Natural Language Text Classification and Filtering with Trigrams and Evolutionary Nearest Neighbour Classifiers

W.B. Langdon
Centrum voor Wiskunde en Informatica,
Kruislaan 413, NL-1098 SJ Amsterdam
W.B.Langdon@cwi.nl http://www.cwi.nl/~bill/
Tel: +31 20 592 4093, Fax: +31 20 592 4199

Abstract

N grams offer fast language independent multi-class text categorization. Text is reduced in a single pass to ngram vectors. These are assigned to one of several classes by a) nearest neighbor (KNN) and b) genetic algorithm operating on weights in a nearest neighbour classifier. 91% accuracy is found on binary classification on short multi-author technical English documents. This falls if more categories are used but 69% is obtained with 8 classes.

Zipf law is found not to apply to trigrams.[1]

1 INTRODUCTION

It is increasingly important that organisation are able to direct electronic documents rapidly to the right person. Therefore automatic and particularly adaptive mechanisms which match documents to the interest profile of individuals are particularly interesting. Ngrams offer a means of storing and updating user interest profiles and rapid language independent means of comparing profiles and documents.

We show trigrams (3-grams) in conjunction with a nearest neighbours (KNN, $K = 1$) classifier is a fast and reasonably accurate way of classifying natural language documents of modest length. N grams have the advantage of being language independent, but naturally new classifiers are needed for each new language. The introduction of a weight per trigram, i.e. a weight vector, which is trained using a genetic algorithm, improves accuracy further at modest additional computational effort but with potential risk of over training.

This work follows up that by Daniel Tauritz [Tauritz and Sprinkhuizen-Kuyper, 1999a, Tauritz and Sprinkhuizen-Kuyper, 1999b] and uses a modified version of his n gram C++ library revision 0.30 and the Reuters 21578 collection of newswire stories.

Section 2 describes n grams. In Section 3 we describe two experiments using n grams to classify documents. Section 3.1 describes the Reuters 21578 collection and our use of it. The first experiment (Section 3.2) uses nearest neighbours techniques, while the second (Sections 3.3 and 3.4) combines nearest neighbours with genetic algorithms. Section 4 discusses trigram distributions with respect to Zipf's Law. Section 5 discusses these techniques and our results. While Section 6 briefly considers the applicability of improving accuracy by removing suffixes from words. Our conclusions are given in Section 7.

2 N GRAMS

An n gram is a sequence of n consecutive symbols. In text processing the symbols are letters of the alphabet etc. and there is a fixed number of them (say 27). Therefore the number of possible n gram is also fixed (actually being $\leq 27^n$). A text is converted to an n gram vector by counting the number of times each of the possible n grams actually occurs in the text and putting this count in the element of the vector corresponding to the n gram.

One way to count the number of n gram in a document is to consider a movable window n characters wide which is slid across the whole text one character at a time. Each time it stops, the n characters it covers correspond to one n gram and so the count for this n gram is incremented. In European languages it is sufficient to consider only the alphabetic characters in the text regardless of whether they are capitals or lowercase letters. All layout and non-letter characters are treated as separation characters. Two or more separation characters are treated as a single space. Figure 1

[1]GECCO'2000 Late Breaking paper, 8-12 July 2000 Las Vegas, Nevada, USA L. Darrell whitley, et al. (eds.)

