Character-Aware Neural Morphological Disambiguation

Alymzhan Toleu **Gulmira Tolegen** **Aibek Makazhanov**
National Laboratory Astana, Nazarbayev University
53 Kabanbay batyr ave., Astana, Kazakhstan
alymzhan.toleu@gmail.com, gulmira.tolegen.cs@gmail.com,
aibek.makazhanov@nu.edu.kz

Abstract

We develop a language-independent, deep learning-based approach to the task of morphological disambiguation. Guided by the intuition that the correct analysis should be "most similar" to the context, we propose dense representations for morphological analyses and surface context and a simple yet effective way of combining the two to perform disambiguation. Our approach improves on the language-dependent state of the art for two agglutinative languages (Turkish and Kazakh) and can be potentially applied to other morphologically complex languages.

1 Introduction

Morphological disambiguation (MD) is a long standing problem in processing morphologically complex languages (MCL). POS tagging is a somewhat related problem, however in MD, in addition to POS tags, one typically has to predict lemmata (roots hereinafter) that surface forms stem from and morphemes[1] they bear. For example, depending on the context, a Turkish word *adam* can be analyzed as: (i) *a man* – $[adam]_1$+$[Noun]_2$+$[A3sg+Pnon+Nom]_3$ or (ii) *my island* – $[ada]_1$+$[Noun]_2$+$[A3sg+P1sg\ +Nom]_3$ (Hakkani-Tür et al., 2002). Thus, if one counts analyses as tags, MD can be cast as a tagging problem with an extremely large tagset. This fact discourages direct application of the state of the art approaches designed for small fixed tagsets.

To develop a language independent dense representation of the analyses, we segment[2] an analysis into (i) the root, (ii) its POS and (iii) the morpheme chain (MC). We then proceed to jointly learn the embeddigns for the root and the POS segments and to combine them and the MC segment representation into a single dense representation. MC segments are represented as binary vectors that, for a given analysis, encode presence or absence of *each* morpheme found in the train set. This ensures language independence and contrasts previous work (at least on Turkish and Kazakh), where only certain morphemes are chosen as features depending on their position (Assylbekov et al., 2016; Hakkani-Tür et al., 2002) or presence (Makhambetov et al., 2015) in an analysis, or the authors' intuition (Yildiz et al., 2016; Tolegen et al., 2016; Sak et al., 2007).

Apart from the sparseness of analyses distribution MCL notoriously raise free word order and long dependency issues. Thus, decoding analysis sequences using only the leftmost context may not be enough. To address this we leverage the rightmost context as well. We model the left- and rightmost surface context in two ways: using (i) BiLSTM (Greff et al., 2015) with a character-based sub-layer (Ling et al., 2015) and (ii) with a feed forward network on word embeddings. We then entertain the idea that given a word with multiple analyses and its surface context, the correct analysis might be "closer" to the context. Following our intuition, we have tried computing the distance between the analysis and the context representations, and a simple dot product (as in unnormalized cosine similarity) has yielded the best performance.

We evaluate our approach on Turkish and Kazakh data sets, using several baselines (including the state of the art methods for both languages) and a variety of settings and metrics. In terms of general accuracy our approach has achieved a nearly 1% improvement over the state of the art for Turkish and a marginal improvement for Kazakh.

[1] We use the term *morpheme* for its universal recognition within the community. A more appropriate term might be *grammeme*, i.e. a value of grammatical category.

[2] Such a segmentation is denoted by the squared brackets numbered in the respective order (cf. Turkish example).

Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 666–671
Vancouver, Canada, July 30 - August 4, 2017.
https://doi.org/10.18653/v1/P17-2105

Our contribution amounts to the following: (i) a general MD framework for MCL that can be analyzed in <root, POS, MC> triplets; (ii) improvement on language-dependent state of the art for Turkish and Kazakh.

2 Models

In this section we describe our approach to encoding morphological analyses and the context into the embeddings and combining them to perform morphological disambiguation.

2.1 Morphological Representation

We treat a morphological analysis as a combination of three main constituents: the root, its POS and the morpheme chain. These constituents are represented as d_r, d_p, and d_m-dimensional vectors respectively. The former two vectors correspond to dense word embeddings (Collobert et al., 2011), and the latter is a binary vector which encodes the presence of a certain morpheme in the chain. The size of the binary vector, d_m, is equal to the size of the morpheme dictionary obtained from the data.

