Investigation of Generalised Nearest Neighbour in Machine Learning

James MITCHELL
Supervisor: Brent MARTIN

November 15, 2004

Contents

Abstract

Instance-based learning is a machine learning that classifies new examples by comparing them to previously seen examples. Non Nested Generalised Exemplars is one such learning algorithm which combines generalisation to provide support for large and small disjuncts. This paper looks at improving this learners tolerance to noise, introducing several possible techniques. Problems were encounted in the implementation of the extensions, preventing the study of the effect of the extensions.

Chapter 1

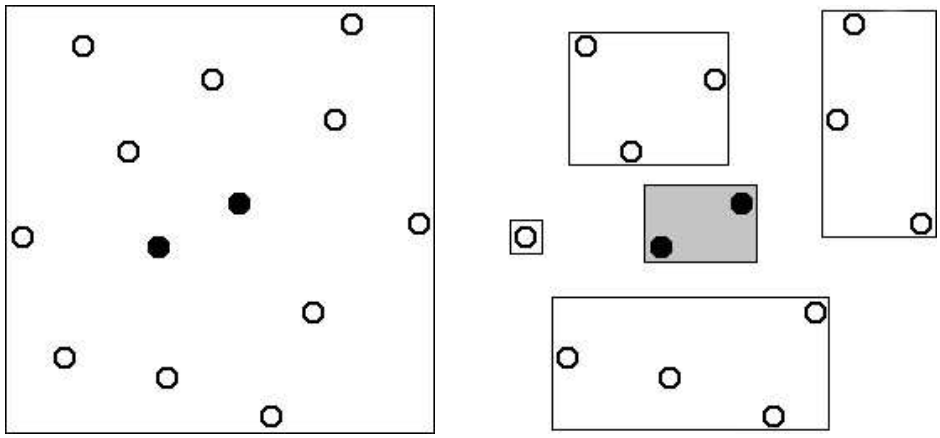
Introduction

Machine learning can be defined as a computer program improving its performance with experience for a given set of tasks. The key concept behind this approach is that a future event with a certain outcome, will exhibit similar properties to an event in the past, that also resulted in the same outcome. More simply, machine learning is using the past to predict the future.

Machine learning has spent a long time in the hands of researchers, and only recently has it begun to spread into commercial products and applications. The Mozilla Project is an example of one application that is currently pursuing the use of machine learning to make it a more "intelligent browser" ((Organization 2004)). Their problem is with auto-completion in the URL-navigation field. Their aim is to rank the URLs in the history using information from history sessions, providing the most relevant URLs at the top of the list.

1.1 Instance-Based Learning

Instance-Based learning is the family of algorithms that learn/classify instances based on their similarity to previously seen instances. The assumption of this approach is that related instances will be in close proximity to one another. Instance-based learners are 'lazy' in the sense that they expend more effort in classifying new instances than learning from the dataset. The measure of similarity is most often described by a Euclidean distance function. The distance between two numeric values is just the subtraction of one from the other, however the distance between colours is more difficult to quantise. This is a problem common to all distance-based similarity functions, and is addressed differently depending on the algorithm.



(a) Ungeneralised exemplars (b) Generalised exemplars

Figure 1.1: Representations of 2 attribute domain

1.1.1 Nearest Neighbour

The most simple of all instance-based learning algorithms is nearest neighbour. It is a purely lazy learner, simply storing the entirety of the training set inside memory, postponing all computation until classification time. In its most basic form, nearest neighbour classifies new examples as the same class as that of its nearest neighbour. While simple, it suffers from over-fitting of the data, in the sense that it essentially learns the dataset, including any noisy instances that it may contain. This prevents the algorithm from learning the correct concepts, decreasing classification accuracy. Another problem with pure nearest neighbour is that the bias of the algorithm is often inappropriate. Nearest neighbour favours small disjuncts or concepts, over large disjuncts. This is because the algorithm does not attempt to generalise larger concepts from the dataset.

