01 Jan 1996

Optimization of the Discriminatory Power of a Trigram Based Document Clustering Algorithm Using Evolutionary Computation

Daniel R. Tauritz
Missouri University of Science and Technology, tauritzd@mst.edu

Optimization of the discriminatory power of a trigram based document clustering algorithm using evolutionary computation

by
Daniel Tauritz

Last revised April 30th, 1996

Index

Paragraph 1 - Introduction

This paper was written to document the practical work done by the author for the course *Evolutionary Algorithms* at the Department of Computer Science, Leiden University, The Netherlands. Besides demonstrating the effectiveness of evolutionary algorithms for solving complex optimization problems, the paper should also provide an easy introduction to trigram based document[1] clustering.

The work described in this document was motivated by the rapidly increasing importance of document clustering as a tool for finding information among the ever-growing quantities of data we generate. The pertinence of document clustering is based on the notion that documents can be described by a set of characteristics which show with high probability a certain similarity when the documents are semantically similar. For example, if two documents both contain many medical terms, one would expect the probability to be high that these documents have a similar subject, namely, medicine. In this paper an algorithm which takes as input sets of characteristics and produces as output a set of clusters and a mapping of each input set to a cluster in such a way that the input sets mapped to the same cluster have a certain similarity is called a *clustering algorithm*.

It is often the case that the set of characteristics chosen to represent documents contains some elements which are more important than others in the clustering process. For example, if we were to represent a document by the set of its word frequencies, the element indicating the frequency of the word ‘the’ would most probably be quite insignificant, while the element indicating the frequency of the word ‘clustering’ should be very significant! The question then is, how do we determine the relative importance of each of the elements of the set of characteristics chosen to represent documents? Even if we only allow two values of importance, namely ‘important’ and ‘not-important’, which would be extremely crude, and limit our set of characteristics to just thirty elements, we would still have over a billion possibilities... In this paper it will be demonstrated that through the use of an evolutionary algorithm it is feasible to find an optimal or near optimal solution to this problem with a search space as large as 10^{3000}!

1. only textual documents are considered in this paper

Paragraph 2 - A trigram based document clustering algorithm

There are many clustering algorithms. They all differ with respect to complexity, performance, and clustering properties. It was deemed sufficient for the research described here to use a simple clustering algorithm to reduce the overall complexity. The (meta) algorithm is as follows:

generic document cluster

Step 1 - Initialization
- Start with no clusters

Step 2 - Apply new document
- Let D := f(next document)

Step 3 - Find the closest cluster
- Let C := g(closest cluster)

Step 4 - Check if C is too far from D
- If C is too far from D, or if there are no clusters yet, then create a new cluster with distance zero to D; goto step 2

Step 5 - Update the matched cluster
- Update C by moving it closer to D; goto step 2

To obtain an actual algorithm it is necessary to create an instantiation of *generic document cluster* by defining the terms ‘document’, ‘cluster’, ‘closest’, ‘too far’, ‘distance’, and ‘closer’ and the functions f and g. Let us start with the term ‘document’. In the rest of this paper a document will be considered to be a list of one or more lines, each line in its turn being composed of words separated by exactly one space (acting as separator symbol) and each word consisting of one or more lower-case letters of the Latin alphabet. We further restrict this simple class of textual documents by requiring the documents to be syntactical and semantically correct English non-fiction (such as the paper you are currently reading). The term ‘distance’ clearly indicates the necessity of defining a distance measure for the space of documents. As it is not clear how one would define the distance between two documents it is common to transform the space of documents into a space on which a distance measure may be more readily imposed. This transformation is performed by the function f. In order to be able to compare C with D it is necessary for C to be in the same space as D. This is accomplished by applying function g.

