Definition Characterisation through Genetic Algorithms

Claudia Borg
Dept. of Artificial Intelligence
University of Malta
claudia.borg@um.edu.mt

Mike Rosner
Dept. of Artificial Intelligence
University of Malta
mike.rosner@um.edu.mt

Gordon J. Pace
Dept. of Computer Science
University of Malta
gordon.pace@um.edu.mt

Abstract

The identification of definitions from natural language texts is useful in learning environments, for glossary creation and question answering systems. It is a tedious task to extract such definitions manually, and several techniques have been proposed for automatic definition identification in these domains, including rule-based and statistical methods. These techniques usually rely on linguistic expertise to identify grammatical and word patterns which characterize definitions. In this paper, we look at the use of machine learning techniques, in particular genetic algorithms, to enable the automatic extraction of definitions. Genetic algorithms are used to determine the relative importance of a set of linguistic features which can be present or absent in definitional sentences as a set of numerical weights. These weights provide an importance measure to the set of features. In this work we report on the results of various experiments carried out and evaluate them on an eLearning corpus. We also propose a way forward for discovering such features automatically through genetic programming and suggest how these two techniques can be used together for definition extraction.

Index Terms

Definition extraction, Genetic Algorithms, Natural Language Processing

1. Introduction

Definitions give meaning to a particular word or phrase by explaining its meaning through synonyms or by giving the context in which it is normally used. Definitions aim to assist a person conceptualise the meaning of a word and identify in what ways it can be used in language. Within an eLearning context, definitions are important as they allow a student to understand, formalise and quickly refer to new concepts of the domain. The extraction of definitions can be useful in the domains of question answering and glossary creation for the building of dictionaries. The manual identification of definitions is a tedious and laborious task and it is thus desirable to automate the task.

In this paper we will be looking at the automatic identification of candidate definitional sentences in a non-technical corpus. Rule-based approaches for definition extraction use linguistic features such as part-of-speech sequences to identify definitional sentences from non-definitional ones. However, such features do not always occur in definitions, and thus we need a combination of features. We want to fine-tune rule-based approaches by assigning relative importance to features expressed as a set of weights. Our approach selects particular features present or absent in definitions and assigns weights to these features indicating their importance as definitional classifiers. A Genetic Algorithm (GA) is used to learn the relative importance of a set of features.

In section 2 we will give a background on GAs and the describe the different techniques which can be used in the implementation of a GA. In section 3 we describe the GA implemented for this work. In this experiment we try out various different selection algorithms and fitness functions. In section 4 we describe the experiments carried out and the results obtained. We also present various observations made on the different techniques implemented. In section 5 we present work carried out in the area of definition extraction and compare the results obtained with our results. The final conclusions and future work is discussed in section 6.

This work is done in collaboration with an EU-

funded FP6 project LT4eL[1]. The project is described in more detail [MLS07], and aims at enhancing LMSs by using language technologies and semantic knowledge.

2. Background

A Genetic Algorithm (GA) ([Hol75], [Gol89]) is a search technique which emulates natural evolution, attempting to search for an optimal solution to a problem by mimicking natural selection. By simulating a population of individuals represented as strings (with a particular interpretation) GAs try to evolve better solutions by selecting the best performing individuals (through the use of a *fitness function*), allowing them to survive and reproduce. This is done using two operations called *crossover* and *mutation*. Crossover takes two individuals (parents), splits them at a random point, and switches them over, thus creating two new individuals (children, offspring). Mutation takes a single individual and modifies it, usually in a random manner. The fitness function measures the performance of each individual, which is used by the GA to decide which individuals should be selected for crossover and mutation, and which individuals should be eliminated from the population. This process mimics survival of the fittest, with the better performing individuals being given higher chances of reproduction than poorly performing ones. There are various considerations to be taken when implementing a GA. Below we will discuss some of the issues and choices arising when implementing a GA.

The encoding of an individual: Since Holland's work ([Hol75], most GA implementations use a fixed-length binary string to encode individuals in a population. Other possible representations include many-character encodings (such as a protein structure), real-value encodings and tree encodings[2] (such as John Koza's [Koz92] work to represent computer programs).

Fitness function: The fitness function is a problem specific way of evaluating an individual's performance, and is used by the selection method to chose those who will reproduce and move on to future generations.

