A Genetic Approach to Tuning Compact Trie Clustering

Richard Elling Moe, Snorre M. Davøen
Department of information science and media studies
University of Bergen

Abstract

The Compact Trie method for document clustering is sensitive to the kind of text it is applied to, but contains various parameters that may be tuned for adaptation to specific applications. We implement a genetic algorithm for optimizing these parameters and apply it to a corpus of texts to demonstrate the feasibility of using genetic algorithms for tuning.

1 Introduction

Suffix Tree Clustering is a method for content based document clustering introduced by Zamir and Etzioni [18] and further discussed in [4, 7]. Some key properties of this method is that word order matters, that clusters may overlap and that it works well on small excerpts (snippets) from the documents. These were attractive features when the Norwegian Newspaper Corpus[14] was to be analyzed for detecting reuse and overlap in the flow of news [13, 9]. However, when adopted, it became clear that the method can be sensitive to the kind of documents it is applied to. Several modifications to the original algorithm proved beneficial in context of the new corpus. For instance using alternatives to suffixes, such as n-grams, and identifying the parts of an article to make up its snippet. Some of these are discussed further in [12, 11]. In the process, a generalized method emerged, referred to as *Compact Trie Clustering*, having Suffix Tree Clustering as a special case.

In these investigations, including the original paper, the setup of the algorithm was tuned manually, relying on researchers' expertise, intuition and qualified guesses to optimize the performance. A mechanized process for tuning the algorithm would allow the method to be adapted to a further range of corpora, without the need for such tacit expert competencies. Furthermore, experimental approaches to tuning are prone to error when intricate dependencies between different parameters is not fully understood by the researchers, leaving potentially fruitful value-combinations unexplored. Mechanized tuning could avoid such pitfalls by operating systematically and exhaustively.

Within the field of Information Retrieval, Zakos et al [17] and Chuan et al [3] discuss similar kinds of tuning. Both apply genetic algorithms successfully, which suggests that a similar approach might be viable in our context.

The remainder of this section presents the aim of the paper. Section 2 describes the Suffix Tree Clustering method while section 3 identifies the parameters being subject to change. Section 4 presents a genetic algorithm for tuning the parameters. Evaluation and results follows in sections 5 and 6 before section 7 concludes.

This paper was presented at the NIK-2014 conference; see http://www.nik.no/.

In this paper we aim to demonstrate the feasibility of tuning of Compact Trie Clustering automatically by means of genetic algorithms. Tuning is the process of finding the best options for certain parameters that may affect performance. We use the word 'parameter' in a wide sense to mean an intrinsic aspect of the algorithm that can be changed. This may range from tweakable numbers, such as limits and thresholds, to core properties such as the mentioned use of suffixes.

The concept of feasibility naturally involves aspects of effectiveness but also of practicality. We broadly take this to mean that the tuning should be accomplished within certain boundaries of time and resources. Given the possible scope of applications and diversity of users we can not pinpoint these boundaries. Generally, we can not expect every kind of user to have access to extraordinarily powerful equipment. Genetic algorithms can be time-consuming but for the purpose of tuning, being a one-off event, the tolerance for running time is probably quite high.

2 Compact Trie Clustering (CTC)

Zamir and Etzioni [18] demonstrate that documents can be clustered by applying the Suffix Tree method to short excerpts from them, referred to as *snippets*, and claim that it can outperform a number of other well-known clustering techniques.

More recently, Eissen et al [4] present a more nuanced picture. They point out that the technique has some weaknesses but maintains that these have little impact when applied to shorter texts and therefore represent no great problem in our specific context.

The current investigation builds on the previous reports [13] on the initial charting of territory, [12] exploring the potential for improving the Suffix Tree Clustering in general terms and [11] tuning the algorithm for further adaption to our specific corpus.

The backbone of the method is the data structure known as a *compact trie* [16] which is a tree for storing sequences. Each arc is labelled with a sequence of tokens and a path from the root represents the concatenation of labels along the path. This simple structure effectively represents sequences and their subsequences as paths whereas the branching captures shared initial sequences. A *stored* sequence will have an end-of-sequence mark attached to its final node. Figure 1 shows the compact trie for the sequences *aa*, *ab*, *abab*, *abc*, *babb*, *bc*, *cbba*, and *cbbc*, with the S_i as end-of-sequence marks.