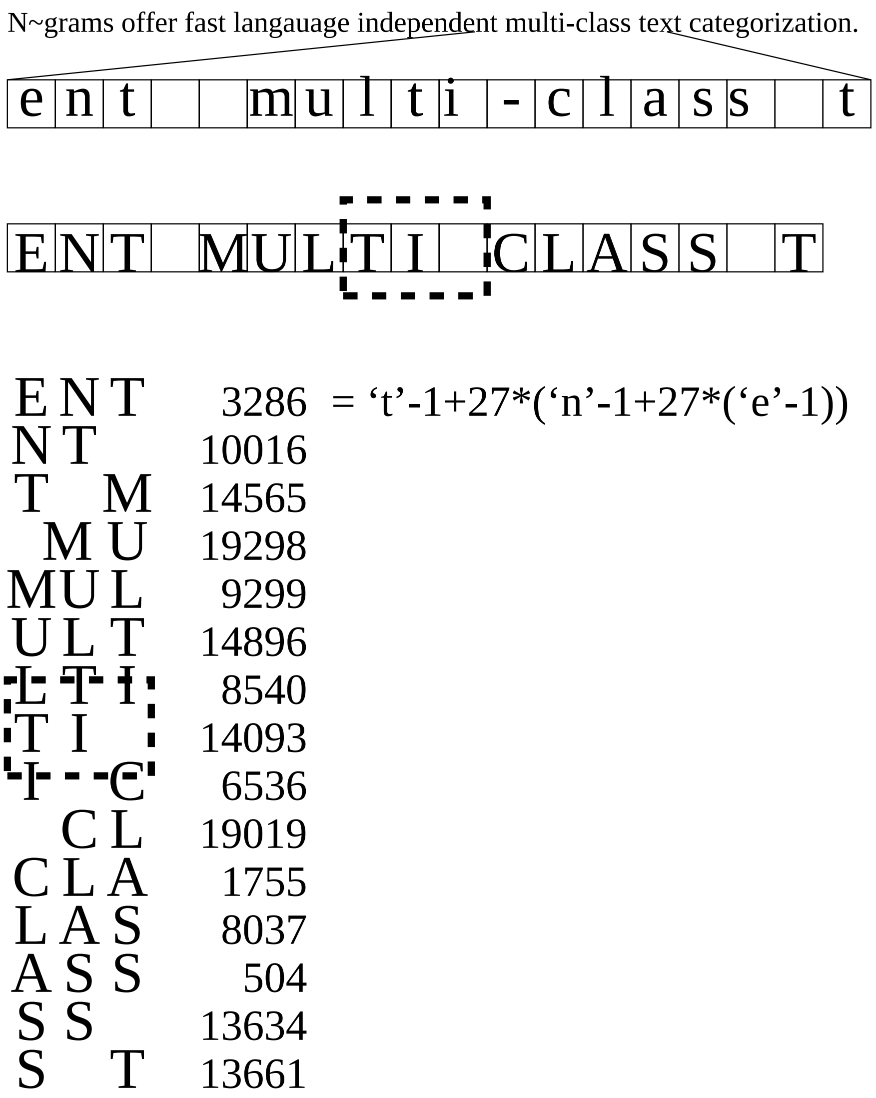

Figure 1: Converting a fragment of text to a trigram vector. Numbers to right are numeric value of each trigram. $'a' = 1$, $'b' = 2$ etc.

shows an example.

In practice most of the possible n grams do not occur and considerable savings in memory and time can be obtained by using sparse data structures. Here we have used `map` data structures but hash tables are commonly used (particularly if $n > 6$). (Commonly exact counts are not needed and so it may be possible to avoid special processing associated with "hash clashes". Instead distinct n grams which happen to overlap in the hash table can be combined in a single count. With suitable hash parameters this will happen infrequently. This avoids time and space complications of rehashing or handling overflow chains.)

3 EXPERIMENTS

3.1 THE REUTERS 21578 COLLECTION

The Reuters 21578 text categorization test collection consists of two parts 1) documents that appeared on the Reuters newswire in 1987 and 2) classifications of the topic of each documents. The collection is distributed *for research purposes only* by David D. Lewis of AT&T Labs via `http://www.research.att.com/~lewis/reuters21578.html` `reuters21578.tar.gz`.

The 8 subject areas we use are Coffee, Crude Oil, Interest Rates, Money/Foreign Exchange, Money Supply, Shipping, Sugar, and Trade. These are also the same as [Tauritz and Sprinkhuizen-Kuyper, 1999a, Table 1].

The documents are of modest length. In the subset we used each newswire story is on average 114 words long but vary greatly in length.

