Arabic POS Tagging: Don't Abandon Feature Engineering Just Yet

Kareem Darwish **Hamdy Mubarak** **Ahmed Abdelali** **Mohamed Eldesouki**
Qatar Computing Research Institute (QCRI)
HBKU, Doha, Qatar
{kdarwish,hmubarak,aabdelali,mohamohamed}@hbku.edu.qa

Abstract

This paper focuses on comparing between using Support Vector Machine based ranking (SVM^{Rank}) and Bidirectional Long-Short-Term-Memory (bi-LSTM) neural-network based sequence labeling in building a state-of-the-art Arabic part-of-speech tagging system. Using SVM^{Rank} leads to state-of-the-art results, but with a fair amount of feature engineering. Using bi-LSTM, particularly when combined with word embeddings, may lead to competitive POS-tagging results by automatically deducing latent linguistic features. However, we show that augmenting bi-LSTM sequence labeling with some of the features that we used for the SVM^{Rank}-based tagger yields to further improvements. We also show that gains realized using embeddings may not be additive with the gains achieved due to features. We are open-sourcing both the SVM^{Rank} and the bi-LSTM based systems for the research community.

1 Introduction

Part-of-speech (POS) tagging is an important building blocking in many natural language processing applications such as parsing and named entity recognition. An Arabic word is composed of one or more segments (or clitics), which are typically a stem to which prefixes and suffixes may be attached. Arabic POS tagging involves assigning appropriate in-context POS tags to each clitic. Tagging can be done for each clitic in sequence or for all clitics in a word simultaneously. Much work has been done on Arabic POS tagging and many morphological and surface-level features have been shown to improve tagging. Recent work on sequence labeling using deep neural networks, particularly using bidirectional Long-Short-Term-Memory (bi-LSTM) and word embeddings, has been shown to be effective for POS tagging in different languages, without the need for explicit feature engineering. In essence, deep neural networks may be able to capture latent linguistic features automatically. In the context of this work, we compare using a discriminative classification technique, namely Support Vector Machine based Ranking (SVM^{Rank}), that requires significant feature engineering with bi-LSTM neural network with and without feature engineering and word embeddings. We experiment with tagging each clitic in context and with tagging all clitics in a word collectively. We also compare both systems with MADAMIRA, which is a state-of-the-art Arabic POS tagging system. We show that adding explicit features to the bi-LSTM neural network and employing word embeddings separately improve POS tagging results. However, combining both explicit features and embeddings together leads sub-optimal results. For testing, we employ the so-called "WikiNews" test set which is composed of freely available recent news articles in multiple genre (Abdelali et al., 2016). We are making all resultant systems available as open-source systems.

The contributions of this paper are as follows:

- We compare using SVM^{Rank} to using bi-LSTM with and without feature engineering and word embeddigns in Arabic POS tagging. We show that feature engineering improves POS tagging significantly.
- We explore the effectiveness of many features including morphological and contextual features for tagging each clitic or each word in-context.

Proceedings of The Third Arabic Natural Language Processing Workshop (WANLP), pages 130–137,
Valencia, Spain, April 3, 2017.

- We open-source both Arabic POS taggers, both of which are written entirely in Java. The SVM^{Rank}-based system has a load time of 5 seconds and can process about 2,000 word/second on an laptop with Intel i7 processor with 16 GB of RAM.

2 Background

2.1 Challenges of Arabic Language

Arabic language is a Semitic language with complex templatic derivational morphology. The Arabic nouns, adjectives, adverbs, and verbs are typically derived from a closed set of approximately 10,000 roots of length 3, 4, or rarely 5 letters. Arabic nouns and verbs are derived from these roots by applying templates to the roots to generate stems. Such templates may carry information that indicate morphological features of words such POS tag, gender, and number. For example, given a 3-letter root with 3 consonants CCC, a valid template may be CwACC , where the infix "wA" is inserted. This template is typically an Arabic broken, or irregular, plural template for a noun of template CACC or CACCp for masculine or feminine respectively. Further, stems may accept prefixes and/or suffixes to form words. Prefixes include coordinating conjunctions, determiner, and prepositions, and suffixes include attached pronouns and gender and number markers. English POS tagging techniques face a problem when dealing with agglutinative and highly inflected languages such as Arabic. This results in a large number of words (or surface forms) and in turn a high-level of sparseness and a large number of previously unseen words. Further, Arabic words embed morphological information such as gender and number and syntactic information such as case and gender and number agreement.

