Joint Prediction of Morphosyntactic Categories for Fine-Grained Arabic Part-of-Speech Tagging Exploiting Tag Dictionary Information

Go Inoue, Hiroyuki Shindo and Yuji Matsumoto
Graduate School of Information and Science
Nara Institute of Science and Technology
8916-5 Takayama, Ikoma, Nara, 630-0192, Japan
{inoue.go.ib4, shindo, matsu}@is.naist.jp

Abstract

Part-of-speech (POS) tagging for morphologically rich languages such as Arabic is a challenging problem because of their enormous tag sets. One reason for this is that in the tagging scheme for such languages, a complete POS tag is formed by combining tags from multiple tag sets defined for each morphosyntactic category. Previous approaches in Arabic POS tagging applied one model for each morphosyntactic tagging task, without utilizing shared information between the tasks. In this paper, we propose an approach that utilizes this information by jointly modeling multiple morphosyntactic tagging tasks with a multi-task learning framework. We also propose a method of incorporating tag dictionary information into our neural models by combining word representations with representations of the sets of possible tags. Our experiments showed that the joint model with tag dictionary information results in an accuracy of 91.38% on the Penn Arabic Treebank data set, with an absolute improvement of 2.11% over the current state-of-the-art tagger. [1]

1 Introduction

Part-of-speech (POS) tagging is a fundamental task in natural language processing. The granularity of the POS tag set that reflects language-specific information varies from language to language. In morphologically simple languages such as English, the size of the tag set is typically less than a hundred. On the other hand, in morphologically rich languages such as Arabic, the number of theoretically possible tags can be up to 333,000, of which only 2,200 tags might appear in an actual corpus (Habash and Rambow, 2005). One reason for this is that in the tagging scheme for such languages, a complete POS tag is formed by combining tags from multiple tag sets defined for each morphosyntactic category. For example, a complete POS tag for the word *Hb* ("love")[2] can be defined as the combination of a noun from the coarse POS category, a nominative (*n*) from the case category, "not applicable" (*na*) from the mood category, and so on. The enormous number of resulting tags causes fine-grained POS tagging for Arabic to be more challenging.

In order to perform this task, it is beneficial to utilize information from other morphosyntactic categories when predicting a label for one category. For example, if a word is a noun, it should take one of three tags from the case category: nominative (*n*), accusative (*a*), or genitive (*g*), while it should take "not applicable" (*na*) from the mood category since mood is not defined for nominals. However, most of the previous approaches in Arabic did not utilize this information, applying one model for each task (Habash and Rambow, 2005; Pasha et al., 2014; Shahrour et al., 2015). To make use of this information, we propose an approach that jointly models multiple morphosyntactic prediction tasks using a multi-task learning scheme. Specifically, we adopt parameter sharing in our bi-directional LSTM model in the hope that the shared parameters will store information beneficial to multiple tasks. To further boost the performance, we propose a method of incorporating tag dictionary information into our neural models by combining word representations with representations of the sets of possible tags.

Our experiments showed that the joint model

[1] Our code is available at https://github.com/go-inoue/FineGrainedArabicPOSTagger

[2] We use the Buckwalter transliteration scheme (Buckwalter, 2002) to represent Arabic characters.

Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017), pages 421–431,
Vancouver, Canada, August 3 - August 4, 2017.

pos ($n = 35$)	noun, noun_num, noun_quant, noun_prop, adj, adj_comp, adj_num, adv, adv_interrog, adv_rel, pron, pron_dem, pron_exclam, pron_interrog, pron_rel, verb, verb_pseudo, part, part_dem, part_det, part_focus, part_fut, part_interrog, part_neg, part_restrict, part_verb, part_voc, prep, abbrev, punc, conj, conj_sub, interj, digit, latin
gen ($n = 3$)	m (masculine), f (feminine), na (not applicable)
num ($n = 5$)	s (singular), d (dual), p (plural), u (undefined), na
cas ($n = 5$)	n (nominative), a (accusative), g (genitive), u, na
mod ($n = 5$)	i (indicative), j (jussive), s (subjunctive), u, na
asp ($n = 4$)	i (imperfective), p (perfective), c (command), na
per ($n = 4$)	1, 2, 3, na
vox ($n = 4$)	a (active), p (passive), u, na
stt ($n = 5$)	i (indefinite), d (definite), c (constructive/poss/idafa), u, na
prc0 ($n = 10$)	0, na, Aa_prondem, AlmA_detneg, lA_neg, mA_neg, mA_part, mA_rel
prc1 ($n = 27$)	0, na,<i$_interrog, bi_part, bi_prep, bi_prog, Ea_prep, EalaY_prep, fiy_prep, hA_dem, Ha_fut, ka_prep, la_emph, la_prep, la_rc, libi_prep laHa_emphfut, laHa_rcfut, li_jus, li_prep, min_prep, sa_fut, ta_prep, wa_part, wa_prep, wA_voc, yA_voc
prc2 ($n = 9$)	0, na, fa_conj, fa_conn, fa_rc, fa_sub, wa_conj, wa_part, wa_sub
prc3 ($n = 3$)	0, na, >a_ques
enc ($n = 54$)	0, na, 1p_dobj, 1p_poss, 1p_pron, 1s_dobj, 1s_poss, 1s_pron, 2d_dobj, 2d_poss, 2d_pron, 2p_dobj, 2p_poss, 2p_pron, 2fp_dobj, 2fp_poss, 2fp_pron, 2fs_dobj, 2fs_poss, 2fs_pron, 2mp_dobj, 2mp_poss, 2mp_pron, 2ms_dobj, 2ms_poss, 2ms_pron, 3d_dobj, 3d_poss, 3d_pron, 3p_dobj, 3p_poss, 3p_pron, 3fp_dobj, 3fp_poss, 3fp_pron, 3fs_dobj, 3fs_poss, 3fs_pron, 3mp_dobj, 3mp_poss, 3mp_pron, 3ms_dobj, 3ms_poss, 3ms_pron, Ah_voc, lA_neg, ma_interrog, mA_interrog, man_interrog, man_rel, ma_rel, mA_rel, ma_sub, mA_sub

Table 1: The 14 morphosyntactic categories and their possible values used in Pasha et al. (2014). n indicates the size of the tag set.

with tag dictionary information yields the best accuracy on the Penn Arabic Treebank data set with 91.38%, an absolute improvement of 2.11% over the current state-of-the-art.

2 Fined-Grained Arabic POS Tagging

POS tagging takes a sequence of n words $x_{1:n}$ as input and outputs a corresponding sequence of labels $y_{1:n}$, where x_t is the t-th word in a sentence and $y_t \in T$ is the tag of x_t. In English, a POS tag is typically taken from a single tag set T. By contrast, in morphologically rich languages such as Arabic, a complete POS tag is formed by combining tags from multiple tag sets defined for each morphosyntactic category.