Selection method: The selection method is responsible for selecting the individuals which will make-up the next generation. It should allow the fitter individuals to survive into the next generations of the population with the idea that the population as a whole grows stronger (fitter), and converge to a good solution. We will describe a number of standard selection methods which we have implemented to experiment with.

Roulette Wheel: This selection method is a fitness-proportionate technique which was used in Holland's original work on GAs, giving a slice of the 'roulette wheel' proportionately to the individual's score. The Roulette Wheel is then spun N times to select N parents for the next generation.

Stochastic Universal Sampling and Sigma Scaling: Extended from the Roulette Wheel, this technique uses an *Expected Value* rather than the fitness itself to allocate a slice of the wheel. To calculate the expected value we use Sigma Scaling, which is calculated according to the fitness of the individual, the mean fitness of the population and the standard deviation of the population fitness, allowing the population to converge at a slower rate. The wheel is spun only once, selecting N parents according to the Sigma Scaling.

Elitism: This selection method keeps a number of the fittest individuals from the population at each generation, and can also be used with various other selection methods.

Boltzmann Selection: Bolzmann selection uses a 'temperature' variable which controls the rate of selection according to a preset schedule. At the beginning the temperature is set to high, resulting in the selection pressure being low. During this time, less-fit individuals have a good chance to be selected. As the temperature is lowered, the selection pressure increases, becoming stricter on selection (where less-fit individuals are more unlikely to be chosen for selection).

Rank Selection: This method considers the rank rather than the fitness value of an individual. Through this technique, if there is a 'super-individual' with a very high fitness value, this difference will not influence the selection as its rank value is one position away from the next ranking individual. In this way we avoid giving a larger share of offspring to a small group of highly fit individuals and allow for a slower convergence of the population.

Genetic operators: *Crossover* is the operation of mating two individuals (parents) to produce two new individuals (offspring) to appear in the next generation. Two ways in which crossover can be implemented are *One-point crossover* (switching the two parents over at one specific random point) and *Bit crossover* (which takes each bit and randomly selects which child will inherit the bit from which parent).

Mutation introduces slight modifications in an individual by randomly (typically with a low probability) changing the gene representation.

1. Language Technologies for Learning `www.lt4el.eu`

2. Tree encoding is used in Genetic Programming, which is sometimes referred to as a GA

Other issues: One important function that crossover and the fitness function should observe is that when mating two good individuals, these should produce good individuals. We briefly viewed the approaches which have been implemented for our experiment. Further detail about GAs and other technical issues may be found in [Mit98].

3. Definition Extraction using Genetic Algorithms

Rule-based definition extraction techniques use linguistic features to identify definitional sentences from non-definitional ones. A feature can be a part-of-speech sequence (such as a determiner followed by a noun and a verb), or simply the presence of certain key words or phrases (such as 'is a' or 'is referred to'). Different features can increase or decrease the probability of whether a sentence is a definition, and thus each feature has a different level of importance for the task of definition identification.

We try to learn the relative importance which can be assigned to features used in definition extraction by using a corpus of non-technical texts and use a GA to learn weights (indicating the relative importance) of a set of features.

3.1. Feature Description

A feature is a test which, given a sentence s, returns whether a particular structure, word or linguistic object is present in the sentence (thus can be categorised as a definition or a non-definition). An example of a feature would be that of a POS sequence that may capture a definition — (e.g. DT→NN→VBZ→DT→NN→IN→NNS) or whether the sentence contains the verb 'to be', returning 1 or 0, indicating the presence or otherwise of the feature.

Given a set of n basic features, f_1 to f_n, and n numeric constants, α_1 to α_n, one can produce a compound feature combining these basic features in a linear fashion:

$$F(s) = \sum_{i=1}^{n} \alpha_i \times f_i(s)$$

The combined feature can be used to determine whether it is able to classify a definitional sentence or not, and if so, with what confidence. There are several ways in which we can interpret $F(s)$. The vanilla version of this technique simply determines whether a sentence is, or is not, a definition by taking zero as the cut-off point (s is a definition if $F(s) > 0$). A more elaborate interpretation is to use the value as the confidence by which one can categorise the sentences as a definition (s_1 is more likely to be a definition than s_2 if $F(s_1) > F(s_2)$). Furthermore, using the zero value as the cut-off point is arbitrary and can be set to any particular value τ (where s is a definition if $F(s) > \tau$ for a fixed value of τ).

