Enhancing Information Retrieval by Automatic Acquisition of Textual Relations using Genetic Programming

Agneta Bergström
Department of Linguistics
Göteborg University
Box 200
SE-405 30 Gothenburg
+46 8-39 08 46
cl4aberg@cling.gu.se

Patricija Jaksetic
PLAY: Applied research on art and technology
Viktoria Institute
Box 620
SE-405 30 Gothenburg
+46 31-773 55 39
patricia@viktoria.informatik.gu.se

Peter Nordin
Complex Systems
Chalmers University of Technology
SE-412 96 Gothenburg
+46 31-772 31 59
nordin@fy.chalmers.se

ABSTRACT

We have explored a novel method to find textual relations in electronic documents using genetic programming and semantic networks. This can be used for enhancing information retrieval and simplifying user interfaces. The automatic extraction of relations from text enables easier updating of electronic dictionaries and may reduce interface area both for search input and hit output on small screens such as cell phones and PDAs (Personal Digital Assistants).

Keywords

Genetic programming, machine learning, natural language processing, semantic networks, information retrieval.

1. INTRODUCTION

The amount of electronic information available is growing exponentially. We talk about our society as the information society and an ever-increasing flood of information is pouring over us in both professional and private contexts. The phenomenon is sometimes referred to as information overload (cf. [10]). However, while the technology to produce, duplicate, spread and store information has leapt forward enormously, the capabilities for filtering and visualizing information have not made corresponding progress. The Internet is growing every day and new, often inexperienced, users are continuously joining the on-line community. This puts an extra strain on simplification of user interfaces and on the use of powerful information retrieval methods. There have for instance been several attempts made to cluster information syntactically and graphically to show relevant search hits [2,7,13].

However not only the number of users is increasing, the number of situations where we would like to have information access is increasing as well, with the introduction of network mobile devices such as cell phones, PDAs, and watches. The new forums places higher constraints on user interfaces for both input and output and graphical user interfaces may be harder to realize and fit on small screens. Imagine searching for information using your favorite search engine but in a WAP (Wireless Application Protocol, [12]) -phone display that is only capable of showing 40 times 3 characters. The work presented here aims at enhancing information retrieval by automatically extracting relations from text masses putting less stress on the user interface. In the scenario below we give an example of how this technique can be useful.

2. SCENARIO

Assume that you are using a WAP mobile phone, which is connected to the Web, and you would like to find out the information regarding a new mobile phone. You could then input your query on a small screen interface as in Figure 1.

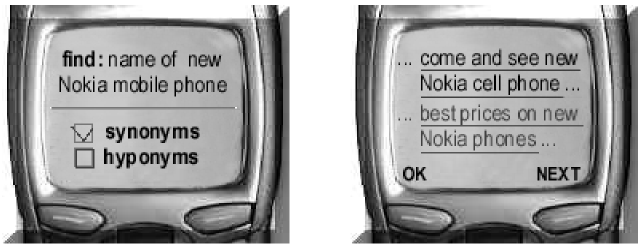

Figure 1: An example on a search interface. Highly relevant matches can be presented on small screen areas through the use of the relations in the texts.

Let us further assume that the relevant WEB/WAP pages only contain “cell phone” not “mobile phone”. If the search engine could include semantic relations, such as synonyms, and hyponyms, in the search then the right pages can be found and presented in the search results and by an appropriate weighting of results they could be presented to the user.

Somewhat paradoxically you can decrease the needed interface space by increasing the number of hits using synonyms and a good sort of the results. Sorting the results is not the bottleneck, by weighting the results in proportion to words, correct words and nearness we could get accurate first hit if we use hyponyms and other semantic relations in the search. Hyponyms, for instance, are available in semantic networks such as the on-line WordNet database (http://www.cogsci.princeton.edu/ ~wn/) along with many other relations. However, the technological progress, among other things, is pushing the terminology evolution and also global integration are moving too fast for any dictionary group, therefore many new words, relations, concepts and phrases are not present in dictionaries. One solution is to try to automatically extract relations such as hyponyms from existing texts. We present work aimed at extracting relations efficiently and with very little human intervention.

By automatically extracting semantically close words we will obtain more accurate search results putting less stress on the graphical user interface (GUI), especially on small screens.