The number of documents about each topic also varies greatly, from a minimum `money-supply` with 97 to `crude` with 355. From each directory we random selected (without replacement) 30 documents for "training" and from the remaining 67–325 we randomly selected (without replacement) 50 more as a "verification" set, i.e. to use to test generalisation. This was done ten times, to create ten such pairs for each subject area.

3.2 NEAREST NEIGHBOURS

The nearest neighbours algorithm is deterministic and fast. The trigram vector for each document in the training set is calculated and normalised (by dividing the number of times each trigram appears in a document by the total number of trigrams in the document).

Each document from the verification set is classified by assigning it the class of the document in the training set to which its normalised trigram vector is the closest. The Euclidean distance between between normalised trigram vectors is used. (There are between 50×2 and 50×8 documents to be classified).

Table 1: Validation performance of Nearest Neighbours Trigrams on Reuters 21578 collection. Means and (standard deviations) of 10 independent runs.

Topics	Neighbours	% Correct		Secs	
Coffee, trade	60	91	(4)	3	(0.4)
+ crude	90	79	(7)	6	(0.3)
+ money-fx	120	77	(7)	10	(0.6)
+ sugar	150	76	(5)	14	(0.3)
+ money-supply	180	76	(4)	18	(0.7)
+ ship	210	72	(4)	25	(0.7)
+ interest	240	69	(3)	33	(2.4)

The percentage of documents correctly classified is given in Table 1. Table 1 also gives the average run time (columns headed "Secs") and the number of reference ngram vectors "Neighbours" each document that is classified is compared with.

3.3 GA NEAREST NEIGHBOURS HYBRID

In the previous section the trigram vectors are unweighted and the distance measure is directly calculated using all (non-zero) trigram frequencies. Naturally this means frequently occurring n grams dominate this distance calculation and so classification but (as Section 4 will show) many n grams occur infrequently and consequently are all but ignored. Here each n gram is allocated a weight. When the distance between two n gram frequency vectors is calculated each corresponding element is multiplied by its weight before the distance is calculated. (Note this is slightly different from [Tauritz and Sprinkhuizen-Kuyper, 1999a, page 5] where frequency vectors are "normalised" again after multiplying by the weights (so the sum of vector elements is unity). Both ours and [Tauritz and Sprinkhuizen-Kuyper, 1999a]'s approaches are slightly different from standard KNN approach where the angle between (weighted) vectors is used as the distance metric. The cosine of this angle is the same as the Euclidean distance between the vectors if both are normalised to have unit *length*. I.e. the sum of the squares of their components comes to unity. We suspect these differences do not produce large differences in performance.)

We follow Tauritz and use a genetic algorithm to determine the actual value of each weight. For our GA we use the object orientated C++ library QGAME [Dekker, 1995]. This is a standard GA which we just use to determine the weights. We have previously been used QGAME with a combinatorial scheduling problem for the electricity transmission grid in the UK [Langdon, 1995, Langdon, 1997, Langdon, 1998].

In the GA each weight is represented on the chromosome by a `float` (we use single rather than double precision for speed). Like Tauritz, weights are constrained to be between zero and 1 and are initialised uniformly at random between these ranges.

The population of twenty weight vectors undergoes non-overlapping generations and elitism is not used. However QGAME does keep track of the highest fitness individual found in the whole GA run and this is used after the GA has finished by the nearest neighbours algorithm to classify the documents in the test set.

On average 60% of each new generation is created by uniform crossover (50% per locus), the others are copies of the previous generation. However all children are randomly subjected to Gaussian mutation (sigma 1.0) at a rate of 3% per gene. Should a mutation attempt to modify a weight to make it negative, the weight is set to zero. If it should try to increase it above one, it is set to one. We use Stochastic Universal Sampling (SUS) [Mitchell, 1996, pages 166–167] with fitness linearly rescaled so the worst member of the population has fitness 1.0 and the best 2.0. The GA continues until either it finds an individual for which nearest neighbours correctly classifies the whole training set or until 100 generations. Each new individual in the GA population is assigned a fitness by using it to classify the whole training set. Its fitness is the correlation coefficient between the category calculated using with nearest neighbours and the correct category. Parameters are summarised in Table 2.