For example, a complete POS tag for the word *Hb* ("love") can be defined as the combination of a noun from the coarse POS category, a nominative (*n*) from the case category, "not applicable" (*na*) from the mood category, and so on. Formally, the fine-grained POS tag y_t^{fine} for a word x_t is defined as the conjunction of the tags $y_t^{(1)} \wedge y_t^{(2)} \wedge ... \wedge y_t^{(k)}$ from k tag sets $T^{(1)}, T^{(2)}, ..., T^{(k)}$. Our purpose is then to predict all morphosyntactic categories for each word — in other words, this can be seen as a multi-class and multi-label sequential labeling problem.

In this paper, we use the 14 morphosyntactic categories[3] used in Pasha et al. (2014), a framework widely used in modern Arabic NLP tools (Pasha et al., 2014; Shahrour et al., 2015; Khalifa et al., 2016). The 14 categories and their possible values are shown in Table 1.

3 Model

In this section, we first briefly describe bi-directional LSTMs. We then present our models which use bi-LSTMs for fine-grained Arabic POS tagging[4]. We also propose a method of incorporating tag dictionary information into our neural models by combining word representations with representations of the sets of possible tags.

3.1 Bi-directional LSTMs

Recurrent neural networks (RNN) (Elman, 1990) are a class of neural networks that are capable of handling sequences of any length. An RNN can be seen as a function that reads the input vector x_t at time step t and calculates a hidden state $\mathbf{h}_t$ using x_t and the previous hidden state $\mathbf{h}_{t-1}$. In classification tasks, the vector $\mathbf{h}_t$ is then fed into the output layer and produces a probability distribution over the possible classes. One of the drawbacks of basic RNNs is their difficulty to train due to the so-called vanishing gradient problem. Long short term memory (LSTM) networks (Hochreiter and Schmidhuber, 1997) address this issue by in-

[3]The categories are: coarse POS (pos), gender (gen), number (num), case (cas), mood (mod), aspect (asp), person (per), voice (vox), state (stt), four proclitics (prc0, prc1, prc2, prc3), and one enclitic (enc).

[4]We do not consider a model that directly predicts full complex tags, since complex tags only found in the test set cannot be predicted by such a model.

troducing memory cells and gate units that capture long-term dependencies.

A bi-directional LSTM network (Graves and Schmidhuber, 2005) is an extension of an LSTM network that allows modeling of past and future dependencies in arbitrary-length input sequences. The output vector $\mathbf{h}_t$ of a bi-LSTM is calculated by concatenating the output vector of the forward directional LSTM that reads the sequence from beginning to end with the output vector of the backward directional LSTM that reads the sequence in the reverse direction.

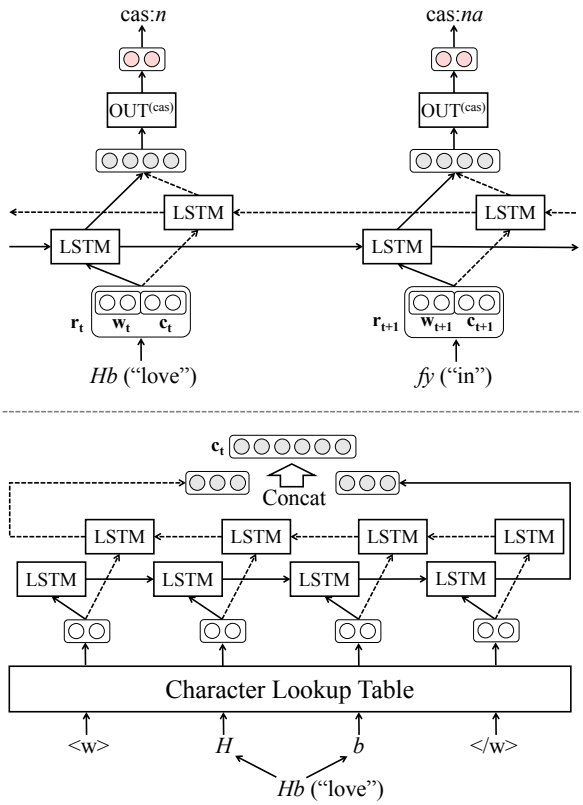

Figure 1: Top: Our baseline model for the category "cas". We have one model for each category, resulting in 14 models in total. Bottom: How to create character-level embeddings. <w> and </w> indicates the beginning and the end of a word.

3.2 Independent Prediction Model

For our baseline method, we use a model that independently predicts each morphosyntactic category using bi-LSTMs. Our baseline is similar to the basic model in Plank et al. (2016). The top part of Figure 1 illustrates an overview of our baseline model. Given a sequence of n words $x_{1:n}$, we encode each word x_t into a vector representation $\mathbf{r}_t = [\mathbf{w}_t; \mathbf{c}_t]$, which is the concatenation of the word embedding $\mathbf{w}_t$ and the character-level embedding $\mathbf{c}_t$. The character-level embedding is computed by concatenating hidden states of the character-level forward LSTM and those of the backward LSTM as depicted in the bottom part of Figure 1.

The vector representation $\mathbf{r}_t$ is then fed into our bi-LSTM model, giving the forward hidden state $\overrightarrow{\mathbf{h}}_t$ and the backward hidden state $\overleftarrow{\mathbf{h}}_t$. Both hidden states are concatenated into single vector $\mathbf{v}_t = [\overrightarrow{\mathbf{h}}_t; \overleftarrow{\mathbf{h}}_t]$ and fed into the output layer. Finally, we obtain the output label y_t by performing a softmax over the tag set vocabulary. We train models separately for each morphosyntactic category, resulting in 14 models in total.

3.3 Joint Prediction Model

Our baseline model does not share any information between morphosyntactic prediction tasks, as it is trained separately. However, it is beneficial to utilize information from other morphosyntactic categories when predicting a label for one category. In order to do this, we adopt a multi-task learning approach (Collobert et al., 2011; Yang et al., 2016; Søgaard and Goldberg, 2016; Bingel and Søgaard, 2017; Martínez Alonso and Plank, 2017). Specifically, we use parameter sharing in the hidden layers of our bi-LSTM model so that we can generate a unified model that can carry information beneficial to each task.