Given a sentence and the j-th surface word form with N analyses, we represent the k-th analysis as:

$$\mathbf{A}_j^k = tanh(\mathbf{W}^r r_k + \mathbf{W}^p p_k + \mathbf{W}^m m_k) \quad (1)$$

where $A_j \in \mathbf{R}^{d_h \times |N|}$, d_h is the dimension of each analysis embedding, $r_k \in \mathbf{R}^{d_r \times 1}, p_k \in \mathbf{R}^{d_p \times 1}, m_k \in \{0,1\}^{d_p \times 1}$ are constituent vectors of the k-th analysis, and $\mathbf{W}^r \in \mathbf{R}^{d_h \times d_r}$, $\mathbf{W}^p \in \mathbf{R}^{d_h \times d_p}$,$\mathbf{W}^m \in \mathbf{R}^{d_h \times d_m}$ are the model parameters. The bias term was left out for clarity. This representation is shown on Figure 1 (bottom).

2.2 Recurrent Neural Disambiguation

The model architecture is shown on Figure 1. It consists of two main blocks that learn the surface context (top) and the morphological analyses representations (bottom).

When it comes to modeling context via word embeddings for morphologically complex languages, it is impractical to actually store vectors for all words, since majority of words in such languages has a large number of surface realizations. Our solution to this problem is to construct a surface word representation from characters that not only reduces data sparseness, but also help in dealing with the out-of-vocabulary (OOV) words (Ling et al., 2015). We represent each character of each word as a vector $x_i \in \mathbf{R}^{d_c}$ and the

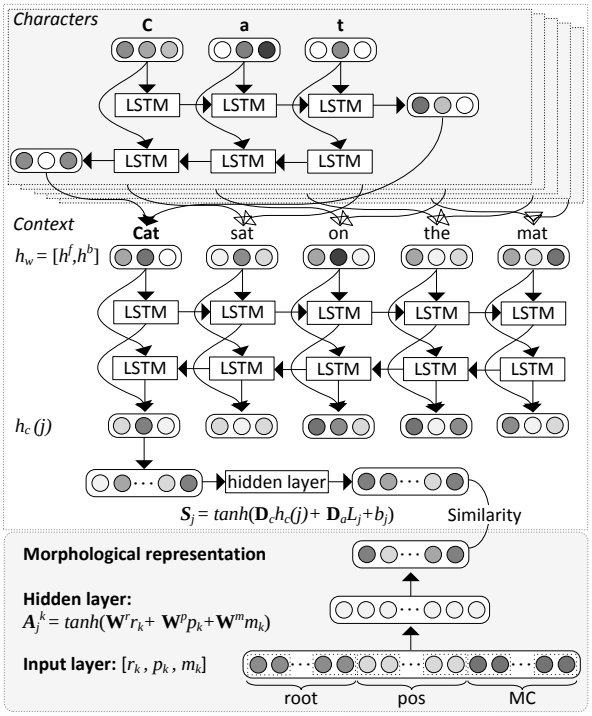

Figure 1: Model architecture

entire embedding matrix as $\mathbf{E}^c \in \mathbf{R}^{d_c \times |C|}$, where C is the character vocabulary extracted from the training set including alphanumeric characters and other possible symbols. Given an input surface word w_i with its character embeddings $x_1, ..., x_n$, the hidden state h_t at the time step t can be computed via the following Vanilla LSTM calculations:

$$i_t = \sigma(\mathbf{W}^i x_t + \mathbf{U}^i h_{t-1} + b^i) \quad (2)$$
$$f_t = \sigma(\mathbf{W}^f x_t + \mathbf{U}^f h_{t-1} + b^f) \quad (3)$$
$$o_t = \sigma(\mathbf{W}^o x_t + \mathbf{U}^o h_{t-1} + b^o) \quad (4)$$
$$z_t = tanh(\mathbf{W}^z x_t + \mathbf{U}^z h_{t-1} + b^z) \quad (5)$$
$$c_t = f_t \odot c_{t-1} + i_t \odot z_t \quad (6)$$
$$h_t = o_t \odot tanh(c_t) \quad (7)$$

where $\sigma(\cdot)$ and $tanh(\cdot)$ are the non-linear functions. i_t, f_t, o_t are referred to three gates: *input*, *forget*, *output* that control the information flow of inputs. Parameters of the LSTM are W^*, U^*, b^*, where $*$ can be any of $\{i, f, o, g\}$. The peephole connections were left out for clarity.

