Case based Arabic Morphological Analysis with Compositional Non-deterministic Automata

Abstract

Natural language processing uses morphological analysis to abstract and annotate text. Often times, and in particular in the context of the Arabic language, annotations resulting from an ongoing branch of morphological analysis may not be appropriate for the case study at hand. In this paper we present Sarf, a case based morphological analyser for Arabic text. Sarf allows a case-based controller to control and refine the state of the analysis on the fly. It uses compositional non-deterministic finite state machines to analyse a stream of text. Each alive machine corresponds to a valid analysis. Sarf uses concatenative analysis based on recursive affixes to better preserve part of speech information. We automated the analysis of three books of Islamic narrations (hadith) using Sarf where we abstracted each book into a vector of complex structures. Our results show high accuracy and efficiency compared to state of the art analyzers.

1 Introduction

Automated analysis of Arabic data sets, including texts, publications, records and digital media is essential with the huge digital Arabic content available nowadays. Morphological analysis is key in current automated analysis techniques for Arabic text. Current morphological analyzers (Al-Sughaiyer and Al-Kharashi, 2004) use concatenative analysis to analyze an Arabic word and consider its internal structure composed of several *morphemes*. A morpheme can be a *stem*, or an *affix*. An affix can be a *prefix, suffix,* or an *infix*. The analysis of one word may lead to several possible morphological solutions. For instance the word أَحمَده may have two valid morphological analyses where in the first أَ is a prefix and the word means "I praise him", and in the second أَ is part of the stem أَحمَد (a proper noun) and the word means "his Ahmad".

The solutions suffer from accuracy due to inherent difficulties of morphological analysis of the Arabic language. For example, it is common practice to write Arabic text without short vowels. Arabic letters can also have up to four different shapes corresponding to their position in a word, i.e, begining of a word, middle of a word, end of word, and separated. This allows two consecutive words such as الىالمدرسة to be visually recognizable as two separate words الى and المدرسة without the need of a delimiter such as a space in between. This specifically happens when the first word ends with a non-connecting letter and is referred to as the "run-o" words problem (Buckwalter, 2004). Such forms occur in text and greatly increase the difficulty of tokenization.

Current morphological analyzers such as Buckwalter (2002), Xerox (Beesley, 2001), SAMA (Seth Kulick and Maamouri, 2010), and ElixirFM (Smrž, 2007) exist. They take as input

white space delimited tokens (Seth Kulick and Maamouri, 2010), consider them as words, and enumerate all possible solutions. This exhaustive enumeration may hurt performance and may not be necessary or appropriate in some case studies as noted in (et al., 2010). Moreover, a white space delimited token may have more than one word. Other morphological analyzers such as Amira (Mona Diab and Jurafsky, 2007; Benajiba et al., 2008), MAGEAD (Habash et al., 2005), and MADA+TOKAN (Habash et al., 2009) also use machine learning and support vector machines (SVM) to enhance the accuracy of the morphological analysis at the expense of perfomance.

We hypothesise that many NLP case studies need the morphological analyser to answer simple queries that do not need high complexity and accuracy at the low morphological analysis level and that can often compensate for tolerable inaccuracy at higher level. For example, if the query is interested in proper names and the prefix in question in the analysis can only connect to a verb, we can provide an answer without completing the analysis. We find evidence to our hypothesis in (et al., 2010) where the addition of a new corpus to the Arabic Tree Bank (Maamouri and Bies, 2004) with features demanding resolution at an abstraction level higher than the text itself led to a refined analyzer (Seth Kulick and Maamouri, 2010). We also find evidence in (Habash and Sadat, 2006) that different types of morphological analyses behave differently on the same case study. We strongly beleive and we find evidence in our results that a case-based controller intervening at every decision is necessary to guide, use, prune, and refine the morphological analysis.

1.1 Sarf

In this paper, we present Sarf, a *novel case-based efficient morphological analyzer* that uses parallel compositional non-deterministic automata driven by a case based controller. Each alive machine in Sarf represents a valid morphological analysis. Sarf keeps alive all possible analyses of the text and gives opportunity to the controller to intervene at every single input character. The decisions of the case-based controller can thus prune false positives early in the run. Each alive machine in Sarf takes as input one character at a time from a text stream. Sarf does not assume that the word at hand is a token and performs tokenization on the fly based on morphological correctness. To out knowledge, this is the first morphological analyzer that handles the "run-on" words problem.

