Fuzzy Logic vs Machine Learning
An Investigation

Kevin Pulo, login kpulo
kev@zip.com.au
http://www.zip.com.au/~kev/

June 1999

Abstract

Fuzzy logic has allowed subjective, real-world imprecisions to be modelled with high degrees of accuracy, compactness and ease. However the fuzzy rules (most often used in control systems) have, for the most part, been constructed manually by experts in the problem domain.

This project aimed to investigate the effectiveness of standard machine learning algorithms when compared to human based learning processes, expressed in concrete terms by fuzzy rules. The results highlight several advantages standard machine learning algorithms have over humans, and suggest numerous directions in which fuzzy advances could be made to improve the algorithms.

1 Introduction and Problem Description

1.1 Fuzzy logic

Fuzzy logic is a mathematical methodology (and a philosophical ideology) that is similar in construct to boolean algebra and similar in appearance to probability, but more general than both in fundamental ideas. This generality is the freedom for truth variables to hold any value between 0 and 1 (inclusive), and fuzzy logic proponents claim this generality allows greater flexibility, freedom, accuracy and compactness when representing real world situations. All the usual properties of boolean algebra can be extended to fuzzy logic, and probability's degree of belief in a boolean variable becomes a fuzzy variable's degree of truth.

Fuzzy logic has had a great deal of success where it has been applied in the real world, and is often touted as a means of making machines smarter. However, for the most part, fuzzy logic has actually been used in control systems as a way of providing more human-like behaviour. For example, many household appliances (notably white goods such as washing machines and refrigerators) now contain fuzzy control systems that allow the machine to adjust to the specific circumstances currently presented to it, with some even learning patterns of usage. While this does improve the machines and their behaviour, it isn't really an advance in the areas of general intelligence on arbitrary domains and machine learning. Only (relatively) recently is the area of fuzzy machine learning being developed, with the most popular approach being fuzzy neural networks (often abbreviated to simply neuro-fuzzy) to allow machine intelligences to learn from arbitrary empirical data and experiences.

1.2 Fuzzy rules

Fuzzy rules may be expressed in terms such as "If the room gets hotter, spin the fan blades faster" where the temperature of the room and speed of the fan's blades are both imprecisely (fuzzily) defined quantities, and "hotter" and "faster" are both fuzzy terms. Fuzzy logic, with fuzzy rules, has the potential to add human-like subjective reasoning capabilities to machine intelligences, which are usually based on bivalent boolean logic. [Wang, 1993] provides an excellent introduction to fuzzy logic and how it can be applied to AI, and [Kosko, 1993] is an excellent popular science book describing all aspects of fuzzy logic.

For the most part, the fuzzy rules that are used in control systems are hand-crafted by the designers of the systems, and machine learning is rarely employed[1]. As such, it can be argued that this human input into the system's design constitutes a homunculus, and that such systems can never be independently intelligent[2].

This project aimed to exploit this in order to investigate the effectiveness of machine learning algorithms when compared to human learning processes. The fuzzy rules served well as they provide an excellent means of compactly representing what the human has learned in a way that is both accurately human-like and subjective, and can be used in controlled experiments to be compared against standard machine learning algorithms.

The compactness of the rules is desirable because there exists evidence suggesting smaller rules perform better [Holte, 1993], with reasons essentially the same as those for overfitting in decision trees – the underlying structure of the process generating the data is captured rather than the superficial structure of the training data. Fuzzy logic appears to be very well suited to the creation of small rules, as fuzzy rules have a higher "information density", so to speak – each rule encapsulates a richness of information and meaning.

2 Method

The project comprised a series of controlled experiments on the performance of various algorithms on a standardised set of datasets from [Blake et al, 1998,

[1]Most likely because the cost of devising and implementing learning systems is prohibitive, especially for the somewhat limited and disjoint problem domains.

[2]The other school of thought is that initial input into the design of a system does not constitute a homunculi, and that systems without independent intelligence require a homunculus to be present at all times. Both of these viewpoints are debatable.

the UCI ML dataset repository]. The primary performance measure considered was accuracy – the number of correct classifications an algorithm infers based on the given set of training data with classifications[3]. The training time was also considered as a somewhat informal secondary performance measure.