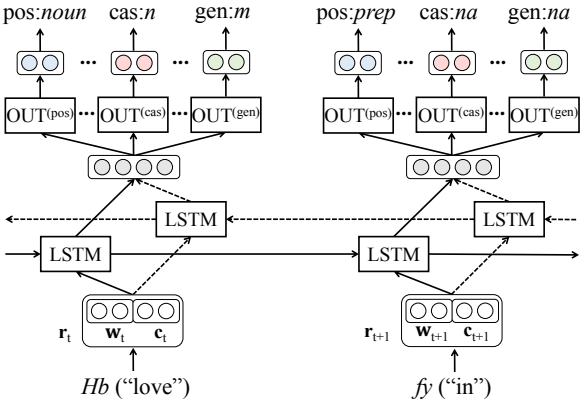

Figure 2: Multi-task bi-directional LSTM model for fine-grained Arabic POS tagging.

Figure 2 shows an overview of our joint model. The output vectors of the bi-LSTMs are fed into multiple output layers, each performing a corresponding morphosyntactic prediction task. Our model trains to minimize the cross-entropy loss

averaged across all the tasks. The loss function for each input word is defined as follows:

$$L(\hat{y}^{fine}, y^{fine}) = \frac{1}{|M|} \sum_{m \in M} L(\hat{y}_m, y_m)$$

where $M = \{pos, cas, gen, ...\}$ is the set of morphosyntactic prediction tasks and $L(\hat{y}_m, y_m)$ is the cross-entropy loss for the category m.

3.4 Encoding Tag Dictionary Information

One of our contributions is to incorporate tag dictionary information into our neural models by combining word representations with representations of the sets of possible tags. Unlike previous approaches that use tag dictionary information provided by a morphological analyzer as a hard constraint (Habash and Rambow, 2005; Pasha et al., 2014; Shahrour et al., 2015), we use it as a soft constraint, as well as an additional feature for our model.

The drawback of using a morphological analyzer in a pipeline fashion is that the model cannot find the correct tag in the disambiguation step if the analyzer does not return any tag candidates. Habash et al. (2016) report in their error analysis that 31.3% of their tagging errors were due to this problem. To cope with this issue, we propose a method of encoding tag dictionary information into our neural models instead of using a morphological analyzer in a pipeline fashion. As such, the output of our tagger is not restricted by the output candidates that are generated by the analyzer, and our method can be applied to POS tagging with an arbitrary tag set.

The bottom part of Figure 3 illustrates how to encode tag dictionary information for the word *Hb* ("love"). First, the input word is given to a tag dictionary that generates sets of possible tags for each morphosyntactic category. The outputs from the dictionary are then fed into the corresponding lookup tables, giving vector representations for possible tags. For each category, we sum over the outputs from the lookup table and then concatenate all the summed vectors into a single vector.

Formally, the encoded vector representation $\mathbf{d}_t$ for the input word x_t is computed by concatenating all the sub-vectors defined for each morphosyntactic category m:

$$\mathbf{d}_t = [\mathbf{d}_t^{(pos)}; ... ; \mathbf{d}_t^{(cas)}; ... ; \mathbf{d}_t^{(gen)}]$$

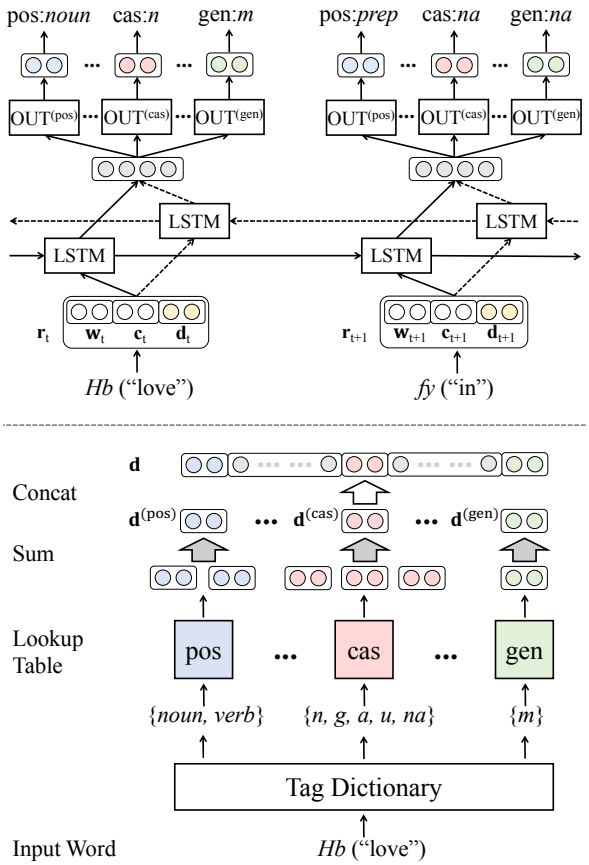

Figure 3: Top: An overview of our proposed model with tag dictionary embeddings. Bottom: Example of how tag dictionary information is encoded.

The sub-vector $\mathbf{d}_t^{(m)}$ is computed with the following equation:

$$\mathbf{d}_t^{(m)} = \sum_{d \in D_t^{(m)}} \mathbf{W}^{(m)} \mathbf{e}_d^{(m)}$$

where $D_t^{(m)}$ is the set of possible tags for the category m given the word x_t, $\mathbf{W}^{(m)}$ is the embedding matrix for the category m, and $\mathbf{e}_d^{(m)}$ is a one-hot vector representing the tag d for the category m. Finally, the resulting vector $\mathbf{d}_t$ is concatenated with the word embedding $\mathbf{w}_t$ and the character-level embedding $\mathbf{c}_t$, forming the input word representation $\mathbf{r}_t = [\mathbf{w}_t; \mathbf{c}_t; \mathbf{d}_t]$ for our model. The top part of Figure 3 illustrates the overall architecture of our proposed model.

4 Experiments

In this section, we present our experimental setup and results. We report tagging accuracy on two data sets: the Penn Arabic Treebank (PATB) data

set and the Arabic Universal Dependencies Treebank (UD Arabic) data set. We also report the effects of tag dictionary information in both data sets.

4.1 Experimental Setup

4.1.1 Implementation Details

We implement all bi-LSTM models using the DyNet library (Neubig et al., 2017). We use the same hyperparameters throughout the independent and joint models, i.e., Adam with cross entropy loss, mini-batch size of a single sentence, 100 dimensions for word embeddings, 50 for character-level embeddings, 10 for each morphosyntactic dictionary embedding, 500 hidden states, 100 dimensions for output layers, random initialization for the embeddings, and no dropout regularization. We do not use external resources for the word embeddings in order to emulate the data availability of earlier work as much as possible. The number of epochs is optimized based on evaluation over the development set, to a maximum of 10 epochs. We use ALMOR (Habash, 2007), which is part of the MADAMIRA distribution (Pasha et al., 2014), alongside the SAMA database (Maamouri et al., 2010c) to create the tag dictionary.