Traditional Arab linguists divide Arabic words into three classes, namely: nouns, verbs, and particles. Such coarse categorization is not suitable for many higher NLP tasks such as parsing. Therefore, more comprehensive tagsets have been created to capture the morphological and syntactic aspects of the words in Arabic. For most, the number of Arabic clitic-level POS tags is small, while the number of valid composite word-level tags is typically large. The proposed clitic-level tagsets range from simplified tagsets such as the *CATiB* tagset (Habash and Roth, 2009; Habash et al., 2009a) which has only six POS tags to more complex tagsets such as that of the *Penn Arabic Treebank* (ATB), which has 70 tags (Maamouri et al., 2004). In our work, we elected to use the tagest proposed by Darwish et al. (2014) which is a simplified version of ATB tagset and uses 18 tags only.

2.2 Arabic POS Tagging

Most recent work on Arabic POS tagging has used statistical methods. Diab (2009) used an SVM classifier to ascertain the optimal POS tags. The classifier was trained on the ATB data. Essentially, they treated the problem as a sequence-labeling problem. Another popular system for Arabic POS tagging is MADAMIRA, which uses an underlying morphological analyzer and is also trained on the ATB (Habash et al., 2009b; Pasha et al., 2014). We use MADAMIRA to compare to our work. Darwish et al. (2014) introduced the use of stem templates to improve POS tagging and to help ascertain the gender and the number of nouns and adjectives. They reported an accuracy of 98.1% on ATB data when using gold segmentation and employing different features such as word surface forms, list matching, and stem-templates.

In recent developments, deep neural networks were used to develop taggers that achieve good POS tagging accuracy. Plank et al. (2016) used bi-LSTM neural network to build taggers for 22 languages. The models achieved significant results for morphologically complex languages including Arabic. The models were built using the Universal Dependencies project v1.2 (Nivre et al., 2015) data. Ling et al. (2015) used bi-LSTM (Hochreiter and Schmidhuber, 1997) combining words and characters vector representations to achieve comparable results to state-of-the-art English POS tagging. Wang et al. (2015) used only word-embeddings on a bi-LSTM neural network to train a POS tagger; their approach achieved 97.26% accuracy on WSJ testset. The highest accuracy reported on this testset was 97.25% by Huang et al. (2012) .

3 Our Part-of-Speech Taggers

Our Arabic part-of-speech (POS) tagging uses the simplified ATB tag set proposed by (Darwish et al., 2014) and shown in Table 1. The POS tagger attempts to find the optimal tag for each word in a sentence. We present here two different approaches for POS tagging. The first uses SVM^{Rank} to guess POS tags at the level of cli-

POS	Description	POS	Description
ADV	adverb	ADJ	adjective
CONJ	conjunction	DET	determiner
NOUN	noun	NSUFF	noun suffix
NUM	number	PART	particles
PREP	preposition	PRON	pronoun
PUNC	punctuation	V	verb
ABBREV	abbreviation	CASE	alef of tanween fatha
JUS	jussification attached to verbs	VSUFF	verb suffix
FOREIGN	non-Arabic as well as non-MSA words	FUT_PART	future particle "s" prefix and "swf"

Table 1: Part-of-speech tag set of Farasa.

tics or words using clitic and word level features as well as context-level features. The second uses Bi-LSTM recurrent-neural-network with clitic level features to guess POS tags at clitic level.