Two different categories of algorithms were examined. These were standard non-fuzzy ML algorithms, such as C4.5, and hand-crafted fuzzy rules based on a human's examination of the training data, processing the validation dataset with the aid of a fuzzy logic software toolkit. It was hoped that a third category, that of neuro-fuzzy ML algorithms could be analysed, but time restrictions prevented this.

The Delve environment [Rasmussen et al, 1996] was considered as the experimental environment, but was rejected as it is excessively large and overfeatured for the small scale of this project, and this additional complexity would merely have hindered the project. In addition, although it does aim to be general in its goals and application areas, it isn't very well suited to the idea of testing a "human" algorithm.

2.1 Choice of dataset

In choosing the datasets to be used as the basis for the comparison, several factors had to be evaluated. Obtaining existing results for the standard ML algorithms was of considerable importance, to avoid the peripheral task of sourcing existing standard ML algorithms and running them on the datasets. Fortunately, [Lim et al, 1999] provides an excellent and current summary of the performance of numerous standard ML algorithms – twenty two decision tree, nine statistical and two neural network – on sixteen datasets from [Blake et al, 1998, the UCI ML dataset repository], both with and without added noise.

The other main criteria for the datasets was a restriction in the size and complexity of the dataset. This is because when constructing the fuzzy rules the human will ultimately be examining all the training data, and the scale of this project was limited. Thus, datasets were preferred to have a small to medium number of training instances (no more than, say 1000), covering all of the attribute space if possible, small to medium number of attributes (no more than, say 20), and discrete attributes and classes, each with a small number of possible values (no more than, say 10). The reasons behind some of these requirements will be discussed in Section 3.2.1.

The datasets were split sequentially into a training set of the first 70% of the instances and a validation set of the last 30% of the instances. The reasons and ramifications of this will be discussed in Section 3.2.2.

The dataset chosen as the best for this work was the 1984 United States Congressional Voting Records Database (voting-records) from [Blake et al, 1998, the UCI ML dataset repository]. This dataset records the votes of 435 US Congressmen on 16 key questions, where each attribute (question) can have the value "yea", "nay" or "abstained", and each Congressman is classified as a Democrat

[3]In other words, only supervised learning was considered, not unsupervised learning.

or Republican. Further, this dataset has results in [Lim et al, 1999].

Unfortunately, an initial underestimate of the time required to create the fuzzy rules and the generally limited scale of the project precluded other datasets from being tested. The most desirable datasets for testing were the Mushroom Database from [Blake et al, 1998, the UCI ML dataset repository] (difficult because of its large number of instances and attributes) and the Car Evaluation Database from [Blake et al, 1998, the UCI ML dataset repository], which would have required results for one or more standard ML algorithms (such as C4.5) to be run on it.

2.2 Hand crafted fuzzy rules

2.2.1 Human details

When constructing the fuzzy rules, the human was actively encouraged to take advantage of all human thought processes and percepts, as this forms the real basis of the project – if these human characteristics can be used to gain an advantage over standard ML techniques. However, the visualisation techniques available were limited in sophistication, due to the limited scale of the project. The techniques that were used were sorting of the training set on different fields (though usually with the class sorted first) and subspace plots, where two or three of the attributes were plotted on a plane or in a volume along with their classifications. For boolean valued attributes, such as those in the Voting Records dataset, subspace projections are essentially statistical, multidimensional histograms. It was originally hoped that several sophisticated approaches to the visualisation of the training set could be applied, such as scientific visualisation software such as AVS and IBM Data Explorer (IBM DX), and full statistical makeup properties of the dataset, however, these proved to be too large to be properly incorporated into the project. This subject is discussed more fully in Section 4.2.3.

It was also important that the human attempted to learn the patterns in the dataset, as opposed to using external knowledge base information. This was important because we were testing the human's ability to learn patterns, not their knowledge on the particular domain. For example, with the Congressional Voting Database, a US Political Scientist or Analyst would almost certainly be able to correctly predict a Congressman's party (Democrat or Republican) given the questions and the Congressman's votes on those questions. However, this is because they know what it means to be a Democrat or Republican, rather than simply observing how Democrats and Republicans vote. This is far from the objectives of this project, and luckily the human involved had no prior knowledge on what Democrats and Republicans believe in (and therefore would be likely to vote for). This subject is discussed further in Section 4.2.4.