k-nearest neighbour is an extension to pure nearest neighbour, instead considering the *k* closest instances during classification. The class with the highest proportion of examples is deemed the best class for the unclassified instance. This solves, to some extent, the amount of over-fitting in the algorithm. Noisy instances should be outweighed by correct instances for a given value of k, reducing the effect of noise on the learner. Choosing an appropriate value for *k* is a problem, with each domain suiting different values of *k*. Most often this is achieved using a trial and error process. Unfortunately, *k*-nearest neighbour also suffers from the same bias as nearest neighbour, reducing the effectiveness of this learner.

1.1.2 Generalised exemplars

Nested Generalised Exemplars (NGE) ((Salzberg 1991)) was an early attempt to introduce generalisation into the instance-based learning domain. It generalised instances into hyper-rectangles, allowing these to nest or overlap. Initial results showed the learner performed well, although further study showed that the results were fortuitous (Wettschereck & Dietterich 1995). They cited the allowance for the nesting and overlapping as the primary cause of the poor performance. (Martin 1995) proposed a new algorithm, based of NGE, however disallowed overlap between exemplar. Hyper-rectangles that conflicted with a new instance, were pruned, preserving the constraint that no exemplar nest or overlap. This approach introduced the ability to represent large disjuncts with generalised exemplar, or hyper-rectangles, while maintaining the bias of nearest neighbour by avoiding overlap. This approach showed promise, with classification accuracies equalling that of C4.5.

1.2 Contributions and outline

(Martin 1995) introduced the learning method of "non-nested generalised exemplars", or NNGE, demonstrating it as a practical solution using generalised exemplars in instance-based learning. It was found that the learner suffered decreased classification performance in noisy datasets. This report investigates this problem, suggesting possible improvements to the algorithm with the aim of improving NNGEs tolerance to noise.

This paper explores non-nested generalised exemplar's approach to the over-fitting of generalisations, as well as the pruning of over-fitted generalisations. It proposes several techniques for this, including different mechanisms to prevent overgenerlaisation, and modifications to the pruning of overgeneralised exemplar.

Chapter Two reviews current techniques for tolerating noise, focusing on instance-based learners. Chapter Three introduces the suggested modifications, providing hypotheses of the results where necessary. Chapter Four discusses the effect of the modifications on the classification accuracy of the learner. Lastly, Chapter Five summarises the results of the research and suggests future work.

Chapter 2

Related Work

Noisy data is not a new phenomenon, often requiring a unique approach that is tailored for each particular learning algorithm. The focus is primarily in an instance-based learning domain.

2.1 Noise

Noise is an unavoidable. Many factors influence noise. Mis-reading of data, errors during data-entry are human-related influences of noise. For example, the recording of the temperature of a room could be influenced by the heaters turning on at a given time. Noise tends learners toward over-fitting of the training data. This is because the training data does not represent the concepts, but the fake concepts brought about as a result of noise.

2.2 Nearest neighbour

Pure nearest neighbour classifies a new instance to the class of the single closest exemplar. This approach degrades in performance in noisy datasets, with an instance containing noisy attribute-values misrepresenting the surrounding space.

k-nearest neighbour is simple approach to reduce over-fitting in the algorithm. By choosing the most popular class-value of the *k*-nearest neighbours, noisy instances can be exposed by the other close neighbours, and avoided. This calculation is performed without a significant increase to the complexity of the algorithm.

2.3 Classification records

IB3, a member of IB family, is a noise tolerant extension of IB2. It maintains classification records for all saved instances (ie their number of correct and incorrect classifications of subsequently presented training instances). The instances with significantly good records are used in subsequent classifications.This is an incremental process, with instances coming and going from the set of 'useful' instances. The acceptance level is if the instance's classification accuracy is statistically, significantly greater than than their class' observed frequency. This also allows for smaller storage requirements in noisy domains, as the poor classifiers will be removed from memory.