3.1 SVM^{Rank}-based POS Tagger

The POS tagger uses SVM^{rank} (Joachims, 2006) with a linear kernel to determine the best POS tag for each word. It was trained on parts 1 (v. 4.1), 2 (v. 3.1), and 3 (v. 2) of the ATB (Maamouri et al., 2004). Instead of testing the POS tagger on a subset of the ATB, which may lead to artificially-high results due to its limited lexical diversity, we tested our system on the WikiNews test set, which includes 70 WikiNews articles from 2013 and 2014 and composed of $18,300$ words that are manually-segmented and POS tagged (Darwish and Mubarak, 2016). The WikiNews test set covers a variety of topics, namely: politics, economics, health, science and technology, sports, arts, and culture.We followed two paths for POS tagging, namely:

- (Clitic) we guess the POS tag for each clitic in a word, and then we combine the tags of the clitics of a word.
- (Word) we guess the compound POS tag for the whole word.

In both paths, we constructed a feature vector for each possible POS tag for each clitic or word. We supplied these vectors to SVM^{Rank} indicating which vector should rank highest given feature values. We then use SVM^{Rank} (Joachims, 2006) to learn feature weights. We use a linear kernel with a trade-off factor between training errors and margin equal to 100 (parameters tuned on offline experiments carried out over a development set that was set aside from ATB). All possible POS tags for a clitic or a word are scored using the classifier, and the POS with the highest score is picked.

3.1.1 Tagging Clitics

Given a sentence composed of the clitics $c_{-n} \dots c_0 \dots c_m$, where c_0 is the current clitic and its proposed POS tag, we train the classifier using the following features, which are computed using the maximum-likelihood estimate on our training corpus:

- $p(POS|c_0)$ and $p(c_0|POS)$.
- $p(POS|c_{-i}..c_{-1})$ and $p(POS|c_1..c_j)$; $i, j \in [1, 4]$.
- $p(POS|c_{-i_{POS}}..c_{-1_{POS}})$ and $p(POS|c_{1_{POS}}..c_{j_{POS}})$; $i, j \in [1, 4]$. Since we don't know the POS tags of these clitics *a priori*, we estimate the conditional probability as:

  $$\sum p(POS|c_{-1_{possible\_POS}}..c_{-i_{possible\_POS}}) \ .$$

  For example, if the previous clitic could be a NOUN or ADJ, then $p(POS|c_{-1}) = p(POS|NOUN) + p(POS|ADJ)$.

If the clitic is a stem, we also compute the following features:

- $p(POS|stem\_template)$. Arabic words are typically derived from a closed set of roots that are placed in so-called stem templates to generate stems. For example, the root `ktb` can be fit in the template `CCAC` to generate the stem `ktAb` (book). Stem templates may

conclusively have one POS tag (e.g., `yCCC` is always a V) or favor one tag over another (e.g., `CCAC` is more likely a NOUN than an ADJ). We used Farasa to determine the stem template (Abdelali et al., 2016).

- $p(POS|prefix)$ and $p(POS|suffix)$. Some prefixes and suffixes restrict the possible POS tags for a stem. For example, a stem preceded by DET is either a NOUN or an ADJ.
- $p(POS|prefix, prev\_word\_prefix)$, $p(POS|prev\_word\_suffix)$ and $p(POS|prev\_word\_POS)$. Arabic has agreement rules for noun phrases and idafa constructs that cover definiteness, gender, and number. Both these features help capture agreement indicators.
- $p(POS|MetaType)$. We assign each clitic a "meta types". The meta types can help the classifier identify different POS tags. The meta types are:
  - *NUM*: If a clitic is a sequence of numerals or matches a gazetteer of numbers spelled out in words.
  - *FOREIGN*: If all characters are Latin.
  - *PUNCT*: If it is composed of non-letters.
  - *ARAB*: If composed of Arabic letters only.
  - *PREFIX*: If it ends with "+" after segmentation (ex. "Al+").
  - *SUFFIX*: If it starts with "+" after segmentation (ex. "+h").