We use both forward and backward LSTM to learn word representations obtained by concatenation of the last states in both direction h^f and h^b, e.g. $h_w = \left[h^f, h^b\right]$. Character-based word embeddings obtained in this manner do not yet con-

tain the context information on a sentence level. Thus, we adopt another LSTM to learn context-sensitive information for each word in both directions. We denote the concatenation of the embeddings learned from the forward and backward LSTM states as $h_c(j) \in \mathbf{R}^{2hs \times 1}$ (where hs is the output size for the j-th word), and represent surface context as:

$$S_j = tanh(\mathbf{D}_c h_c(j) + b_j) \quad (8)$$

where $S_j \in \mathbf{R}^{d_h \times 1}$ is a hidden layer output, $h_c(j) \in \mathbf{R}^{2hs \times 1}$is a context vector of the j-th word, and $b_j \in \mathbf{R}^{d_h \times 1}$ is a bias term.

For the final prediction, we score each analysis by computing the inner product between its representation and the context's representations:

$$P_j^k = S_j \cdot \mathbf{A}_j^k \quad (9)$$

where S_j and A_j^k are computed as equations (8) and (1) respectively. We normalize the obtained scores using softmax and choose the analysis with the maximum score as the correct one. In what follows we refer to this model as **BiLSTM**.

Finally, in a separate setting, in addition to the surface context in the hidden layer we also incorporate the immediate (left and right) morphological context in the form of the average of the analyses representations:

$$S_j^* = tanh(\mathbf{D}_c h_c(j) + \mathbf{D}_a L_j + b_j) \quad (10)$$

where $L_j \in \mathbf{R}^{2d_h \times 1}$ is concatenation of averaged representations of the leftmost and rightmost analyses, and $\mathbf{D}_c \in \mathbf{R}^{d_h \times 2hs}$ and $\mathbf{D}_a \in \mathbf{R}^{d_h \times 2d_h}$ are the model parameters. This advanced variation is referred to as **BiLSTM***.

2.3 Alternative Context Representation

We also experiment with an alternative context model that uses a feed-forward NN architecture (Collobert et al., 2011; Zheng et al., 2013). In this model word embeddings of fixed window size are fed to the hidden layer, and the output represents the context. The remaining parts of the architecture stay the same: we use the same morphological representation and choose the correct analysis exactly as we did for BiLSTM model. As in the case with BiLSTM, we leverage morphological context, by performing a Viterbi decoding conditioned on the leftmost analysis. We refer to this model as **DNN** (Deep NN), an advanced variation of which uses the averaged rightmost morphological context as well, and is referred to as **DNN***.

Lang.	Train	Test	OOV	ΔAT
Kazakh	16,624	2,324	43.9%	2.85
Turkish	752,332	20,536	10.24%	1.76

Table 1: Corpora statistics: ΔAT denotes the average number of analysis per token.

2.4 Training

In all models, the top layer of the networks has a softmax that computes the normalized scores over morphological candidates given the input word. The networks are trained to minimize the cross entropy of the predicted and true morphological analyses. Back-propagation is employed to compute the gradient of the corresponding object function with respect to the model parameters.

3 Experiments and Evaluation

3.1 Data Sets

We conduct our experiments on Kazakh (Assylbekov et al., 2016) and Turkish (Yuret and Türe, 2006) data sets[3]. Table 1 shows the corpora statistics. Kazakh data set is almost 50 times smaller than that of Turkish, with four times the OOV rate and almost twice as many analyses per word on average. Given such a drastic difference in the resources it would be interesting to see how our models perform on otherwise similar languages (both Turkic). Lastly, while the corpora provide train and test splits, there are no tuning sets, so we withdraw small portions from the training sets for tuning hyper-parameters[4].

3.2 Baselines

We compare our models to three other approaches. For Kazakh we use an HMM based tagger and its version extended with the rule-based constraint grammar (Assylbekov et al., 2016), which is considered the state of the art for the language. We

[3]For Turkish, we used a test set that was manually re-annotated by Yildiz et al. (2016).

[4]The following hyper-parameters are used in all the experiments: character embedding size $d_c = 35$, character and context LSTM states are 50, root and POS embedding sizes are all set to 50, hidden layer size $d_h = 200$, learning rate is set to 0.01. The window size of DNN is set to 5. For regularization we use dropout (Srivastava et al., 2014) with probability 0.5 on the hidden layers. We further constrain the norm of gradient to be below 2 by using gradient clipping.