3. RELATED WORK

Extraction of relations can be performed with varying degree of automation. One can for instance use various grammar formalisms (cf. [5]) and try to extract relations from texts by syntactically parsing them. However, grammars are brittle, which makes it very difficult to write complete grammars as people often are ungrammatical or use words and concepts which disqualifies the use of full grammatical rules. Another approach would be to try to find short phrases or patterns which often identifies a certain relation. For instance, the phrase “a/an X is the same as Y” might extract a synonym while “X is a kind of Y” or “X such as Y” might indicate a hyponym. Such patterns and phrases have been detected by hand in previous work by for instance Hearst [6] who used hand-coded patterns to acquire hyponyms and then used them to extend a database called WordNet [3]. The problem with hand-coded patterns is that it requires a lot of hard work and that it is difficult to find good coverage of useful patterns and to adapt to changes in vocabulary and phrasing. In our approach we use a machine learning technique to automatically induce feasible patterns/phrases for extraction of semantic relations. We are thus not only extracting new relations in unknown text, but we extract, induce or learn patterns for the relation extraction itself.

As machine learning technique we use genetic programming (GP) a relatively new general method for machine learning [1, 9]. GP is a very broad technique that previously has been used for text classification [11] and database datamining [4] but semantic relation extraction is not very well explored in this field.

4. METHOD

Below follows a brief introduction to the method used for the system, and also a description of the system.

4.1 Genetic Programming

We chose GP for our system since it is rare among “soft-computing” machine learning methods in its ability to accept symbolic input and return symbolic output in contrast to e.g. neural networks. GP is often used to breed computer programs as an instance of automatic programming. However, GP can be used to breed any structure that can be represented in computer memory and in our case we use GP to breed pattern used to extract relations. In the most common variants of GP, tree structures are evolved. Tree structures are used since it is easy to design genetic operators that ensure syntactic closure on tree structures, which is very desirable since it will prevent any occurrence of structures that are not well formed. The most common genetic operators in

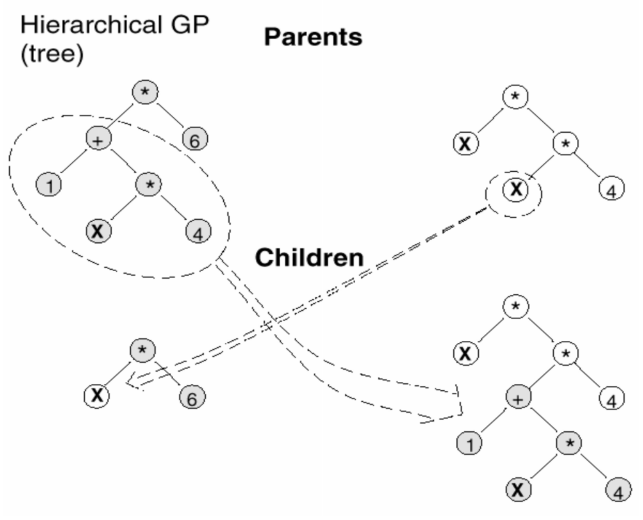

Figure 2: Crossover in genetic programming.

evolutionary algorithms, EAs, are mutation and crossover. Crossover (see Figure 2) is performed by exchanging two arbitrary subtrees in the two parent tree structures to obtain two offspring tree structures. Mutation is performed on a single individual often using a crossover with a individual that is not part of the population and built with a random function for just this single purpose. Another genetic operator called reproduction is used for making copies of especially competent individuals.

In GP, a population of solution candidates are evolved or bred according to a goodness criterion often called fitness function. The initial population consists of tree structures randomly generated from a set of functions and terminals. The best performing individuals (according to the goodness criterion) are selected for mating through the genetic operators to constitute a new generation. In this way new generations of successively more fit individual structures are evolved. The evolution ends when a certain criterion is met or when maximum number of evolved generations is reached.

4.2 System Implementation

The original GP systems were written in the LISP language for its string and meta processing capabilities, but most GP systems today are written in the C language for efficiency. We have chosen PROLOG (SICStus prolog, http://www.sics.se) as implementation language, since it is very well suited for natural language tasks and also because WordNet offers an additional lexical database (http://www.cogsci.princeton.edu/~wn/) implemented in PROLOG. This database is well suited for the kind of fitness calculation we want to perform.

4.2.1 The Architecture of the system

Our system is based on a steady-state GP algorithm (also known as the tournament selection model) and consists of five sequential steps, where the last four are iterated.

1. Initialization

2. Selection

3. Fitness evaluation

4. Tournament

5. Genetic operators

First the system initializes a population by building individuals out of functions and terminals into data tree structures. The evolution is started by selecting four individual structures that will participate in a tournament. The four selected tournament participants must first be evaluated by the fitness function to obtain a fitness value and the higher value obtained the better the individual will perform in the competition. The four individuals compete in pairs and the genetic operators will process the two winners to produce offspring. The three genetic operators used are crossover with a possibility of 60%, reproduction with a possibility of 40%, and one of the two offspring produced by the crossover or the reproduction will have a 40% possibility to mutate. The two offspring will take the two losers' place in the population and then evolution will go on until maximum number of tournaments has been reached.