4.1.2 Data Sets

The PATB Data Set

In order to compare our models with the current state-of-the-art tagger, we use the Penn Arabic Treebank (PATB, parts 1, 2 and 3) (Maamouri et al., 2010a, 2011, 2010b) with the same partitioning as Diab et al. (2013). The statistics of the data set are shown in Table 2. The data sets are pre-processed as in Pasha et al. (2014) to correct annotation inconsistencies and to obtain the morphosyntactic feature representation for each word. All the Arabic characters are transliterated according to the Buckwalter transliteration scheme (Buckwalter, 2002) and each numerical digit is substituted with 0.

	Train	Dev	Test
# Sentences	15789	1986	1963
# Words	502991	63136	63168
# Tags	2028	1034	1069

Table 2: Number of sentences, space-delimited words, and fine-grained POS tags in the Penn Arabic Treebank data set.

The UD Arabic Data Set

In order to evaluate the performance of our models on different data in a different tagging scheme, we use the Arabic portion of Universal Dependencies Version 1.4 (Nivre et al., 2016) with the provided gold tokenization. We assume gold tokenization for the sake of simplicity. The statistics of the data set are shown in Table 3.

	Train	Dev	Test
# Sentences	6174	786	704
# Tokens	225853	28263	28268
# Tags	327	214	213

Table 3: Number of sentences, tokens, and fine-grained POS tags in the UD Arabic data set.

For the fine-grained POS tag set, we use the universal POS tags and 16 of the morphological features defined in the UD Arabic data set. The annotations in the UD Arabic data set are automatically converted from the Prague Arabic Dependency Treebank (Smrž et al., 2008). Table 4 shows the lists of possible values for each morphosyntactic category. The annotations in UD Arabic are different from those in PATB with regard to the choice of categories and their granularity, although there are some overlaps in categories such as gender and person. For pre-processing, each numerical digit is substituted with 0.

POS ($n = 17$)	ADJ, ADP, ADV, AUX, CONJ, DET, INTEJ, NOUN, NUM, PART, PRON, PROPN, PUNCT, SCONJ, SYM, VERB, X
Gender ($n = 3$)	Fem, Masc, EMPTY
Number ($n = 4$)	Dual, Plur, Sing, EMPTY
Case ($n = 4$)	Acc, Gen, Nom, EMPTY
Mood ($n = 5$)	Imp, Ind, Jus, Sub, EMPTY
Aspect ($n = 3$)	Imp, Perf, EMPTY
Person ($n = 4$)	1, 2, 3, EMPTY
Voice ($n = 3$)	Act, Pass, EMPTY
Definite ($n = 5$)	Com, Cons, Def, Ind, EMPTY
Abbr ($n = 2$)	Yes, EMPTY
AdpType ($n = 2$)	Prep, EMPTY
Foreign ($n = 2$)	Yes, EMPTY
Negative ($n = 2$)	Negative, EMPTY
NumForm ($n = 3$)	Digit, Word, EMPTY
NumValue ($n = 4$)	1, 2, 3, EMPTY
PronType ($n = 4$)	Dem, Prs, Rel, EMPTY
VerbForm ($n = 2$)	Fin, EMPTY

Table 4: The 17 morphosyntactic categories in the UD scheme (i.e., the universal POS tags and 16 morphological features) and their possible values. n indicates the size of the tag set.

4.1.3 Evaluation

Tagging Accuracy on the PATB data set

We report tagging accuracy over the 14 morphosyntactic categories and their combination,

	pos	gen	num	cas	mod	asp	per	vox	stt	prc0	prc1	prc2	prc3	enc	**All**
CamelParser	96.78	99.41	99.43	92.68	99.13	99.27	99.23	99.08	97.54	99.67	99.63	99.59	99.90	99.61	89.27
Independent	96.31	99.05	99.26	93.17	99.07	99.08	99.10	98.80	97.23	99.62	99.64	99.73	**99.97**	99.44	87.74
+*Dict*	97.07	99.33	99.51	94.70	99.31	99.34	99.35	99.18	98.11	99.48	99.78	**99.78**	**99.97**	99.68	90.17
Joint	96.24	99.27	99.16	93.48	99.18	99.19	99.20	98.91	97.70	99.66	99.64	99.68	**99.97**	99.58	89.49
+*Dict*	**97.21**	**99.50**	**99.59**	**94.76**	**99.41**	**99.44**	**99.47**	**99.25**	**98.24**	**99.71**	**99.81**	99.73	99.96	**99.71**	**91.38**

Table 5: Tagging accuracies on the PATB data set. **All** is the percentage where all categories were correct (i.e., the fine-grained POS tag). +*Dict* indicates the use of the tag dictionary embeddings. Best results are in boldface.

	pos	gen	num	cas	mod	asp	per	vox	stt	prc0	prc1	prc2	prc3	enc	**All**
Joint	96.24	99.27	99.16	93.48	99.18	99.19	99.20	98.91	97.70	99.66	99.64	99.68	99.97	99.58	89.49
+*pos*	**+0.96**	**+0.25**	+0.27	**+1.00**	**+0.25**	+0.21	+0.23	**+0.38**	+0.46	+0.04	+0.09	+0.09	0.00	+0.08	**+1.48**
+*gen*	+0.35	+0.10	+0.18	+0.34	+0.12	+0.12	+0.09	+0.21	+0.19	0.00	-0.06	0.00	-0.01	+0.02	+0.33
+*num*	+0.36	+0.10	**+0.43**	+0.45	+0.06	+0.07	+0.08	+0.17	+0.13	+0.03	-0.02	+0.02	-0.01	+0.01	+0.63
+*cas*	+0.51	+0.13	+0.25	+0.82	**+0.25**	+0.22	+0.23	+0.32	+0.41	-0.01	+0.08	+0.04	0.00	+0.06	+0.99
+*mod*	+0.38	+0.10	+0.14	+0.77	+0.23	+0.23	+0.21	+0.31	+0.39	-0.01	+0.04	+0.05	-0.01	+0.06	+0.82
+*asp*	+0.47	+0.12	+0.22	+0.48	+0.22	+0.22	+0.24	+0.33	+0.33	+0.02	+0.06	+0.03	0.00	+0.03	+0.68
+*per*	+0.26	+0.16	+0.18	+0.72	+0.24	**+0.28**	**+0.29**	+0.36	+0.32	+0.01	+0.08	+0.06	0.00	+0.07	+0.78
+*vox*	+0.27	+0.13	+0.15	+0.65	+0.21	+0.21	+0.19	+0.31	+0.29	+0.01	-0.07	-0.01	-0.01	+0.04	+0.60
+*stt*	+0.60	+0.12	+0.20	+0.87	+0.23	+0.23	+0.22	+0.35	**+0.47**	+0.03	+0.07	+0.05	-0.01	+0.05	+0.99
+*prc0*	+0.31	+0.10	+0.16	+0.56	+0.06	+0.08	+0.08	+0.16	+0.16	**+0.06**	+0.06	+0.05	0.00	0.00	+0.56
+*prc1*	+0.40	+0.09	+0.21	+0.50	+0.06	-0.02	+0.06	+0.14	+0.11	+0.02	**+0.15**	+0.02	0.00	0.00	+0.69
+*prc2*	+0.23	+0.04	+0.16	+0.23	0.00	-0.01	+0.04	+0.12	+0.05	+0.04	-0.09	**+0.10**	-0.01	-0.02	+0.35
+*prc3*	+0.14	+0.05	+0.16	+0.33	+0.07	+0.04	+0.04	+0.15	+0.09	+0.01	-0.05	+0.05	-0.01	+0.01	+0.28
+*enc*	+0.26	+0.02	+0.12	+0.53	+0.09	+0.07	+0.07	+0.21	+0.22	+0.02	0.00	+0.04	-0.01	**+0.12**	+0.63
+*all*	+0.97	+0.23	+0.43	+1.28	+0.23	+0.25	+0.27	+0.34	+0.54	+0.05	+0.17	+0.05	-0.01	+0.13	+1.89