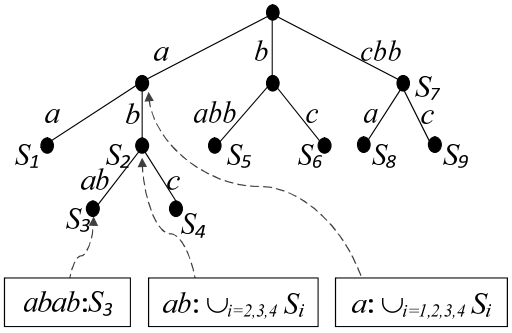

Figure 1: A compact trie and base-clusters

Given a set of *phrases*, initially snippets from a document, these are expanded into an extended set of phrases that will represent the document and each of them is inserted into a compact trie. That is, the arcs are labeled with sequences of *words*. Furthermore, the end-of-sequence mark now also registers the set of documents that the phrase occurs

in. The *Suffix* Tree employed by Zamir and Etzioni is the case where each phrase in the snippet is expanded into all its suffixes.

The compact trie forms the basis for constructing clusters of documents. Each node in the trie corresponds to a *base-cluster*. A base-cluster $\sigma : S$ is basically the set S of documents associated with the subtree rooted in the node. The *label* σ is composed of the labels along the path from the root to the node in question. Figure 1 illustrates three examples of base-clusters.

The base-clusters will be further processed to form the final clusters. That is, they are merged on grounds of being similar. Specifically, two base-clusters $\sigma : S$ and $\sigma' : S'$ are similar if and only if $\frac{|S \cap S'|}{|S|} > \omega_{sim}$ and $\frac{|S \cap S'|}{|S'|} > \omega_{sim}$, where ω_{sim}= 0.5.

Now consider the *similarity-graph* where the base-clusters are nodes and there is an edge between nodes if and only if they are similar. The final clusters then correspond to the connected components of the similarity-graph. That is, a cluster is the union of the document-sets found in the base-clusters of a connected component. Originally the cluster is not given a designated label of its own but we add one by collecting the words from the base-cluster labels and sort them by their frequencies therein.

Clearly, the construction of final clusters requires every base-cluster to be checked for similarity with every other base-cluster. This is a bottleneck in the process but Zamir and Etzioni circumvent the problem by restricting the merging to just a selection of the base-clusters. For this purpose they introduce a *score* and form the final clusters from from only the 500 highest scoring base-clusters. We refer to this limit as ω_{top} , i.e. in [18] we have ω_{top} =500.

The score of a base-cluster $\sigma : B$ is defined to be the number $|B| \times f(\sigma)$ where the function f returns the *effective length* of σ. The effective length of a phrase is the number of words it contains that are neither too frequent, too rare nor appear in a given *stop-list* of irrelevant words. Specifically, 'too frequent' means appearing in more than a percentage ω_{max} of the (total collection of) documents whereas a word is too rare if it appears in less than ω_{min} documents. Furthermore, f penalizes phrases shorter than ω_{scrMin}, is linear for phrases with lengths between ω_{scrMin} and ω_{scrMax} and is constant beyond that. In [18, 7] these thresholds are set to ω_{min} =3, ω_{max} =0.4, ω_{scrMin}=2 and ω_{scrMax}=7.

Variations on the scheme

A number of modifications to the original algorithm has proved beneficial in our context [13, 12, 11]. The more substantial changes include the use of n-grams in stead of suffixes. Except from some uninteresting special cases, the number of words contained in the n-grams of a phrase is strictly fewer than the number of words in the corresponding suffixes. With fewer words to process the algorithm works faster but there is a the concern that the information held by the data then becomes impoverished. Therefore, n was originally set to maximize the number of words inserted into the trie. This is achieved by expanding a phrase of length k into its $\lceil k/2 \rceil$-grams.

Alternative measures of similarity have also been introduced. The original similarity-measure takes only the overlap of sources into account. Making use of other characteristics such as labels and word frequencies leads to improvement.

Finally, we believe the original scoring is somewhat arbitrary and sensitive to the kind of text it is applied to [13]. The improvements reported in [12, 11] are partly results of experimentation with different scoring-functions.

3 Parameters for Tuning

We are faced with a wide range of options for setting up the clustering process. Here, we confine ourselves to variations considered in the investigations reported in [13, 12, 11] and [18]. Roughly speaking, they vary along five dimensions:

1. *Snippet-extraction:* There are many conceivable selections of text that could make up the snippets, ranging from the entire document to some specific, smaller, parts of it.