A practical class of transformations is for instance the aggregate transformation. It transforms objects in document space to objects in a vector space where each element of a vector can be expressed as a function on a set of characteristics of the objects in document space. It is practical because there are many well-known ways of defining distance measures between vectors. One of the most often used is the *euclidean metric* which is the method employed in the research described here. The next step is to decide what to aggregate. Counting the number of symbols in a document is clearly not very relevant for determining its contents. A more sophisticated method might be to count the number of occurrences of each distinct symbol. As there are 27 distinct symbols in our documents, namely the 26 letters of the Latin alphabet and the space, this method would yield vectors consisting of 27 elements. And if we normalized those elements by dividing each sum of occurrences by the total number of symbols in that document, the length of a document would have no influence on its representation. This method is called normalized 1-gram analysis.

The significance of the number of occurrences of a distinct symbol (e.g. the letter ‘a’) is, however, neligible. How about the significance of the number of occurrences of a distinct group of two symbols (e.g. ‘th’)? This would yield vectors consisting of 729 (27*27) elements. But according to [Scholtes93] (page 143) the discriminatory power (a measure of the quality of the discrimination between documents) of 2-gram analysis is normally insufficient whereas that of 3-gram analysis is normally acceptable. Better still is 4-gram analysis, which, however, requires vectors of over half a million elements making it currently not very practical for daily use, though certainly feasible for research. The research described in this paper used 3-gram analysis, further called trigram analysis, yielding vectors of 19683 (27*27*27) elements.

With ‘document’ having been defined and all aspects of distance being expressed in terms of euclidean distance this leaves us with the term ‘cluster’. Clearly it should be of the same type as the representation of the documents to be clustered. The vectors representing clusters shall further be called *prototype vectors*. The clusters are hyperglobes with the prototype vectors indicating the centroids and a parameter *c* indicating the radius. Using these clusters, the normalized trigram analysis, and the euclidean metric, we obtain the following instantiation of *generic document cluster*:

cluster euclidean

```
Step 1 - Initialisation
• Start with no cluster prototype vectors

Step 2 - Apply new input vector
• Let I = $\left( \boldsymbol{I_1}, \boldsymbol{I_2}, \ldots, \boldsymbol{I_n} \right)$ := [next normalized trigram input vector]

Step 3 - Find the closest cluster prototype vector (if any)
• Find the P = $\left( \boldsymbol{P_1}, \boldsymbol{P_2}, \ldots, \boldsymbol{P_n} \right)$ to minimize d(P,I) = $\sqrt{\sum_{x=1}^{n} \left( \boldsymbol{P_x} - \boldsymbol{I_x} \right)^2}$

Step 4 - Check if P is too far from I
• If d(P,I) > c, or if there are no cluster prototype vectors yet, then create a new cluster, with
  prototype vector equal to I; goto step 2

Step 5 - Update the matched prototype vector
• Let P := $(\boldsymbol{1} - \lambda) \cdot \boldsymbol{P} + \lambda \cdot \boldsymbol{I}$ ; goto step 2
```

This algorithm does have some unwelcome clustering properties, examples of which can be found in paragraph 3 of [Heins95] in which ART (Adaptive Resonance Theory) is shown to overcome these unwelcome properties.

Now let us look at some experimental results using *cluster euclidean* on two sets of documents. All the documents were selected from [Microsoft94], see appendix A for an exact enumeration. The first set consists of documents about chemicals and documents about medicine, the second of documents about fish and documents about musical instruments. The vectors used in the research described here were created by filtering the documents in such a manner that only the previously discussed 27 distinct symbols remained. This was done by converting upper-case letters to lower-case and all non-letters between two letters to a single space. Also, over a hundred common words such as 'the' and 'it' were discarded. The number of distinct trigrams in the first document set was 2897 and in the second document set 2268, thus reducing the size of the vectors from 19683 to 2897 and 2268 respectively.

