Part of Speech Tagger for Tunisian Arabic

Karen McNeil
Georgetown University, Washington, D.C.
km1542@georgetown.edu

Abstract

This paper presents a comparison of several different part-of-speech taggers trained on a hand-annotated Tunisian Arabic sample of 6,000 words in 450 sentences. Despite the small size of the annotated corpus, the Trigram Tagger and Brill Tagger performed nearly as well as a custom rule-based tagger. This work not only contributes a new resource to Tunisian Arabic—a severely under-resourced language—but also provides insight for developing materials for other under-resourced languages.

1 Introduction

One of the main goals of my work developing the Tunisian Arabic corpus, Tunisiya (McNeil, 2019) was enabling the production of a bidirectional Tunisian-English dictionary. Modern dictionaries are produced using large corpora to analyse usage examples in context. To be able to see the different senses and constructions each word is used in, a word sketch (Kilgarriff and Tugwell, 2001) or similar syntactic analyzer is necessary (see Arts and McNeil, 2013 for a description of this process in the development of the *Oxford Arabic Dictionary*). But in order for a word sketch to be created, a corpus needs to be annotated for part of speech. This is a non-trivial task, given the ambiguity of natural language in general—and of Arabic in particular.

In this work, I hand-annotate a 6,000-word section of the Tunisiya corpus for part-of-speech in order to evaluate the performance of part-of-speech classifiers: a rules-based morphological segmenter plus POS tagger and two machine learning classifiers: a backoff n-gram tagger and a Brill transformation tagger. I discovered that, despite the small size of the corpus, the machine learning taggers performed almost as well as the rules-based tagger, achieving 83.93% accuracy on the held-out test data. While is far behind the curve for Standard Arabic systems (such as Freihat et al., 2017, who achieved accuracies above 98% for segmentation and part-of-speech annotation), it compares favorably with previous work on dialectical Arabic, a less standardize and more variable data set.

2 Arabic NLP

Arabic is a challenging language for NLP, for several reasons. Chief among these is its morphological complexity: there are many affixes and clitics, to the point that an entire sentence can be contained in one word. For example, the Tunisian Arabic word ومياكتبوهاليش (*w* + *mā* + *y*+ *kitb* + *ū* + *hā* + *l* + *ī* + *š*) means 'they don't write it for me' and is composed of nine morphemes: two proclitics (+و *u*+, 'and,' plus the negative particle +ما *mā*+), one verbal inflection prefix (+ي *y*+, '3rd person'), the verb stem *ktib*, one verbal inflection suffix (+و *ū*+, 'plural') and four enclitics (the pronominal direct object +ها *hā*+, 'her;' the preposition +ل *l*+, 'to', the pronominal indirect object +ي ī +, 'me; and the second part of the negation circumfix +ش š+). This is an extreme example, since Tunisian cliticizes elements (such as the dipartite negation and indirect objects) that cannot be cliticized in Standard Arabic. Yet it is not uncommon for orthographic words in Standard Arabic as well to be composed of five or six elements (see Habash, 2010:39-42 for a discussion of Arabic morphological complexity).

In addition to the complex morphology, the Arabic script itself can be problematic. There are no capital letters and internal short vowels are not written, which means that a single orthographic form like `ktb` could have multiple readings, such as *kteb* ('he wrote') or *kutub* ('books').

3 Challenges of Tunisian NLP

Much of the Arabic NLP work that has been done has been on Modern Standard Arabic (MSA). Though it is the official language throughout the Arab world, MSA is not the spoken language of any community — it is learned at school and is nobody's mother tongue. Rather, Arabs speak one of many colloquial varieties (often called 'dialects'), which form a continuum from Morocco to Iraq and which vary to the point of unintelligibility, depending on speakers' prior exposure to each other's dialects. The dialects also vary within each country: I refer to the subject of this study as 'Tunisian Arabic' for convenience, but it is more precisely the prestige variety of the capital city, Tunis.

