Second Generation AMIRA Tools for Arabic Processing: Fast and Robust Tokenization, POS tagging, and Base Phrase Chunking

Mona T. Diab

Center for Computational Learning Systems,
Columbia University
475 Riverside Drive, Suite 850,
New York, NY 10115
mdiab@ccls.columbia.edu

Abstract

In this paper, we address the problem of processing Modern Standard Arabic. We present the second generation of tools that process Arabic (AMIRA). AMIRA is a successor suite to the ASVMTools. The AMIRA toolkit includes a clitic tokenizer (TOK), part of speech tagger (POS) and base phrase chunker (BPC) - shallow syntactic parser. The technology of AMIRA is based on supervised learning with no explicit dependence on explicit modeling or knowledge of deep morphology. AMIRA is based on using a unified framework casting each of the component problems as a classification task. The underlying technology employs Support Vector Machines in a sequence modeling framework using the YAMCHA toolkit. The system is very fast and robust and allows for a number of variable user settings depending on the disambiguation granularity. The AMIRA toolkit has been widely used for different NLP (MT, IE, IR, NER, etc.) applications due to its speed and high performance.

1. Introduction

The Arabic language raises many challenges for natural language processing (NLP). First, Arabic is a morphologically complex language. Second, Arabic is written with optional diacritics that would primarily specify short vowels. Finally, Arabic, in its current use, is a collection of varieties (aka dialects). The primary variety, Modern Standard Arabic (MSA), is used in formal settings yet it is no ones native tongue. The various dialects are commonly used informally and with increasing presence on the web. The different Arabic dialects display a wide variation in morphology, syntax and their lexicon akin to the situation with Romance languages. The dialects, for primarily political and historical reasons have no common standard orthography. As a result, spelling inconsistencies abound leading to high sparsity and high ambiguity for NLP systems.

In this paper, we address the problem of processing MSA. We present the second generation of tools that process Arabic (AMIRA). AMIRA is a successor suite to the ASVMTools (Diab et al., 2007). The AMIRA toolkit includes a clitic tokenizer (TOK), part of speech tagger (POS) and base phrase chunker (BPC) - shallow syntactic parser. We added more functionality and flexibility to the AMIRA toolkit over the previous versions as it can take in text in different encodings and produce the output in the input encoding, accordingly, the user may submit UTF8 encoding and the processed text output will be in UTF8.

The technology of AMIRA is based on supervised learning with no explicit dependence on knowledge of deep morphology, hence, in contrast to systems such as MADA (Habash and Rambow, 2005; Roth et al., 2008), it relies on surface data to learn generalizations. In general the tools are based on using a unified framework casting each of the component problems as a classification problem. The underlying technology uses Support Vector Machines (deterministic classification) in a sequence modeling framework using the YAMCHA toolkit.[1] The system is very fast and robust and allows for a limited number of variable user settings depending on the disambiguation granularity. The AMIRA tools have been widely used for different NLP applications due to its speed and high performance. It has been used for preprocessing for the purposes of Machine Translation, Information Retrieval, Parsing, Named Entity Recognition, Information Extraction, etc. The different components of the tool suite could be invoked together, taking raw text in any encoding and producing clitic tokenized, POS tagged and/or chunked data, or the different components could be applied directly on some given texts,

[1]http://chasen.org/ taku/software/YamCha/

for example POS tagging could be applied on raw text directly without necessarily the need to explicitly invoke tokenization. It is worth noting that the AMIRA tools are trained on MSA, however, we have adapted them on a shallow level to handle dialectal Arabic. We contrasted the adapted version of the tools against training the tools on MSA text and applying them to dialectal text without any tuning of the parameters.
Since this paper is more of a system description, we will refer the reader to our previous relevant publications that discuss the details of the different components. This paper is organized as follows: Section 2. reviews tokenization within the context of the AMIRA 2.0 suite of tools; Section 3. presents the POS tagging as it is implemented in the tools; finally, Section 4. discusses Base Phrase Chunking.