2.2.2 Fuzzy logic toolkit

The fuzzy logic toolkit selected for the project was [Hillegas, Fuzzy Java]. This toolkit was written in a good language (Java), with reasonable documentation and good primitives for building fuzzy rules. Fuzzy rules in this toolkit took the form of "tune" statements, that are akin to a fuzzy version of the "switch" or "case" statements of traditional programming languages. In these statements, all the cases in the tune statement are executed to some degree, this degree is based on how true their condition is and their output is weighted accordingly.

A supporting program was written in Java to read in a dataset, apply the hand constructed, hard coded fuzzy rules to each instance, and then apply the defuzzification procedure to obtain the classification for that instance. The hard coding of the fuzzy rules proved adequate, but limited some possibilities; this is discussed in more detail in Section 4.2.2.

3 Results

3.1 Results obtained

[Lim et al, 1999, page 13] reports that the Congressional Voting dataset used is one of the easiest to classify, with error rates between 4% and 6%. Further, from [Lim et al, 1999, page 19] we find that most algorithms have training time less than 30 minutes, with the best being less than 3 minutes.

The fuzzy rules hand developed for this project were extremely difficult to obtain with the short amount of time and resources available for the project. This is borne out by the error rates – greater than 40% in all attempts, with the worst case being around 60% – and the training time – between 2 and 6 hours for each attempt.

Each attempt had a different set of fuzzy rules, with a different approach to solving the classification problem. Further, once the human had deemed that the rules were as desired (based on the human's satisfaction with the rules and their performance on the training set) the rules were run once on the validation set. The reasons for this requirement are discussed in Section 3.2.2.

The relationship was observed that the more time and effort put into analysing the data, the better the results were. In particular, the more sophisticated the visualisation techniques, and the higher the abstraction of the representation away from the raw data, the easier it was to obtain rules (and further, to obtain rules which subsequently performed slightly better). The problem was that sophisticated visualisation techniques were harder to implement and took more time than simple ones. This is discussed in detail in Section 4.2.3.

Unfortunately, the large amount of time spent on the Congressional Votes dataset meant that other datasets couldn't have rules constructed for them, as described in Section 2.1.

3.2 Problems encountered

Numerous problems were exposed during this project, these are detailed here. Where appropriate, they are also discussed in Section 4.

3.2.1 Restrictions on datasets

It was noted earlier in Section 2.1 that the datasets to be analysed required several restrictions, most notably on their number of instances and attributes, and on the number of possible values each attribute may take. The reason is that since this project has limited scale, and that a human was required to analyse the data, overly complicated data would simply "clog" the human's mind, making the issue of finding rules for the dataset substantially harder.

For example, a dataset such as the www.microsoft.com web server log files (also from [Blake et al, 1998, the UCI ML dataset repository]) has approximately 160 attributes, most of which are nominally valued. For a human to be able to make sufficient sense of these to be able to make accurate classifications would be an immensely formidable task, well beyond the reach of this project.

There is another dilemma concerning the complexity of the datasets, relating to whether datasets should have continuous or discrete attributes and classes. Fuzzy logic has the greatest advantage over bivalent techniques when applied to continuous values; when applied to discrete values it can be reduced to being a somewhat advanced version of bivalent logic. However, datasets with continuous values are considerably more complex and involved than those with discrete values. Thus, as above, discrete valued datasets were chosen in the interests of restraining complexity.

3.2.2 Problems splitting the dataset into training and validation sets

In their comparison of datasets, [Lim et al, 1999] used a ten-fold cross validation technique. That is, in basic terms, they split the dataset into a 90% training set and 10% validation set for 10 different disjoint set splits, then running the algorithm over each of the 10 splits and averaging the results. However, there is certainly not enough time for the human to perform the rule construction 10 times over, especially when in theory the rules for the dataset should be the same (and the human is aware of this fact). As such, it is virtually impossible for the human to run several truly distinct iterations of the rule generation with the different training sets (as required for the cross validation), as human memory will skew and bias the later results.