Colloquial varieties of Arabic, like Tunisian, share the NLP challenges of Arabic in general, but also have some additional ones. The first major challenge is the lack of standardization. Because it is not a written language, Tunisian has no standard orthography, and fewer models of normative usage. So people often spell things however they sound, leading to quite a lot of variation. For example, a search of the Tunisiya corpus yields nine different spellings of the common function word *ʾašniya* ('what')—three of them quite common (McNeil, 2019: 42–43).

A larger problem is the lack of resources available for natural language processing of Tunisian Arabic. There have been a few lexical resources developed; a Tunisian WordNet (TunDiaWN, Bouchlaghem et al., 2014) contains 32,848 words but is not publicly available, while the few other lexicons that exist are very small. There are several corpora that have been produced, mostly raw text but a few with various kinds of annotations: sentiment (TSAC, Mdhaffar, 2017), dialect/not dialect (Younes et al., 2015), and morphosyntactic (STAC, Zribi et al., 2015). The largest corpora available are a 870,904-word SMS and social media corpus (Masmoudi et al., 2015) and the 881,964-word mixed-source corpus Tunisiya (McNeil, 2019). The latter is marred by the admixture of MSA and French and its lack of annotation and standardization, but is easily accessible online.[1] The former is also unannotated, but its orthography is normalized following the CODA-TUN standard (Zribi et al., 2014). See Mekki et al. (2018) for a detailed overview of available resources.

4 Related work

Arabic is under-resourced in NLP tools, but there has been a concentrated effort over the past couple of decades to improve the state of NLP for Standard Arabic. Unfortunately, tools and resources developed for Standard Arabic are not readily applicable to colloquial varieties. Previous studies (including Duh and Kirchhoff, 2005; Habash and Rambow, 2006; Habash et. al., 2012) have shown accuracy rates of only about 65–70% for processing Egyptian or Levantine Arabic with Standard Arabic analyzers.

Researchers have obtained better results by creating tools for colloquial Arabic from scratch, rather than trying to leverage existing Standard Arabic tools. Al-Sabbagh and Girju (2012) hand-annotated a 423,691-word Twitter corpus of Egyptian Arabic to train a Brill transformation-based part-of-speech tagger (Brill, 1994, discussed further below). They achieved very good results, with a segmentation F-score of 94.5% and a POS-tagging F-score of 87.6%.

Of the tools for colloquial Arabic that have been developed, very few are for Tunisian Arabic or other North African varieties. Instead—for various historical and cultural reasons—the Eastern varieties like Egyptian and Levantine have received more attention. In just the last few years, however, a flurry of activity—much of it out of the University of Sfax in Tunisia—has begun to fill this gap. Inés Zribi and her team created a 42,388-word transcribed corpus of spoken Tunisian Arabic (STAC, Zribi et al., 2015), and have been developing a suite of tools. These include a conventional orthography for Tunisian Arabic (CODA-TUN, Zribi et al., 2014), sentence boundary detection for transcribed speech (Zribi et al., 2016), a morphological analyzer (Al-Khalil-TUN, Zribi et al., 2013), and a POS tagging system (TAMDAS, Zribi et al., 2017).

These last two works are clearly the most relevant to my current project. In Zribi et. al.,

[1] http://tunisiya.org

(2016), she and her team adapted an MSA morphological analyzer, al-Khalil (Boudlal et al., 2010) by creating a Tunisian lexicon for its root list and replacing the MSA clitics with their Tunisian equivalents. Using this system, dubbed al-Khalil-TUN, they were able to obtain an F-score of 88.86% when applied to their STAC corpus.

In their 2017 paper, Zribi and her team tested three different machine learning techniques for disambiguation and POS tagging: the rule induction algorithm RIPPER (Cohen, 1995), a partial decision-tree algorithm, PART, based on Mohamed et al., 2013, and a Support Vector Machine (SVM, Vapnik, 1995). They selected features related to the morphology of each word, its position in the sentence, and the part-of-speech tags preceding it, and used the STAC corpus to train and test their model, using 10-fold cross-validation. They achieved an accuracy of 87.32% (Zribi et al., 2017).

