Automatic Grammar Rule Extraction and Ranking for Definitions

Claudia Borg, Mike Rosner, Gordon J. Pace

University of Malta
claudia.borg, mike.rosner, gordon.pace@um.edu.mt

Abstract

Learning texts contain much implicit knowledge which is ideally presented to the learner in a structured manner - a typical example being definitions of terms in the text, which would ideally be presented separately as a glossary for easy access. The problem is that manual extraction of such information can be tedious and time consuming. In this paper we describe two experiments carried out to enable the automated extraction of definitions from non-technical learning texts using evolutionary algorithms. A genetic programming approach is used to learn grammatical rules helpful in discriminating between definitions and non-definitions, after which, a genetic algorithm is used to learn the relative importance of these features, thus enabling the ranking of candidate sentences in order of confidence. The results achieved are promising, and we show that it is possible for a Genetic Program to automatically learn similar rules derived by a human linguistic expert and for a Genetic Algorithm to then give a weighted score to those rules so as to rank extracted definitions in order of confidence in an effective manner.

1. Introduction

Definitions provide information on the meaning of a term to enable its comprehension and can be useful in several scenarios. In an eLearning context, definitions allow learners to assimilate a term's meaning, thus facilitating the learning process faced by students. In an ideal situation, definitions are available collectively in a glossary to enable quick referencing and access to terms' meanings so that students would not have to sieve through texts to find such information. One way of composing such glossaries is for tutors to copy definitions manually into a glossary as they create the learning content. However, tutors generally do not have the time to do such manual tasks. Ideally, once content is prepared and uploaded on to a Learning Management System, the tutor is offered additional functionality to automatically create such glossaries and additional metadata which could help students in their learning path.

Identifying definitions in an automatic manner is generally based on manually crafted rules which attempt to separate definitions from non-definitions by distinguishing linguistic structure or certain key words or phrases present exclusively in definitions. However, the effectiveness of such a technique depends on several factors, such as the type of text from which the definitions are being extracted — technical and medical texts tend to be more well-structured when defining terms and thus make it easier to create rules for their extraction, whilst non-technical texts tend to be more informal when describing terms, making it more difficult to discriminate between definitions and non-definitions.

Further approaches augment the results by applying machine learning techniques, were different types of algorithms are used to classify definitions from non-definitions, improving results over manually crafted rules. Such techniques are seen as worth investigating and usually focus on identifying features that could be used as a dividing line between the two sets of sentences (definitions and non-definitions).

In this paper, we explore the use of evolutionary techniques to learn sentence classifiers which discriminate between definitions and non-definitions in non-technical texts. We split the approach into two distinct phases (i) using genetic programming techniques to automatically discover grammar rules which act as effective discriminators; and (ii) using genetic algorithms to combine rules learnt through different runs of the genetic programming module and learn their relative importance. In this manner we can analyse a corpus and identify definitions in a fully automated manner. The results achieved are very encouraging, despite the fact that the approach has little human expert intervention.

In section 2 we provide the background as the basis of our work. Section 3 describes the experiments carried out using genetic programming to discover grammar rules automatically, whilst section 4 applies genetic algorithms to learn the relative importance of the rules used to classify definitions. In section 5 we demonstrate how these two techniques can be combined with the aim of achieving a complete automatic definition extractor tool. In section 6 we discuss the results achieved compared to related work and provide future directions of our experiments in section 7.

This work was done in collaboration with an EU-funded FP6 project LT4eL[1]. The project is described in more detail in (Monachesi et al., 2007), and aimed at enhancing Learning Management Systems by using language technologies and semantic knowledge.

2. Background

Rule-based approaches to definition extraction tend to use a combination of linguistic information and cue phrases to identify definitions. (Muresan and Klavans, 2002; Storrer and Wellinghoff, 2006) use technical texts where definitions are more likely to be well-structured. Other work attempts definition extraction from the Internet (Klavans et al., 2003) and

[1] Language Technologies for eLearning `www.lt4el.eu`

eLearning texts (Westerhout and Monachesi, 2007; Przepiórkowski et al., 2007), where machine learning is then used to improve results. Non-technical texts tend to contain definitions which are ambiguous, uncertain or incomplete compared to technical texts.

