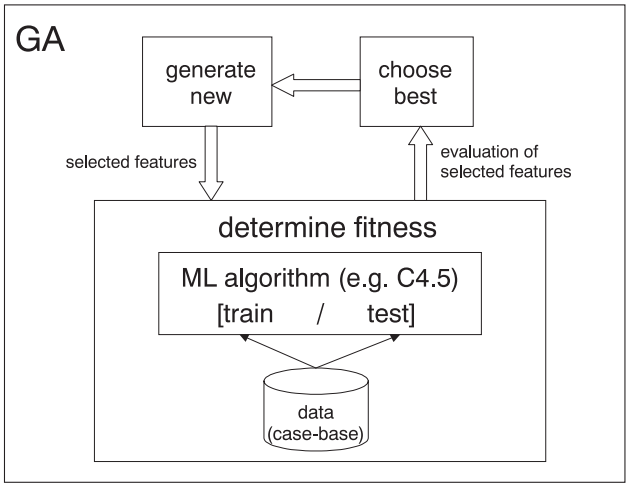

Figure 3: GA wrapper method for feature selection.

search or Genetic Algorithms (GAs) are often used. In Figure 3 we have chosen a GA to search for the best selection of features with the ML algorithm providing the GA's fitness function.

Aha [1998] uses the same binary classification into filter and wrapper methods when discussing feature-weighting methods used in lazy learning. Blum & Langley [1997] further propose distinguishing *embedded* methods, in addition to filter and wrapper methods; these are characterised by being an integral part of the ML algorithm.

We have investigated both filter and wrapper methods for feature selection and feature weighting. A wrapper approach should give better results than a filter method, since it adapts itself to the inherent biases of the ML algorithm to be used. However a wrapper approach is likely to have larger computation costs which may make it prohibitive. Since we are seeking an approach that fits well with the indexing and similarity matching algorithms already embedded in CBR systems, we do not consider embedded methods further.

3 The Problem Domain

The target application for the feature selection and weighting algorithms we develop is tablet formulation [Rowe, Craw & Wiratunga 1999]. In this paper we are concerned with only a subproblem of the full tablet formulation task; we must choose an appropriate filler to be used in a tablet for a given dose of a specific drug. Full tablet formulation chooses a binder, disintegrant, lubricant and surfactant in addition to the filler and also proposes quantities for each of these excipients.

The feature vector for a new tablet formulation task contains 26 features: 5 are physical properties of the drug, 20 are chemical properties of the drug with the excipients, and the last is the required dose. Available domain knowledge tells us that some of these features are

irrelevant (for filler selection) and some are derived from others. We have access to two datasets: tablet formulations for 39 drugs at 8 different doses, and a subset corresponding to 13 drugs. Given the number of features even the complete 312-formulation dataset can be considered sparse.

A hand crafted prototype system for tablet formulation, CBRTFS, was developed in ReCall. Details of its implementation appear in [Craw, Wiratunga & Rowe, 1998], together with results comparing its performance to a nearest-neighbour (NN) algorithm on the 13-drug dataset. CBRTFS did not include any feature selection prior to building the index and only "dose" feature was given an increased weighting for similarity matching. However, after the retrieval, voting was applied to choose excipients, and also some adaptation knowledge was available and used in the system.

CBRTFS's decision-tree index, generated by ReCall, contained about half of the available features, thereby confirming that many of the features may be irrelevant. One of our goals is to determine an optimal subset of relevant features. Moreover, correctly weighting the features for the similarity function can be expected to be important. In particular, the experiments have shown that increasing the weighting for the "dose" feature recognised its importance, and improved problem solving. But it was obvious that further adjustment of the similarity weightings was needed to obtain improved results.

4 Selecting Relevant Features

We wish to identify relevant features of the case-base so that the induction algorithm ignores irrelevant features when it selects the most discriminating feature for each node in the decision tree, which is then to be used as an index for the case-base.

4.1 Filter Method

A filter method suitable for our data should be capable of handling continuous valued features, multiple target classes, and noisy data; RELIEF-F [Kononenko 1994] satisfies these requirements. Because it is a filter method we ran first RELIEF-F on the data to determine the relevant features and then used the selected features to induce an index from the data using C4.5, see Figure 2.

For each feature, RELIEF-F returns a numerical measure of its importance. A threshold (usually set to zero) then determines which features are irrelevant. Table 1 shows RELIEF-F's relevance values for the tablet formulation features at the left, calculated using K=1 and K=4 nearest hits/misses. However, on our data only a few importance values were below the normal zero threshold. Even increasing the threshold, and thus ignoring further features (these are italicised in Table 1) prior to applying C4.5, gave no improvement over applying C4.5 to all the feature vectors. One explanation for these unsatisfactory results is John et al.'s [1994] comment that in real domains many features have high correlations, and thus many are (weakly) relevant, and so RELIEF-F cannot recognise them as irrelevant.