5 Corpus for Present Study

For the current project I make use of Tunisya, a 881,964-word corpus of Tunisian Arabic available online at tunisiya.org (McNeil, 2019). This corpus is composed of both traditional sources of written Tunisian (folktales, songs, screenplays), as well as materials made possible by new technology: blogs, forum postings, SMS and Facebook messages. Due to diglossia in the Arab world and Tunisian Arabic's status as an 'unwritten' language, many of the media sources typically leveraged for corpus-construction are not available: all news and published works, both fiction and non-fiction, are written in Standard Arabic. (See McNeil 2019 for more about the construction of the Tunisiya corpus.) However, there has recently been a development in the usage scope of Tunisian Arabic: in the last few years there have been four novels published entirely in Tunisian Arabic, two of which have so far been added to the corpus.

To create a subcorpus for the present study, I filtered the corpus to remove text that seemed to be predominately MSA or French. I then selected 450 of the remaining Tunisian sentences at random for hand-annotation; this sample makes up the corpus discussed below.

6 Method overview

The goal of this study is to evaluate the accuracy of several part-of-speech tagging systems, with the long-term goal of implementing one of them on the whole Tunisiya corpus. The main steps for the part-of-speech tagging were:

1. Preprocessing
2. Light stemming with the Tunisiya parser
3. Annotation of training and testing data
4. Training and evaluating the classifiers

These steps are discussed one-by-one below.

6.1 Preprocessing

The first task was to perform sentence segmentation on the corpus. In some ways, this presents less of a challenge for Arabic than Western for languages because abbreviations are uncommon. So erroneously breaking in the middle of a sentence is not a concern. But it is common for conjunctions to be used instead of punctuation as a sentence delimiter, leading to some very long sentences with dozens of words, containing five or six complete phrases. This is a problem because the n-gram taggers (described in §6.6) are not intended to consider context across sentence boundaries.

Such an issue cannot be solved programmatically without syntactic and semantic analysis; my solution was simply to segment sentences with a regular expression tokenizer, and break up excessively long sentences during the hand-annotation phase.

After sentence tokenization, the next step of preprocessing was to insert a space between Arabic characters and foreign (English or French) characters. (It is common for Tunisians to code-switch, even in the middle of a word, for example by using a French word with an Arabic definite article). For this purpose I exploited Arabic characters' lack of case to determine if a character was in Arabic or latin script.

An important step in any Arabic NLP process is spelling normalization, in order to decrease noise and sparsity issues (see Habash, 2010:22–23). I would like in the future to normalize the corpus following the CODA-TUN orthographic standards (Zribi et al., 2014), but in the meantime I simply removed or collapsed problematic characters. This included:

- **Removal of *tatwīl*:** this is a decorative mark that serves to increase the space between letters but has no meaning
- **Removal of diacritics:** optional marks representing short vowels and gemenation that are used rarely and inconsistently
- ***Hamza* normalization:** The Arabic letter *hamza* (ء), representing a glottal stop, has complex spelling rules and is such a common source of errors that it's standard practice to conflate all of them into two forms: a bare *alif* at the beginning of words (ا < آ أ إ) and a bare *hamza* in other positions (ء < ئ أ ء ؤ).

Lastly, the words in each sentence were tokenized on whitespace.

I should note here that I did *not* transliterate the text into latin characters as part of the preprocessing: all of the work below was done on the native Arabic script. When I originally built the software for the corpus, I transliterated all of the text into a modified form of Buckwater transliteration (Buckwater, 2004) before any processing. This was necessary because I was working in Python 2, but with native unicode support in Python 3, I preferred to leave the text as it was written. See Habash (2010:20–22) for a discussion of the pros and cons of transliteration, and the variety of systems available.