As such, a split of 70% training data, 30% validation data was used in an attempt to partially compensate for these different validation results – the larger validation set should provide a better test of the created rules without sacrificing too many training instances.

A related problem was that the dataset was split sequentially – that is, the training set was simply the first 70% of the instances, the validation set the last 30%. It isn't known if the data is listed with any drifts or trends in it (but there is no reason to assume so). The instances in the dataset should have been randomized before being split into the training and validation set, however it is believed that any effect not doing this may have had is negligible when compared with the overall results.

A related problem is that of the feedback from the training set often not being particularly helpful, for the same reasons as when overfitting. Once the rules have been run on the validation set, the results cannot be used to further improve the rules, as this would simply merge the validation set into the training set. Thus the rules must be run on the validation set once only, at which point they cannot be further developed. This leaves the human in a very awkward and counter-intuitive position of having little information with which to decide if the rules are complete. The training set results are further dubious because of overfitting – large errors do not necessarily indicate that the rules are particularly bad[4]. More importance must be placed on how the human feels about the rules, that is, if the human thinks they accurately represent the underlying structure of the problem, usually in terms of their elegance, their being sufficiently large (and yet not overly large), and other measures of the human's satisfaction, and these can be hard to gauge.

4 Discussion

This project has had several major failings in its attempts to use fuzzy logic and human inference to learn under supervision from a standard dataset. However, much knowledge has been gained about the advantages that standard ML has over human learning processes, and suggestions for future experiments in this area.

4.1 Advantages of standard ML

Standard ML algorithms, as with many computer algorithms, are very well suited to performing mechanical manipulations of very large datasets very quickly. This is excellent for problems such as the www.microsoft.com web server logs, with many attributes. Only for datasets with extremely large numbers of instances, with intractable running times of standard ML algorithms would the cost of hand crafting fuzzy rules be acceptable.

In addition to this, the training time is often much shorter than for humans. Often ML algorithms can be very accurate with only a very small expenditure in terms of their training time. Again, for all but the very largest datasets, this would outweigh the effort required for a human to construct fuzzy rules.

Humans perform much better when they are able to interpret and gain meaning, understanding and information from the training data. This is when they can generalize best and draw the best conclusions. ML algorithms, however, have no such requirement, and can apply techniques such as decision trees and Bayesian inference to obtain results approaching the probabilistic optimum without any need to comprehend anything.

[4]For example, one of the initial rule sets constructed had a large error on the training set of ~ 70%, yet had an error of ~ 50% on the validation set. This could be coincidence but could also be valid, and this is the whole problem – when faced with a training set error of 70%, how is the human to know if this reflects bad rules or bad luck?

4.2 Suggested new methods and future work

These are the techniques that have been perceived during this work as the best way forward, that is, what would liked to have been attempted given sufficient time.

4.2.1 Fuzzy Decision Trees

The fuzzy rules that were generated were "flat", in the sense they were all applied simultaneously. While this makes more sense with fuzzy rules than with standard bivalent rules, the natural extension is to take a cue from decision trees and attempt to construct fuzzy decision trees. These would be identical to standard decision trees except that the decisions at each branching point would be fuzzy, rather than bivalent. This makes calculating the best split somewhat harder (continuous functions for the information gain must be maximized), but would be more rewarding by allowing smaller trees with less leaves and internal nodes to encapsulate a richer amount of information.

Several papers have been written on this topic; [Boyen, 1995] describes the application of fuzzy decision trees to assessing the stability of power systems, while [Heinz, 1995] describes a hybrid method of adaptive fuzzy neural trees, combining fuzzy logic with both neural networks and decision trees.

4.2.2 Fuzzy Instance Based Methods

This is a method that is similar to standard proximity style instance based learning methods, differing in that it uses fuzzy logic to determine the "proximity" of a given instance to the training set's instances. This would be done (at least initially) by the time and space expensive technique of storing each of the instances of the training set as a disjoint fuzzy rule. Validation instances are then fuzzily compared to each of the rules and weighted according to their "proximity" in feature space (how true rule is given the instance). While storing every training instance would be too expensive, the results to this would prove particularly interesting in suggesting possible ways to trim down the number of instances required.