In our work, we focus on definition extraction from non-technical eLearning English texts in the field of ICT. The corpus consists of a collection of learning objects gathered as part of the LT4eL project (Monachesi et al., 2007) which were collected from several tutors in different formats, and standardised in XML format. It is generally recognised that part-of-speech information, which can be extracted automatically from natural language texts is crucial to enable effective discrimination, and the corpus is thus annotated with linguistic information, using the Stanford part-of-speech tagger (Toutanova and Manning, 2000).

The corpus was manually annotated with definitions, to be used as a training set for the definition extraction task. Manually crafted grammars were created in the project to extract definitions, however the results were not satisfactory (Borg, 2007). From observation it was noted that the structure of definitions does not always follow a regular *genus et differentia* model and different styles of writing pose a major challenge for the identification of definitions. The solution adopted was to categorise the definitions into different classes, and engineer definition recognisers for each of the classes separately. This reduces the complexity, by attempting to identify a grammar focusing for each type of definition. The types of definitions observed in the LT4eL texts were classified as follows:

1. Copula: Definitions containing the verb 'to be' as a connector. E.g.: 'A joystick is a small lever used mostly in computer games.'

2. Verb: Definitions containing other verbs as connectors such as 'means', 'is defined' or 'is referred to as'. E.g.: 'the ability to copy any text fragment and to move it as a solid object anywhere within a text, or to another text, usually referred to as cut-and-paste.'

3. Punctuation: Definitions containing punctuation features separating the term being defined and the definition itself. E.g.: 'hardware (the term applied to computers and all the connecting devices like scanners, telephones, and satellites that are tools for information processing and communicating across the globe).'

Three further categories have been identified and used in the LT4eL project, but were not considered for our experiments due to the difficulty of applying machine learning in those instances.

3. Learning Grammatical Structures

Definitional sentences can be classified on the basis of their linguistic structure, keywords and other identifying features. However, to identify these structures manually is challenging and requires linguistic expertise. Manually crafted rules are typically expressed as complex grammars or regular expressions ranging over parts-of-speech. In our work, we propose to explore as many potential grammatical structures/rules as computationally possible in an automatic manner by using genetic programming. Unlike a human expert, the technique has no preconceived knowledge of what a correct grammar rule should consist of. Thus, the genetic program will produce ungrammatical rules that would not match any sentence (whether definitional or non-definitional). However, these will be discarded as the genetic program evolves and learns better preforming rules.

3.1. Linguistic Rules

Linguistic rules can be viewed as objects which, given a sentence, return a boolean value indicating whether or not the sentence matched the rule. Such rules can easily be expressed as regular expressions, e.g. `noun·is·a·noun` where we find a mixture of part-of-speech classes (noun) or specific words (is). The linguistic content of such rules can vary, and we can use either specific parts-of-speech tags (such as NN for noun, common, singular or mass) or general categories (such as the class noun to refer to all POS tags in the noun category), or even specific words.

A set of linguistic rules can be used as a complete grammar to capture definitional sentence. However, we also need a way of evaluating such a rule to decide how effective in its task it is. Given a corpus of sentences (definitions and non-definitions) each time a rule is tested on the whole corpus of sentences, we end up with 4 figures which are (i) true positives (definitional sentences classified correctly), (ii) false negatives (definitional sentences classified as non-definitions), (iii) false positives (non-definitional sentences classified as definitions) and (iv) true negatives (non-definitional sentences classified correctly). Through these figures we are able to calculate precision, recall and f-measure, with the latter metric used to evaluate how well a rule is able to classify definitional sentences.

3.2. Genetic Programming

Genetic programs (GP) is an evolutionary algorithm introduced by (Koza, 1992) mimicking natural selection. A GP aims at learning instances of a language (typically computer programs) automatically by searching for a potential optimal solution. It starts by creating a random population of individuals (potential solutions) represented as tress, and tries to evolve better solutions by selecting the best performing individuals (through the use of a *fitness function*), allowing only the best individuals to survive into the next generation through reproduction. 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

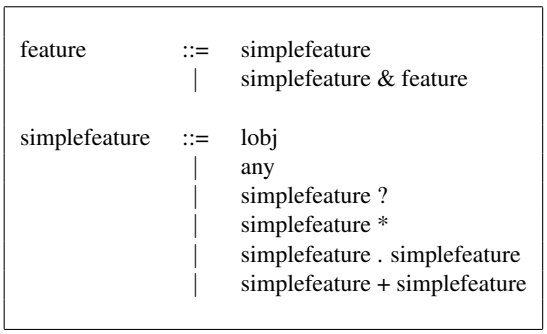

Figure 1: Specification of the representation of individuals

each individual[2], which is used by the GP 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, and thus their winning characteristics are passed on to future generations.