6.3 Light stemming with Tunisiya parser

When I created the Tunisiya corpus and its management software, I implemented morphological segmentation to make the keyword in context (KWIC) searches on the website more useful for researchers using it: so that searching for the Tunisian word for 'house,' for example, would also return results for 'the-house,' 'and-house,' 'to-house- my,' (and the dozen other possible inflections). I defined an extensive grammar of affixes and their various combinations into word types, then used the pyparsing module to determine which of these word types a given word could be parsed as. Because of the ambiguity of written Arabic, most words produce multiple possible parses; I used the concept of affix similarity (Dasgupta and Ng, 2007) to determine which of the possible parses was most likely to be correct.

An updated version of this part-of-speech tagger is the basis of comparison for the supervised learning taggers implemented below. The initial parsing obtained a word-level segmentation accuracy of 92.0%.

6.4 Selection of part-of-speech tagset

The selection of a part-of-speech tagset is difficult, because the NLP work on dialectical Arabic has not yet coalesced around a single standard, so there are many options available (see Zeroual, 2016). The tags used here were selected for practicality: word types that were morphologically distinct were given a separate tag, yielding tags for: conjunctions, determiners, demonstrative pronouns, emphatic pronouns, foreign words, interrogative pronouns, nouns, negative particles, negative copular pronouns, prepositions, particles, pronouns (both independent and affixed), punctuation/numbers, relative pronouns, past-tense verbs, and present-tense verbs. During the manual annotation, I also tagged adjectives and adverbs (which are morphologically similar to other word classes and so could not easily be identified with the parser, but behave differently syntactically).

6.4 Manual annotation of data

Once the updated parser was run on the 450 Tunisian sentences, I created the gold-standard file by going through the parse results, correcting any erroneous parses or part-of-speech tags.

Some such errors are unavoidable, due to the ambiguity inherent in the language, and caused only minor problems for the annotation. For example, the plural suffix of verbs (ـو -*ū*) is identical to the direct object enclitic for 'him' (ـو -*ū*). So the word *kteb-ū* may be analyzed by the parser as `[('ktb', 'VBD'), ('w', 'PRO')]`, even though the correct parse may be `[('ktbw', 'VBD')]` 'they wrote'— or even `[('ktb', 'N'), ('w', 'PRO')]` 'his books'. Even for a human annotator, there's no way to tell which it is without seeing the word in context.

The parser also produced parses that don't correspond to valid words: for instance the conjunction-pronoun phrase *wa-hiyya* 'and-she' ('C', 'PRO') being parsed as *wah-y* *wah-my ('N', 'PRO'). This kind of segmentation error must be corrected manually.

After correcting the segmentation and POS tags (including adding the adjective and adverb tags, which were not produced by the parser at all), I determined that the Tunisiya parser tagged part-of-speech with an accuracy of 84.3%.

It's worth noting that the annotation of the gold standard tags was done by starting with the output of the Tunisiya parser and then correcting its mistakes. This may lead the accuracy rates of the parser to be slightly higher than they would be, had the work been done by scratch: when assigning part-of-speech tags, there are often judgements to be made and such judgements are likely to favor an already existing tag. I should also note that the annotation was done by a single annotator (myself) who is a non-native speaker of the language—ideally annotation would be performed by at least two annotators (so cross-annotator consistency could be calculated) who are native speakers with linguistic training.

6.5 Division into training and test data

Before implementing the supervised learning algorithms on my now-annotated data, I divided the data into training and test portions, using cross-validation. Ten-fold cross-validation is standard, but larger numbers of folds—including the maximum number of folds in the leave-one-out method—are also possible. Previous research (Kohavi, 1995) has suggested that on typical data sets, ten-fold cross validation is superior, even if the computational resources for more folds is available. In order to explore both options, I implemented each tagger using both 10-fold splits and leave-one-out (i.e. 449-fold) splits. I found that leave-one-out performed better on this data than 10-fold cross validation, improving the accuracy by over 0.5%.