4.2 Wrapper Method

We chose a Genetic Algorithm (GA) for the wrapper method to search for relevant features, see Figure 3. A GA is suited to this type of problem because it performs a randomised search and is not very susceptible to getting stuck in local minima. Moreover, the crossover operator has the effect of merging solutions whilst preserving the already successful feature selections. Yang & Honavar [1998] and Skalak [1994] adopt a similar approach, although Skalak uses random-mutation hill-climbing algorithms instead of a fully-featured GA.

We code the GA population (the chromosomes) as simple vectors of binary genes, where 1s represent relevant features. The "fitness" of solutions is evaluated by running C4.5 on the data from the case-base using only the features corresponding to 1s in the chromosome, and returning the C4.5's estimated accuracy as the fitness. No cross-validation using a test set is done to obtain the value of the fitness, though theoretically it could lead to better results (and is suggested in Figure 3), because of the computational cost. However, the final results are evaluated using cross-validation (see Section 6.1).

Table 1 shows the effect on C4.5 of adding selection. Features labelled "used" were decision nodes in a C4.5 tree induced from all the features. The C4.5+S column includes selection: features marked "use" are selected by the wrapper GA and those with "(d)" appear as nodes in the index. Though C4.5 with selection uses fewer features, it gives improved classification accuracy, as results in Section 6.1 show.

5 Determining Similarity Weights

We now turn our attention to similarity and ways to determine weights that emphasise features influential to useful similarity and ignore the irrelevant features.

5.1 Filter Method

We explored a Value Difference Metric (VDM) as a filter method to determine feature weights. Since standard VDM demands discrete features, and all of the features in CBRTFS are continuous, we use its continuous variant Windowed VDM (WVDM) [Wilson & Martinez 1997]. Payne & Edwards [1998] show that the implicit feature selection property of VDM is able to ignore irrelevant features for a nearest-neighbour algorithm. However, because it assigns importances to the matched features, we consider it here as a feature weighting, not a feature selection, method.

For our problem, WVDM's accuracy was lower than a simple NN algorithm with unweighted Euclidean distance (see Section 6.2). This may have two explanations: first, one of the 8 possible fillers dominates WVDM's probability distributions that are the basis for feature comparison; and second, the attribute values are not evenly

distributed over their domains, and this distorts the probability distributions. Moreover, because our data set is small, the statistics calculated by the WVDM are probably not reliable. Thus the problems are related particularly to our dataset, so no general comments about VDM for CBR feature weighting can be made.

5.2 Wrapper Method

A GA wrapper method determines feature weights for the similarity measure. The GA implementation is largely the same as the feature selection GA, except that genes are real-valued and represent the weights for the features in the similarity measure; Oatley et al. [1998] describe a similar approach. Mutation cannot now flip a gene but changes its value by adding an increment drawn at random from a Gaussian distribution. All other operators are identical to those for standard binary GAs. The fitness measure is the accuracy of a NN classifier using the weights (as defined in the chromosome) in its Euclidean distance measure. The accuracy is determined using leave-one-drug-out cross-validation (see Section 6).

Table 1: Feature selection and weighting results.

		Selection				Weighting	
	Feature	RELIEF-F (K=1)	RELIEF-F (K=1)	C4.5	C4.5+S	A	B
Physical	P1	*0.019*	*0.018*	used		0.15	0.13
	P2	0.035	0.029		use	0.74	0.82
	P3	0.053	0.076	used		0.77	0.56
	P4	0.036	0.059	used	use(d)	0.36	0.80
	P5	0.025	0.038	used	use(d)	1.00	0.89
Chemical	C1	0.098	0.085			0.24	0.28
	C2	0.057	0.050	used	use(d)	*0.00*	0.19
	C3	0.049	0.064			0.21	0.70
	C4	0.102	0.115	used	use(d)	0.57	0.55
	C5	0.074	0.089		use	0.15	0.75
	C6	0.056	0.057		use	0.18	0.94
	C7	0.032	0.039			0.67	0.62
	C8	*0.005*	*0.017*			0.02	0.39
	C9	*0.008*	*0.005*	used	use(d)	0.08	0.82
	C10	0.088	0.086	used		0.67	0.66
	C11	*0.021*	0.025			0.34	*0.00*
	C12	0.030	0.028	used	use(d)	0.54	1.00
	C13	0.030	0.028			0.60	0.41
	C14	0.035	0.032	used	use(d)	0.22	0.06
	C15	*0.017*	*0.017*			*0.00*	*0.00*
	C16	0.071	0.079			0.12	0.32
	C17	*0.004*	*0.003*			0.48	0.03
	C18	*0.014*	*-0.003*			0.52	0.28
	C19	*-0.015*	*-0.019*		use	0.02	0.34
	C20	-	-		use	0.37	0.98
	Dose	0.213	0.157	used	use(d)	0.95	0.10