3.1.2 Tagging Words

In this setup, we attempt to tag the entire word at once instead of tagging each clitic separately. Similar to the tagging of clitics in subsection 3.1.1, we train SVM^{Rank} using word-level features. Given a word sequence $w_{-n}...w_0...w_m$, we used the following features:

- $p(w_0|POS)$ and $p(POS|w_0)$
- $p(POS|w_{0word\_template})$ – The word-template here is the stem-template plus the prefixes and suffixes. For example, the stem of the "Al+ktAb" (the book) is "ktAb" with the stem-template "fEAl", and the word-template is "Al-fEAl".
- $p(POS|MetaType)$ – This is the meta type defined earlier with clitics, except that "PREFIX" and "SUFFIX" meta types are excluded.
- $p(POS|w_{0prefixes})$ – The prefixes are just the prefixes that are attached to the word.
- $p(POS|w_{0suffixes})$ – The suffixes are just the suffixes that are attached to the word.
- $p(POS|w_{0prefixes}, w_{-1prefixes})$ – This helps in capturing gender and number agreement.
- $p(POS|w_{0prefixes}, w_{-1prefixes}, w_{-1POS})$ – This also helps in capturing gender and number agreement.
- $p(POS|w_{-1suffixes})$
- $p(POS|w_{-1POS})$ – Since we don't know the POS tags of words *a priori*, we estimate the conditional probability using the same method we employed for clitics.
- $p(POS|w_{-2POS}, w_{-1POS})$
- $p(POS|w_{1POS}, w_{2POS})$
- $p(POS|w_{1POS}, w_{2POS}, w_{3POS})$
- $p(POS|VerbOrNot)$ – For this feature, we automatically analyzed all the unique words in ten years worth of Aljazeera.net articles using Al-Khalil morphological analyzer (Boudchiche et al., 2016). The articles contains 95.4 million tokens including 613k unique tokens. Given the different analyses of Al-Khalil, if it analyzed a word as a verb only, this feature is set to "V". If it appears as possibly a verb or some other POS, this feature becomes "possible-V". Otherwise, the feature is "not-V". Al-Khalil attempts to provide all the possible analysis of a word, but does not provide any ranking of the solutions. Since this is a word-level feature and not a clitic-level feature, we only used it in this setup.
- $P(POS|NounOrNot)$ – As with VerbOrNot, this feature is also based on the Al-Khalil analyzer, where the feature assumes the values "Noun", "possible-Noun", or "not-Noun".

- Word context features: $p(POS|w_{-1})$, $p(POS|w_1)$, $p(POS|w_{-2}, w_{-1})$, $p(POS|w_{-3}, w_{-2}, w_{-1})$, and $p(POS|w_{-4}, w_{-3}, w_{-2}, w_{-1})$

3.1.3 OOVs and pre-Filtering

For both clitic and word tagging, In case we could not compute a feature value during training (e.g., a clitic was never observed with a given POS tag), the feature value is assigned a small ϵ value equal to 10^{-10}. If the clitic is a prefix or a suffix, then stem-specific features are assigned the same ϵ value.

In order to improve efficiency and reduce the choices the classifier needs to pick from, we employ some heuristics that restrict the possible POS tags to be considered by the classifier: (*i*) If a word is composed of one clitic, and the clitic is a number, restrict to "NUM". We check if the clitic is composed of digits or matches a gazetteer of numbers spelled out in words.
(*ii*) If a word is composed of Latin letters, restrict to "FOREIGN".
(*iii*) If punctuation, restrict to "PUNCT".
(*iv*) If a clitic is a stem and we can figure out the stem-template, restrict POS tags to those that have been seen for that stem-template during training. Similarly, if we can figure out the word-template, we restrict POS tags to those that have been seen for the word-template during training.
(*v*) If a clitic is a stem, restrict to POS tags that have been seen during training given the prefixes and suffixes of the word.