Due to the length of the chromosome it was felt necessary to use highly disruptive genetic operators (i.e. uniform crossover and a high mutation rate and step size). This appears to be the correct choice as the GA does learn at a reasonable rate. The large mutation step size probably means many mutation will be truncated to the legal range of the genes (zero and one). Thus the exact form of the mutation operator (Gaussian in our case) is probably not important.

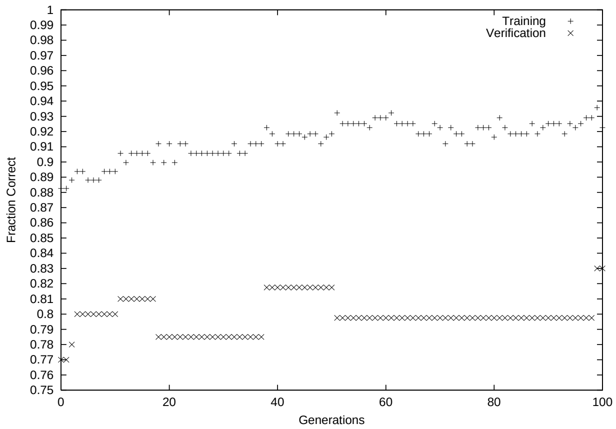

Figure 3: Evolution of Best of Population Individual's score of training and verification sets. (First run with 8 classes)

3.4 GA KNN RESULTS

Figure 2 plots the mean performance on the best of run individual on the test set across ten runs, at the end of the initial random population, generation 20 and generation 100. For comparison it also contains the results given in Table 1 and "system" performance given in [Tauritz and Sprinkhuizen-Kuyper, 1999a, Table 2]. Performance is on average 3% worse than Tauritz reports but this may not be statistically significant. It appears while performance with the best of twenty randomly chosen weights is better than the unweighted system, performance does not improve during the GA run.

The GA does tune the weights and performance on the training documents steadily increases. However performance on the verification set does not improve indicating *over training*. Figure 3 show the best of generation performance on the training set for one GA run together with the performance on the verification set of the same individuals.

The failure of the GA to usefully abstract from the training documents leads to investigations of the expected behaviour of the distribution of trigrams within natural language text. We hope a model may be found and then the GA could learn parameters of this model and then these could be used to classify the texts.

4 ZIPF'S LAW

It has been shown that Zipf's law fits the distribution of many human activities such as the distribution of human populations in towns and cities [Zipf, 1949]. In particular it has been found to fit the dis-

Table 2: QGAME Parameters for Classifying Natural Language Text

Objective:	Find a set of weights that nearest neighbour classifier can use to predict class of documents given its trigram vector.
Representation:	floating point value between 0 and 1
Fitness cases:	30 randomly chosen documents of each category
Fitness:	Correlation between predicted and actual classification on training set
Selection:	SUS with linear rescaling between 1 and 2, non-elitist, generational
Wrapper:	Weights used by nearest neighbour classifier
Pop Size:	20
Individual length:	Number non-zero trigrams in whole of training set
Initial pop:	Each weight randomly set between 0..1
Parameters:	60% Uniform crossover (each weight equally likely to come from either parent). Gaussian mutation N(0,1.0) (weights kept in 0..1)
Termination:	An individual scores 100% on training set or maximum number of generations 100

Table 3: Percentage correct classification. QGAME KNN Trigram Hybrid on Reuters 21578 collection. Means and (standard deviations) of 10 independent runs.