A *solution* is defined as a set of instances of the parameters of a particular clustering algorithm. The *rating* of a solution is defined as the maximal rating minus a penalty for incorrect clustering. Maximal rating is defined as the square of the number of vectors to be clustered. Incorrect clustering is clustering differing from those shown in appendix A. The penalty for incorrect clustering is defined as the sum of the penalties for each separate vector. Each vector was penalized for:

- Incorrect neighbours - vectors assigned to the same cluster when they should not have been
- Missing neighbours - vectors which should have been assigned to the same cluster but were not

The focus of the research described here was to find solutions with the highest possible ratings with the lowest possible computational effort.

The distance between vectors of the first document set is in the range 0.036 to 0.060, between vectors of the second document set in the range of 0.047 to 0.075. Applying *cluster euclidean* to the two document sets with c chosen within the above mentioned distance ranges produces the following results:

Table 1: *cluster euclidean* on first document set

c	λ	rating	#clusters
0.040	0.1	60	8
0.040	0.9	72	7
0.045	0.1	70	6
0.045	0.9	80	6
0.050	0.1	56	3
0.050	0.9	56	3

Table 2: *cluster euclidean* on second document set

c	λ	rating	#clusters
0.050	0.1	72	7
0.050	0.9	72	7
0.055	0.1	64	6
0.055	0.9	64	6
0.060	0.1	56	3
0.060	0.9	56	3

The optimal values found using the optimization techniques which will be introduced in the next paragraph were: c = 0.047 & λ = 0.9 producing a rating of 86 (4 clusters) for the first document set and c = 0.060 & λ = 0.999 producing a rating of 86 (4 clusters) for the second document set. For both sets *cluster euclidean* seems to be incapable of finding solutions with the maximal rating of 100 (2 clusters).

Paragraph 3 - Optimization using evolutionary computation

As mentioned in the introduction, some elements of the set of characteristics of a document are more significant than others with respect to discriminating between different documents. In the present case this means some trigrams contribute more towards the discriminatory power of trigram analysis than others. This observation can be expressed in the form of weights associated with each trigram or with each difference between trigrams, the last being employed here. Step 3 of *cluster euclidean* (further called *weighted cluster euclidean*) then becomes:

Step 3 - Find the closest cluster prototype vector (if any)

- Find the P = (P_1, P_2, ..., P_n) to minimize d(P,I) = $\sqrt{\sum_{x=1}^{n} \left((P_x - I_x) \cdot W_x \right)^2}$

with W = (W_1, W_2, ..., W_n) being the weight vector.

To maximize the discriminatory power of *weighted cluster euclidean* all we have to do is to find the optimal values for the weight vector (and, of course, for c and λ), or, put another way, all we have to do is solve an optimization problem with 19685 free variables! The rest of this paper will be devoted to demonstrating how evolutionary computation can be used to find optimal or near-optimal values for the weight vector. For a good introduction to evolutionary computation see [Michalewicz94]; chapter 8 is especially relevant to this paper.

The concept behind evolutionary computation involves the maintainance of a population of solutions on which the process of evolution is performed until either an optimal (or near-enough optimal) solution has been found, or, the decision is made to give up. The process of evolution involves selecting one or more individuals from the population and creating new individuals by applying genetic operators such as mutation and crossover to the selected individuals. A new population is then formed by selecting a number of individuals from the old population and from the new individuals based on their fitness. The fitness of an individual is a measure of the quality of the solution it represents.

A description of the evolutionary algorithm employed follows. Two selection mechanisms, uniform random and exponential ranking, were applied. With uniform random selection each member of the population has an equal chance of being selected. Using exponential ranking individuals with a higher fitness have an exponentially higher chance of being selected. This causes an effect called selective pressure. Selective pressure results in the higher ranking solutions dominating the population with the hope that faster convergence can be obtained in this way. The drawback is that the chance of premature convergence in a local optimal solution is much larger. The genetic operators were gaussian mutation and uniform crossover. Gaussian mutation was applied by selecting one individual and adding some gaussian noise with mean zero[1] to each of its elements. Uniform crossover was applied by selecting two distinct individuals and selecting uniform randomly each element of the new individual from them. The fitness of an individual was defined as its rating. An updated population was formed after each creation of a new individual by adding the new individual to the old population and then eliminating the individual with the lowest fitness.
In the experiments various parameter values were tried: population sizes of 50 and 100, crossover rates of 0 and 0.2 and various selection methods. The crossover rate indicates the ratio of individuals created by crossover against individuals created by mutation. A crossover ratio of 0

