Interactive learning of syntax-based natural language queries

Henrik Björklund

Umeå University
901 87 Umeå
henrikb@cs.umu.se

Abstract

We propose an approach to syntax-driven text search based on tree transducers. The main idea is to use interactive learning to let users design syntax-based queries without having to know anything about either natural language syntax or tree transducers. This method has previously been successfully applied in the area of web wrapper generation.

1. Introduction

We all use keyword-driven text search on a daily basis and it has become quite refined. Still, for some tasks, it is not very practical. Imagine, for example, that you want to pose queries such as the following.

- How frequently does author A use dependent clauses?
- Find all examples of sentences in text T where the word "well" is used as an adjective. (As in, e.g., "she looks well".)
- In which contexts is the Swedish verb "lämna" used as intransitively?

For such a search to be efficient, it has to be *syntax-driven*.

We propose a system for just such queries. In order to make it useful even for non-linguists the inner workings of the system will be hidden to the user. Assume that the user wants to search for (parts of) sentences that have property P. She would then load a text into the system. This text will internally be parsed and annotated with syntax trees. The user would then highlight a number of sentences that have property P. Based on this, the system produces a tree transducer that generalizes the given set of sentences, searches the text and presents the user with all sentences that are selected by the transducer. Most likely, the system will not find all sentences with property P and will, additionally, select some sentences that do not have property P. The user can now tell the system which selected sentences are fase positives and also provide it with additional sentences that have property P. Based on this, the tree transducer is refined and the process is iterated until the user is satisfied with the result. In the end, the user will have trained the transducer to recognise property P or at least something very close to P. This transducer can now be used to search for sentences with property P in other texts.

In other words, the system we propose is based on *interactive learning* of *node-selecting tree transducers*. This approach has previously been used by Carme et al. (2007) in the area of web wrapper generation.

The goal of the current work is to evaluate the feasibility and usefulness of an interactive learning approach based on tree transducers in NLP. One of the main questions is whether the encouraging results achieved in the area of web wrapper induction can be transferred to search in less structured domains, such as syntax trees and what adaptations are necessary. For instance, in the web wrapper setting, unranked trees are the natural choice, while in the NLP setting, it is unclear whether ranked or unranked trees are more suitable in practice.

The intended users of the proposed system are scholars in the humanities and social sciences, as well as language educators and students. Such users can be expected to be interested in, and know something about, grammatical phenomena, but not necessarily familiar with the grammatical frameworks used by linguists and computational linguists.

2. Interactive learning

Interactive learning is an approach to query generation that aims at combining automated learning with the knowledge of a domain expert (the user in the above example scenario). The goal is to produce queries of high quality while using the knowledge of the domain expert as efficiently as possible, since human time is normally a very expensive resource. The expert annotates some data, but not necessarily very much. The annotated data is used to learn a query. The expert then views the results of running the query on a test data set and gives additional input to the learning system. This process is iterated until the expert is satisfied.

The concept of iterative learning has been studied extensively by, e.g., Small (2009), whose thesis looks at how interactive learning can be applied to a number of different learning tasks, focusing on NLP applications. It concludes that interactive learning can substantially assist a domain expert in encoding world knowledge into the learning process. Importantly, Small writes that "interactive encoding of modelling information through feature engineering often leads to better performance than simply acquiring additional labeled data." He demonstrates the effectiveness of interactive learning on a semantic role labeling and an information extraction task.

3. Node-selecting tree transducers

Syntactic information that has been extracted from natural language is generally tree-shaped. Whether phrase structure parsers or dependency parsers are used, the resulting structures are trees, see, e.g., (Klein and Manning, 2003; de Marneffe et al., 2006; Kübler et al., 2009).

There is already an extensive literature on machine learning for regular tree languages, see, e.g., (Drewes and Högberg, 2003; Drewes, 2009). Much less has been written, on the topic of interactive learning for these structures but there are some notable exceptions, primarily from the field of web wrapper induction.

In an article on *Interactive learning of node selecting tree transducers*, Carme et al. (2007) tackle the problem of learning web wrappers. Their approach is to limit the power of the query language. In this setting, a node-selecting tree transducer is a deterministic finite unranked tree automaton except that it marks all nodes that have been visited in an accepting state. The subtrees of such nodes are then selected and extracted.

The authors present two algorithms, one for learning a node-selecting transducer from fully annotated data and one for learning interactively from partially annotated data. By fully annotated data, they mean a set of HTML documents in which all elements that should be selected by the extractor are marked (and no other elements are marked). Given such data, an RPNI-style[1] algorithm is employed to infer a deterministic tree automaton. The standard RPNI algorithm is modified slightly to make sure that the merging of states is conditional not only on preservation of determinacy, but also on the preservation of functionality. Internally, the system works with *stepwise tree automata* (Carme et al., 2004). These are automata that work on the so-called curried binary encoding of unranked trees. The reason for this is that it is difficult to find a suitable notion of bottom-up determinism for unranked tree automata, particularly one that allows for unique minimisation (Martens and Niehren, 2007). In a setting where ranked trees are sufficient, standard deterministic tree automata can be used instead.