Topics	Neighbours	Chrome	Gen 0			End of Run		
		length	Training	Test	Secs	Training	Test	Secs
Coffee, trade	60	3200	100 (0)	99 (1.3)	2	100 (0)	99 (1.3)	2
+ crude	90	3600	99 (0.8)	94 (2.1)	4	100 (0.4)	93 (3.1)	41
+ money-fx	120	3800	98 (1.2)	92 (2.8)	6	100 (0)	91 (2.4)	89
+ sugar	150	3900	98 (1.1)	90 (2.9)	8	100 (0.4)	91 (1.9)	210
+ money-supply	180	4000	92 (2.0)	87 (1.6)	10	96 (2.1)	89 (1.9)	745
+ ship	210	4200	91 (2.1)	85 (2.1)	13	95 (1.5)	87 (2.5)	1034
+ interest	240	4200	88 (2.3)	81 (2.3)	17	93 (2.0)	82 (3.0)	1337

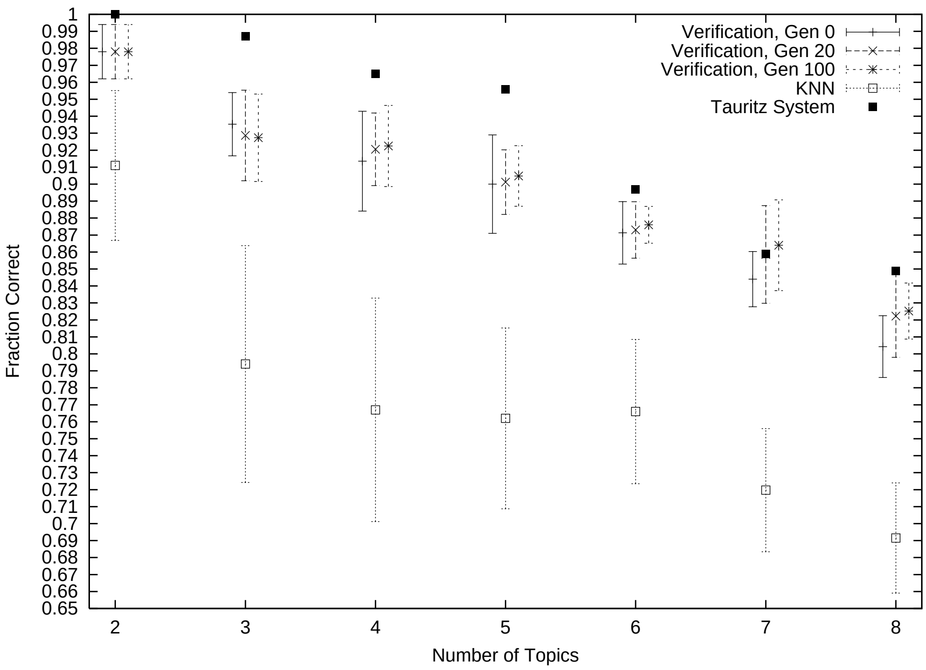

Figure 2: Performance of Trigram classifiers on Test set. Mean and standard deviations of 10 runs

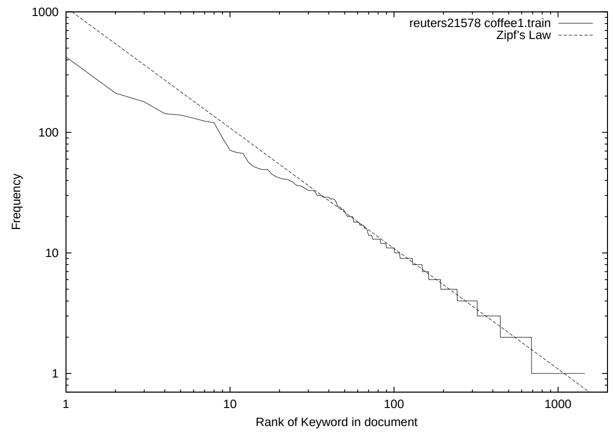

Figure 4: Distribution of words in the Reuters 21578 collection (Thirty randomly chosen documents about coffee)