2. Tokenization

The Arabic language is a morphologically rich language which leads to significant challenges in processing due to the extensive variability in surface forms. Therefore, reducing the sparsity is a necessary step especially when there is a relatively limited amount of data. Tokenization is one such process of normalization with the goal of reducing the sparsity of observed forms.
The task of tokenization can be defined in several ways as expressed by (Sadat and Habash, 2006; Habash and Sadat, 2006). In this work, we are adhering to the definition of clitic tokenization as characterized by the Linguistic Data Consortium (LDC). The LDC defines a commonly accepted standard of clitic tokenization: separating conjunctions, affixival prepositions and pronouns, future marker clitics, and definite articles. For example, given a word such as *wbHsnthm*,[2] and by their virtue, the output could be *w b Hsnt hm* or *w+b+Hsnt#hm*. The former case increases the token count in the data while the latter preserves the input surface token count. In the previous version of the ASVMTools, the user was not given the choice of tokenization scheme. In the current version of AMIRA 2.0, the user can choose among different choices of:

Token Count: either maintaining the surface token count or not;

Type of Clitic Tokenization: the user could choose not to separate the definite article, or the future marker, for example.

[2]We present all the Arabic data in this paper using the Buckwalter transliteration schema.

The modeling of TOK in AMIRA 2.0 is exactly the same process as that used in ASVMTools and described in detail in (Diab et al., 2007). We will review it briefly here. In modeling TOK, we apply a chunking scheme on the character level casting the problem as a chunk boundary and chunk classification problem. We use an IOB annotation scheme, every character in our data (including punctuation) is annotated as: inside a chunk (I), beginning of a chunk (B) or outside a chunk (O). For the I and B tags, we have five possible classes: Prefix 1, Prefix 2, Prefix 3, Word, Suffix. This leads to a total of 11 classes in the data: O, B-PRE1, I-PRE2, B-PRE2, I-PRE2, B-PRE3, I-PRE3, B-WORD, I-WORD, B-SUFF, I-SUFF.
Once the test item is chunked into a series of tokens indicating whether they are of prefix clitic types, stems, or suffixes. Our goal is to produce text where all the word tokens (modulo the clitics) are bona fide words in a dictionary and increasing the level of regularity in the text as a means of normalization. Hence, we apply some heuristics to reverse the effect of morphotactics such as the loss of *A* in the definite article *Al* when in the context of proclitic preposition *l* meaning *for*. For example the input word *wllblAd*, **and for the countries**, is clitic tokenized as *w+l+l+blAd* in the system output from the classification, then a post hoc fix is applied to ensure the consistency of the output, hence the final output is rendered *w+l+Al+blAd*. Most of these morphotactic restorations are deterministic rendering the application of the post hoc rules straightforward. However, there exist several cases where the morphotactics are not deterministic such as the feminine marker, alef maqsuura, and the word final hamzas. In AMIRA 2, we handle the former two cases.

a) Taa Marbuuta: the word final feminine marker on nominals indicating syntactic gender, `taa marbuuta` or *p* in the Buckwalter transliteration scheme. The word final *p* feminine marker alternates with a *t* when an enclitic suffix is added to a word. This alternation creates a problem for POS tagging later on since the suffix is typically a pronoun that could either be a possessive pronoun or an object pronoun where the pronouns look the same. Hence, the stem ending with a *t* could either be a verb or a noun. Therefore, in our example *w+b+Hsnt#hm* above, *Hsnt* is an underlying *p* word finally as in *Hsnp* which is a noun as opposed the verb *Hsnt*.

b) Alef Maqsuura: The Alef Maqsuura word finally alternates between a *y* and an *A*. This distinction is a lexical distinction and it relates to the underlying root

of the word. Even though this does not change the POS tagging for the words, yet it renders the tokens more consistent.
Since these are non deterministic features, we apply another layer of learning to the problem of classifying word final letters *t* as either to remain regular *t* in the case of taa marbuuta, or the *A* is convertible to a *Y* alef maqsuura or not. Accordingly the total TOK process results in valid surface words. At this stage, the AMIRA system does not handle inflectional morphology. The tokenization system has an F score measure of 99.2%.

3. POS Tagging

The POS tagging system optionally produces the PATB standard tag set of 25 tags (RTS) as well as an extended tag set (ERTS) that encodes some overt morphological markers including gender, number and definiteness (Diab, 2007b). ERTS has 72 tags. In RTS nouns maybe expressed as NN or NNS indicating number. In ERTS, nouns could be represented using definiteness, and gender in addition to the default number characterization. In fact we add the dual to the number repertoire. We adopt an SVM based classification approach using character n-grams as features in our sequence models. Both POS taggers with their different POS tag sets perform at over 96% accuracy, for ERTS is 96.13% and RTS 96.15%. This suggests that our choice of information to tag explicitly in the ERTS tag set reflects a natural division in the syntactic space. In (Diab, 2007b; Diab, 2007a), we show that using ERTS improves the quality of higher up processing such as Base Phrase Chunking. In either case, the input to the tagger could be raw text or clitic tokenized text. The user has the flexibility to request tokenized or non tokenized POS tagged output.