This method was suggested because during the construction of the fuzzy rules, the human found himself feeling as though he was overfitting the data by adding rules for many different cases in the dataset, some covering relatively few instances. However, the fuzzy nature of the rules meant that these rules weren't weighted as much as more important rules, since they are less true. This then suggested the approach of having a fuzzy rule for each instance.

The main reason this wasn't attempted in this project was the fact that the fuzzy logic toolkit used ([Hillegas, Fuzzy Java]) required the fuzzy rules to be hard coded into the program. Allowing instance based rules would have required modifying the toolkit to suit this, which could not have been feasibly achieved within the scale of this project.

4.2.3 Improved Dataset Visualisation and Data Mining

Relationships observed in the experiment (Section 3.1) suggested that the more sophisticated the visualisation techniques, the better the results of the human's fuzzy rules. As such, it is anticipated that the visualisation of the data set is the key to allowing the human to properly "digest" the information contained in the underlying structure of the data. An experiment to test this would involve several artificially constructed datasets, each with the same number of instances, attributes and so on, but each generated from different rules of equal complexity. A human would then apply several visualisations of varying complexity to these different datasets and observe the resulting performances.

4.2.4 Exploiting an Expert's Knowledge Base

It was remarked earlier (Section 2.2.1) that a US Political Analyst would "obviously" be able to determine a Congressman's party given their votes. However, would this necessarily be true? Put another way, would this expert's knowledge base necessarily yield better results than a learning algorithm operating on an observed dataset? After all, there are always exceptions to rules, and people can be detrimentally biased by their knowledge base, so does the knowledge base really help? The person does, however, have the advantage that if they don't know the answer to a given question, they can attempt to learn what is required to answer it.

This then begs the question of can eager learning algorithms discover what they need to learn in order to make correct conclusions? An interesting experiment would be to compare the results of a human expert with standard ML algorithms and eager learning expert systems.

References

[Blake et al, 1998] C. Blake, E. Keogh and C. J. Merz. *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.

[Boyen, 1995] X. P. Boyen and L. Wehenkel. Fuzzy Decision Tree Induction for Power System Security Assessment. *Proceedings of SIPOWER '95, IFAC Symposium on Control of Power Plants and Power Systems*, Cancun, Mexico, 1995.

[Heinz, 1995] A. P. Heinz. *Adaptive Fuzzy Neural Trees.* Institut fur Informatik, Albert-Ludwigs-Universitat, Freiburg, Germany, 1995.

[Hillegas, Fuzzy Java] R. Hillegas. *Fuzzy Java Toolkit.* `http://www.slip.net/~bcondon/adj/top.html`

[Holte, 1993] R. C. Holte. Very Simple Classification Rules Perform Well on Most Commonly Used Datasets, *Machine Learning*, 11:63-91, 1993.

[Kosko, 1993] B. Kosko. *Fuzzy Thinking, The New Science of Fuzzy Logic.* Flamingo (Harper Collins), London, 1993.

[Lim et al, 1999] T.-S. Lim, W.-Y. Loh and Y.-S. Shih. *A Comparison of Prediction Accuracy, Complexity, and Training Time of Thirty-three Old and New Classification Algorithms.* Department of Statistics, University of Wisconsin, Madison, WI, 1999.

[Rasmussen et al, 1996] C. E. Rasmussen, R. M. Neal, G. E. Hinton, D. van Camp, M. Revow, Z. Ghahramani, R. Kustra and R. Tibshirani. *The DELVE Manual, Version 1.1.* (Data for Evaluating Learning in Valid Experiments, `http://www.cs.utoronto.ca/~delve/`) University of Toronto, Toronto, Ontario, Canada, 1996.

[Wang, 1993] P. Wang. *The Interpretation of Fuzziness.* Center for Research on Concepts and Cognition, Indiana University, 1993.

[Zheng, 1993] Z. Zheng. A Benchmark for Classifier Learning. *Technical Report 474, Basser Department of Computer Science, University of Sydney, Australia,* 1993.