Proceedings of the 1st Joint SLTU and CCURL Workshop (SLTU-CCURL 2020), pages 258–264
Language Resources and Evaluation Conference (LREC 2020), Marseille, 11–16 May 2020

Voted-Perceptron Approach for Kazakh Morphological Disambiguation

Gulmira Tolegen, Alymzhan Toleu, Rustam Mussabayev
Institute of Information and Computational Technologies
Almaty, Kazakhstan
gulmira.tolegen.cs@gmail.com, alymzhan.toleu@gmail.com, rmusab@gmail.com

Abstract

This paper presents an approach of voted perceptron for morphological disambiguation for the case of Kazakh language. Guided by the intuition that the feature value from the correct path of analyses must be higher than the feature value of non-correct path of analyses, we propose the voted perceptron algorithm with Viterbi decoding manner for disambiguation. The approach can use arbitrary features to learn the feature vector for a sequence of analyses, which plays a vital role for disambiguation. Experimental results show that our approach outperforms other statistical and rule-based models. Moreover, we manually annotated a new morphological disambiguation corpus for Kazakh language.

Keywords: Morphological Disambiguation, Voted Perceptron, Kazakh Language

1. Introduction

Morphological analysis and disambiguation play a vital role in handling the problems of i) reducing the complexity of the word structures and ii) alleviating the data sparsity issue. A morphological analyzer can decompose any raw word into a sequence of morphological tags and it produces more than one analysis per word. An example is given in Table 2, where a simple Kazakh phrase is analyzed and each word has more than one analysis. Morphological disambiguation (MD) is the task of selecting the correct analysis among the candidate analyses by leveraging the context information. Kazakh is an agglutinative language with rich morphology. A root/stem in Kazakh may produce hundreds or thousands of new words. It is apparent from below that Kazakh has large unique tokens than English, which leads to the data sparsity problem.

Corpus size	Kaz uni. tok.	Eng uni. tok.
948,612 (News)	91,495	57,017
25,327,611 (Wikipedia)	873,693	427,980

Table 1: Comparison of Kazakh and English corpora. *uni. tok.* denotes the number of unique tokens.

Developing an accurate disambiguation approach is appealing because it can alleviate the data sparseness problem caused by rich morphology. Most researchers investigating Kazakh MD have utilised Hidden markov model (HMM) (Assylbekov et al., 2016; Makhambetov et al., 2015) as the statistical model. There are several problems with the use of this model: i) the strong assumption of HMM makes it not flexible to use arbitrary features; ii) the complexity of the task itself makes the model not tractable in practice when using a full analysis as labels (breaking-down an analysis into smaller units may work for this case, but the cost may be a loss of accuracy); iii) it cannot capture the long-distance dependency.

In this paper, we present an approach of voted perceptron for MD with a new manually-annotated corpus. We treat an analysis *kala n nom e cop aor p3 pl* as a combination of three main constituents: root, POS, and morpheme chain:

$$\underbrace{kala}_{root}\ \underbrace{n}_{POS}\ \underbrace{nom\ e\ cop\ aor\ p3\ pl}_{morpheme\ chain} \quad (1)$$

As we can see that a full analyses have a complex structure, which means the model must correctly predict every single tag in these three parts including the root. The idea behind of these constituents is that we try to represent a sequence analysis with feature vectors. To learn the feature vectors for each sequence of analysis, we present a voted-perceptron approach for MD. The underlying hypothesis is that we need to train the model as follows: the feature vector of the extracted sequence analysis in the correct path should have a large value than those in the non-correct path. In order to improve the model's performance, we use a set of features and assess how these features affect the results. In the experiment, we try to evaluate how the breaking-down technique of analysis affects the model performance and evaluate what is the optimal length of morpheme chain (MC) for disambiguation. The proposed approaches do not need to specify the hand-rules (like constrained grammars (CGs) does), and the approach achieves better results than both the statistical and rule-based models.

2. Related Work

In general, the two tasks morphological disambiguation and morphological tagging are similar to each other. The difference between morphological disambiguation and the morphological tagging is that the latter one only makes prediction through the surface word and it is harder than MD. The former is able to access the possible candidates of analysis, which more designed to solve ambiguity of analysis produced by an analyzer.

2.1. Morphological Disambiguation

Several approaches have been applied for the morphological disambiguation and can be categorized as follows: rule-based, statistical-based model with discrete features, neural network-based and hybrid approaches. Makhambetov et al. (2015) presented a data-driven approach for Kazakh morphological analysis and disambiguation that was based

Kala	Kelbeti	Zhana	...
kala n nom			
kala n attr		zhana adj	
kala n nom e cop aor p3 pl	kelbet n px3sp nom	zhana adv	
kala n nom e cop aor p3 sg	kelbet n px3sp nom e cop aor p3 pl	zhana adj advl	...
kal v iv prc_impf	kelbet n px3sp nom e cop aor p3 sg	zhana adj subst nom	
kala v tv imp p2 sg		...	
kal vaux prc_impf			

Table 2: Example of morphological analysis for a Kazakh sentence: *Kala kelbeti Zhana* (the appearance of the city is new).

on the Hidden Markov Model (HMM). The authors conducted 10 cross-validated evaluations and obtained 86% accuracy on the test data. Kessikbayeva and Cicekli (2016) presented a rule-based morphological disambiguator for Kazakh language and it achieved 87% accuracy on the test data (about 15,000 words). In the same direction, Assylbekov et al. (2016) presented a hybrid approach that applied constrained grammar (CG) with HMM tagger. The authors reported that the HMM tagger achieved 84.55% accuracy and the hybrid approach achieved 90.73% accuracy on the test data.