In our work, the individuals of the population consists of potential grammar rules which could match definitional or non-definitional sentences. The fitness function of each individual is based on the f-measure metric which gives an indication not only how definitional sentences are classified, but also how non-definitional sentences are classified as well, thus having a more complete view of an individual's performance.

3.3. Encoding the Individual

In GPs, an individual is generally represented as a tree, with most work focusing on the representation of computer programs. In this case, we are interested in learning grammar rules which can be used to identify definitions. At the same time we would like the rules to be resembling closely to manually crafted rules which are generally represented as sequences of POS or regular expressions. We chose to represent individuals as extended regular expressions, defining a language over which our rules are created.

These regular expressions would range over the grammar shown in figure 1. Linguistic objects ($lobj$) are the terminal elements in the grammar, resulting in the use of either parts-of-speech or other linguistic components used to create the grammatical rule. Note also that to enable more complex rules, we allow not only the usual regular expression operators (optional inclusion, repetition, catenation and choice), but also allow the conjunction of regular expressions at the top most level (thus controlling the computational complexity of matching the regular expression).

The framing of basic features as instances of this language of regular expressions, enables us to formulate the task of the learning algorithm as that of learning an instance of this language (of regular expressions) which is effective when used for definition extraction. For the choice of linguistic objects, we chose to either use specific part-of-speech tags such as NN (noun, common, singular or mass) or to generalise these tags into one class and refer to them as nouns.

This framework allows a certain degree of flexibility in that the linguistic objects can be different for each experiment run. Thus this will allow us to use different POS tags for the different categories that we are working on. Of course, the linguistic objects chosen will effect the resulting rules since the GP can only use structures that it knows about. The regular expression any matches any linguistic element.

3.4. Results

When using machine learning techniques, it is usually necessary to reduce the problem size into a manageable one, so as to facilitate the learning process. In the case of definition extraction, it is possible to run different experiments for each separate category of definitions so that the grammar rules learnt would focus on the characteristics of a particular category. We focused on the three main categories described in section 2, and we also compare the results achieved to particular experiments that attempt learning a grammar for all types of definitions. By restricting the learning process to one particular category we were able to obtain improved results over the learning of all definitions, since this allows a reduced search space of potential solutions, thus resulting in an 'easier' problem to solve. Our results are thus divided over the different definitional categories that we worked upon (copula, verb, punctuation) with a final comparison of the results achieved when attempting to learn a grammar for all definitions.

3.4.1. Copula category

In this category, we find definitions containing the words 'is a', usually showing that a term is being described within the sentences. The challenge of this category is that there are several instances where the term 'is a' does not necessarily translate into a definitional sentence (e.g.: John *is a* tall man). This means that we also need the surrounding structure of the sentence in order to be able to distinguish between definitions and non-definitions.

Our experiments were split in to two. The first experiment was to test the applicability of the use of the component any. The experiment showed that with a minimal set of linguistic objects (the word *is*, noun, adjective, adverb, modal) the GP was able to learn rules like *noun·is·any·noun*, with any replacing the determiner. The average f-measure obtained by the GP using five simple linguistic objects was around 22%, with the best experiment achieving 26%. In this set of experiments we tried various different settings with respect to the technical aspects of the GP, such as population size and number of generations. We observed that although the experiments with larger populations and more generations did produce better performing rules (such as the one achieving a 26% f-measure), the actual rules

[2]The fitness of an individual is the measure of how good this candidate solution is at solving the problem being tackled.

were large and cumbersome, with the worst instance taking up over 10 pages of print. In experiments were the number of generations and population size were smaller, we achieved a two-point lower f-measure, but with a rule resulting in no longer than four lines of print. Thus we favoured such settings in further experiments in order to have more 'legible' rules.
In the next set of experiments we introduced a larger set of linguistic objects and ran several experiments. Overall, the results slightly improved over the previous experiments, with an average of 26% f-measure achieved. Table 1 shows the results achieved on average for a number of experiments, and the best result from these.