2. *Snippet-expansion:* Given a snippet, how should it be expanded into the phrases that goes into the trie? Suffixes is obviously an option while the use of n-grams offer various alternatives. There is also a question of *granularity*, i.e. should the snippet be expanded as a whole or is it composed of multiple *units*, say the sentences, to be expanded one by one?

3. *Limits and thresholds:* There is a number of parameters built into the algorithm, including the frequency-thresholds ω_{min} and ω_{max} and the limit ω_{top} on the number of base-clusters to be fed into the merging phase.

4. *Base-cluster selection:* The choice of base-clusters that proceed to the merging phase can have great impact on the end result. Scoring is the main mechanism for this purpose. Possibly with additional filtering based on cluster characteristics

5. *Similarity measure:* Which base-clusters should be merged?

We have identified a total of 19 parameters. Throughout, these are either written in capitalized TYPEWRITER-font or as subscripted ωs. For lack of space we can not present them all in full detail. Some parameters are components of complex parameters and these are presented only when they apply.

Each parameter received an ample range of relevant values. These were determined by reference to literature, specifically Zamir and Etzioni's original paper [18] and the investigations concerning our own application [13, 12, 11], occasionally with some additional experimentation. For example, tests indicated that the full range of 25378 base-clusters for ω_{top} could be reduced significantly which undoubtedly saved considerable running-time.

In our context, a snippet will typically consist of some sentences and each of them becomes a unit for expansion. The question remains, what do we pick from each article to form its snippet? Moe and Elgesem [13] demonstrate that smaller snippets can do better than the whole text. Clearly, the choice of snippets matters and we consider this to be a tunable parameter ranging over a sample of ways to extract the snippets. We refer to it as EXT . For news articles, candidate snippets can be readily available in the form of front-page matter such as headlines, captions and ingresses. The range of EXT thus comprise the combinations of such article-parts. Specifically: headline, ingress and byline taken from the front-page and/or the article itself as well as some initial portion of the article-text specified by an additional parameter ω_{txt} ranging from 0 to 1.

Alternatives to suffixes as way of snippet expansion could be the use of n-grams. In [13, 12, 11] this is shown to be an improvement, for a specific choice of n. However, other uses of n-grams are easily conceivable. Hence we let EXP be a parameter to be tuned over a range of ways to expand a snippet of length k, including its suffixes and various uses of n-grams:

- n-grams for $0 \leq n \leq 10$
- $\lceil k/2 \rceil$-grams, as in [13, 12, 11] .
- n-grams for a range $k\omega_{rMin} \leq n \leq k\omega_{rMax}$
  where $0.0 \leq \omega_{rMin} \leq 0.9$ and $0.1 \leq \omega_{rMax} \leq 1.0$.

Our earlier investigations [13, 12, 11] has demonstrated the potential of tweaking $\omega_{\min}$, $\omega_{\max}$ and ω_{top} and hence we treat these as tunable parameters. with ranges including Zamir and Etzioni's original values and those used in [13, 12, 11]. Specifically, $0 \leq \omega_{\min} \leq 150$, $0.05 \leq \omega_{\max} \leq 1.00$ and $100 \leq \omega_{\text{top}} \leq 10000$

Behind the results reported in [13, 12, 11] lies experimentation with a number of different measures for similarity and scoring. As different measures are reported to give different performance we let these too constitute tunable parameters, referred to as `SIM` and `SCR` respectively.

For `SCR` we consider the two scoring-functions $\uparrow$ and $\downarrow$ found in [12] ($\downarrow$ being Zamir and Etzioni's original score.) with $0 \leq \omega_{scrMin} \leq 20$ and $3 \leq \omega_{scrMax} \leq 25$ whereas `SIM` ranges over 4 different similarity-measures, including those described in [18, 12, 11].

In the following we assume that b and b' are base-clusters $\sigma : S$ and $\sigma' : S'$ respectively. We write $\hat{s}$ to denote the set of words occurring in a sequence s, whether it be a label or an entire document.