6.6 Classifier implementation

I then used this hand-annotated data to train the supervised-learning classifiers. The two classifiers I implemented were the n-gram backoff tagger, and the Brill transformation tagger, both part of the Natural Language Toolkit (NLTK) package (Bird et al.). The results are presented in Figure 1.

The n-gram trainer uses a backoff chain of taggers to default to a less sophisticated model whenever the more advanced model is unable to assign a tag. The most basic of these taggers is the Default Tagger, which assigns the same tag to all words (here 'N', since nouns are the most frequent class). This tagger, unsurprisingly, did not perform very well, tagging just over a quarter of the words correctly. A step up from this is the unigram tagger, which assigns each word the most common tag for that word. This increased accuracy to 83.59%.

The bigram tagger, in addition to considering the word, also considers the part of speech of the previous word; the trigram considers the part of speech of the two previous words. The bigram and trigram taggers, when used alone, have very poor accuracy: if an unknown word is encountered, the tagger is not only unable to tag that word, but is also unable to tag the rest of the sentence (since a POS tag of 'None' in the context will have not been seen before). This is why such taggers are used in a backoff chain: rather than returning a 'None' tag, a 3-gram sequence can be backed off to a 2-gram sequence, then a 1-gram and—if the word is out of vocabulary—just tagged with the default tag of 'N' (Bird et al., §5.3-5.4). In my data, I saw a small improvement in accuracy for each additional word considered: the trigram tagger achieved a top accuracy of 83.81%.

The Brill tagger, on the other hand, is not a backoff tagger, but rather a transformation classifier. A drawback of the n-gram taggers described above is that they only consider the part-of-speech tags of the previous words: they are unable to consider the previous words themselves, or the word and tags following the word being tagged. In addition, it is not possible to inspect their model and determine how they are making their decisions, since their models consist of huge probability tables of word and tag combination.

The Brill tagger, on the other hand, is implemented as a series of simple, human-readable rules, such as `Word([-1])` and `Pos([1])`, for the previous word and the following word's part-of-speech tag, respectively. As Brill (1994) describes, the algorithm transforms the rules based on errors observed at each stage of the iterative process, promoting rules that decrease the error-rate and demoting rules that increase it. The text is first run through an initial tagger, which could be something as simple as a default tagger that gives each word the same tag, or as complex as the trigram backoff tagger I used here. "Once text has been passed through the initial-state annotator, it is then compared to the *truth* [i.e. a manually-annotated gold standard] and transformations are learned that can be applied to the output of the initial state annotator to make it better resemble the *truth*" (Brill 1994:2).

Results: Accuracy of Classifiers

Classifier	Cross-Validation Method	
	10-fold	*Leave-one-out*
Default	26.61%	26.98%
Unigram	82.53%	82.59%
Bigram	83.12%	83.37%
Trigram	83.24%	83.81%
Brill	83.30%	**83.93%**
Tunisiya	**84.3%** *(cross-val n/a)*	

Figure 1

As seen in Figure 1, the Brill tagger provided only a slight improvement in accuracy over the trigram parser. But the ability to introspect the rules was interesting: in examining the highest-ranked rules, we can see that the most important context for the tagger were the previous two words, and the part-of-speech of the previous word and the following word.

7 Evaluation and discussion

Due to the small size of the annotated data set, this is clearly an exploratory study only, with limited practical applications. Nonetheless, the study has produced some interesting conclusions. The main—and somewhat surprising—conclusion is that supervised learning methods can be valuable, even for under-resourced languages with tiny annotated corpora. The total training and test data size for this project was just 450 sentences (about 6,000 words / 70 Kilobytes), yet the tools trained on it produced results comparable to a laboriously-produced rules based parser.

In this specific case, the supervised learning taggers are not particularly useful, since they require that the data be previously segmented, and the rules-based segmenter used here determines the part of speech of each word as a consequence of finding the correct parse. However, this work is still exciting in demonstrating that, if a supervised-learning method for segmentation (such as the Brill-based method used by Al-Sabbagh and Girju 2012) is developed, the part of speech tagging can be performed subsequently at a fair accuracy.