	F-measure	**Precision**	**Recall**
Average	0.25	0.20	0.33
Best	0.28	0.22	0.39

Table 1: Results for Copula Category

Overall in these experiments, the most obvious rule (*noun is determiner noun*) was always learnt quickly. However, the results improved only when this rule was slightly modified, say by adding optional linguistic components which might also capture other sentences apart from this rigid sequence. Thus the GP allowed a perfect setting to explore small changes to the discovered rules and to evaluate their performance. When a rule performs better that other rules, it is always kept and its components spread over to the remaining rules.

3.4.2. Verb category

In this category we find definitions which contain connector verbs such as 'define', 'call' and 'refer'. For this category we decided to split the grammar learning in two separate experiments. One experiment was to contain all different verb part-of-speech tags so as to see if it is possible to learn a grammar based solely on linguistic information rather than specific keywords. The other experiment was to contain these keywords specifically as linguistics objects in their base form. Table 2 shows the best result obtained from these different experiments.

	F-measure	**Precision**	**Recall**
Verb POS	0.11	0.16	0.09
W Keywords	0.20	0.33	0.14

Table 2: Results for the Verb Category

Relying on simple part-of-speech information for this category was not sufficient, and it is clear from the result of an 11% f-measure that the GP was not able to learn satisfactory rules. At times the rules did not contain any verb POS tags, and it seemed as though it did not help the language learning process at all to have all verb POS as part of the linguistic objects. On the other hand, an improvement was registered once the verb POS tags were all grouped as one category, and we introduced specific keywords found in these type of definitions (mean, define, relate, call, consist, know). However, when the learning progress through the lifetime of any experiment was analysed, we noticed that although some of these words were actually present in the random generation of the initial population, they were soon discarded because they were contained within ungrammatical rules. This means that if the GP at the beginning of an experiment created a rule containing the verb 'call' in an ungrammatical rule, the rule would perform badly, and be discarded together with the word 'call'. Thus, the GP would not have the opportunity to investigate rules containing 'call', unless it is reintroduced through mutation[3]. This clearly shows that search space exploration is very important, and several different experiments must be carried out in order to obtain more comprehensive set of rules.

3.4.3. Punctuation category

The punctuation category contains sentences were punctuation delimits the sentence in such a way so as to indicate an explanation present in the sentence. The most obvious example is that of when a keyword is followed by a colon and its explanation. We ran two experiments in this category, and the results were more positive than in the other categories. In table 3 we show the results of these experiments, with the best f-measure achieved was at 30%, and the rules learnt were concise and clear.

F-measure	**Precision**	**Recall**
0.27	0.23	0.33
0.30	0.25	0.36

Table 3: Results for the Punctuation Category

Possible reasons as to why we managed to obtain better results in this category are that the sentences might be rather similar in structure and the number of distinctive linguistic objects in this category is rather small. In fact most of the rules learnt revolved around the use of a colon. The fact that the rules learnt were very simple in structure could also indicate that the learning task was easier than the other two categories. However, we also observed that the use of the comma was not well explored by the GP — this could mean that when commas are used in definitional sentences, the structure of the sentence itself might be too similar to non-definitional sentences, thus making it difficult for the GP to learn rules which would distinguish the two types of sentences.

4. Relative Importance of Rules

One of the aspects of rule-based methods in definition extraction is that all rules are treated equally when

[3]It is possible for any linguistic element to be introduced through mutation, however there is no preference given between linguistic objects already present and those not present, so the chances remain rather low.

used to classify definitions from non-definitions. This means that if there is one rule that is able to classify definitions better, its classifications are at par with other candidate sentences. One way of introducing some form of preference on the classification is to consider how many of the rules a sentence matches, with sentences matching a higher number of rules being displayed in the result list before other sentences. However, not all rules are equally effective at capturing definitions. Ideally, we have some form of relative importance attached to each rule which indicates how effective the rule is at distinguishing definitions from non-definitions. The weight of the rule that captures a candidate sentence would affect how the sentence is ranked, and thus we can introduce a ranking mechanism presenting candidate definitions according to the level of confidence in their classification. In order to learn such weights, we applied evolutionary algorithms, namely a genetic algorithm, to learn the relative importance of a set of rules.