First, we let sim_1 be the Zamir and Etzionis original similarity-relation described in section 2. Further we let sim_2 be the similarity used in [11], defined by

$$sim_2(b,b') \text{ iff } sim_1(b,b') \text{ and } |\hat{\sigma} \cap \hat{\sigma}'| \geq \omega_{\cap} \text{ and } \frac{\sum_{w \in \hat{\sigma} \cup \hat{\sigma}'} cf(w)}{|\hat{\sigma} \cup \hat{\sigma}'|} \leq \omega_{\text{freq}}$$

where $0 \leq \omega_{\cap} \leq 50$, $5 \leq \omega_{\text{freq}} \leq 500$ and $cf(w)$ denotes the *corpus frequency* of the word w, i.e. the total number of times w occurs in our documents.

In [12], similarity is given in terms of the *label profile* for a base-cluster $\sigma : S$ which is the function $\vec{v}$ defined by:

$$\vec{v}(w) = \begin{cases} \Theta(w, S^+, C) & \text{if } w \in \hat{\sigma} \\ 0 & \text{otherwise} \end{cases}$$

where Θ is the term frequency - inverse document frequency which reflects the weight of a word w in a document relative to a corpus of documents. In our case, the document in question is a concatenation S^+, of the documents contained in the base-cluster whereas the corpus is our entire collection C of articles. Specifically, $\Theta(w, S^+, C)$ equals

$$tf(w, S^+) * \log \frac{|C|}{1 + |\{d \mid d \in C \text{ and } w \in \hat{d}\}|}$$

where the term frequency $tf(w, S^+)$ denotes how many times w occurs in S^+.

Two base-clusters are now considered similar if, and only if, they satisfy the original similarity-measure in conjunction with their label profiles having a *cosine-similarity* above a threshold ω_{cos}. That is, we define

$$sim_3(b,b') \text{ iff } sim_1(b,b') \text{ and } \frac{\Sigma_w(\vec{v}(w) * \vec{v}'(w))}{\sqrt{\Sigma_w \vec{v}(w)^2} * \sqrt{\Sigma_w \vec{v}'(w)^2}} \geq \omega_{cos}, \text{ where } 0 \leq \omega_{cos} \leq 1$$

Finally we try a variation of sim_1 often referred to as the Jaccard measure:

$$sim_4(b,b') \text{ iff } \frac{|S \cap S'|}{|S \cup S'|} > \omega_{Jac}, \text{ where } 0 \leq \omega_{Jac} \leq 1$$

Further filtering of base-clusters can be applied: [13] observes that base-cluster with a singleton label, i.e. having a label consisting of a single word, are often large and inaccurate so that the net effect of dropping them would be positive. This corresponds to a boolean parameter `SGL` for which *true* signifies that base-clusters with singleton labels are retained, and $false$ that they are discarded. There is a similar parameter `SGD` specifying whether or not base-clusters containing only a single document are filtered out.

4 A GA for tuning CTC

A genetic algorithm (GA) [6, 8, 10] maintains a population of individuals, each represented by its *chromosome*. A chromosome is a sequence of genes, each encoding a parameter whose value/state may influence the fitness of the individual. Based on fitness, a set of individuals can be selected for reproduction to yield a new generation in the population. Simultaneously, some individuals may vanish from the population. Reproduction is a pairwise combination of individuals producing two offspring whose chromosome is a crossover of those of the parents. The reproductive propagation of gene-values can be overridden by *mutation*, i.e. random non-hereditary changes in genes.

Starting from a random initial population, the GA mimics an evolutionary process, successively producing new generations until a stop-criterion has been met, ideally converging on a population whose chromosomes have reached optimal fitness.

When applying a GA for optimizing a process, each gene represents one of its parameters. That is, each individual chromosome corresponds to a possible solution for the optimization. Fitness is a measure of the performance being optimized, typically expressed in terms of the process' effectiveness and/or efficiency.

Genes

The set of parameters described above make up the gene-pool for CTC-tuning. That is, a chromosome is a sequence of values for these genes. Each chromosome then represents a specific *configuration* of the CTC-algorithm. Hence, the ultimate goal of optimizing the algorithm translates into finding highly fit individuals.

Reproduction

Evolution is driven by the perpetual cycle of reproduction. In each round:

- The $k\%$ fittest individuals are kept for reproductive duties. k is a given number referred to as the *keep size*. The remaining individuals are considered too unfit and are evicted from the population.
- The sufficient number of individuals are selected to become parents for a new generation so that the population-size is re-established. The selection is made using the Roulette Wheel algorithm, by which any individual may be selected but highly fit ones are more likely to be chosen [8]. The selected individuals are paired up randomly in couples.