Table 6: Performance comparison of the different models, each of which uses a single morphosyntactic category in its tag dictionary embeddings, on the PATB data set. +*m* in the leftmost column indicates the use of the category *m* to form the tag dictionary embeddings. +*all* indicates the use of all categories to form the tag dictionary embeddings. Boldfaced numbers represent the largest improvement in the category to predict (minimum of 0.05% absolute).

i.e., the fine-grained POS tag (**All**). For comparison, we use CamelParser (Shahrour et al., 2015), the current state-of-the-art tagger. CamelParser is an improved version of the previous state-of-the-art tagger MADAMIRA (Pasha et al., 2014), which ranks the possible analyses provided by a morphological analyzer using SVMs. CamelParser adjusts the outputs of MADAMIRA by utilizing case-state classifiers that incorporate additional syntactic information provided by a dependency parser and hand-written rules. The tag set used in CamelParser is compatible with the 14 morphosyntactic categories we use.

Tagging Accuracy on the UD Arabic data set
For the UD Arabic data set, we report tagging accuracy over the 17 morphosyntactic categories (i.e., the universal POS tags and 16 morphological features) and their combination (**All**). We use independent models with and without tag dictionary information and joint models with and without tag dictionary information for this data set.

Most Influential Categories
For both data sets, we conduct additional experiments to investigate which morphosyntactic category in the tag dictionary embeddings contributes most to the performance. Specifically, instead of using all morphosyntactic categories to create the tag dictionary embeddings, we use only one at a time. In other words, we skip the last step of concatenating all the sub-vectors defined for each morphosyntactic category, and use only one of the sub-vectors for the tag dictionary embeddings.

4.2 Results

4.2.1 The PATB Data Set

Our Models vs CamelParser

Table 5 illustrates our experimental results on the PATB data set. The best performing model was the joint model with tag dictionary embeddings (+*Dict*), achieving an accuracy of 91.38% on the strictest metric "**All**" (i.e., the fine-grained POS tag) with an absolute improvement of 2.11% over CamelParser, the current state-of-the-art tagger. This model outperforms CamelParser in every morphosyntactic category. Among these categories, the most notable improvement is the case category (cas) with an absolute improvement of 2.08% over the current state-of-the-art system. Leaving out the dictionary embeddings (+*Dict*) reduces the performance by 1.89% absolute, but still outperforms CamelParser without using any addi-

	POS	Gender	Number	Case	Mood	Aspect	Person	Voice	Definite
Independent	95.15	97.28	96.38	93.76	99.56	99.35	99.37	99.14	96.40
+Dict	96.08	98.06	97.23	94.86	99.68	99.51	99.47	99.16	97.09
Joint	95.92	97.96	96.69	94.60	99.67	99.50	99.45	99.21	96.67
+Dict	**96.64**	**98.32**	**97.47**	**95.43**	**99.69**	**99.58**	**99.59**	**99.32**	**97.35**

	Abbr	AdpType	Foreign	Negative	NumForm	NumValue	PronType	VerbForm	**All**
Independent	99.88	99.75	99.16	99.99	99.88	99.80	99.76	99.69	86.45
+Dict	**100.00**	99.84	99.58	99.99	**99.90**	99.80	99.79	99.73	89.17
Joint	99.99	99.85	99.47	99.99	**99.90**	**99.98**	99.81	99.78	90.36
+Dict	99.99	**99.86**	**99.66**	99.99	99.89	**99.98**	**99.84**	**99.84**	**91.68**

Table 7: Tagging accuracies on the UD Arabic data set. **All** is the percentage where all categories were correct (i.e., the fine-grained POS tag). *+Dict* indicates the use of the tag dictionary embeddings.

tional resources such as a morphological analyzer or a dependency parser, indicating the effectiveness of joint modeling of morphosyntactic categories. On the other hand, the independent model gives an accuracy of 87.74%, which is 1.53% absolute worse than CamelParser. However, adding dictionary embeddings (*+Dict*) enhances the performance with an absolute improvement of 2.43% and yields the second-best accuracy, showing the impact of the additional dictionary feature.

Most Influential Categories

Which morphosyntactic category in the tag dictionary embeddings contributes most to the performance? Table 6 compares the performance of the different models, each of which uses a single morphosyntactic category in its tag dictionary embeddings. The category that contributes most in the tag dictionary embeddings is the coarse POS category (*+pos*) with an absolute improvement of 1.48% on the metric "**All**". It is worth mentioning that case and state categories are tied for the second most contributing category, which supports CamelParser's idea that improving the prediction of case and state categories will provide further performance gains.

Looking at the effects on each category to predict, the embeddings for coarse POS (*+pos*) give the best improvement in 5 categories: coarse POS (pos), gender (gen), case (cas), mood (mod), and voice (vox). We can see that the information carried by the coarse POS category plays a central role for predicting other morphosyntactic categories, especially for the case category. On the other hand, in 8 categories, the best improvement was achieved when the category used for the tag dictionary embeddings was the same as the category to predict. The 8 categories were: coarse POS (pos), number (num), person (per), state (stt), three of the proclitics (prc0, prc1, prc2), and enclitic (enc). This result suggests that the tag dictionary embeddings of a given category behave as a soft constraint when predicting the same category, which makes intuitive sense.