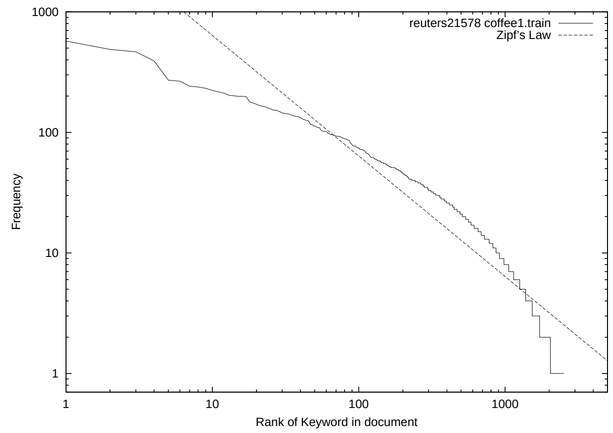

Figure 5: Distribution of trigrams in the Reuters 21578 collection (Same documents as in Figure 4)

tribution of words used in human generate texts in many languages. The next section shows it holds in the Reuters 21578 collection. It has been suggested that it would also describe the distribution of n grams. The following section (Section 4.2) shows that trigrams don't fit Zipf's law but there is regularity in the distribution which might be used.

4.1 ZIPF'S LAW AND KEYWORDS

Zipf showed in a variety of languages authors have a strong pattern in the way they use words. A few words are frequently used while many others are infrequently used. Suppose the number of times each word is used is counted and then the words are ordered so the most frequently used is given rank 1, the second rank 2 and so up to the least frequently used work rank n. Zipf showed frequency is related to rank harmonically, i.e. the frequency of a word is approximately proportional to the inverse of its rank. Therefore if they are plotted against each other on a log-log scale a straight line with gradient -1 results (excepting both rank and frequency must be integers resulting in discrete steps in the line). Figure 4 shows Zipf's law essentially applies.

4.2 ZIPF'S LAW AND TRIGRAMS

We had expected that the distribution of n grams would also obey Zipf's law. But as Figure 5 shows their is systematic disagreement between the actual distribution and Zipf's law. (Such disagreement may not be present with larger n values). The middle ranking trigrams are much more frequent than Zipf's law would suggest.

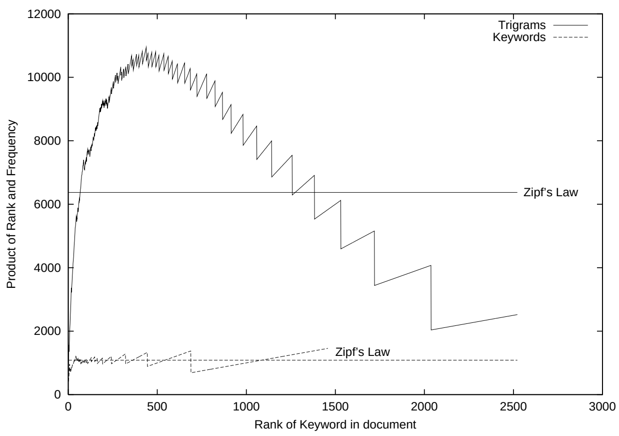

Figure 6: Distribution of keywords and trigrams in the Reuters 21578 collection (Same documents as in Figures 4 and 5)

Zipf's law predicts that frequency $\times$ rank will be a horizontal line. Figure 6 shows this is true (baring the discrete nature of frequency) for keywords but not for trigrams.

As Figures 5 and 6 show there is clearly some structure in the distribution of trigrams. We observed both the disagreement and also this structure in the Reuters 21578 collection and various unrelated LaTeX documents. In future it may be possible to use this structure as a "base line" from which to learn. I.e. the classifier would start not with the n gram distribution but its difference from the standard distribution. This might also be used to suppress noise.

5 WHY DO KNN AND TRIGRAMS WORK?

Analysis of a limited number of nearest neighbours results shows that distance between the document to be classified and the reference documents is surprisingly variable and documents from different classes overlap in their distances. (It is surprising how often the closest neighbour *is* of the required class). Also the distance measure is dominated by one trigram and only a few make any discernible contribution. It appears random adjustment of the importance (by using the weight vector) of a few trigrams is sufficient to boost the performance of the nearest neighbours algorithm. While more systematic variation by the genetic algorithm simply leads to better results of the training data but no gain in performance on unseen data. It has been suggested that using the closest three or four neighbours, rather than just the single closest neighbour, might further improve performance by giving some noise suppression. (The class is then the majority of the three of four votes of these k neighbours, $k = 3$ or 4).