- Each of these couples mate to produce two offspring whose genes are crossover combinations of the parents' genes. Given two parent chromosomes c_1 and c_2, the crossovers are constructed by choosing a random *crossover-point* p. That is, $c_1 = st$ and $c_2 = uv$ where the subsequences s and u are both of length p. Now, the crossover chromosomes are the recombined sequences sv and ut.

- Mutation is inflicted on m randomly chosen genes from a set P of individuals, either the entire population or just the new generation of offspring. $m = \textit{mutation rate} \times |P| \times |\textit{Genepool}|$. The mutated gene receives a new value depending on its type. Boolean genes are negated. A number-valued gene is adjusted randomly, up or down, by some value from a limited interval within its range. Otherwise, the new value is picked randomly from the range.

In principle, the evolutionary process could be set to aim for some sort of global convergence. For instance when all reproduction literally re-produces chromosomes or when the process reaches a population identical to an earlier population. However, this could easily be too time-consuming. Instead, we monitor evolution by keeping track of the fittest individuals over time. When the top candidate is unchallenged for an extensive period, the process terminates on the assumption that this individual is highly fit. Specifically, our GA stops when the highest fitness remains unchanged for 10 consecutive rounds of reproduction.

Fitness

The fitness of an individual relates directly to the corresponding configuration's performance, i.e. the quality of clusters the CTC algorithm delivers when configured accordingly. Having defined a 'correct' clustering, referred to as the *ground truth*, the clusters produced by the algorithm can be measured up against it and the quality quantified in terms of precision and recall.

Precision/Recall

The notion of *relevance* is fundamental to precision/recall [2]. For a clustering algorithm, relevance pertains to individual clusters delivered by the algorithm. A cluster being relevant basically means that the cluster appears in the ground truth clustering. Precision and recall then measures to what degree the algorithm succeeds in producing relevant clusters.

Assuming that C is the set of clusters generated by the CTC algorithm and R the set of relevant clusters, precision would measure the proportion $\frac{|C \cap R|}{|C|}$ of relevant clusters among the clusters generated by the algorithm. Recall measures the extent to which the algorithm will recreate the set of relevant clusters, i.e. $\frac{|C \cap R|}{|R|}$.

The two measures are often combined into one. Given precision p and recall r, the (balanced) *F-measure* is their harmonic mean $2\frac{pr}{p+r}$.

The corpus

Since 2006 the Norwegian Newspaper Corpus [14] has downloaded the front pages of the 8 largest online newspapers and stored them in HTML format. From this, a sample corpus consisting of the daily 10 top stories from December 7 to 18, 2009 [9], had been extracted and prepared for experimentation. A total of 960 articles had been manually coded based on categories that are used by media scholars to classify news, cf [1] and [5]. Each article

received a tag, consisting of five categories, characterizing the content of the article. For example

$$International - Economy - FinanceCrisis - Debt - Dubai$$

The data had been further pre-processed by reducing words to their lemmas and keeping only certain kinds of words: nouns, verbs, adjectives and adverbs. This was achieved by marking up the text with syntactic information using the Oslo-Bergen tagger [15] and subsequently filter the document to leave only the desired words in the desired form.

Relevance

The manually tagged portion of our corpus can serve as ground truth for evaluation in terms of precision and recall. Since the tags represent a human judgement as to what the document is about we think it is fair to assume that a high degree of overlap in tags will indicate overlap in content. Therefore, we use tags to define relevance.

A *ground truth cluster* consists of all documents having identical tags, and only those documents. Thus, ground truth clusters are identifiable by tags.

Here, the basic idea of relevance is that a good cluster contains the documents sharing a tag. That is, a cluster is considered relevant if it matches some ground truth cluster. The question is, should we require a *perfect* match? Consider the cluster containing three articles, two of which constitutes a ground truth cluster

$$International - Politics - Climate - Obama - Copenhagen$$

and one with the tag

$$International - Politics - Climate - Draft - Copenhagen$$

These articles are all about the 2009 Copenhagen Climate Change Conference, and the cluster would appear to be good. However, there can be no matching ground truth cluster because of the discrepancy of one word in the tags. Is it reasonable to deem this cluster irrelevant? This is largely a matter of the intended use and human opinion so the question has no definite answer. However, by incorporating a degree of perfection we get the flexibility that might allow for the cluster to be considered relevant.