Sarf uses a *recursive* concatenative analysis with a novel refinement. Sarf introduces recursive affixes in order to retain better part of speech information and enhance the efficiency of affix matching.

We validate our hypothesis using a case study from the Islamic literature. A *hadith* is a narration related to the prophet Mohamad through a *sanad* or a sequence of narrators. The authentication of a hadith higly depends on the credibility of each of the narrators as reported in separate biography books. In this paper we consider the problem of automatically segmenting a hadith book into narrations, then segmenting each narration into its content or *matn* and its sanad. We also would like to partition the sanad accurately into the separate narrators so that we can later look each one of them up in the biography books.

The hadith case study is interesting for Sarf since the controller considers most of the analyses where we have a concentration of proper names in the text, and ignores most of the analyses as long as they are valid where we do not have proper names. The controller is also interested in a small subset of words that relate persons to each other such as بن or "the sun of" or words that mean narrate such as قال or "said". With Sarf, we successfully automated the analysis of three books of Islamic narrations (**?**; **?**; **?**) and extracted from each one of them a vector of a three level deep complex structure.

We make the following key contributions.

1. **Case-based analysis:** We deisgned and implemented a case-based morphological analyzer where a case-based controller machine controls and guides the analysis.

2. **Exhaustive analysis:** Since we have a case-based controller that can decide on the fly whether an analysis is accepted or not, we can afford to keep all valid analyses modulo those pruned by the case controller. We do that using non-deterministic FSAs.

3. **Recursive affixes:** We refine the concatenative analysis of Buckwalter (2002) and use recursive affixes. This allowed Sarf to retain better compositional part of speech tags.

4. **Hadith case study:** We evaluated Sarf using the hadith case study. We were able to efficiently extract the sequence of narrators into a complex data structure with three levels of hierarchy with high accuracy.

2 Background

3 Related Work

Many morphological analyzers exist (Al-Sughaiyer and Al-Kharashi, 2004). Some use specialized part of speech tag sets (Khoja, 2001; **?**).

Buckwalter: used lexicon, added proper nouns, word based, recursive affixes

MAGEAD

Beeseley: many compiled FSAs compared to compositions of three FSAs

SAMA: word based, treebank specific,

MADA+TOKAN: no controller, we do not need a tokenizer

AMIRA: machine learning...

ElixirFM:

4 Sarf

4.1 Recursive Affixes

4.2 Motivating Example

To help illustrate the strength of the method we imploy, we clarify it by walking through an example explaining the stages that lead to morphological analysis in SARF. Figure 1 shows a partial snapshot of the FSM's needed to analyze the following words: وسيلعبها, سيلبسهما and اللّاعبون.

4.3 Affix Linear FSM

4.4 Stem Linear FSM

4.5 Non-deterministic Composition of FSMs

5 Islamic Literature Case Study

5.1 Hadith Segmentation

5.2 Chain of Narrators

5.3 Controller

6 Results

6.1 Efficiency Comparison Against Buckwalter and SAMA

6.2 Case Study Accuracy and Efficiency Results

7 Future Work

Acknowledgments

References

Imad A. Al-Sughaiyer and Ibrahim A. Al-Kharashi. 2004. Arabic morphological analysis techniques: a comprehensive survey. *American Society for Information Science and Technology*, 55(3):189–213.

Keneth R. Beesley. 2001. Finite-state morphological analysis and generation of arabic at xerox research: Status and plans. In *Workshop Proceedings on Arabic Language Processing: Status and Prospects*, pages 1–8, Toulouse, France.

Yassine Benajiba, Mona Diab, and Paolo Rosso. 2008. Arabic named entity recognition using optimized feature sets. In *EMNLP '08: Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 284–293, Morristown, NJ, USA.

Mohamed Maamouri et al.. 2010. From speech to trees: Applying treebank annotation to arabic broadcast news. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May.

Timothy Buckwalter. 2002. Buckwalter arabic morphological analyzer version 1.0. Technical report, LDC catalog number LDC2002L49.

Timothy Buckwalter. 2004. Issues in arabic orthography and morphology analysis. In *Semitic '04: Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages*, pages 31–34, Morristown, NJ, USA.