4. Base Phrase Chunking

Base phrase chunking is the process by which a sequence of adjacent words are grouped together to form syntactic phrases such as NPs and VPs. An English example of base phrases would be *[I]*$_{NP}$ *[would eat]*$_{VP}$ *[red luscious apples]*$_{NP}$ *[on Sundays]*$_{PP}$. BPC is the first step towards shallow syntactic parsing. Many high end applications such as information extraction and semantic role labeling in English have been proven to benefit tremendously from BPC at a relatively low loss in performance when compared to the use of deep syntactic parsing.
In the current version of AMIRA, the BPC module produces the longest possible base phrases with not much internal recursion. The internal recursion is done as a deterministic post process. We have modified the BPC rules to be more appropriate for the Arabic language (Diab, 2007a). 9 types of chunked phrases are recognized using a phrase BIO tagging scheme (described earlier); Inside *I* a phrase, Outside *O* a phrase, and Beginning *B* of a phrase. The 9 chunk phrases identified for Arabic are *ADJP, ADVP, CONJP, PP, PRT, NP, SBAR, INTJ, VP*. Thus the task is a one of 19 classification task (since there are I and B tags for each chunk phrase type, and a single O tag). The training data is derived from the `Arabic Treebank` using the `ChunkLink` software.[3] `ChunkLink` flattens the tree to a sequence of base (non-recursive) phrase chunks with their BIO labels. For example, a token occurring at the beginning of a noun phrase is labeled as *B-NP*.
At this stage the system internally uses ERTS POS tag set however if the user requested the RTS as the POS tag set, we have an internal mapping process from ERTS to the RTS POS tagset. For the BPC system, we use both POS tags and character n-grams as features. We adopt an IOB annotation scheme. Our approach uses a sequence model over SVMs. The AMIRA BPC component is very fast and extremely accurate. The system yields an F1 measure (harmonic mean) of 96.33%. Different research groups have shown the utility of BPC for different applications such as Machine Translation and information extraction specifically in Named Entity Recognition (NER) (Benajiba et al., 2008; Benajiba et al., 2009).

5. Acknowledgments

The author would like to acknowledge the partial support for this work by the DARPA GALE project. Also we would like to thank three anonymous reviewers for their useful comments.

6. References

Yassine Benajiba, Mona Diab, and Paolo Rosso. 2008. Arabic named entity recognition using optimized feature sets. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 284–293, Honolulu, Hawaii, October. Association for Computational Linguistics.

Yassine Benajiba, Mona Diab, and Paolo Rosso. 2009. Arabic named entity recognition: A feature-driven study. July.

[3]http://ilk.uvt.nl/ sabine/chunklink

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky, 2007. *Arabic Computational Morphology: Knowledge-based and Empirical Methods*, chapter Automated Methods for Processing Arabic Text: From Tokenization to Base Phrase Chunking. Kluwer/springer edition.

Mona Diab. 2007a. Improved arabic base phrase chunking with a new enriched pos tag set. In *Proceedings of the 2007 Workshop on Computational Approaches to Semitic Languages: Common Issues and Resources*, pages 89–96, Prague, Czech Republic, June. Association for Computational Linguistics.

Mona Diab. 2007b. Towards an optimal pos tag set for modern standard arabic processing. In *Proceedings of Recent Advances in Natural Language Processing (RANLP)*, Borovets, Bulgaria.

Nizar Habash and Owen Rambow. 2005. Arabic tokenization, part-of-speech tagging and morphological disambiguation in one fell swoop. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 573–580, Ann Arbor, Michigan, June. Association for Computational Linguistics.

Nizar Habash and Fatiha Sadat. 2006. Arabic preprocessing schemes for statistical machine translation. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, pages 49–52, New York City, USA, June. Association for Computational Linguistics.

Ryan Roth, Owen Rambow, Nizar Habash, Mona Diab, and Cynthia Rudin. 2008. Arabic morphological tagging, diacritization, and lemmatization using lexeme models and feature ranking. In *ACL 2008: The Conference of the Association for Computational Linguistics; Companion Volume, Short Papers*, Columbus, Ohio, June. Association for Computational Linguistics.

Fatiha Sadat and Nizar Habash. 2006. Combination of arabic preprocessing schemes for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 1–8, Sydney, Australia, July. Association for Computational Linguistics.