3.2 bi-LSTM Part-of-Speech Tagger

Bi-LSTM neural networks has been shown to be very effective for tagging sequential data, e.g. language modeling, speech utterances (Zen and Sak, 2015), handwritten text (Messina and Louradour, 2015), and scene text recognition (Hassanien, 2016). Further, word embeddings have demonstrated their potential for capturing statistical properties of natural language (Sutskever et al., 2011; Wang et al., 2015; Palangi et al., 2016). Along these directions, we modeled POS tagging as a sequence to sequence learning problem. We used a bi-LSTM neural-network model (Ling et al., 2015) to learn the expected tagset given an input for the model as a sequence of features $f_1, ..., f_n$ that could include word representations –embeddings– as well. The expected output of the network feed-forward states S_i^f contains the tag sets information for the parts 0 to i, while the backforward state S_i^b contains the information for the part $i+1$ to n. The forward and backward states are combined, for each output i as follows:
$l_i = tanh(L^f S_i^f + L^b S_i^b + b_l)$
where L^f , L^b and b_l denote the parameters for combining the forward and backward states.
We experimented with a number of settings where the clitic sequence was augmented with a subset of features that includes character sequences, word meta type, stem template (Darwish et al., 2014), and also combined with 200 dimension word embeddings learned over the aforementioned collection of text containing 10 years of Al-Jazeera articles[1]. To create the embeddings, we used word2vec with continuous skip-gram learning algorithm with an 8 gram window (Mikolov et al., 2013)[2]. For the bi-LSTM experiments, we used the Java Neural Network Library[3], which is tuned for POS tagging(Ling et al., 2015). We extended the library to produce the additional aforementioned features.

4 Evaluation and Discussion

4.1 SVM Approach

We trained the POS tagger using the aforementioned sections of the ATB (Maamouri et al., 2004). Testing was performed on the WikiNews dataset (Darwish and Mubarak, 2016). Table 2 reports on the accuracy of our POS tagger on the WikiNews dataset and compares it to MADAMIRA. The word-level SVM-based system beats the clitic-level system by 0.4% accuracy and achieves nearly identical results to MADAMIRA (with less than 0.005% difference). Using the word-level system has the advantage of being able to capture more context than the clitic-level system. We classified all the errors from our best system (word-based segmentation). The breakdown of the errors listed in Table 3 shows that confusion between ADJ and NOUN is the most common mistake type with a combined 41.1% of the errors followed by mistakes in segmentation. Common reasons for confusion between ADJ and NOUN include:

- Some words can assume either tag. For example, the word "AstrAtyjyp" could mean "strategy" or "strategic".

[1] aljazeera.net
[2] code.google.com/archive/p/word2vec/
[3] https://github.com/wlin12/JNN

State-of-the-art: MADAMIRA	95.3	
System	Truth Segmentation	Farasa Segmentation
SVM^{Rank} (Clitic)	95.9	94.9
SVM^{Rank} (Word)	96.2	95.3
bi-LSTM (Clitic)	94.5	93.5
bi-LSTM (Clitic) w/embeddings	95.0	92.4
bi-LSTM (Clitic) w/features	96.1	95.0
bi-LSTM (Clitic) w/features + embeddings	95.5	94.7

Table 2: The accuracy of our POS tagger on the WikiNews dataset (Darwish and Mubarak, 2016) against Madamira

- Arabic allows nouns to be omitted and adjectives assume their syntactic roles. For example, the word "AlErby" ("the Arab") could be used in the context of "qAl AlErby ("the Arab said") where it is implied that "the Arab man said", where the word "man" is omitted.
- on some occasions, adjectives may precede the nouns they modify as in the words "¿kbr" ("bigger than").
- the adjective is separated from the noun it modifies by several words.

Error Type	Percentage
ADJ → NOUN	26.3
Segmentation Errors	23.0
NOUN → ADJ	14.8
V → NOUN	11.2
NOUN → V	5.5
PREP → PART	3.0
NUM → ADJ	2.2
CONJ → PART	1.8
NUM → NOUN	1.6

Table 3: Most common errors for best SVM^{Rank} configuration

Verbs are often mislabeled as nouns or vice versa. This is more problematic than the confusion between nouns and adjectives, as the mislabeling verbs can have a bigger impact on downstream applications such as parsing. Much of the errors stem from either: words that could assume either POS tag such as "tqy" meaining either "to protect from" or "righteous"; and verbs that were not observed in training, where the tagger would naturally prefer the more common tag of "NOUN". As the results in Table 2 show, using perfect segmentation leads to improved POS tagging accuracy. This is reflected in Table 3 where segmentation errors accounts for 23% of the errors.