8 Next steps

Though the results of this study were encouraging, there were many limitations which, if addressed, would strengthen the conclusions and possibly improve the accuracy of the machine learning classifiers. Addressing the size limitations of the annotated data is clearly the first priority: this could be done by annotating more data and—more importantly—implementing a standard orthography like CODA-TUN so that multiple occurances of the same word can be indexed together, even if originally spelled differently.

Also, the spelling normalization implemented here was chosen based on what is normally done when processing Arabic (Habbash, 2010:21–23), but itself was not tested to see how it affected the parsing and POS tagging accuracy. There are other normalizations that can be done (for instance, conflating the feminine ending +ة with the homophonic +ه, or ي with ى). Such changes may improve the parser's accuracy, but may not. They would need to be evaluated.

A further improvement of this work would be to bring the part-of-speech tags in line with other existing tag standards. There is not yet an accepted standard tagset for colloquial Arabic, and there are different systems used for different purposes. But the tag system used here is an ad-hoc one, determined mostly by word types that are morphologically transparent. Bringing the tags in line with the other major tagsets used in the field will make the results more directly comparable to previous work done. It will also make the tagged corpus more valuable to other researchers, when the tagger is eventually implemented on the public version of the Tunisiya corpus.

9 References

Arts, T., and McNeil, K., 2013. Corpus-based lexicography in a language with a long lexicographical tradition: The case of Arabic. Workshop on Arabic Corpus Linguistics, Lancaster University (available online).

Al-Sabbagh, R., Girju, R., 2012. A supervised POS tagger for written Arabic social networking corpora. *Proceedings of KONVENS 2012 (Main Track: Oral Presentations)*, Vienna, pp. 39–52.

Bird, S., Klein, E. and Loper, E. (undated). "Categorizing and Tagging Words," in *Natural Language Processing with Python – Analyzing Text with the Natural Language Toolkit*. http://www.nltk.org/book/ch05

Bouchlaghem, R., Elkhlifi, A., and Faiz, R., 2014. *Tunisian dialect Wordnet creation and enrichment using web resource sand other Wordnets. EMNLP Workshop on Arabic Natural Language Processing*, 2014, pp. 104–113.

Boudlal, A., Lakhouaja, A., Mazroui, A., Meziane, A., Ould Abdallahi Ould Bebah, M., Shoul, M., 2010. Alkhalil Morpho SYS1: a morphosyntactic analysis system for Arabic texts. *International Arab Conference on Information Technology*, pp. 1–6.

Boujelbane, R., Ellouze, M., Béchet, F., Belguith, L., 2014. De l'arabe standard vers l'arabe dialectal: projection de corpus et ressources linguistiques en vue du traitement automatique de l'oral dans les médias tunisiens. *Revue TAL.*

Brill, E., 1994. *Some advances in transformation-based part of speech tagging.* arXiv preprint cmp-lg/9406010.

Cohen, W.W., 1995. Fast effective rule induction. *Proceedings of the Twelfth International Conference on Machine Learning*, 115–123 101.1.50.8204.

Dasgupta, S., and V. Ng. 2007. High-performance, language-independent morphological segmentation, *Proceedings of Human Language Technology (NAACL)*.

Duh, K., Kirchhoff, K., 2005. POS tagging of dialectal Arabic: a minimally supervised approach. *Proceedings of the ACL Workshop on Computational Approaches to Semitic Languages*, pp. 55–62.

Freihat, A.A., Bella, G., Mubarak, H., & Giunchiglia, F., 2018, April. A single-model approach for Arabic segmentation, POS tagging, and named entity recognition. *In 2018 2nd International Conference on Natural Language and Speech Processing (ICNLSP)*, pp. 1–8. IEEE.

Habash, N.Y., 2010. *Introduction to Arabic natural language processing*. Synthesis Lectures on Human Language Technologies, 3(1):1–187.