3.2. Learning Weights

The problem is now: given a fixed set of features, how can we calculate a good set of weights so as to maximise the effectiveness of the combined features as a definition classifier? We use a GA with the different possible interpretations of a compound feature described above. The values learnt would thus correspond to the relative effectiveness of the individual features as classifiers of definitions. Before starting the experiment, a predefined set of features is adopted and remain static throughout the experiment.

The individual will be encoded as a list of real numbers of length equal to the number of predefined features. Thus, the ith individual ($1 \leq i \leq$ *populationsize*) will have the structure:

$$g_i = \langle \alpha_{i,1}, \alpha_{i,2} \ldots \alpha_{i,n} \rangle$$

Note that n corresponds to the number of predefined features. An individual g_i scores a sentence s using the compound feature formula given earlier:

$$\mathit{value}_i(s) = \sum_{j=1}^{n} f_j(s) \times \alpha_{i,j}$$

The initial population will consist of genes with random weights assigned to each feature.

3.3. Fitness Function

Given that we have a training corpus available, we can define a fitness function that is based on how many definitions and non-definitions an individual manages to classify correctly. F-Measure, precision and recall are popular metrics used in retrieval and classification domains.

Precision is the percentage of correctly classified definitions from all sentences being proposed as definitions by the learning system. This percentage measures the quality of the definitions being proposed.

$$\mathit{Precision} = \frac{\mathit{truePositives}}{\mathit{truePositives} + \mathit{falsePositives}}$$

Recall is the percentage of correctly classified definitions from all the set of positively marked sentences in

the training data. This percentage measures how much of the actual definitions we managed to capture.

$$Recall = \frac{truePositives}{truePositives + falseNegatives}$$

F-measure is a metric that uses precision and recall together with an *alpha* value, which gives the relative importance to be assigned to precision or recall. The alpha value is used to give more importance to either one of the two metrics. Our approach in this experiment is to use F-Measure as a fitness function, with alpha being equal to one (no preference to either precision or recall).

$$F_\alpha = \frac{(1+\alpha^2) \cdot (Precision \cdot Recall)}{(\alpha^2 \cdot Precision + Recall)}$$

Typically, *truePositives*, *falsePositives* and *falseNegatives* used in these formulae are calculated based on the count of how the sentences are classified. We have two ways of classifying sentences as definitions based on $value_i(s)$:

1) by taking a sentence s to be a definition if $value_i(s) > 0$ (we refer to this method as `CountZero`)
2) by computing an optimal τ_i threshold value for the i^{th} individual and take s to be a definition if $value_i(s) > \tau_i$ (we refer to this method as `CountShifted`)

An alternative way of using the formulae is by taking the sum of the squares of the distances above or below τ_i (or zero), rather than simply the count. For example, a sentence classified correctly as a definition with value 17 would contribute not 1, but $(17-\tau_i)^2$ to the value of *truePositives*. This encourages well separated positive and negative classification of sentences. We refer to these methods as `DistanceZero` and `DistanceShifted` respectively.

4. Experiment Description and Results

The purpose of this experiment is to ensure that the GA is able to deliver a set of weights that will perform well when applied to the rules to extract definitions from text. We focus our efforts on definition extraction from non-technical English texts. The corpus consists of a collection of LOs (Learning Objects) gathered as part of the LT4eL project in standard XML format. These were converted from several LOs created by different tutors and originally in different formats. The corpus is annotated with linguistic information, using the Stanford POS tagger and a the Stanford named entity recogniser.

Manually, definitions in the LT4eL corpus were annotated (amounting to 450 definitions), and split into six different categories (described in more detail in [Bor07]). To discriminate between definitions and non-definitions in an automatic manner, it is important to identify features which are present or absent in definitions and could thus be used to classify definitions. Features may range from a simple rule stating *contains the verb "to be"*, to more complex rules, such as POS sequences.

Manually crafted features, based on human observation, obtained a 17% precision and 58% recall in the *containing the verb "to be"* category, and a 34% precision and 32% recall in the *other defining verbs* category[3], when applied to a subset of the corpus. By learning to identify definitions in the separate categories, we reduce the size and complexity of the search space. In this work we will concentrate on the category *containing the verb "to be"*.