1. with unit variance for weights, 0.1 x unit variance for λ, and 0.01 x unit variance for c

indicates solely mutations and a ratio of 1 indicates solely crossovers.

Paragraph 4 - Experimental results

What results are to be expected? First of all, will solutions with the maximal rating (fitness) of 100 be found? The answer is yes, for both document sets solutions with maximal ratings were obtained. Secondly, which combination of parameters of the evolutionary algorithm minimized the computational effort to find solutions with maximal ratings? Theory would suggest that it is wisest to strike a balance between exploration (genetic diversity) and exploitation (convergence). Larger populations contribute towards higher genetic diversity, selective pressure towards faster convergence. However, the higher the selective pressure, the greater the chance of premature convergence in local optima.
The stop condition for this experiment was finding the maximal rating, with a maximum of 20,000 evaluations. The results shown in the following tables are averages over 20 runs, except where a number between brackets indicates otherwise.

Table 3: First document set with uniform random selection

Population size	Crossover rate	Average # evaluations
50	0	497
50	0.2	469
100	0	426
100	0.2	620

Table 4: Second document set with uniform random selection

Population size	Crossover rate	Average # evaluations
50	0	823
50	0.2	1079
100	0	1197
100	0.2	1457

Table 5: First document set with exponential ranking

Population size	Cross over rate	Selective pressure	Average # evaluations
50	0	0.1	520
50	0	0.5	472 (70x)
50	0.2	0.1	617 (70x)
50	0.2	0.5	1331[1] (50x)
100	0	0.1	252
100	0	0.5	288
100	0.2	0.1	282
100	0.2	0.5	667

1. average includes one time that the maximum of 20000 evaluations was reached

Table 6: Second document set with exponential ranking

Population size	Cross over rate	Selective pressure	Average # evaluations
50	0	0.1	596 (50x)
50	0	0.5	294 (50x)
50	0.2	0.1	686 (50x)
50	0.2	0.5	2286 (50x)
100	0	0.1	549 (50x)
100	0	0.5	847 (50x)
100	0.2	0.1	593
100	0.2	0.5	871 (50x)

Bearing in mind that these figures are all approximations of the mean of distributions with very large variances, a few observations can be made. Larger populations only seem to help when using a selection mechanism causing selective pressure. This can be explained by assuming that the increase in genetic diversity produced by larger populations is needed to prevent the premature convergence facilitated by selective pressure. It is also clear that too high a selective pressure has a negative effect as it greatly increases the chances of premature convergence. The use of the genetic operator crossover seems to work counterproductively, though perhaps a different ratio could be beneficial. Comparing the two selection mechanisms it is clear that exponential ranking with a low selective pressure is superior to uniform random selection.

For those who are still uncertain about the effectiveness of evolutionary computation table 7 presents the results of random search averaged over 20 runs:

Table 7: Random search

Document set	Average # evaluations
1	2030
2	5184[1]

1. average includes 4 times that the maximum of 10000 evaluations was reached

Clearly evolutionary computation is superior to random search.

Now that the problem of finding suitable weights has been solved one might want to know if there are particular trigrams which contribute considerably more than average to the discriminatory power of the clustering algorithm. Preliminary results indicate not, but more research is needed to determine the answer to this question with any certainty.