4.2 bi-LSTM Approach

Similar to the evaluation setup of the SVM^{Rank}-based system, we modeled the ATB data into a sequence of clitics and the target was to learn the POS tags. The clitics were obtained using either gold ATB segmentation or from the Farasa Segmenter (Abdelali et al., 2016).

We augmented the input sequence with additional features that included the surface form of the clitic, leading and trailing characters, word meta type, and stem template. In additional experiment, we included the word embeddings learned for aforementioned corpus of Aljazeera.net, that was segmented using the Farasa segmenter. Table 2 shows the results for our bi-LSTM experiments with gold and Farasa segmentation. As expected, bi-LSTM was able to deliver competitive results by capturing complex non-linear and non-local dynamics in sequences (Hochreiter and Schmidhuber, 1997). Results in Table 2 show that:

- Not surprisingly using non-gold segmentation decreased POS tagging accuracy. However, the drop is more pronounced than the drop seen for the SVM^{Rank}-based system, particularly when using embeddings where the drop in accuracy was 2.6%.
- Though using either embeddings or features lead to overall improvements, features lead to bigger improvement than embeddings with greater robustness in the presence of segmentation errors.
- Using both features and embeddings together lead to worse results.

- The best bi-LSTM setup edged the SVM^{Rank} clitic setup by 0.1%.

Table 4 summarizes the error types we observed when using the best bi-LSTM system (using features only and Farasa segmentation). The error trends and the reasons for the errors for bi-LSTM are similar to those of the SVM^{Rank}. We attempted to extend bi-LSTM to perform word-level tagging, but the results were very low (below 82% accuracy). We plan to investigate the reasons for such a drop.

Error Type	Percentage
Segmentation errors	21.8
NOUN → ADJ	17.6
ADJ → NOUN	15.5
NOUN → V	9.3
V → NOUN	7.7
ADJ → NUM	7.0
NUM → NOUN	1.6
CONJ → PART	1.3
NOUN → NUM	1.0

Table 4: Most common errors for best bi-LSTM configuration

5 Conclusion

This work presents two open source state-of-the-art POS tagging systems that are trained using standard ATB dataset (Maamouri et al., 2004) and evaluated on the WikiNews test set (Abdelali et al., 2016). In building the system we explored two approaches using Support Vector Machines (SVM) and Bidirectional Long Short-Term Memory (bi-LSTM). While the first is heavily dependent on linguistically engineered features that are modeled on linguistic knowledge, the second approach has the ability to induce latent linguistic features. Our experiments show that generic approaches might reach considerably high results, but using linguistic features may achieve higher results by encoding domain knowledge and nuances that are difficult to induce from the data alone. Further, using embeddings may lead to improved results, but not as much as hand crafted features.

References

Ahmed Abdelali, Kareem Darwish, Nadir Durrani, and Hamdy Mubarak. 2016. Farasa: A Fast and Furious Segmenter for Arabic. pages 11–16, San Diego, CA, June. Association for Computational Linguistics.

Mohamed Boudchiche, Azzeddine Mazroui, Mohamed Ould Abdallahi Ould Bebah, Abdelhak Lakhouaja, and Abderrahim Boudlal. 2016. Alkhalil morpho sys 2: A robust arabic morpho-syntactic analyzer. *Journal of King Saud University-Computer and Information Sciences*.

Kareem Darwish and Hamdy Mubarak. 2016. Farasa: A New Fast and Accurate Arabic Word Segmenter. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*, Paris, France, may. European Language Resources Association (ELRA).

Kareem Darwish, Ahmed Abdelali, and Hamdy Mubarak. 2014. Using stem-templates to improve arabic pos and gender/number tagging. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Reykjavik, Iceland, may. European Language Resources Association (ELRA).