Models	Kazakh					Turkish				
	tok. acc.	tok. amb.acc.	OOV acc.	OOV amb.acc.	sen. acc.	tok. acc.	tok. amb.acc.	OOV acc.	OOV amb.acc.	sen. acc.
HMM	83.82	74.98	80.90	73.89	32.41	-	-	-	-	-
MANN	85.56	77.66	81.68	74.96	32.80	91.35	82.32	86.72	74.09	40.38
Voted Perceptron	88.41	82.07	86.19	81.12	40.31	91.89	83.47	87.98	76.87	41.52
DNN	86.33	78.86	84.13	78.31	40.71	92.24	84.14	87.05	74.74	41.16
DNN*	87.25	80.28	85.99	80.85	40.31	92.22	84.12	87.24	75.11	40.38
DNN*‡	88.26	81.85	86.77	81.92	39.52	**92.32**	**84.32**	87.95	76.50	**41.55**
BiLSTM	87.49	80.65	85.21	79.78	39.52	91.37	82.39	86.91	74.46	38.13
BiLSTM*	90.92	85.95	88.73	84.60	50.19	92.03	83.73	88.01	76.60	40.54
BiLSTM*‡	**91.06**	**86.40**	**88.93**	**85.27**	**50.98**	92.16	84.01	**88.24**	**77.06**	41.01
HMMCG	90.39	85.88	88.83	86.07	53.75	-	-	-	-	-
BiLSTM*‡+CG	91.74	87.45	90.00	86.74	54.54	-	-	-	-	-

Table 2: Results: here, *tok. acc.* and *tok. amb. acc.* denote the accuracy over all and ambiguous tokens respectively. Same goes for *OOV acc.* and *OOV amb. acc.*. Sentence accuracy is denoted as *sen. acc.*.

refer to these baselines as HMM and HMMCG. Another baseline is a voted perceptron (Collins, 2002) based tagger. We use our implementation of this baseline for Kazakh and the model developed by Sak et al. (2007) for Turkish. Lastly, we use a neural network model proposed by Yildiz et al. (2016), which is considered state of the art for Turkish. For this baseline too we use our own implementation (for both languages) and refer to it as MANN[5].

3.3 Experimental Setup

As described in the previous section, each of our models has two settings: the one that does not incorporate *surrounding* morphological context and the one that does (the starred one). In addition to that we use pre-trained embeddings, by training *word2vec* (Mikolov et al., 2013) skip-gram model on Wikipedia texts. This setting is denoted by a double dagger (‡).

We perform a single run evaluation in terms of token- and sentence- based accuracy. We consider four types of tokens: (i) all tokens; (ii) ambiguous tokens (the ones with at least two analyses); (iii) OOV tokens; (iv) ambiguous OOV tokens. Thus, we use a total of five metrics. In terms of strictness we deem correct only the predictions that match the golden truth completely, i.e. in root, POS and MC (up to a single morpheme tag).

[5]Note that all of the baselines are language dependent to a certain degree, with MANN being the least dependent and HMMCG the most. The latter baseline employs hand-engineered constraint grammar rules to perform initial disambiguation, followed by application of the HMM tagger, which cherry-picks the most informative grammatical features.

3.4 Results and Discussion

The results are given in Table 2. Unless stated otherwise we refer to the general (all tokens) accuracy when comparing model performances.

For Kazakh, DNN conditioned on the leftmost analysis yields 86.33% accuracy. DNN* that in addition uses the rightmost analysis embeddings, improves almost 1% over that result (87.25%). On the other hand BiLSTM, whose context representation uses surface forms only, performs even better (87.49%). When this model incorporates immediate morphological context, it (BiLSTM*) performs at 90.92% and beats the HMMCG baseline. However, the latter being a very strong language dependent baseline still outperforms our model in ambiguous OOV and sentence accuracy. When we evaluate our model under equal conditions (BiLSTM*‡+CG) it beats HMMCG on all of the metrics. We separate this comparison from the rest because of a language-dependent set up.

In contrast, for Turkish DNN models outperform BiLSTM on seen tokens and yield an almost equal 92.2% accuracy regardless of using the rightmost morphological context. This performance is also higher than that of all baselines, including the state of the art MANN. However BiLSTM* is still better than DNN* in OOV token accuracy, both overall and ambiguous.