2.4 NNGE

NNGE uses a similar approach by maintaining classification records, however does not drop any instances if they become poor. NNGE maintains records for exemplars, not just each instance. The difference to IB3 is the classification records alter the similarity between two exemplars. Poor classifying exemplar increase the distance, thus making it more difficult for the exemplar to have any effect on classification. Because of the requirement for exemplars to store all instances within it, there can be no storage benefit as in IB3

2.5 Pruning decision trees

Noise is not common to only instance-based learners. Rule induction systems are also susceptible to noisy datasets. The presence of noise in a dataset can cause decision trees such as ID3 to branch on the wrong attribute, leading to an overspecialised tree. Rule post-pruning is a technique aimed at increasing classification performance. Rule post-pruning works by first converting the tree into a set of rules, and then pruning each rule by removing any preconditions that result in improving its accuracy. The rules are then sorted by their accuracy and are considered in this sequence during classification. The reasons for converting the tree into rules it to eliminate the limitations of modifying rule-trees. Each path in the tree is a separate rule, and can be pruned accordingly, however should it have to be pruned as a tree, there are only two choices, prune the node, or not, possibly also pruning other rules. Also reduces the problems when pruning at the root of the tree.

Chapter 3

Research

This chapter introduces the specifics of the NNGE algorithm, before doing stuff.

3.1 NNGE algorithm

Non-nested generalised exemplars.

3.1.1 Hyper-rectangles

Hyper-rectangles are represented as a class value and the bounds on each attribute that define its borders. For continuous attributes, the maximum and minimum attribute-values are stored. These maximum and minimum vales describe the range of values covered by the hyper-rectangle. For symbolic attributes, a list is maintained of the attribute-values covered by the hyper-rectangle.

To add an example to the hyper-rectangle, the attribute bounds that do not overlap are extended. For symbolic attributes, the value is added to the list of covered values. For continuous attributes, the maximum value is increased, or minimum decreased until the new example is covered by the hyper-rectangle.

3.1.2 Classification

NNGE classifies new examples by finding the closest exemplar using a modified Euclidean distance function. The modifications required allow the correct computation of distances from hyper-rectangles. The distance between two continuous points is the difference in values, divided by the range of possible values. The division normalises the distance, so to not favour attributes with a small range over attributes with a large range. The distance between two symbolic values id defined

as a distance of 1. A hyper-rectangle and an example that lies within its bounds is treated as having a distance of 0.

NNGE treats missing values by ignoring them. Should either the example or hyper-rectangle contain missing attribute-values, then they are skipped, not counting toward the distance function. Finally the distance is divided by the number of non-missing attributes, as to not punish those with all attribute-values present.

3.2 Modifications

The modifications are listed below. They describe the reason for change, along with the changes necessary to the algorithm. The positive and negative effects of each change are also hypothesised.

3.2.1 Preventing over-generalisation

The way in which NNGE prevents over-generalisation with respect to conflicts was defined as a weakness. While NGE allowed exemplar overlap, NNGE does not allow for any nesting or overlap. For a noisy instance that conflicts with a hyper-rectangle, the hyper-rectangle is pruned. This means that the noisy instance has destroyed the original hyper-rectangle, thus losing the generalisation power of the original hyper-rectangle. In other words, NNGE treats all instances equally during generalisation, leading to rogue instances to disrupt the generalisation process.

The problem was narrowed down to the constraint which must be satisfied before pruning can commence. The original algorithms constraint stated that no two exemplar of differing class may nest or overlap. This constraint is too strict, not allowing for an error in attribute-values. Below are listed two alternatives, both of which employ nesting to hide unwanted instances.

k-NN based constraint

This constraint is an attempt to use a proven effective instance-based learning approach, modifying it to determine if the hyper-rectangle should be split. Conflicting instances are stored inside the exemplar, and split according to the following heuristic:

> A hyperactives should be pruned if there exists a conflicting instance that has another conflicting instance as one of its k-nearest neighbours.