Mona Diab. 2009. Second generation amira tools for arabic processing: Fast and robust tokenization, pos tagging, and base phrase chunking. In *2nd International Conference on Arabic Language Resources and Tools*.

Nizar Habash and Ryan M Roth. 2009. Catib: The columbia arabic treebank. In *Proceedings of the ACL-IJCNLP 2009 conference short papers*, pages 221–224. Association for Computational Linguistics.

Nizar Habash, Reem Faraj, and Ryan Roth. 2009a. Syntactic annotation in the columbia arabic treebank. In *Proceedings of MEDAR International Conference on Arabic Language Resources and Tools, Cairo, Egypt*.

Nizar Habash, Owen Rambow, and Ryan Roth. 2009b. Mada+ tokan: A toolkit for arabic tokenization, diacritization, morphological disambiguation, pos tagging, stemming and lemmatization. In *Proceedings of the 2nd international conference on Arabic language resources and tools (MEDAR), Cairo, Egypt*, pages 102–109.

Ahmed Mamdouh A Hassanien. 2016. Sequence to sequence learning for unconstrained scene text recognition. *arXiv preprint arXiv:1607.06125*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural computation*, 9(8):1735–1780.

Liang Huang, Suphan Fayong, and Yang Guo. 2012. Structured perceptron with inexact search. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 142–151. Association for Computational Linguistics.

Thorsten Joachims. 2006. Training Linear SVMs in Linear Time. KDD '06, pages 217–226, New York, NY. ACM.

Wang Ling, Tiago Luís, Luís Marujo, Ramón Fernandez Astudillo, Silvio Amir, Chris Dyer, Alan W Black, and Isabel Trancoso. 2015. Finding function in form: Compositional character models for open vocabulary word representation. *arXiv preprint arXiv:1508.02096*.

Mohamed Maamouri, Ann Bies, Tim Buckwalter, and Wigdan Mekki. 2004. The Penn Arabic Treebank: Building a Large-Scale Annotated Arabic Corpus. In *NEMLAR conference on Arabic language resources and tools*, volume 27, pages 466–467.

Ronaldo Messina and Jerome Louradour. 2015. Segmentation-free handwritten chinese text recognition with lstm-rnn. In *Document Analysis and Recognition (ICDAR), 2015 13th International Conference on*, pages 171–175. IEEE.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '13, pages 746–751, Atlanta, GA, USA.

Joakim Nivre, Željko Agić, Lars Ahrenberg, and et al. 2015. Universal dependencies 1.2. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University in Prague.

Hamid Palangi, Li Deng, Yelong Shen, Jianfeng Gao, Xiaodong He, Jianshu Chen, Xinying Song, and Rabab Ward. 2016. Deep sentence embedding using long short-term memory networks: Analysis and application to information retrieval. *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, 24(4):694–707.

Arfath Pasha, Mohamed Al-Badrashiny, Mona T Diab, Ahmed El Kholy, Ramy Eskander, Nizar Habash, Manoj Pooleery, Owen Rambow, and Ryan Roth. 2014. MADAMIRA: A Fast, Comprehensive Tool for Morphological Analysis and Disambiguation of Arabic. LREC 2014, pages 1094–1101. European Language Resources Association (ELRA).

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual part-of-speech tagging with bidirectional long short-term memory models and auxiliary loss. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, pages 412–418.

Ilya Sutskever, James Martens, and Geoffrey E Hinton. 2011. Generating text with recurrent neural networks. In *Proceedings of the 28th International Conference on Machine Learning (ICML-11)*, pages 1017–1024.

Peilu Wang, Yao Qian, Frank K Soong, Lei He, and Hai Zhao. 2015. A unified tagging solution: Bidirectional lstm recurrent neural network with word embedding. *arXiv preprint arXiv:1511.00215*.

Heiga Zen and Haşim Sak. 2015. Unidirectional long short-term memory recurrent neural network with recurrent output layer for low-latency speech synthesis. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 4470–4474. IEEE.