Two of the many sets of weights found are shown on the right of Table 1. A peculiarity of our results is that quite different sets of weights lead to the same overall filler-prediction accuracy (though the details of the results are not the same). Even the dose weightings from A and B are very different, despite dose being the only feature that had an increased weighting in CBRTFS and this was substantial. This variation in competitive weightings suggests the possibility of retrieving cases using several sets of weights and then combining the results in the adaptation stage. Even a simple voting method might be sufficient to exploit this redundancy.

6 Experimental Results

In this paper, we restrict attention to filler prediction rather than the complete formulation. Also, selection and weighting are evaluated separately, though ideally they should be evaluated together as a part of CBR retrieval stage; this will be the next step in our research.

For feature selection we use C4.5's standard cross-validation, a *leave-one-(tablet)-out* approach. For feature we weighting and earlier CBRTFS testing we use *leave-one-(drug)-out* where formulations for all but one drug form the case-base and formulations for the excluded drug are predicted – different doses of a drug may require different excipients.

6.1 Feature Selection

As already mentioned in Section 4.2 the fitness measure used by the GA for feature selection was very simple. It had the advantage of being relatively fast to compute, as it was based on the C4.5 accuracy estimated only from the training set. A better evaluation of the accuracy would, however, require cross-validation. This is how the final feature selection was evaluated, and the results are presented in this section.

The goal of this testing is to judge the effectiveness of the GA wrapper for feature selection. In particular we wish to compare the performance of decision-tree indexes created from selected features with those that have a completely free choice of features for the decision nodes. We assess the accuracy of the index by using the decision tree to predict the filler for the new tablet. Of course the index is really used to retrieve all the cases at the leaf node rather than used directly to make a prediction. Table 2 presents the output from C4.5 after running the cross-validation script on the 39-drug dataset. The indexes built with the selected features have a reduced error for both the training formulations (case-base) and the testing formulations (leave-one-tablet-out). The estimated error is a prediction for unseen tablets.

Table 2: Feature selection improves C4.5 classification.

Features		Before pruning #nodes	Before pruning Error	After pruning #nodes	After pruning Error	Estimated error
All	Train:	30.7	14.3%	30.5	14.3%	22.3%
	Test:		26.3%		25.6%	
Selected	Train:	34.5	13.5%	31.5	13.7%	22.1%
	Test:		20.2%		20.2%	

It should be noted that doing the evaluation using leave-one-tablet-out is not the best possible choice, because usually there is very close similarity between tablets using the same drug, making the results too "optimistic". However this approach was used because it is directly supported by C4.5 cross-validation script. To compensate for the presence of 8 very similar tablets for each drug we required C4.5 to produce tree leaves having at least 8 examples (the -m 8 option of C4.5) both during GA search and during the evaluation of the results.

6.2 Feature Weighting

Table 3 compares the filler prediction of CBRTFS [Craw et al. 1998] with the predictions of an NN algorithm and various feature-weighting methods for the 13-drug dataset; the 39-drug results are shown in brackets. We should note that CBRTFS includes an indexing mechanism, and some adaptation, whereas the other NN systems have neither indexing nor adaptation.

The GA wrapper method provided such good feature weights that not only did the weighted nearest-neighbour algorithm substantially improve on non-weighted versions but also the improvement over CBRTFS is considerable, despite the adaptation available in CBRTFS. WVDM's poor performance was discussed in Section 5.1.

Table 3: Filler prediction with feature weightings.

CBRTFS	NN + VDM	NN	NN + GA
62% (-)	50% (66%)	56% (67%)	65% (74%)

Though the results from the NN classifier might seem worse than the results obtained using C4.5 decision tree, one should remember that here the more realistic leave-one-drug-out cross-validation was used and not the leave-one-tablet-out cross-validation.

7 Conclusions & Future Work

We have presented a practical example of a generic approach to feature selection and feature weighting. A GA wrapper method achieved effective feature selection with a resulting improved predictive accuracy for C4.5. A similar GA wrapper method tuned feature weighting for an improved accuracy with nearest-neighbour retrieval. Also, the computational cost of the GA wrapper is not a problem, at least for tasks of the size presented here. Although filter methods did not perform well here, this is probably due to characteristics of our data set.