In other words, add conflicting examples into the existing hyper-rectangles, with no effects. After adding the conflicting example, check that there does not

exist a conflicting example where there is another similar example within the k-nearest neighbours for that hyper-rectangle.

This approach will tolerate small amounts of conflicts within a hyper-rectangle before pruning occurs. Should the conflicts contain noisy attribute-values, then the learner will have benefited by not pruning a correct generalisation. However, should the conflicting exemplar represent a small disjunct, or a disjunct that is under-represented in the dataset, the learner will have failed in learning a concept in the data.

Nesting constraint

If you could create a hyper-rectangle inside another, it suggests that there could be a small disjunct inside the generalisation. The nested hyper-rectangle would have to exist such that there were no instances with a conflicting class inside its bounds.

There are potential problems with this approach. An instance might restrict the creation of a hyper-rectangle within another. Should that instance contain noise, then the learner would fail to represent a possible small disjunct.

3.2.2 Exemplar pruning

The introduction of exemplars containing conflicting examples until it is certain that the exemplar must be pruned. This fundamental change to the representation of an exemplar requires a change in how the hyper-rectangle is pruned. Several approaches have been outlined below.

Preserve class

Preserve the class of the original hyper-rectangle, allowing conflicting instances inside hyper-rectangles. It is hypothesised there could be problems in initial stages where noisy instances could create a noise-based generalisation that the algorithm would be fighting against for the rest of learning phase.

Free for all

Destroy the hyper-rectangle, and rebuild new rectangle's that do not contain instances of other classes. In the presence of high levels of noise, one would expect there to be a loss of classification power.

Overthrow

Do not favour the original class, and attempt to create generalisations that cover the class of the example that forced the pruning. This is extreme, and probably not work, but mentioned for the sake of completion. This might be beneficial early in the training, as hyper-rectangles that form because of noise could be hard to split later on, i.e. the algorithm will be constantly trying to prune the noisy hyper-rectangles.

Chapter 4

Results

It can often be difficult to compare results of one learner with another, as details surrounding the testing are often ommitted. It was for this reason that WEKA (Witten & Frank 1999) was used as a workbench to carry out tests. The algorithm was developed to integrate with WEKA, utilising the file readers and analysation tools that are provided. WEKA also maintains implementations of other learners, such as IBk and J48, an implementation of C4.5.

4.1 Test domains

Database	# classes	# attributes	# instances
Iris	3	4	150
Labor	2	16	57
Weather	2	4	14
Soybean	35	19	683

Table 4.1: Test domain details

4.2 Test results

The results in the above table illustrate serious deficiencies with the extended version of NNGE. This is believed to be due to program error, such as not growing/pruning the hyperrectangles correctly.

Domain	**J48**	**NNGE**	**NNGEx**
Iris	96	96	76
Labor	73	77	80
Weather	64	57	57
Soybean	91	91	32

Table 4.2: Test domain details

Chapter 5

Summary and Conclusions

Since the implementation of the extensions contain errors, it is impossible to determine what effect, if any, the suggested modifications would have made. Further work will be carried out to finish the implementation for the benefit of future work.

Bibliography

Martin, B. (1995), Instance-based learning: nearest neighbour with generalisation, Master's thesis, University of Waikato.

Organization, T. M. (2004), 'applying machine learning to autocomplete'. *http://www.mozilla.org/projects/ml/autocomplete/

Salzberg, S. (1991), 'A nearest hyperrectangle learning method', *Mach. Learn.* **6**(3), 251–276.

Wettschereck, D. & Dietterich, T. G. (1995), 'An experimental comparison of the nearest-neighbor and nearest-hyperrectangle algorithms', *Mach. Learn.* **19**(1), 5–27.

Witten, I. H. & Frank, E. (1999), *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*, Morgan Kaufmann.