Habash, N., Rambow, O., 2006. MAGEAD: a morphological analyzer and generator for the Arabic dialects. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics,* pp. 681–688.

Habash, N., Eskander, R., Hawwari, A., 2012. A morphological analyzer for Egyptian Arabic. In *NAACL-HLT 2012 Workshop on Computational Morphology and Phonology (SIGMOR- PHON2012)*, pp. 1–9.

Kilgarriff, A. and Tugwell, D., 2001. Word sketch: Extraction & display of significant collocations for lexicography. In *Proceedings of the 39th ACL & 10th EACL-workshop 'Collocation: Computational Extraction, Analysis and Exploitation'*, p. 32–38, Toulouse.

Kohavi, R., 1995, August. A study of cross-validation and bootstrap for accuracy estimation and model selection. In *Ijcai* Vol. 14(2):1137–1145.

Masmoudi, A., Habash, N., Ellouze, M., Estève, Y., and Belguith, L.H., 2015. Arabic Transliteration of Romanized Tunisian Dialect Text: A Preliminary Investigation, in: *Proceedings of CICLing 2015, Part I*, Cairo, Egypt, 2015, pp. 608–619.

McNeil, K., 2019. Tunisian Arabic Corpus: Creating a Written Corpus of an 'Unwritten' Language. In McEnery, T., Hardie, A., & Younis, N. (Eds.). *Arabic Corpus Linguistics.* Edinburgh: Edinburgh University Press.

Mdhaffar, S., Bougares, F., Eve, Y., and Hadrich-Belguith, L., 2017. Sentiment Analysis of Tunisian Dialect: Linguistic

Resources and Experiments. In *Proceedings of the Third Arabic Natural Language Processing Workshop (WANLP)*, Valencia, Spain, 2017, pp. 55–61.

Mohamed, W.N.H.W., Salleh, M.N.M., Omar, A.H., 2013. A comparative study of reduced error pruning method in decision tree algorithms. *Proceedings – 2012 IEEE International Conference on Control System, Computing and Engineering, ICCSCE 2012*, pp. 392–397.

Vapnik, V.N., 1995. *The Nature of Statistical Learning Theory.* Springer. http://dx. doi.org/10.1109/TNN.1997.641482.

Younes, J., Achour, H. and Souissi, E., 2015. Constructing Linguistic Resources for the Tunisian Dialect Using Textual User-Generated Contents on the Social Web. *Current Trends in Web Engineering - 15th International Conference, ICWE 2015* Rotterdam, The Netherlands, pp. 3–14.

Zeroual, I., Lakhouaja, A., & Belahbib, R. (2017). Towards a standard Part of Speech tagset for the Arabic language. *Journal of King Saud University—Computer and Information Sciences*, 29(2):171–178.

Zribi, I., Boujelbane, R., Masmoudi, A., Ellouze, M., Belguith, L.H. and Habash, N., 2014. A Conventional Orthography for Tunisian Arabic. *Proceedings of LREC 2014, European Language Resources Association (ELRA)*, Reykjavik, Iceland, pp. 2355–2361.

Zribi, I., Ellouze, M., Belguith, L.H., Blache, P., (2015) Spoken Tunisian Arabic Corpus "STAC": Transcription and Annotation, *Research in computing science* 90.

Zribi, I., Ellouze, M., Belguith, L.H., Blache, P., 2017. Morphological disambiguation of Tunisian dialect. *Journal of King Saud University—Computer and Information Sciences,* 29(2):147–155, Arabic Natural Language Processing: Models, Systems and Applications.

Zribi, I., Kammoun, I., Ellouze, M., Belguith, L.H. and Blache, P. 2016. Sentence boundary detection for transcribed Tunisian Arabic. *Proceedings of the 12th Edition of the Konvens Conference*, Bochum, Germany, 2016.

Zribi, I., Khemakhem, M.E., Belguith, L.H. 2013. Morphological Analysis of Tunisian Dialect. *IJCNLP 2013*, Nagoya, Japan, pp. 992–996.