It is important to make a distinction between a tournament and a generation (cf. [8]). When using tournament selection you need as many tournaments as the sum of the population divided by four, to make a new generation, e.g. if you have a population of 1000 individuals you need 250 tournaments to evolve a new generation called a steady-state generation.

4.2.2 Fitness evaluation using WordNet

The most important part of a GP system is the fitness function, which decides what individuals will be allowed to have offspring.

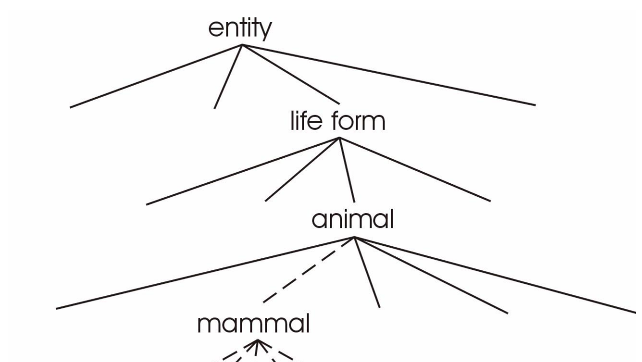

Figure 3: Part of a noun hierarchy.

At the moment the test system deals with the hyponym relation. The hyponym relation can be described as a subordinate relation between two nouns such as cow-mammal and this can be pronounced “the cow is a kind of mammal” or “a mammal such as cow”. This can graphically be viewed as a tree (see Figure 3) where the noun “mammal” is the parent node and all the hyponyms of mammal would be the node of the children. The hyponym relation is transitive and that gives us the possibility to arrange nouns into noun hierarchies. Another feature, given that you have access to a big enough noun hierarchy, is that words appear as many times as they have senses. Thus by finding a hyponym relation we can recognize what sense the word has in that specific occurrence.

We use a lexical database called WordNet [3] to measure the individuals' fitness. WordNet is an electronic lexical database used frequently by researchers in computational linguistics, and other related areas. The theories of psycholinguistics and human lexical memory have influenced the design of WordNet, resulting in that the four lexical categories used - nouns, verbs, adjectives, and adverbs - are organized into synonym sets, each representing one underlying lexical concept. The WordNet prolog database, used in our implementation, contains 15 different relations derived from these synonym sets.

An individual in our system picks out pairs of words. Different text predicates, for instance predicates defining order of words, are part of the tree structures of the bred patterns. In the initial population there are randomly constructed tree structures and any word pairs are likely to be output. The problem is to find a goodness criterion that will grade the individuals in this domain. We use the WordNet database for the fitness calculation and the principle behind the fitness is: if an individual processes a lot of word-pair of the right relations (according to WordNet) while giving very few false picks (also according to WordNet) then we are willing to accept output from that individual on word-pairs where WordNet has no information. In case those individuals does not produce proper hyponym pairs, the training needs to be enhanced.

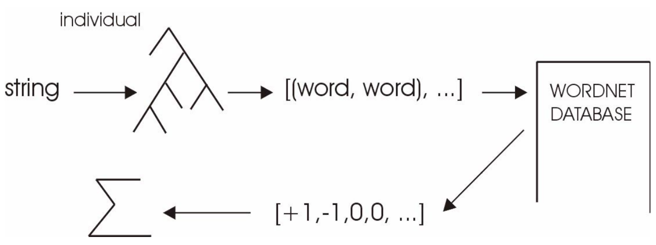

Figure 4: The fitness function.

If one noun pair matches an instance of a hyponym relation in WordNet it will be scored one point else if it matches an instance of another relation it will be scored minus one point. The last possibility is if no match is found, then the individual is given zero points.

All the scores, according to the rated pairs, for one individual are added and used as fitness value for this particular individual. When this is done with four individuals, the tournament can begin.

5. EXPERIMENT AND INITIAL RESULTS

The experiments where performed on pre-processed texts from the Web. Part of texts where selected where we knew that hyponyms existed and that they also could be found in WordNet. In our experiments we wanted to achieve a “proof-of concept” by evolving patterns which could extract hyponyms from Web-texts. Initial results (see Figure 5) confirm the feasibility of the method and we have evolved patterns that detect trivial types of hyponyms in natural language.