4.2.2 The UD Arabic Data Set

Results of Our Models

Table 7 illustrates our experimental results on the UD Arabic data set. The independent model gives an accuracy of 86.34% on the metric "**All**" (i.e., the fine-grained POS tag). Adding the tag dictionary embeddings (*+Dict*) improves the accuracy with an absolute improvement of 2.72%. Unlike the PATB data set, the joint model outperformed both independent models regardless of the use of the tag dictionary embeddings. The best performing model was the joint model with the tag dictionary embeddings (*+Dict*), achieving an accuracy of 91.68%. We can observe that the overall results show similar tendencies to the results on the PATB data set in spite of the different annotation schemes.

Most Influential Categories

Table 8 compares the performance of the different models, each of which uses a single morphosyntactic category in its tag dictionary embeddings, on the UD Arabic data set. As in the results on the PATB data set, the coarse POS category (*+pos*) is the category that contributes the most in the tag dictionary embeddings, giving an absolute improvement of 0.92% on the metric "**All**". It also gives the best improvement in 8 categories: POS, Aspect, Case, Definite, Foreign, Gender, Number, Person, and Voice. This result confirmed that the possible tag information from the POS category is more effective than information from the other categories.

On the other hand, unlike in the PATB data set, we do not observe a relationship between the cat-

	POS	Gender	Number	Case	Mood	Aspect	Person	Voice	Definite
Joint	95.92	97.96	96.69	94.60	99.67	99.50	99.45	99.21	96.67
+*pos*	**+0.55**	**+0.30**	**+0.49**	**+0.58**	+0.04	+0.07	**+0.14**	**+0.15**	**+0.56**
+*gen*	-0.20	+0.01	-0.07	+0.05	**+0.06**	**+0.09**	+0.09	+0.13	-0.01
+*num*	+0.12	+0.06	+0.44	+0.32	+0.04	+0.01	+0.13	+0.04	+0.25
+*cas*	+0.19	+0.01	+0.33	+0.33	+0.02	+0.02	+0.08	+0.04	+0.37
+*mod*	+0.15	-0.09	+0.24	+0.26	+0.02	+0.07	+0.13	+0.14	+0.19
+*asp*	+0.19	0.00	+0.23	+0.33	-0.02	+0.06	+0.11	+0.09	+0.29
+*per*	+0.20	+0.03	+0.26	+0.38	+0.01	+0.07	+0.12	+0.07	+0.28
+*vox*	+0.08	+0.01	+0.17	+0.13	-0.01	+0.05	+0.09	+0.14	+0.21
+*stt*	+0.08	-0.06	+0.25	+0.48	-0.04	+0.04	+0.08	+0.07	+0.41
+*prc0*	-0.03	-0.06	+0.27	+0.10	+0.04	+0.02	+0.08	-0.02	+0.31
+*prc1*	-0.01	-0.01	+0.18	+0.20	+0.03	+0.02	+0.06	+0.07	+0.14
+*prc2*	-0.17	-0.12	+0.21	+0.15	+0.03	0.00	+0.01	-0.03	+0.22
+*prc3*	+0.07	-0.14	+0.29	+0.21	-0.02	+0.02	+0.08	+0.06	+0.25
+*enc*	-0.01	-0.22	+0.40	+0.10	-0.02	-0.04	-0.03	-0.05	+0.28
+*all*	+0.72	+0.36	+0.78	+0.83	+0.02	+0.08	+0.14	+0.11	+0.68

	Abbr	AdpType	Foreign	Negative	NumForm	NumValue	PronType	VerbForm	**All**
Joint	99.99	99.85	99.47	99.99	99.90	99.98	99.81	99.78	90.36
+*pos*	+0.01	-0.01	**+0.16**	0.00	+0.01	-0.02	+0.03	+0.03	**+0.92**
+*gen*	+0.01	0.00	+0.03	0.00	0.00	0.00	+0.01	+0.04	+0.08
+*num*	0.00	-0.02	+0.06	0.00	+0.02	0.00	+0.01	+0.03	+0.34
+*cas*	+0.01	0.00	+0.08	0.00	+0.02	0.00	0.00	+0.03	+0.30
+*mod*	0.00	-0.04	+0.03	0.00	0.00	-0.01	+0.01	+0.04	+0.33
+*asp*	+0.01	-0.01	+0.07	0.00	-0.01	0.00	+0.01	+0.02	+0.48
+*per*	0.00	-0.01	+0.16	0.00	+0.02	0.00	0.00	+0.02	+0.50
+*vox*	+0.01	-0.03	+0.09	0.00	-0.01	0.00	+0.01	+0.01	+0.06
+*stt*	+0.01	-0.03	+0.06	0.00	-0.01	-0.01	+0.01	+0.01	+0.28
+*prc0*	+0.01	-0.01	+0.09	0.00	0.00	-0.01	+0.01	+0.03	+0.13
+*prc1*	+0.01	-0.01	+0.12	+0.01	-0.03	-0.01	+0.01	0.00	+0.30
+*prc2*	+0.01	-0.02	+0.11	0.00	-0.02	-0.01	0.00	0.00	-0.02
+*prc3*	+0.01	-0.03	-0.02	0.00	0.00	-0.03	0.00	-0.02	-0.01
+*enc*	+0.01	-0.04	+0.03	-0.01	0.00	0.00	0.00	+0.01	-0.02
+*all*	0.00	+0.01	+0.19	0.00	-0.01	0.00	+0.03	+0.06	+1.32

Table 8: Performance comparison of the different models, each of which uses a single morphosyntactic category in its tag dictionary embeddings, on the UD Arabic data set. +*m* in the leftmost column indicates the use of the category *m* to form the tag dictionary embeddings. +*all* indicates the use of all categories to form the tag dictionary embeddings. Boldfaced numbers represent the largest improvement in the category to predict (minimum of 0.05% absolute).

egory used for the tag dictionary embeddings and the category to predict, presumably because of the difference in the annotation schemes.

5 Related Work

Diab et al. (2004) proposed a segmentation-based approach, in which they tag each clitic-segmented token using SVMs. Mohamed and Kübler (2010) proposed a word-based approach which takes space-delimited words as inputs and uses memory-based learning. Their experiment showed that the word-based approach performed better than the segmentation-based approach, avoiding segmentation error propagation. Zhang et al. (2015) proposed joint modeling of segmentation, POS tagging, and dependency parsing using a randomized greedy algorithm. The aforementioned studies were focused on tagging with reduced POS tag sets whose sizes ranged from 12 to 993. However, we use one of the most fine-grained POS tag sets, with about 2,000 tags appearing in our training set.