4.1. Genetic Algorithms

A Genetic Algorithm (GA) (Holland, 1975; Goldberg, 1989) is a search technique similar to the GP described above. It simulates a population of individuals which evolve into better solutions through the process of crossover and mutation. It also uses a fitness function whereby it evaluates how well an individual performs and thus deciding whether the individual should be part of future generations or not. The main difference between GAs and GPs is the representation of the individual, where we now have individuals represented as strings.

4.2. Combining Features

A rule is considered to be a test which, given a sentence s, returns a boolean value stating whether a particular structure, word or linguistic object is present in the sentence — essentially, characteristics that may be present in sentences. We have so far spoken about grammatical rules that were learnt automatically by the GP, but these could vary, from rendering information (bold, italic), to the presence of keywords, or part-of-speech sequences that could identify the linguistic structure of a definition. We use the term feature to capture not only grammar rules, but other identifying aspects of definitions. Features such as the presence of a bold word can be used to indicate the likelihood of whether a sentence is indeed a definition, and thus relying not only on a sentence's linguistic structure, but also other potential aspects of definitional sentences.

In order to learn weights to a set of features, we need to combine them together so that the weights learnt are relative to the full set. Thus we can describe that given a vector of n basic features, $\overline{f} = \langle f_1, \ldots f_n \rangle$, and numeric constants, $\overline{\alpha} = \langle \alpha_1, \ldots \alpha_n \rangle$, we define a compound feature combining them in a linear manner:

$$F_{\overline{\alpha}}^{\overline{f}}(s) = \sum_{i=1}^{n} \alpha_i \times f_i(s)$$

Given a sentence, a vector of features and their respective weights, we can thus calculate a numeric value of the sentence by combining the features accordingly. One would also have to identify a threshold value τ such that only sentences scoring higher than this value would be tagged as definitions i.e. s is tagged as a definition if and only if $F_{\overline{\alpha}}^{\overline{f}}(s) \geq \tau$.

4.3. Learning Weights

We use a GA to identify a good set of weights and the threshold value for a given set of features. Each individual in the population of the genetic algorithm is represented as the vector of numeric weights. Crossover between individuals simply consists of splitting the vector of the two parents at a random position, and joining the parts, thereby creating two new individuals for the next generation.

What the GA learns is determined by the fitness function, which, given an individual, returns a score of how 'good' the individual is. The fitness function takes an individual (vector of weights) and runs through the whole corpus using the combined feature function and calculates how many definitions are correctly classified, and how many are incorrectly tagged as non-definitions. Similarly, we compute the values for the non-definitional sentences. Through these figures we are then able to extract precision, recall and f-measure in the same manner we did for the GP.

One aspect that differs from the GP is in the actual classification of the sentences themselves. Through the use of weights we produce a score which should indicate a level of confidence in the classification. For instance, if we consider the threshold to be zero, anything scoring above would be a candidate definition, anything below a non-definition. However the choice of using zero as the threshold is rather arbitrary. We are able to compute the optimal value for the threshold with respect to the corpus using an efficient (linear) algorithm.

Two experiments were run, one with a fixed threshold value of zero, and another using optimal (individual specific) thresholds, with the latter achieving far better results.

4.4. Initial Results

The GA experiments focused on the copula category, using different techniques within the algorithm mechanics. The best selection algorithm was SUS with sigma scaling (Mitchell, 1998). Here we present a summary of the best and most interesting results of this work. During the set up of the GA, we used a simple set of ten features which were hand-coded and inputted into the GA for it to learn their relative importance. Following is the set of features used:

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 tag indicating a foreign word - in the example "The process of

bringing up the operating system is called booting", booting is tagged as an FW.)

4. has possessive pronoun (I, we, you, they, my, your, it)
5. has punctuation mark in the middle of the sentence (such as a hyphen 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

These features were purposely simplistic when compared to the manually crafted rules in the LT4eL project for definition extraction. This enabled us to analyse the relative weights assigned and to be able to allow more focus on the algorithmic aspects of the GA. These features were used throughout all the experiments discussed in this section.