With a population of 1000 individuals we had the system evolve 1500 steady-state generations and the best performance came from the individual shown in Figure 5. It consists of the function

followed_by and the terminals **first**, **such**, **as**, and **last**. **first** matches anything that is followed by the word such and **last** matches anything following the word as. In this sentence, which is taken from the training set, *"By hardwood, I mean any broadleaf tree such as maple, almond, ash, alder, hickory, cherry, etc."* the individual will find one match and therefore produce one word-pair (tree, maple). In this case the individual gets one point for the match since the hyponym exists in WordNet. Clearly this individual does not capture all the hyponyms present in the sentence, but it proves that it is possible to evolve structures representing hyponyms.

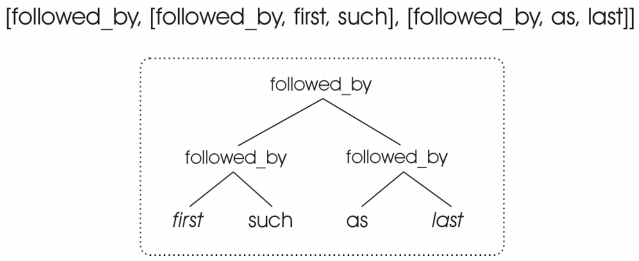
[followed_by, [followed_by, first, such], [followed_by, as, last]]

Figure 5: An individual evolved by the system, matching one simple pattern representing hyponyms.

6. FUTURE WORK

There are several important directions for future work and the most crucial one is to enable the system to build more complex individuals in order to cover more complex structures such as the sentence from our training set, showed above. Further directions are implementing other textual relations and evolving not only patterns for extraction but functions for confidence grading of output.

Also, the training time is extensive due to the programming language with several days of training on real world texts. In the experiment it took almost 90 hours to evolve such individuals as Figure 5 shows, although the training set was minimized and hyponym intense. If we implement parts of the system in C instead and use parallel computers the training time could be reduced.

7. CONCLUSION

We have used state of the art machine learning techniques (genetic programming) and proven that it is possible to automatically evolve patterns for relation extraction from Web-texts. The technique could be an important countermeasure against information overload and also be used to enable feasible user interfaces on small screens.

8. REFERENCES

1. Banzhaf, W., Nordin, P. Keller, R. E., and Francone, F. D. *Genetic Programming: An Introduction On the Automatic Evolution of Computer Programs and Its Applications*. Morgan Kaufmann, Germany, 1997.
2. Broder, A., Glassman, S., Manasse, M., and Zweig, G. Syntactic clustering of the Web. In *Proceedings of the Sixth International World Wide Web Conference*, pp. 391-404, 1997.
3. Fellbaum, C. *WordNet - An Electronic Lexical Database*. MIT Press, Cambridge MA, 1998.
4. Freitas, A. A. A genetic programming framework for two data mining tasks: Classification and generalized rule induction. In *Genetic Programming 1997: Proceedings of the Second Annual Conference*, pp. 96-101, Stanford University, CA, USA, 1997.
5. Gazdar, G., and Mellish, C. *Natural Language Processing in PROLOG: An Introduction to Computational Linguistics*. Addison-Wesley Publishing Company, Wokingham, England, 1994.
6. Hearst, M. Automatic Acquistion of Hyponyms from Large Text Corpora. In *Proceedings of the Fourteenth International Conference on Computational Linguistics*, Nantes, France, 1992.
7. Hearst, M., and Pedersen, J. Reexamining the Cluster Hypothesis: Scatter/Gather on Retrieval Results, In *Proceedings of the 19th Annual International ACM/SIGIR Conference*, Zurich, 1996.
8. Kinnear, Jr., K. E. Generality and difficulty in genetic programming: Evolving a sort. In *Proceedings of the 5th International Conference on Genetic Algorithms, ICGA-93*, pp. 287-294, Urbana-Champaign, IL, USA, July 1993.
9. Koza, J. R. *Genetic Programming: On the Programming of Computers by Means of Natural Selection*. Cambridge, MA. The MIT Press, 1992.
10. Nelson, M. R. We Have the Information You Want, But Getting It Will Cost You: Being Held Hostage by Information Overload. *Crossroads* 4-1: Fall 1997.
11. Masand, B. Optimizing Confidence of Text classification by Evolution of Symbolic Expressions. In *Advances in Genetic Programming*, MIT Press, USA, 1994.
12. Wireless Application Protocol Forum. Available at http://www.wapforum.org, 1999-07-03.
13. Zamir, O. and Etzioni, O. Web Document Clustering: A Feasibility Demonstration. In *21st International ACM SIGIR Conference on Research and Development in Information Retrieval*, pp. 46-54, Melbourne, Australia, 1998.