In the context of fine-grained POS tagging, Mueller et al. (2013) presented an approximated higher-order CRF for morphosyntactic tagging across six languages, assuming gold clitic segmentation. Pasha et al. (2014) used an analyze-and-disambiguate approach, in which they ranked the possible analyses provided by a morphological analyzer for each space-delimited word. The state-of-the-art tagger (Shahrour et al., 2015) extended their model by adjusting the outputs of Pasha et al.'s tagger by utilizing case-state classifiers that incorporate additional syntactic information provided by a dependency parser and hand-written rules.

Compared to their approaches, our model is simple but powerful: It does not assume gold clitic segmentation, since segmentation is also modeled as part of the morphosyntactic categories, nor does it require the additional pipeline process of syntactic parsing. Nonetheless, it is more accurate than the current state-of-the-art.

Another related line of work tackles sequential labeling problems using multi-task learning with deep neural networks and investigates situations where multi-task learning leads to improvements in performance (Søgaard and Goldberg, 2016; Bingel and Søgaard, 2017; Martínez Alonso and Plank, 2017). Although our main focus is not on investigating the most effective task combination, it can be worth experimenting with various configurations in our settings.

With regard to the use of outputs from a morphological analyzer as additional features, our work is closely related to Bohnet et al. (2013) and Shen et al. (2016). Bohnet et al. (2013) presented a joint approach for morphological and syntactic analysis for morphologically rich languages, integrating additional features that encode whether a tag is in the dictionary or not. Shen et al. (2016) proposed an approach in which they encode a sequence of possible morphosyntactic tags provided by a morphological analyzer using bi-directional LSTMs. In contrast, we provide an alternative way of encoding this information, as well as an analysis on the most influential categories in the encoded tag embeddings.

6 Conclusions

We presented an approach for fine-grained Arabic POS tagging that jointly models each morphosyntactic tagging task using a multi-task learning framework. We also proposed a method of incorporating tag dictionary information into our neural models by combining word representations with representations of the sets of possible tags. The joint model with tag dictionary information results in the best accuracy of 91.38% with an absolute improvement of 2.11% over the current state-of-the-art tagger. In addition, our experiments showed that the proposed method of encoding tag dictionary information improves the tagging accuracy even on a data set with different annotations.

One potential future direction to explore is domain adaptation to Arabic dialects, since our approach is easily applicable as it does not require construction of a morphological analyzer for each dialect. Another direction is to make use of publicly available dictionaries such as Wiktionary to construct a tag dictionary.

Acknowledgments

We would like to thank the anonymous reviewers, Hiroki Ouchi, Yuichiro Sawai, Taishi Ikeda, Hitoshi Manabe, and Michael Wentao Li for their valuable comments and suggestions. We also appreciate Nizar Habash and Salam Khalifa for providing the pre-processed data and running the CamelParser experiment.

References

Joachim Bingel and Anders Søgaard. 2017. Identifying beneficial task relations for multi-task learning in deep neural networks. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. Association for Computational Linguistics, Valencia, Spain, pages 164–169. http://www.aclweb.org/anthology/E17-2026.

Bernd Bohnet, Joakim Nivre, Igor Boguslavsky, Richárd Farkas, Filip Ginter, and Jan Hajič. 2013. Joint Morphological and Syntactic Analysis for Richly Inflected Languages. *Transactions of the Association for Computational Linguistics* 1:415–428.

Tim Buckwalter. 2002. Buckwalter Arabic Morphological Analyzer Version 1.0 LDC2002L49. Linguistic Data Consortium (LDC, Philadelphia US).

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research* 12(Aug):2493–2537.

Mona Diab, Nizar Habash, Owen Rambow, and Ryan Roth. 2013. LDC Arabic Treebanks and Associated Corpora: Data Divisions Manual. In *arXiv preprint arXiv:1309.5652*.

Mona Diab, Kadri Hacioglu, and Daniel Jurafsky. 2004. Automatic Tagging of Arabic Text: From Raw Text to Base Phrase Chunks. In *Proceedings of the 2004 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Short Papers*. Association for Computational Linguistics, Boston, Massachusetts, USA, pages 149–152. http://anthology.aclweb.org/N/N04/N04-4038.pdf.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive Science* 14(2):179–211.

Alex Graves and Jürgen Schmidhuber. 2005. Framewise Phoneme Classification with Bidirectional LSTM and Other Neural Network Architectures. *Neural Networks* 18(5):602–610.

Nizar Habash. 2007. Arabic Morphological Representations for Machine Translation. *Arabic Computational Morphology: Knowledge-based and Empirical Methods* pages 263–285.

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)*. Association for Computational Linguistics, Ann Arbor, Michigan, USA, pages 573–580. https://doi.org/10.3115/1219840.1219911.

Nizar Habash, Anas Shahrour, and Muhamed Al-Khalil. 2016. Exploiting Arabic Diacritization for High Quality Automatic Annotation. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*. pages 4298–4303. http://www.lrec-conf.org/proceedings/lrec2016/pdf/878_Paper.pdf.

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

Salam Khalifa, Nasser Zalmout, and Nizar Habash. 2016. YAMAMA: Yet Another Multi-Dialect Arabic Morphological Analyzer. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 223–227. http://aclweb.org/anthology/C16-2047.

Mohamed Maamouri, Ann Bies, Seth Kulick, Fatma Gaddeche, Wigdan Mekki, Sondos Krouna, Basma Bouziri, and Wadji Zaghouani. 2010a. Arabic Treebank: Part 1 v 4.1. Linguistic Data Consortium (LDC, Philadelphia US).

Mohamed Maamouri, Ann Bies, Seth Kulick, Fatma Gaddeche, Wigdan Mekki, Sondos Krouna, Basma Bouziri, and Wadji Zaghouani. 2011. Arabic Treebank: Part 2 v 3.1. Linguistic Data Consortium (LDC, Philadelphia US).

Mohamed Maamouri, Ann Bies, Seth Kulick, Sondos Krouna, Fatma Gaddeche, and Wadji Zaghouani. 2010b. Arabic Treebank: Part 3 v 3.2. Linguistic Data Consortium (LDC, Philadelphia US).

Mohamed Maamouri, David Graff, Basma Bouziri, Sondos Krouna, Ann Bies, and Seth Kulick. 2010c. LDC Standard Arabic Morphological Analyzer (SAMA) Version 3.1 LDC2010L01. Linguistic Data Consortium (LDC, Philadelphia US).

Héctor Martínez Alonso and Barbara Plank. 2017. When is multitask learning effective? semantic sequence prediction under varying data conditions. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*. Association for Computational Linguistics, Valencia, Spain, pages 44–53. http://www.aclweb.org/anthology/E17-1005.

Emad Mohamed and Sandra Kübler. 2010. Is Arabic Part of Speech Tagging Feasible Without Word Segmentation? In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, Los Angeles, California, USA, pages 705–708. http://www.aclweb.org/anthology/N10-1105.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient Higher-Order CRFs for Morphological Tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 322–332. http://www.aclweb.org/anthology/D13-1032.