A cluster C *matches ground truth with discrepancy* $5-d$ if and only if d is the maximal number for which there is a ground-truth cluster G such that $G \subseteq C$ and d is the number of words common to all tags in $C \cup G$.

Intuitively, discrepancy 0 is a perfect match while 5 means that there is no category that appears in all tags. Now, relevance can be defined as matching ground truth within some appropriate level of discrepancy. However, just what is meant by 'appropriate' depends on the application so it is hard for us to pinpoint a level which distinguishes relevant from irrelevant. For our analysis we consider only the extremes, regarding discrepancy 0 as ultimate relevance while a cluster with discrepancy 5 is held to be irrelevant (but see the remarks below).

Finally, fitness can be defined as the F-measure with a skew for discriminating against disproportionate clustering. If the ratio *number of clusters produced / number of ground truth clusters* is between 0.8 and 3 then 0.1 is added to the F-measure, otherwise 0.1 is subtracted to obtain the fitness.

5 Evaluation

To evaluate the feasibility of automatic tuning of CTC by GAs we mainly address the issue of effectiveness but also some considerations concerning practicality.

Effectiveness relates to the algorithm's ability to tune for the better, i.e. that the process constitutes a pursuit for higher performance. Preferably, it should also be capable of matching the performance obtained by manual tuning, thus eliminating the need for expert intuition and tacit competencies. We evaluate this by comparing configurations found by GAs to

- The average F-measure of 100 random configurations. This checks if the GA will tune for better performance.
- The original, manually tuned, configuration of Zamir and Etzioni and one of the manual configurations described in [13].

Practicality is mainly a question of time and machine-resources. In the bigger picture, the process as a whole can involve many time-consuming activities, such as preparing a ground truth. For this discussion the running-time of the GA is the natural focus. In principle, GAs can be distributed to run in parallel on several computers, which can speed up program-execution. In terms of machinery, practicality basically means being able to use readily available computers. Obviously, this depends on the circumstance but we can reasonably imagine users who possess only a limited number of non-homogenous run-of-the-mill computers.

Some remarks

Our previous studies [13, 12, 11] focus mainly on precision but the running-time of configurations also receive some attention. One reason for this is that the parameter ω_{top} is vital to recall, i.e. higher values will improve recall. However, increasing ω_{top} also floods a computational bottleneck in the CTC-algorithm so boosting recall comes at a cost in terms of efficiency. In this study there is no particular call for precision to carry more weight than recall, or vice versa, so we remain neutral by defining fitness in terms of the balanced F-measure. The running-time of a CTC-configuration could be taken as an aspect of its fitness but fair timing can be difficult to achieve when the GA-execution is distributed over a non-homogenous array of computers. Our experiments include a distributed run and therefore we do not incorporate time-efficiency into the fitness.

Relevance is a relative concept which could vary greatly between different applications and among different users. This is briefly discussed in [13]. They point out that relevance as matching ground truth may be so strict that the values for precision/recall drop to unfairly low levels. They suggest the alternative concept of *tag accuracy* which provides a considerably more liberal notion of relevance. However, tag accuracy has quirks of it own and since our purpose is not to provide an absolute measure of quality, but simply to register improvement, we are not concerned about the scale of our F-measures.

Finally, it must be said that we are a bit cavalier about regarding clusters of discrepancy 5 as plainly irrelevant. That is not necessarily the case. Moe and Elgesem [13] point out that an apparently good cluster, consisting predominantly of articles on a shared topic, could be deemed irrelevant if contaminated by just a single 'junk' article of high discrepancy. Technically, even relevant articles can appear to be junk because articles describing the same event from different perspectives could receive completely different tags. Whether or not contamination by a small proportion of junk should spoil

the relevance of a cluster is a matter of the intended use. It must be for the user to decide if the concept of relevance should be relaxed to allow some junk. We find it unnecessary for our purposes.

6 Results

Our GA has been tested on ordinary off-the-shelf machinery. Two experiments have been conducted in which the GA was set up in different ways. Not surprisingly, they confirm that characteristics such as population size, keep size and mutation rate can influence both effectiveness and practicality.