Note that most trigrams make no real contribution to the distance. With care, it might be possible to still further increase the speed of these techniques by omitting low frequency trigrams.

6 STEMMING

Stemming techniques [Porter, 1980] reduce multiple endings of keywords to a single keyword. E.g. steming removes the `-e` and `-ing` endings of `come` and `coming` so `come` and `coming` are treated as the same keyword rather than separate two words. These can be fast but are not 100% accurate and different algorithms are needed for different languages thus nullifying one of the advantages of trigrams. However they might improve classification accuracy.

7 CONCLUSIONS

It is clear that trigrams combined with simple classifiers, such as (weighted or unweighted) nearest neighbours algorithms offer automatic fast multi-category classification of natural language text documents. However accuracy falls as the number of categories is increased.

Since there are typically thousands of weights it is not surprising that more sophisticated learning techniques such as GAs adjusting trigram weights are liable to over fit the training data. Results obtained from 20 random weight vectors are not significantly worse than Tauritz' approach but our technique is simpler and much faster.

It may be possible to reduce the dangers in over fitting by using knowledge of a "standard" distribution instead of using the normalised trigram frequency. Then only statistically significant differences between the measured distribution and the "standard" distribution would be used in the classifier. This should greatly reduce the number of variables. This should both reduce the chance of over fitting and speed the process.

Acknowledgements

I would like to thank Daniel Tauritz and Sander Bohte for enthusiastic advice and to thank Daniel for his code.

References

[Dekker, 1995]
Laura Dekker. Qgame manual. Technical report, University College, London, 1995. `http://www.cs.ucl.ac.uk/staff/L.Dekker/pubs/qgameman.ps`.

[Langdon, 1995] W. B. Langdon. Scheduling planned maintenance of the national grid. In Terence C. Fogarty, editor, *Evolutionary Computing*, number 993 in Lecture Notes in Computer Science, pages 132–153. Springer-Verlag, 1995.

[Langdon, 1997] W. B. Langdon. Scheduling planned maintenance of the south wales region of the national grid. In David Corne and Jonathan L. Shapiro, editors, *Evolutionary Computing*, volume 1305 of *Lecture Notes in Computer Science*, pages 181–197, University of Manchester, UK, 7-8 April 1997. Springer-Verlag.

[Langdon, 1998] William B. Langdon. *Data Structures and Genetic Programming: Genetic Programming + Data Structures = Automatic Programming!*, volume 1 of *Genetic Programming*. Kluwer, Boston, 24 April 1998.

[Langdon, 2000] W. B. Langdon. Natural language text classification with trigrams and evolutionary nearest neighbour classifiers. In *Late breaking papers of GECCO'2000: Proceedings of the Genetic and Evolutionary Computation Conference*, Las Vegas, 8-12 July 2000.

[Mitchell, 1996] Melanie Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1996.

[Porter, 1980] M. F. Porter. An algorithm for suffix stripping. *Program*, 14(3):130–137, July 1980.

[Tauritz and Sprinkhuizen-Kuyper, 1999a] D. R. Tauritz and I. G. Sprinkhuizen-Kuyper. Adaptive information filtering: Improvement of the matching technique and derivation of the evolutionary algorithm. Technical report, LIACS, April 1999.

[Tauritz and Sprinkhuizen-Kuyper, 1999b] Daniel R. Tauritz and Ida G. Sprinkhuizen-Kuyper. Adaptive information filtering algorithms. In David J. Hand, Joost N. Kok, and Michael R. Berthold, editors, *Advances in Intelligent Data Analysis, Third International Symposium, IDA-99*, volume 1642 of *LNCS*, pages 513–524, Amsterdam, The Netherlands, 9–11 August 1999.

[Zipf, 1949] George Kingsley Zipf. *Human Behavior and the Principle of Least Effort: An Introduction to Human Ecology*. Addison-Wesley Press Inc., Cambridge 42, MA, USA, 1949.