As it can be seen, pre-training boosts the performance of DNN* and BiLSTM* across all metrics. For Kazakh pre-training results in .14% improvement in general token accuracy for BiLSTM*, which amounts to .67% improvement over the state of the art. For Turkish this results in an

almost 1% net improvement in overall token accuracy over MANN, the state of the art[6].

A cross-linguistic comparison reveals that although Kazakh data set is much smaller than that of Turkish and has more analyses per word on average and higher OOV rate, on certain metrics the models perform on par or even better for Kazakh[7]. To investigate this further we have made data sets comparable in size by randomly choosing 20.6K+ and 3.4K from Turkish training and test sets. On this data BiLSTM*‡ yields 91.18, 82.0% general and ambiguous token accuracy and respective scores for OOV are 87.0, 74.6%. This result follows the pattern, where for Turkish only the general accuracy is higher than that of Kazakh. It turns out that Turkish data contains many unambiguous tokens: 49% and 48% for full and small data sets (train + test average), against 36% for Kazakh. This suggests that the higher general accuracy on Turkish data can be explained by the higher rate of the unambiguous tokens. Also Turkish has a more complex derivational morphology, which "lengthens" the analyses, e.g. an average number of morphemes per analysis is higher for Turkish (5.25) than for Kazakh (4.6). This adds sparseness to the morpheme chains and certainly further complicates disambiguation, especially in an OOV scenario.

We also observe that BiLSTM*‡ works best on all metrics for Kazakh, but for Turkish it beats DNN*‡ only on the OOV part. Due to BiLSTM*‡ being computationally prohibitive we ran it with significantly less number of epochs than DNN, and it also being a character-based model, we speculate that it was able to learn character aware context embeddings hence better at OOV.

4 Related Work

A morphology-aware NN (MANN) for MD was proposed by Yildiz et al. (2016), and has been reported to achieve ambiguous token accuracies of 84.12, 88.35 and 93.78% for Turkish, Finish and Hungarian respectively. This approach differs from ours in a number of ways. (i) Our analysis representation treats morpheme tags in a language-independent manner considering every tag found in the training set, whereas in MANN certain tags are chosen with a specific language in mind. (ii) MANN is a feed-forward NN that, unlike our approach, does not account for the surface context. (iii) As we understood, at the decoding step MANN makes use of the golden truth, whereas our models have no need for that.

Although several statistical models have been proposed for Kazakh MD, such as HMM- (Makazhanov et al., 2014; Makhambetov et al., 2015; Assylbekov et al., 2016), voted perceptron- (Tolegen et al., 2016) and transformation-based (Kessikbayeva and Cicekli, 2016) taggers, to our knowledge ours is the first deep learning-based approach to the problem that is also purely language independent.

It is becoming increasingly popular to use richer architectures to learn better embeddings from characters/words (Yessenbayev and Makazhanov, 2016; Ling et al., 2015; Wieting et al., 2016). Ling et al. (2015) used a BiLSTM to learn word vectors, showing strong performance on language modeling and POS tagging. Melamud et al. (2016) proposed *context2vec*, a BiLSTM based model to learn context embedding of target words and achieved state-of-the-art results on sentence completion and word sense disambiguation.

5 Conclusion

We have proposed a general MD framework for MCL that can be analyzed in <root, POS, MC> triplets. We have showed that the surface context can be useful to MD, especially if combined with morphological context. Our next step would be to assess our claims on a larger number of typologically distant languages.

Acknowledgments

This work has been conducted under the targeted program O.0743 (0115PK02473) of the Committee of Science of the Ministry of Education and Science of the Republic of Kazakhstan, and the research grant 129-2017/022-2017 of the Nazarbayev University.

The authors would like to thank Xiaoqing Zheng for tremendously helpful discussions, as well as Eray Yildiz and Zhenisbek Assylbekov for the data sets used in this study and prompt replies to all questions regarding those.

[6]For the un-pretrained model original work reports 84.12% accuracy on ambiguous tokens (Yildiz et al., 2016), which is lower than 84.14% that un-pretrained DNN achieves on this metric.

[7] For instance, BiLSTM*‡ applied to Kazakh performs better than any other model for Turkish in terms of sentence, ambiguous and OOV token accuracy. Moreover all of the models (including the baselines) perform better on Kazakh in terms of ambiguous OOV accuracy.