Effectiveness

In the following '*Z&E*' refers to Zamir and Etzioni's original configuration with a slight adaptation to the kind of text in our corpus. *Z&E* sets the parameters as follows: `EXT`: Everything except the article main text. ω_{txt}: 0, `EXP`: Suffixes, $\omega_{\min}$: 3, $\omega_{\max}$: 0.4, `SCR`: $\downarrow$, ω_{scrMin}: 2, ω_{scrMax}: 7, ω_{top} : 500, `SGL`: False, `SGD`: False, `SIM`: sim_1, ω_{sim}: 0.5

Moe and Elgesem [13] explore some alternative configurations and we choose to consider the one with the best balance between precision and recall, but modify it slightly by replacing sim_1 with sim_2 which actually improves the original performance. Specifically, the configuration '*M&E*' sets the parameters as follows: `EXT`: article headline, introduction and byline ω_{txt}: 0, `EXP`: $\lceil k/2 \rceil$-grams, $\omega_{\min}$: 6, $\omega_{\max}$: 0.5, `SCR`: $\uparrow$, ω_{scrMin}: 2, ω_{scrMax}: 7, ω_{top} : 5000, `SGL`: False, `SGD`: True, `SIM`: sim_2, ω_{sim}: 0.5, $\omega_{\cap}$: 1, ω_{freq} : 5

By '$Random^{100}$' we refer to the average performance of a set of 100 random configurations. This will, alongside the configurations *Z&E* and *M&E* serve as the benchmarks for the performance of configurations produced by our GAs. We test two setups of the GA described above:

GA^{200}: population size = 200, keep size = 80, mutation rate = 0.01 on the entire population. This produced the configuration: `EXT`: Frontpage headline, ingress and byline ω_{txt}: 0, `EXP`: 7-grams, $\omega_{\min}$: 33, $\omega_{\max}$: 0.6, `SCR`: $\downarrow$, ω_{scrMin}: 7, ω_{scrMax}: 8, ω_{top} : 9924, `SGL`: False, `SGD`: True, `SIM`: sim_2, ω_{sim}: 0.86, $\omega_{\cap}$: 2, ω_{freq} : 0

GA^{5000}: population size = 5000, keep size = 50, mutation rate = 0.02 on each new generation of offspring. This produced the configuration: `EXT`: ingress, article headline and introduction ω_{txt}: 0.32, `EXP`: 4-grams, $\omega_{\min}$: 141, $\omega_{\max}$: 0.47, `SCR`: $\uparrow$, ω_{scrMin}: 11, ω_{scrMax}: 17, ω_{top} : 7937, `SGL`: False, `SGD`: True, `SIM`: sim_1, ω_{sim}: 0.99,

Table 1 shows the F-measures obtained by the configurations described above. We have made no attempt to determine the appropriate maximum level of discrepancy for a cluster be deemed relevant. Therefore we present the F-measure for each level separately, in stead of their accumulation up through the levels. This also allows us to interpret the results by considering the opposite ends of the relevance-spectrum, discrepancies 0 and 5.

If we were to rank these configurations GA^{200} is a clear winner, having the highest score for discrepancy 0 and the lowest for 5. The scores are quite similar for *M&E* and GA^{5000}. While the latter does somewhat better on discrepancy 0, it loses to *M&E* on discrepancy 5. We can hardly claim that one is better than the other, but they both rank above *Z&E* and $Random^{100}$. Remarkably, the manually tuned configuration *Z&E* is

Discrepancy	*Z&E*	*M&E*	$Random^{100}$	GA^{200}	GA^{5000}
0	0.138	0.624	0.188	0.735	0.638
1	0.014	0.017	0.015	0.007	0.010
2	0.010	0.011	0.016	0.000	0.004
3	0.028	0.011	0.038	0.000	0.002
4	0.028	0.021	0.087	0.003	0.000
5	0.682	0.313	0.538	0.241	0.327

Table 1: F-measures

outperformed by $Random^{100}$. We believe this is partly due to CTC being highly sensitive to the kind of text it is applied to, which motivated the need for corpus-specific tuning in the first place. In addition, the *Z&E* configuration trades recall for speed. Zamir and Etzioni kept the ω_{top} parameter low in order to reduce the time spent on base-cluster merging, but this causes severe harm to recall and thereby to our F-measures. $Random^{100}$ on the other hand is likely to represent a middle-of-the-range value for ω_{top} and thus be balanced for a better F-measure.