Method	F-measure	Precision	Recall
Experiment 1	**0.57**	0.62	0.52
Experiment 1a	0.62	**0.70**	0.42
Experiment 1b	0.54	0.46	**0.56**
Experiment 2	**0.57**	0.64	0.50
Experiment 3	**0.54**	0.59	0.50

Table 4: Results for best experiments

Table 4 presents the results achieved by the best performing runs, indicating the f-measure, precision and recall achieved by assigning the weights learnt to the set of features. The best runs achieved an f-measure of 57%, with the runner-up achieving 54%. Since we used f-measure as the basis of measuring the weights' effectiveness in classifying definitions, we were also able to influence f-measure to favour precision or recall according to the setting of the alpha value. Experiments 1a and 1b show the results for favouring precision and recall respectively.

Using a small set of simple features, the GA has managed to obtain positive results, especially when comparing to the manually crafted grammars in LT4eL. We have increased precision from 17% to 62%, whilst maintain recall over 50%. Further improvement would probably be achieved had we to include more rules from the manually crafted grammar as part of our set of features.

The possibility of influencing the learning of weights to favour precision or recall is considered a positive facility in this experiment, since the end use of the definition extraction tool could require different settings. In a fully automatic system, precision might be given more importance, whilst in a semi-automatic system, recall is more important since a human expert will verify the correctness of the candidate sentences.

5. Towards Automatic Definition Extraction

With both techniques in place, each with a specific task (producing rules and producing weights), we next wanted to view their effectiveness as a complete tool which could be used as a fully automated definition extractor. Combining the two techniques is shown in figure 2, were we see the different phases of the definition extraction process. Phase one is the creation of an annotated training set and is not dealt with in this work. Given an annotated corpus with definitions, one can then move onto phase two where the GP is applied to learn useful simple features which can be used to distinguish definitions from non-definitions. In phase three the GA is then used to learn weights for the rules learnt by the GP. Using the rules and weights, one can incorporate all this in a definition classification tool in phase four. In this section we present the results achieved from combining phase two and three together.

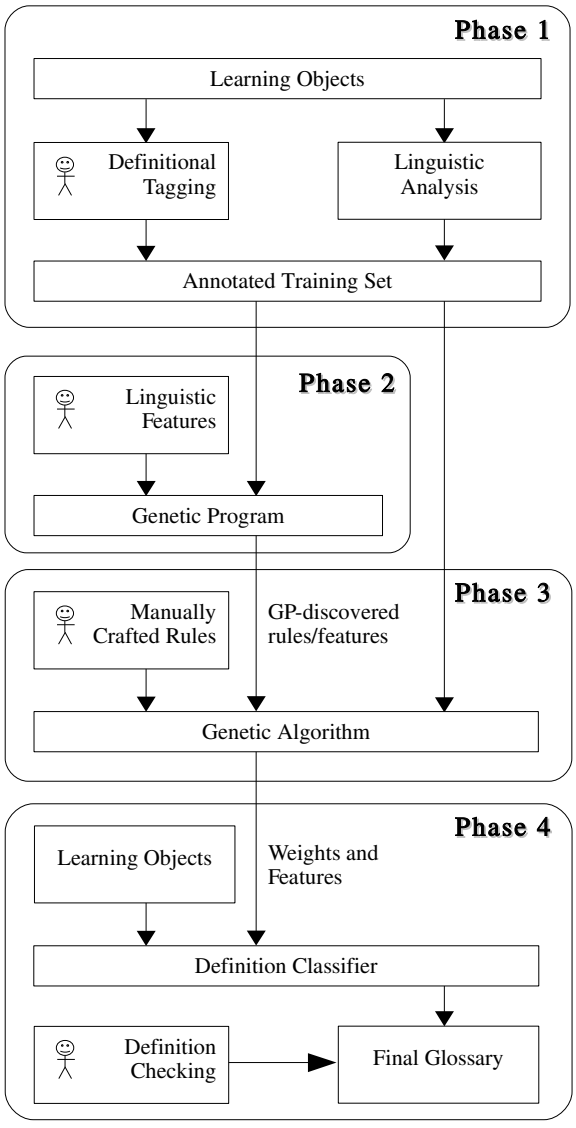

Figure 2: Phases of definition extraction

For the purpose of combining the two phases, we used the best rules learnt by ten different GP experiments in