References

Zhenisbek Assylbekov, Jonathan Washington, Francis Tyers, Assulan Nurkas, Aida Sundetova, Aidana Karibayeva, Balzhan Abduali, and Dina Amirova. 2016. A free/open-source hybrid morphological disambiguation tool for Kazakh. In *TurCLing 2016*. pages 18–26.

Michael Collins. 2002. Discriminative Training Methods for Hidden Markov Models: Theory and Experiments with Perceptron Algorithms. In *Proceedings of the ACL-02 Conference on Empirical Methods in Natural Language Processing - Volume 10*. Association for Computational Linguistics, Stroudsburg, PA, USA, EMNLP '02, pages 1–8.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *J. Mach. Learn. Res.* 12:2493–2537.

Klaus Greff, Rupesh Kumar Srivastava, Jan Koutník, Bas R. Steunebrink, and Jürgen Schmidhuber. 2015. LSTM: A Search Space Odyssey. *CoRR* abs/1503.04069.

Dilek Z. Hakkani-Tür, Kemal Oflazer, and Gökhan Tür. 2002. Statistical Morphological Disambiguation for Agglutinative Languages. *Computers and the Humanities* 36(4):381–410.

Gulshat Kessikbayeva and Ilyas Cicekli. 2016. A Rule Based Morphological Analyzer and a Morphological Disambiguator for Kazakh Language. *Linguistics and Literature Studies* 4(4):96–104.

Wang Ling, Chris Dyer, Alan W. Black, Isabel Trancoso, Ramon Fermandez, Silvio Amir, Lus Marujo, and Tiago Lus. 2015. Finding Function in Form: Compositional Character Models for Open Vocabulary Word Representation. In *EMNLP*. The Association for Computational Linguistics, pages 1520–1530.

Aibek Makazhanov, Zhandos Yessenbayev, Islam Sabyrgaliyev, Anuar Sharafudinov, and Olzhas Makhambetov. 2014. On certain aspects of Kazakh part-of-speech tagging. In *Application of Information and Communication Technologies (AICT), 2014 IEEE 8th International Conference on*. pages 1–4.

Olzhas Makhambetov, Aibek Makazhanov, Islam Sabyrgaliyev, and Zhandos Yessenbayev. 2015. Data-Driven Morphological Analysis and Disambiguation for Kazakh. In *Computational Linguistics and Intelligent Text Processing - 16th International Conference, CICLing 2015, Cairo, Egypt, April 14-20, 2015, Proceedings Part I*. pages 151–163.

Oren Melamud, Jacob Goldberger, and Ido Dagan. 2016. context2vec: Learning Generic Context Embedding with Bidirectional LSTM. In *CoNLL*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed Representations of Words and Phrases and their Compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, Curran Associates, Inc., pages 3111–3119.

Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2007. Morphological Disambiguation of Turkish Text with Perceptron Algorithm. In *Proceedings of CICLing 2007*. volume LNCS 4394, pages 107–118.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* 15:1929–1958.

Gulmira Tolegen, Alymzhan Toleu, and Zheng Xiaoqing. 2016. Named Entity Recognition for Kazakh Using Conditional Random Fields. In *Proceedings of the 4-th International Conference on Computer Processing of Turkic Languages TurkLang 2016*. Izvestija KGTU im.I.Razzakova. TurkLang 2016, pages 122–129.

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2016. Charagram: Embedding Words and Sentences via Character n-grams. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Austin, Texas, pages 1504–1515.

Zhandos Yessenbayev and Aibek Makazhanov. 2016. Character-based Feature Extraction with LSTM Networks for POS-tagging Task. In *Application of Information and Communication Technologies (AICT), 2016 IEEE 10th International Conference on*. pages 62–66.

Eray Yildiz, Caglar Tirkaz, H. Bahadir Sahin, Mustafa Tolga Eren, and Omer Ozan Sonmez. 2016. A Morphology-Aware Network for Morphological Disambiguation. In *Proceedings of AAAI*. AAAI Press, pages 2863–2869.

Deniz Yuret and Ferhan Türe. 2006. Learning Morphological Disambiguation Rules for Turkish. In *Proceedings of the Main Conference on Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*. Association for Computational Linguistics, Stroudsburg, PA, USA, HLT-NAACL '06, pages 328–334.

Xiaoqing Zheng, Hanyang Chen, and Tianyu Xu. 2013. Deep learning for Chinese word segmentation and POS tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Seattle, Washington, USA, pages 647–657.