Basic GAs with simple "fitness" functions achieved good improvements with feature relevance and weighting. We optimised both C4.5 and NN classifiers to minimise the (estimated) classification error, thus benefiting from a wrapper's ability to incorporate the learning algorithm in the GA's fitness. But it is easy to let the GA optimise any criterion; e.g. requiring an index tree with a small error rate but also few nodes. Since the approach can be adapted to other fitness measures, it is likely to be applicable in other domains and to other more sophisticated CBR systems.

The optimisation was done in two separate stages. We first optimised the selection of index features. Secondly, without focusing on the cases retrieved by the index we optimised the feature weighting for the similarity measure (as applied to the whole case-base). The feature weighting optimisation could also be applied to the cases retrieved using a previously optimised index, possibly leading to better results. Of course it is possible to optimise the whole retrieval system, but we must be aware of the potential complexity. The GA algorithms would have to be wrapped around a fully functional CBR retrieval system - which here is implemented in ReCall. From the implementation point of view this could be done because of the ability to interface ReCall with other programs. However, such a construction would be much more computationally intensive, certainly if one considers also that the GA chromosome would be twice as long.

Obviously, a question arises how much of these optimisations are data related and how much of the derived knowledge about feature relevance and the relative importance for similarity reflects real domain knowledge. This will require further more elaborate testing involving other strategies for generating training and testing sets. We must also discuss our results more fully with the domain expert.

In similarity-weight determination using GA the fitness was calculated using only the best match returned by the NN algorithm. However, with adaptation the 2nd or 3rd best match may affect the prediction. This suggests a more elaborate feedback for the quality of retrieved cases. Oatley et al. [1998] use a ranking provided by domain experts, but our goal is to reduce the expert participation as much as possible. Several methods of scoring the ranking of retrieved cases have been used, from simple counts of the positions of good matches in the ranking, to measures taking account of estimated similarities between fillers. However, which of these performs best can be judged only by testing with a prototype CBR system with a functioning adaptation phase. This will be the next stage of this research, together with partially automating knowledge acquisition for adaptation.

Acknowledgments

We thank our expert Ray Rowe and Zeneca Pharmaceuticals for help with tablet formulation. We also acknowledge Isoft's software contribution to the project. We also would like to thank the reviewers of the draft version of this paper for their useful comments.

References

Aha, D. (1998). Feature weighting for lazy learning algorithms, *in* H. Liu & H. Motoda (eds), *Feature Extraction, Construction and Selection: A Data Mining Perspective*, Norwell MA: Kluwer.

Almuallim, H. & Dietterich, T. G. (1992). Efficient algorithms for identifying relevant features, *Proceedings of the Ninth Conference on Artificial Intelligence*, Morgan Kaufman, Vancouver, pp. 38–45.

Blum, A. L. & Langley, P. (1997). Selection of relevant features and examples in machine learning, *Artificial Intelligence* **97**(1–2): 245–271.

Craw, S., Wiratunga, N. & Rowe, R. (1998). Case-based design for tablet formulation, *Proceedings of the Fourth European Workshop on Case-Based Reasoning*, Springer, Dublin, Eire, pp. 358–369.

John, G., Kohavi, R. & Pfleger, K. (1994). Irrelevant features and the subset selection problem, *in* W. W. Cohen & H. Hirsh (eds), *Machine Learning: Proceedings of the 11th International Conference*, Morgan Kaufmann, San Francisco, CA., pp. 121–129.

Kononenko, I. (1994). Estimating attributes: Analysis and extensions of RELIEF, *Proceedings of the European Conference on Machine Learning (ECML-94)*, Catania, April 1994.

Oatley, G., Tait, J. & MacIntyre, J. (1998). A case-based reasoning tool for vibration analysis, *in* R. Milne, A. Macintosh & M. Bramer (eds), *Proceedings of the 18th BCS Expert Systems Conference*, Springer-Verlag, Cambridge, December 1998, pp. 132–146.

Payne, T. R. & Edwards, P. (1998). Implicit feature selection with the value difference metric, *in* H. Prade (ed.), *ECAI 98: 13th European Conference on Artificial Intelligence*, Wiley, pp. 450–454.

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

Rowe, R., Craw, S. & Wiratunga, N. (1999). Case-based reasoning – a new approach to tablet formulation, *Pharmaceutical Technology Europe* **11**(2): 36–40.

Skalak, D. B. (1994). Prototype and feature selection by sampling and random mutation hill-climbing algorithms, *Proceedings of the Eleventh International Conference on Machine Learning*, New Brunswick, New Jersey, pp. 293–301.

Wilson, D. R. & Martinez, T. R. (1997). Improved heterogeneous distance functions, *Journal of Artificial Intelligence Research* **6**: 1–34.

Yang, J. & Honavar, V. (1998). Feature subset selection using a genetic algorithm, *in* Motoda & Liu (eds), *Feature Extraction, Construction and Selection – A Data Mining Perspective*, Kluwer.