the copula category. These individuals were used by the GA to learn their respective weights. The set-up is shown in figure 3 where the final result is a set of rules in the copula category together with their allocated weights indicating the level of effectiveness each weight has. As shown in the previous section, the rules the GP learnt without the application of the weights resulted at best in f-measure being 28%. Once weights were learnt and applied to the definition extraction tool, this increased to 68% f-measure. This improvement shows that learning weights is useful to the classification task since it does matter which rule is actually carrying out the classification of sentences.
Further analysis show that the f-measure is resulting from a 100% precision and a 51% recall. This means that by combining the rules learnt and their associated weights, we succeeded in classifying just over half of the annotated definitions, without classifying any incorrect definitions. There are several factors behind these results:

1. This experiment was carried out on only one corpus, so the rules learnt together with their respective weights, were specific to the corpus used. Achieving such a good result is only indicative that as in any machine learning process, the two algorithms were able to learn rules and weights specific to our corpus.

2. The recall of 51% represents definitions for which the genetic program did not learn rules for. Since these algorithms are searching for solutions in an automatic manner without expert feedback, it is the case that not all possible rules are explored. This can be tackled by including rules from more experiments or by having direct feedback from a linguistic expert (say, injection of good humanly crafted rules into the population).

Notwithstanding the conditions under which they were achieved, the results are very promising.

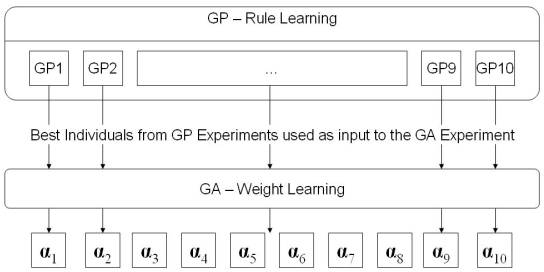

Figure 3: Combining the two experiments

6. Discussion and Related Work

Although the results achieved so far are promising and encourage further investigation of these techniques, it is difficult to provide a fair and just comparison to other techniques. One of the main reasons is that an evaluation using an unseen corpus is required to have a more realistic view of the results achieved using these techniques. To our knowledge there is no other work in definition extraction using evolutionary algorithms to which our results can be directly compared to.

However, there are various attempts at definition extraction using different techniques. DEFINDER (Muresan and Klavans, 2002) is a rule-based system which extracts definitions from technical medical texts so that these can later be used in a dictionary. The rules are primarily based on cue-phrases such as "is called a", with the initial set of candidate sentences being filtered out through the use of POS rules and noun phrase chunking. They manage to obtain a precision of 87% and a recall of 74%. Definition extraction is also considered to extract the semantic relations present in definitions. In (Malaisé et al., 2004), they apply lexico-syntactic patterns in addition to cue phrases, focusing on hypernym and synonym relations in sentences. They obtain 66% precision and 36% recall.

Work carried out in (Storrer and Wellinghoff, 2006), applies valency frames to capture definitional sentences achieving an average of 34% precision and 70% recall across the rules created. A German corpus consisting of legal decisions is used in (Walter and Pinkal, 2006) to extract definitions. They analyse the structure of definitions in this domain, and observe that the German word `dann` can be used as a signal word indicating that a sentence is a definition. There is no equivalent term in English. The rules are crafted manually through observation, and achieve an average of 46% precision. When only the most effective rules are used, precision increases to over 70%, however recall is not discussed since the corpus is not annotated with definitions. Extraction of definitions from eLearning texts is attempted for the Slavic group of languages in (Przepiórkowski et al., 2007), using noun phrase chunking and phrase structure as the potential identifying features in definitions. The best results are achieved for the Czech language with precision at 22% and recall at 46%.

Research in general seems to point out to the need of going beyond rule-based techniques, and trying out machine learning to improve definition extraction. Definitions extracted from the Dutch Wikipedia from medical articles in (Fahmi and Bouma, 2006) first use a rule-based approach using cue-phrases, but further improve their extraction process by using Naïve Bayes, maximum entropy and SVNs. As part of their feature set they include sentence positioning, a feature which cannot be applied to other types of corpora. The best result is from applying maximum entropy, achieving 92% accuracy. Similar experiments by (Westerhout and Monachesi, 2007) on an eLearning corpus obtain 88% accuracy, with the difference in result being due to the type and structure of the corpus used. Similarly (Kobyliński and Przepiórkowski, 2008) obtain an accuracy of 85% using a Balanced Random Forest on an eLearning corpus. These techniques all share the similarity in having improved considerably the results of manually crafted grammars when applying machine learning techniques.