Since the purpose of this experiment is to find the best configuration of selection methods and fitness functions, we run the experiments with the different selection methods to analyse convergence, and overall GA results. We also experiment with one-point crossover and bit-crossover, together with a mutation probability of 0.01 percent.

These experiments will focus on the category *containing the verb "to be"*, where we have 111 definitions and 21,122 non-definitional sample sentences. The GA experiments are run with a population size of 100, and 5000 generations. The initial population, although created randomly, is the same for each experiment so that the evaluation of the overall GA performance is not affected by a super individual which might have been present in the one experiment and not in an other, keeping in mind that the purpose of the experiment is to identify the best overall configuration for future runs.

The best final individual should give the clearest separating threshold between definitions and non-definitions. This will also allow us to identify which of the features in the predefined feature set are most important. For the purpose of this experiment we are limiting the set of features to ten:

1) contains the verb "to be"
2) has sequence "IS A" ("to be" followed by a determiner)
3) has sequence "FW IS" (FW is a foreign word - in the example "The process of bringing up the operating system is called booting", booting is

3. This category constitutes of defining verbs or phrases, such as 'is known as', 'is also called' and 'is defined as'.

Table 1. F-Measure for Various Experiments

Fitness Function	Roulette	Roulette-Bit	SUS	SUS-Bit	Boltzmann	Boltzmann-Bit	Elite	Rank	Rank-Bit
`CountZero`	0.02	0.02	0.03	0.03	0.01	0.01	0.03	0.03	0.03
`DistanceZero`	0.01	0.01	0.02	0.02	0.01	0.01	0.02	0.02	0.02
`CountShifted`	0.50	0.45	**0.57**	**0.57**	0.37	0.41	**0.57**	0.53	0.53
`DistanceShifted`	0.39	0.36	**0.54**	**0.54**	0.52	0.41	**0.54**	**0.54**	0.54

tagged as an FW by the part-of-speech tagger.)
4) has possessive pronoun (I, we, you, they, my, your, it)
5) has punctuation mark in the middle of the sentence (such as a hypen or colon)
6) has a marked term (keyword)
7) has rendering (italic, bold)
8) has a chunk marked as an organisation
9) has a chunk marked as a person
10) has a chunk marked as a location

The variety of the features were chosen as simple part-of-speech sequences or other markings in order to facilitate the understanding of the weights learned. Certain features are obviously more present in the set of definitional sentences than in non-definitional sentences such as the marked term feature. We expect this feature to receive a higher weight than say the first feature, as there are many non-definitional sentences containing the verb "to be", and that particular feature on its own is not sufficient to identify definitional sentences.

4.1. Results

The results in table 1 display the F-Measure obtained by the best individual in each experiment across different fitness functions and selection methods.

From the figures presented in table 1, we note that choosing zero as the separating line between definitions and non-definitions yields very poor results. Although in both `CountZero` and `ShiftedZero` on average the recall is over 90%, precision is around 2%, resulting in a very low F-Measure score for all selection methods. As expected, shifting the separator does result in much better figures in our experiments. The `CountShifted` performs slightly better on average than the `DistanceShifted` given their F-Measure, both able to achieve over 50%. We still have to investigate whether the `DistanceShifted` technique separates with more confidence the definitions from non-definitions, by looking at the scores that the individual sentences obtain when using this technique.

Looking at the different selection methods, both the Roulette Wheel and the Boltzmann selection methods seem to be unable to converge with any of the fitness functions used. We believe that these methods do not promote elitism or ranking of individuals. The best individual in the Boltzmann technique tends to fluctuate throughout the GA's lifecycle. In the case of the Roulette Wheel, the fact that two parents are chosen randomly also seems to produce a fluctuating best fitness in the experiments. We also believe that in the shifted techniques, if two individuals who perform well, but at very different thresholds, then it would be difficult to produce equally good offspring from such varying parents. Within the Elite method, the best individuals are also kept whole and are thus not lost to crossover. A possible solution which could be studied further is to introduce an element of multi-species in which the GA restricts mating based on similar thresholds.