Paragraph 5 - Conclusion

The central theme of the research described in this paper is the search for a document clustering algorithm with maximal discriminatory power and minimal computational effort. As basis for this research trigram analysis and a simple clustering algorithm were chosen. Two document sets were selected and the clustering algorithm was demonstrated to be unable to satisfactorily cluster these. An enhanced version of the clustering algorithm was then proposed. It, however, required the solving of a complex optimilization problem. Evolutionary computation was then introduced as a potential method for solving this problem. It was shown to solve the optimilization problem roughly ten times faster than randomly searching for the solution. The major obstacle in this research was the amount of computational time needed to obtain the results presented in this paper. Using an AMD486DX4-100 it would have taken about 200 hours (very rough estimate). Luckily many of the experiments could be performed on Silicon Graphic workstations. The major reason so much computer time was needed was because the experiments had to be repeated many times to obtain (at all) reliable approximations due to the great variances of the underlying distributions.

Does the success of this research imply that the problem of document clustering is (just about) solved? Not at all. There is one very important step which still has to be made before such a thought can be entertained, and that is demonstrating the generalisability of this research. An optimal set of weights for document set 1 is most probably not at all optimal for document set 2, and if it is, it is by sheer chance. To be of practical use a set of weights is required which is optimized for the environment in which it is going to be used. For instance, one could ask what the optimal set of weights is for clustering all documents included on [Encarta94]. Two questions spring immediately to mind. First of all, will the optimal set be pertinent, or in other words, will the optimal set allow clustering of sufficient quality to be useful? And secondly, assuming the optimal set is pertinent, how large a sample of the contents is sufficient to find the optimal set (or a set near enough)?

The focal point of any extension to this research lies clearly in the area of generalisability. One can hope that merely sampling enough document vectors will produce an optimal set of weights pertinent to a specific environment. However, this might fail, or, the size of the sample might exceed ones computational capabilities, thus posing a scalability problem. To overcome such problems one could improve on the research presented in this paper by:

- More advanced preprocessing (for example the use of a thesaurus or stemming)
- More powerful clustering algorithms
- More sophisticated evolutionary algorithms

References

Heins, L.G., Tauritz, D.R. (1995) “Adaptive Resonance Theory (ART): An Introduction”, Internal Report 95-35, Department of Computer Science, Leiden University, URL: “http://www.wi.leidenuniv.nl./art/paper.ps”

Michalewicz, Zbigniew (1994) “Genetic Algorithms + Data Structures = Evolution Programs”, 2nd extended edition, Springer-Verlag

Microsoft (1994) “Encarta’95”, CD-ROM

Scholtes, Johannes Cornelius (1993) “Neural Networks in Natural Language Processing and Information Retrieval”, PhD thesis, University of Amsterdam, The Netherlands

Appendix A - Document sets

Table 8: First document set

Cluster	Document
Chemistry	Science & Technology / Chemistry / Alcohol
Chemistry	Science & Technology / Chemistry / Alloy
Chemistry	Science & Technology / Chemistry / Arsenic
Chemistry	Science & Technology / Chemistry / Carbohydrate
Chemistry	Science & Technology / Chemistry / Carbon Dioxide
Medicine	Life Science / Medicine / Abortion
Medicine	Life Science / Medicine / Acquired Immune Deficiency Syndrome
Medicine	Life Science / Medicine / Alcoholism
Medicine	Life Science / Medicine / Allergy
Medicine	Life Science / Medicine / Anesthesia

Table 9: Second document set

Cluster	Document
Fish	Life Science / Fish / Barracuda
Fish	Life Science / Fish / Carp
Fish	Life Science / Fish / Catfish
Fish	Life Science / Fish / Cod
Fish	Life Science / Fish / Eel
Musical Instrument	Performing Arts / Musical Instruments / Accordion
Musical Instrument	Performing Arts / Musical Instruments / Bagpipe
Musical Instrument	Performing Arts / Musical Instruments / Banjo
Musical Instrument	Performing Arts / Musical Instruments / Carillon
Musical Instrument	Performing Arts / Musical Instruments / Clarinet