Compared to the average random configuration, our GA clearly tunes for the better. In addition, it can perform favorably compared to manual tuning. We think it is reasonable to conclude that these results demonstrate the effectiveness of tuning by GA for our kind of corpus.

Practicality

Population size is a major difference between GA^{200} and GA^{5000}. One would naturally expect the larger population to require more processing-time so while GA^{200} ran on a single computer, the execution of GA^{5000} was distributed over 10 clients residing on 4 computers. Even so, the time-performances are on different scales. While GA^{200} took hours to complete, GA^{5000} ran into days. Since tolerance for running-time depends heavily on the situation, we can only presume that users in circumstances similar to ours might find the performance of GA^{200} acceptable. If so, since GA^{200} were run on a single standard computer it could be considered practical. GA^{5000} required considerably more resources both in terms of time and machinery and should be deemed significantly less practical than GA^{200}, if not impractical.

7 Conclusion

We find that the GA^{200}-configuration demonstrates the feasibility of tuning CTC by means of GAs, at least for our kind of corpus. Beyond that we can make no claim of external validity. However, there is promise in the fact that a very simple, not to say naive, GA did the job for our corpus. Given the rich body of sophisticated techniques that has developed in the field of GAs we believe that a GA approach can be viable for many kinds of corpora.

References

[1] S. Allern. Newsvalue: On marketing and journalism in ten norwegian newspapers (in Norwegian). IJ Forlaget (publisher) (2001)

[2] R. Baeza-Yates, B. Ribeiro-Neto. Modern information retrieval: the concepts and technology behind search. Addison Wesley (2011)

[3] Chuan, L., Shao-ping, M., Min, Z. Parameter optimization for information retrieval with genetic algorithm. In proceedings of the 2003 ieee international conference on Systems, man and cybernetics (2003)

[4] S. M. zu Eissen, B. Stein, M. Potthast. The Suffix Tree Document Model Revisited. In: Tochtermann, Maurer (Eds.): Proceedings of the I-KNOW05, Graz 5th International Conference on Knowledge Management Journal of Universal Computer Science, pp. 596-603 (2005)

[5] D. Elgesem, H. Moe, H. Sjøvaag, E. Stavelin. The national public service broadcaster's (NRK) news on the internet in 2009 (in Norwegian). Report to the Norwegian Media Authority, Department of information science and media studies, University of Bergen (2010)

[6] D. E. Goldberg. Genetic Algorithms in Search, Optimization and Machine Learning. Addison Wesley (1989)

[7] J. A. Gulla, H. O.Borch, J. E. Ingvaldsen. Contextualized Clustering in Exploratory Web Search. In: H. A. do Prado; E. Ferneda. Emerging Technologies of Text Mining: Techniques and Applications, pp. 184-207. IGI Global (2007)

[8] R. L. Haupt, S. E. Haupt. Practical Genetic Algorithms. John Wiley & Sons (2004)

[9] G. Losnegaard. Automatic extraction of news text from online newspapers. Project report, Department of information science and media studies, University of Bergen (2012)

[10] M. Negnevitsky. Artificial intelligence: a guide to intelligent systems. Addison Wesley (2002)

[11] R. Moe, Clustering in a News Corpus. In: P. Sojka et al. (Eds.): TSD 2014, LNAI 8655, pp. 301–307. Springer International Publishing Switzerland (2014)

[12] R. Moe, Improvements to Suffix Tree Clustering. In M. de Rijke et al. (Eds.): Advances in Information Retrieval, prodeedings of ECIR 2014. LNCS 8416, pp. 662-667 (2014)

[13] R. Moe, D. Elgesem. Compact trie clustering for overlap detection in news. In: Proceedings of the Norwegian Informatics Conference (NIK'13) (2013)

[14] Norwegian Newspaper Corpus, http://avis.uib.no/om-aviskorpuset/english

[15] Oslo-Bergen Tagger, http://tekstlab.uio.no/obt-ny/english/index.html

[16] B. Smyth. Computing Patterns in Strings. Addison Wesley (2003)

[17] Zakos, J., Ping, Z., Verma, B. (2005). Optimization of parameters for effective Web information retrieval using an evolutionary algorithm. In Proceedings of the 2005 ieee international joint conference on Neural networks (IJCNN´05) (2005)

[18] O. Zamir, O. Etzioni. Web Document Clustering: A Feasibility Demonstration. In: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval , pp. 46-54. ACM New York (1998)