Table 2. Results for best experiments

Method	F-Measure	Precision	Recall
SUS (CS)	0.57	0.62	0.52
SUS-Bit (CS)	0.57	0.64	0.50
Elite (CS)	0.57	0.62	0.52
SUS (DS)	0.54	0.59	0.50
SUS-Bit (DS)	0.54	0.59	0.50
Elite (DS)	0.54	0.59	0.50
Rank (DS)	0.54	0.58	0.50
Rank-Bit (DS)	0.54	0.59	0.50

The best performing techniques are `CountShifted` with SUS, SUS-Bit and Elite, and `DistanceShifted` with SUS, SUS-Bit, Elite, Rank and Rank-Bit obtaining close results. We will have to analyse the confidence of sentence classification to be able to judge which fitness function is the better one of the two. In table 2 we present the precision, recall and F-Measure for the best resulting experiments. We emphasise that since the features chosen are not a complete set of possible features the results could improve with a larger feature set for the "is a" category of definitions. When comparing these results to those achieved with the manually crafted rules in the LT4eL project, and considering that the experiment is using a restricted selection of features,

we not only have managed to retain a high recall, but also increased precision to over 60% from just 17% attained using the manually crafted rules.

5. Related Work

There are various attempts at definition extraction for different application. [MK02] use a rule-based approach to extract definitions from technical medical texts to create a non-technical glossary, using cue phrases as well as linguistic information. They manage to obtain a precision of 87% and a recall of 74%. [WP06] extract definitions from legal texts, applying linguistic rules and post-filtering rules, obtaining an average precision of 47%. [FB06] extract definitions from the Dutch medical wikipedia webpages using rule-based followed by machine learning techniques, namely naïve Bayes, maximum entropy and support vector machine, reaching a precision of up to 92%. [LCN03] extracts definitions from the Internet, looking for a particular concept together with cue phrases (e.g. {Concept}{refers to — satisfies}), achieving a precision of 61%.

The results above show that definition extraction obtains varied results, depending much on the corpus being used and the techniques being carried out. Our work is based on non-technical texts where definitions might be in less structured sentences. For instance, when working with encyclopedia articles, the first sentence present is most likely to be a definition. In our case definitions can be present anywhere in the text, making it harder to identify. However, with the inclusion of more complete feature set, we believe that our results will improve and be comparable to other work.

6. Conclusions and Future Work

The results achieved are encouraging, outlining the best selection methods and fitness functions that should be used in future experiments. We plan to experiment further with the `CountShifted` and `DistanceShifted` using SUS and Elite selection methods to explore different alpha values for F-Measure, thus giving more emphasis to precision or recall. We will attempt to use first a set of weights which allows for high recall (thus capturing all possible definitions, including wrong ones), followed by the application of a set of weights with high precision (thus filtering out the wrong definitions captured in the first phase). We also plan to look into the results achieved by the `DistanceShifted` function to see whether using the distance rather than the count does actually separate the positive and negative sentences, leaving ambiguous cases around the shifted zero.

Since one of the challenging elements of this experiment is the identification of appropriate features, we plan to use Genetic Programming techniques to learn new features which can then be used in our GA to improve definition extraction.

References

[Bor07] Claudia Borg. Discovering grammar rules for Automatic Extraction of Definitions. In *Doctoral Consortium at the Eurolan Summer School 2007, Iasi, Romania.*, pages 61–68, 2007.

[FB06] Ismail Fahmi and Gosse Bouma. Learning to Identify Definitions using Syntactic Features. In *Workshop of Learning Structured Information in Natural Language Applications, EACL, Italy*, 2006.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA, 1989.

[Hol75] John H. Holland. *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor, 1975.

[Koz92] John R. Koza. *Genetic Programming: On the Programming of Computers by means of Natural Selection*. MIT Press, Cambridge, MA, 1992.

[LCN03] Bing Liu, Chee W. Chin, and Hwee T. Ng. Mining Topic-Specific Concepts and Definitions on the Web. In *Proceedings of the Twelfth International World Wide Web Conference (WWW'03)*, 2003.

[Mit98] Melanie Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1998.

[MK02] Smaranda Muresan and Judith L. Klavans. A method for automatically building and evaluating dictionary resources. In *Proceedings of the Language Resources and Evaluation Conference*, 2002.

[MLS07] Paola Monachesi, Lothar Lemnitzer, and Kiril Simov. Language Technology for eLearning. In *First European Conference on Technology Enhanced Learning*, 2007.

[WP06] Stephan Walter and Manfred Pinkal. Automatic Extraction of Definitions from German Court Decisions. In *Workshop on Information Extraction Beyond The Document*, pages 20–28, 2006.