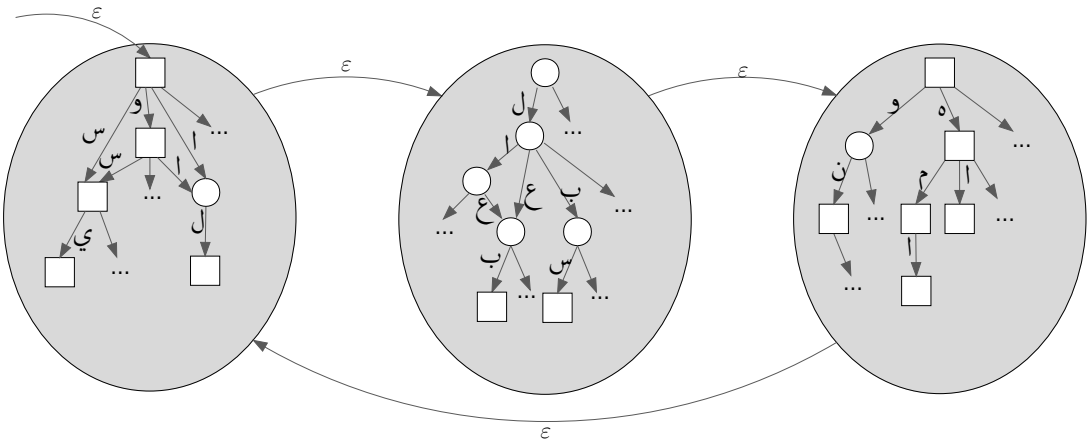

Figure 1: Example FSM.

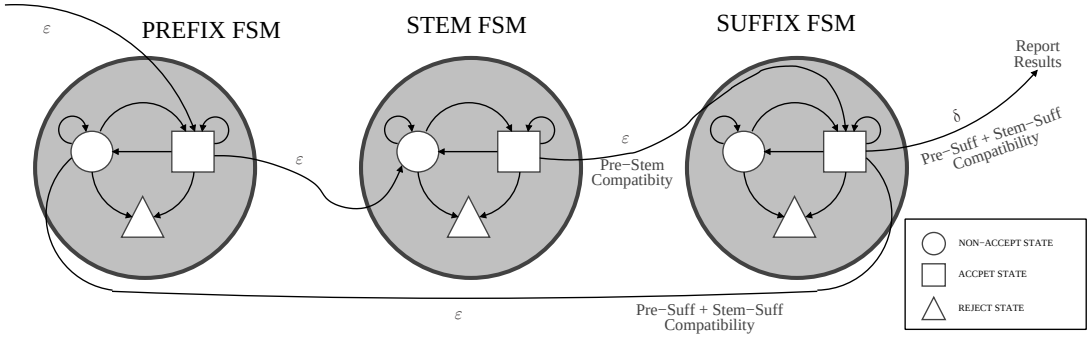

Figure 2: Abstract FSM.

Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the North American Chapter of the Association for Computational Linguistics(NAACL)*, pages 49–52.

Nizar Habash, Owen Rambow, and George Kiraz. 2005. Morphological analysis and generation for arabic dialects. In *Semitic '05: Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pages 17–24, Morristown, NJ, USA. Association for Computational Linguistics.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009. Mada+tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In Khalid Choukri and Bente Maegaard, editors, *Proceedings of the Second International Conference on Arabic Language Resources and Tools*, Cairo, Egypt, April. The MEDAR Consortium.

Shereen Khoja. 2001. Apt: Arabic part of speech tagger. In *NAACL student research workshop*.

Mohamed Maamouri and Ann Bies. 2004. Developing an arabic treebank: methods, guidelines, procedures, and tools. In *Semitic '04: Proceedings of the Workshop on Computational Approaches to Arabic Script-based Languages*, pages 2–9. Association for Computational Linguistics.

Kadri Hacioglu Mona Diab and Daniel Jurafsky, 2007. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, chapter 9, pages 159–179. Springer.

Ann Bies Seth Kulick and Mohamed Maamouri. 2010. Consistent and flexible integration of morphological annotation in the arabic treebank. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC'10)*, Valletta, Malta, May.

Otakar Smrž. 2007. Elixirfm: implementation of functional arabic morphology. In *Semitic '07: Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages*, pages 1–8, Prague, Czech Republic. Association for Computational Linguistics.