7. Future Directions

In this paper, we have presented a methodology for the use of evolutionary algorithms to create sentence discriminators for definition extraction. We have shown how GPs can be used to learn effective linguistic rules, which can then be combined together using weights learnt through the use of a GA. The overall system can, with very little human input, automatically identify definitions in non-technical texts in a very effective manner. Using our approach we have managed to learn rules similar to the manually crafted ones by the human expert in the LT4eL project, and further associate them with weights to identify the definitions in non-technical texts — all performed in an automated fashion. One of the major strong points of the approach is that the (expensive) learning phases is performed once, and the resulting definition discriminator is very efficient, making it viable to be included in other applications.

The final experiment of using both techniques for definition extraction gave surprising results, managing to identify only definitions, achieving a 100% precision, albeit having identified rules to capture only half of the definitional set of sentences. This result is certainly encouraging when considering that the process is fully automated.

There are various directions we plan to explore in the future. Our experiments would need to be evaluated further, experimenting with other corpora in different domains. For instance, medical texts contain several terms which a part-of-speech tagger might not recognise and would tag as 'foreign word'. Thus the rules learnt for our eLearning corpus might not necessarily apply for a medical corpus.

We also intend to evaluate the definition extraction tool over an unseen corpus. Such an evaluation might show that the rules learnt by the GP are not generic enough to cover unseen definitions, a result which is common in such machine learning techniques. It would be ideal to have some form of feedback loop from an expert to the learning algorithm to integrate new knowledge gained over unseen corpora.

We plan to explore and assess the use of weights to go beyond a crisp discriminator, and interpret the results as a fuzzy discriminator, associating a degree of confidence with each sentence, thus enabling us to rank definitions according to how sure the system is that it is a definition. This is crucial if the definitions discovered are to be vetted by a human operator.

Finally, we plan to extend the use of GP to learn rules in an iterative manner. After each iteration of the experiment, the sentences for which it learnt rules are removed from the training corpus, and the experiment repeated. In this way we would be reducing the search space, and forcing the GP to learn new rules. It might be the case that the GP does not learn certain rules as they would classify to many non-definitions to simply capture few definitions. However, by carrying out such an experiment we might be able to learn rules which cover the search space better, and at the same time identify those definitions for which it is difficult to define rules which provide acceptable results.

8. References

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

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

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

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

Judith L. Klavans, Samuel Popper, and Rebecca Passonneau. 2003. Tackling the internet glossary glut: Automatic extraction and evaluation of genus phrases. In *SIGIR'03 Workshop on Semantic Web*.

Łukasz Kobyliński and Adam Przepiórkowski. 2008. Definition Extraction with Balanced Random Forests. In *proceedings of GoTAL 2008*.

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

Véronique Malaisé, Pierre Zweigenbaum, and Bruno Bachimont. 2004. Detecting semantic relations between terms in definitions. In *COLING CompuTerm 2004: 3rd International Workshop on Computational Terminology*, pages 55–62.

Melanie Mitchell. 1998. *An Introduction to Genetic Algorithms*. MIT Press.

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

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

Adam Przepiórkowski, Łukasz Degórski, Miroslav Spousta, Kiril Simov, Petya Osenova, Lothar Lemnitzer, Vladislav Kubon, and Beata Wójtowicz. 2007. Towards the automatic extraction of definitions in Slavic. In *Proceedings of the BSNLP workshop at ACL 2007*.

Angelika Storrer and Sandra Wellinghoff. 2006. Automated detection and annotation of term definitions in german text corpora. In *Language Resources and Evaluation Conference*.

K. Toutanova and C. D. Manning. 2000. Enriching the Knowledge Sources Used in a Maximum Entropy Part-of-Speech Tagger. In *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-2000), Hong Kong*.

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

Eline Westerhout and Paola Monachesi. 2007. Extracting of Dutch Definitory Contexts for elearning purposes. In *CLIN 2007*.