The authors also present an algorithm in the spirit of Angluin's MAT-learning (Angluin, 1987) for learning the transducer from partially annotated data. It introduces so-called *Correct Labeling Queries*. The learner presents the teacher with a tree in which some nodes are selected. The teacher either answers that the annotation is correct or points to a node that is selected although it shouldn't be or a node that is not selected although it should be. There are also equivalence queries similar to those of Angluin.

Rather than working with tables, as the original MAT-learning algorithm (Angluin, 1987) and previous variants for tree languages (Sakakibara, 1990; Drewes and Högberg, 2003), the algorithm gradually builds a selection of fully annotated trees and uses the previously discussed RPNI algorithm to construct hypothesis transducers.

The algorithm is implemented in the SQUIRREL information extraction system. Here, the end-user can design queries using a graphical user interface.

4. Current and future work

The usefulness of a system such as the proposed is best evaluated empirically. We are therefore currently in the process of building a prototype implementation of the system. It is based on using a phrase structure parser, ranked trees and deterministic bottom-up tree automata.

Once this prototype is in place, we will make a first evaluation of its general usefulness. We will also try out variations. In particular, the current architecture works with ranked trees, which may not be the best choice, since trees representing syntactic structure can in most models have unlimited branching. If this turns out to be a problem, we plan on switching to unranked trees and make use of stepwise tree automata. This was used for HTML trees by Carme et al. (2007).

Another question is to which extent the actual words should be taken into account. A user may want to search for phrases of a certain structure, but only those where the main verb is "run". The current architecture only allows us to take all words or no words at all into account. We plan on implementing a feature that lets the user mark, in each example sentence, the words that are of interest.

We can also imagine a scenario where the user wants to search for all sentences of a certain structure that contain, e.g., a color. For this to be possible, the system will have to be able to group words together. In a later version, we plan on integrating ontologies into the system, making it possible, after seeing example sentences that mention, e.g., "green" and "blue", to conclude that a sentence containing the word "red" may also be of interest.

[1] Regular Positive and Negative Inference

Acknowledgements

We gratefully acknowledge the financial support from the Swedish Research Council grant 621-2011-6080 and the EU FP7 MICO (Media in Context) project. The presentation of node-selecting tree transducers above is partially based on the authors own contributions to (Aichroth et al., 2014).

References

Patrick Aichroth, Johanna Björklund, Florian Stegmaier, Thomas Kurz, and Grant Miller, editors, 2014. *State of the Art in Cross-Media Analysis, Metadata Publishing, Querying and Recommendations*, chapter 2.3. http://www.mico-project.eu.

Dana Angluin. 1987. Learning regular sets from queries and counterexamples. *Information and Computation*, 75(2):87–106.

Julien Carme, Joachim Niehren, and Marc Tommasi. 2004. Querying unranked trees with stepwise tree automata. In Vincent van Oostrom, editor, *Proc. 19th International Conference on Rewriting Techniques and Applications*, volume 3091 of *Lecture Notes in Computer Science*, pages 105–118. Springer.

Julien Carme, Rémi Gilleron, Aurélien Lemay, and Joachim Niehren. 2007. Interactive learning of node selecting tree transducer. *Machine Learning*, 66(1):33–67.

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parsers from phrase strucutre parses. In *Proc. 5th International Conference on Language Resources and Evaluation (LREC 2006)*, pages 449–454.

Frank Drewes and Johanna Högberg. 2003. Learning a regular tree language from a teacher. In Z. Ésik and

Z. Fülöp, editors, *Proc. 7th International Conference on Developments in Language Theory (DLT'03)*, volume 2710 of *Lecture Notes in Computer Science*, pages 279–291. Springer.

Frank Drewes. 2009. MAT learners for recognizable tree languages and tree series. *Acta Cybernetica*, 19(2):249–274.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *Proc. 41st Annual Meeting of the Association for Computational Linguistics (ACL'03) – Volume 1*, pages 423–430. The Association for Computational Linguistics.

Sandra Kübler, Ryan McDonald, and Joakim Nivre. 2009. *Dependency Parsing*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool.

Wim Martens and Joachim Niehren. 2007. On the minimization of XML Schemas and tree automata for unranked trees. *Journal of Computer and System Sciences*, 73(4):550–583.

Yasubumi Sakakibara. 1990. Learning context-free grammars from structural data in polynomial time. *Theoretical Computer Science*, 76:223–242.

Kevin Small. 2009. *Interactive Learning Protocols for Natural Language Applications*. Ph.D. thesis, University of Illinois.