Graham Neubig, Chris Dyer, Yoav Goldberg, Austin Matthews, Waleed Ammar, Antonios Anastasopoulos, Miguel Ballesteros, David Chiang, Daniel Clothiaux, Trevor Cohn, Kevin Duh, Manaal Faruqui, Cynthia Gan, Dan Garrette, Yangfeng Ji, Lingpeng Kong, Adhiguna Kuncoro, Gaurav Kumar, Chaitanya Malaviya, Paul Michel, Yusuke Oda, Matthew Richardson, Naomi Saphra, Swabha Swayamdipta, and Pengcheng Yin. 2017. Dynet: The Dynamic Neural Network Toolkit. *arXiv preprint arXiv:1701.03980* .

Joakim Nivre, Željko Agić, Lars Ahrenberg, Maria Jesus Aranzabe, Masayuki Asahara, Aitziber Atutxa, Miguel Ballesteros, John Bauer, Kepa Bengoetxea, Yevgeni Berzak, Riyaz Ahmad Bhat, Eckhard Bick, Carl Börstell, Cristina Bosco, Gosse Bouma, Sam Bowman, Gülşen Cebirolu Eryiit, Giuseppe G. A. Celano, Fabricio Chalub, Çar Çöltekin, Miriam Connor, Elizabeth Davidson, Marie-Catherine de Marneffe, Arantza Diaz de Ilarraza, Kaja Dobrovoljc, Timothy Dozat, Kira Droganova, Puneet Dwivedi, Marhaba Eli, Tomaž Erjavec, Richárd Farkas, Jennifer Foster, Claudia Freitas, Katarína Gajdošová, Daniel Galbraith, Marcos Garcia, Moa Gärdenfors, Sebastian Garza, Filip Ginter, Iakes Goenaga, Koldo Gojenola, Memduh Gökrmak, Yoav Goldberg, Xavier Gómez Guinovart, Berta Gonzáles Saavedra, Matias Grioni, Normunds Grūzītis, Bruno Guillaume, Jan Hajič, Linh Hà M, Dag Haug, Barbora Hladká, Radu Ion, Elena Irimia, Anders Johannsen, Fredrik Jørgensen, Hüner Kaşkara, Hiroshi Kanayama, Jenna Kanerva, Boris Katz, Jessica Kenney, Natalia Kotsyba, Simon Krek, Veronika Laippala, Lucia Lam, Phng Lê Hng, Alessandro Lenci, Nikola Ljubešić, Olga

Lyashevskaya, Teresa Lynn, Aibek Makazhanov, Christopher Manning, Cătălina Mărănduc, David Mareček, Héctor Martínez Alonso, André Martins, Jan Mašek, Yuji Matsumoto, Ryan McDonald, Anna Missilä, Verginica Mititelu, Yusuke Miyao, Simonetta Montemagni, Keiko Sophie Mori, Shunsuke Mori, Bohdan Moskalevskyi, Kadri Muischnek, Nina Mustafina, Kaili Müürisep, Lng Nguyn Th, Huyn Nguyn Th Minh, Vitaly Nikolaev, Hanna Nurmi, Petya Osenova, Robert Östling, Lilja Øvrelid, Valeria Paiva, Elena Pascual, Marco Passarotti, Cenel-Augusto Perez, Slav Petrov, Jussi Piitulainen, Barbara Plank, Martin Popel, Lauma Pretkalnia, Prokopis Prokopidis, Tiina Puolakainen, Sampo Pyysalo, Alexandre Rademaker, Loganathan Ramasamy, Livy Real, Laura Rituma, Rudolf Rosa, Shadi Saleh, Baiba Saulīte, Sebastian Schuster, Wolfgang Seeker, Mojgan Seraji, Lena Shakurova, Mo Shen, Natalia Silveira, Maria Simi, Radu Simionescu, Katalin Simkó, Mária Šimková, Kiril Simov, Aaron Smith, Carolyn Spadine, Alane Suhr, Umut Sulubacak, Zsolt Szántó, Takaaki Tanaka, Reut Tsarfaty, Francis Tyers, Sumire Uematsu, Larraitz Uria, Gertjan van Noord, Viktor Varga, Veronika Vincze, Lars Wallin, Jing Xian Wang, Jonathan North Washington, Mats Wirén, Zdeněk Žabokrtský, Amir Zeldes, Daniel Zeman, and Hanzhi Zhu. 2016. Universal Dependencies 1.4. LINDAT/CLARIN digital library at the Institute of Formal and Applied Linguistics, Charles University. http://hdl.handle.net/11234/1-1827.

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. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*. Reykjavik, Iceland, volume 14, pages 1094–1101. http://www.lrec-conf.org/proceedings/lrec2014/pdf/593_Paper.pdf.

Barbara Plank, Anders Søgaard, and Yoav Goldberg. 2016. Multilingual Part-of-Speech Tagging with Bidirectional Long Short-Term Memory Models and Auxiliary Loss. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 412–418. http://anthology.aclweb.org/P16-2067.

Anas Shahrour, Salam Khalifa, and Nizar Habash. 2015. Improving Arabic Diacritization through Syntactic Analysis. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Lisbon, Portugal, pages 1309–1315. http://aclweb.org/anthology/D15-1152.

Qinlan Shen, Daniel Clothiaux, Emily Tagtow, Patrick Littell, and Chris Dyer. 2016. The Role of Context in Neural Morphological Disambiguation. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. The COLING 2016 Organizing Committee, Osaka, Japan, pages 181–191. http://aclweb.org/anthology/C16-1018.

Otakar Smrž, Viktor Bielicky, and Jan Hajic. 2008. Prague Arabic Dependency Treebank: A Word on the Million Words. In *Proceedings of the Workshop on Arabic and Local Languages (LREC 2008)*. pages 16–23.

Anders Søgaard and Yoav Goldberg. 2016. Deep multi-task learning with low level tasks supervised at lower layers. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. Association for Computational Linguistics, Berlin, Germany, pages 231–235. http://anthology.aclweb.org/P16-2038.

Zhilin Yang, Ruslan Salakhutdinov, and William Cohen. 2016. Multi-Task Cross-Lingual Sequence Tagging from Scratch. *arXiv preprint arXiv:1603.06270* .

Yuan Zhang, Chengtao Li, Regina Barzilay, and Kareem Darwish. 2015. Randomized Greedy Inference for Joint Segmentation, POS Tagging and Dependency Parsing. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Denver, Colorado, USA, pages 42–52. http://www.aclweb.org/anthology/N15-1005.