In recent years, deep learning arguably achieved tremendous success in many research areas such as NLP (Tolegen et al., 2019; Mikolov et al., 2013; Toleu et al., 2017b; Dayanik et al., 2018; Toleu et al., 2019), speech signal processing (Mamyrbayev et al., 2019) and computer vision (Girdhar et al., 2019; Pang et al., 2019). Toleu et al. (2017a) presented a neural network-based disambiguation model, in which the author proposed to measure the distance of the two embeddings: the context and the morphological analyses. In order to measure the distances, the author applies neural networks to learn the context representation from characters and represents the morphological analyses as well. The correct analysis should more similar to the context's embedding; in other words, they are closely arranged in the vector space compared to the other candidate analyses.

2.2. Morphological Tagging

Morphological tagging has been studied extensively for the decade, here we review the work most relevant to this paper Mueller et al. (2013) presented a pruned CRF (PCRF) for morphological tagging and proposed to use coarse-to-fine decoding and early updating to train the higher-order CRF. Experiments on six languages show that the PCRF gives significant improvements in accuracy. Müller and Schütze (2015) compared the performance of the most important representations that can be used for across-domain morphological tagging. One of their findings is that the representations similar to Brown clusters perform best for POS tagging and that word representations based on linguistic morphological analyzers perform best for morphological tagging. Malaviya et al. (2018) combines neural networks and graphical models presented a framework for cross-lingual morphological tagging. Instead of predicting full tag sets, the model predicts single tags separately and modeling the dependencies between tags over time steps. The model is able to generate tag sets unseen in training data, and share information between similar tag sets. This model is about cross-lingual MT and we do not make comparisons with monolingual morphological tagging models. Tkachenko and Sirts (2018) presented a sequence to sequence model for morphological tagging. The model learns the internal structure of morphological labels by treating them as sequences of morphological feature values and applies a similar strategy of neural sequence-to-sequence models commonly used for machine translation (Sutskever et al., 2014) to do morphological tagging. The authors explored different neural architectures and compare their performance with both PCRF (Mueller et al., 2013). Double layer of biLSTMs were applied in those neural architectures as Encoder (Ling et al., 2015; Labeau et al., 2015; Ma and Hovy, 2016). The encoder uses one biLSTM to compute character embedding and the second biLSTM combine the obtained character embedding along with pre-trained word embedding to generate word context embeddings. The output of those neural networks are different: one of the baselines is to use a single output layer to predict whole morphological labels. As the second baseline, the output layer can be changed to predict the different morphological value of tag with separate output layers. An improved version of the second one is to use a hierarchical separate output layers in order to capture dependencies between tags.

3. HMM-based Disambiguation

Let $\mathbf{w} = w_1, w_2, ..., w_n$ be a sentence of length n words and $\mathbf{t} = t_1, t_2, ...t_n$ be corresponding morphological analysis sequence. $\mathbf{c} = (t_1^{c_1}, t_1^{c_m}), ..., (t_n^{c_1}, t_n^{c_m})$ is the candidate analysis of each word. m is the number of candidates, and it can be vary to each word. Morphological disambiguation is the problem of finding the **t** from **c** given the **w**:

$$\begin{aligned} \underset{\mathbf{t}}{\operatorname{argmax}} Pr(\mathbf{t}|\mathbf{w}) &= \underset{\mathbf{t}}{\operatorname{argmax}} \frac{Pr(\mathbf{t})Pr(\mathbf{w}|\mathbf{t})}{Pr(\mathbf{w})} \\ &= \underset{\mathbf{t}}{\operatorname{argmax}} Pr(\mathbf{t})Pr(\mathbf{w}|\mathbf{t}) \end{aligned} \tag{2}$$

where $Pr(\mathbf{w})$ is a constant and can be ignored.

To compute $Pr(\mathbf{t})$ and $Pr(\mathbf{w}|\mathbf{t})$, the first-order HMM assumptions are applied to simplify the analysis transition probability into that the current analysis depends only on previous analysis.

$$Pr(\mathbf{t}) = \prod_{i=1}^{n} Pr(t_i|t_{i-1}) = \prod_{i=1}^{n} \frac{\alpha + c(t_i, t_{i-1})}{\alpha|T| + c(t_{i-1})} \tag{3}$$

#	Features	Description
0	w_i	word context
1	r_i	lemma/stem of the word
2	POS_i	POS tags, such as noun, verb etc.
3	mc_i	full morpheme chain
4	ma_i	a full morphological analysis
5	#t	the number of the tags
6	wc_i	word case
7	ps_i	plural and singular tags

Table 3: Feature category.

where $c(t_i, t_{i-1})$ counts the number of occurrences of t_i, t_{i-1} in the corpus. α is smoothing number. T the unique number of tags in the corpus.

$$Pr(\mathbf{w}|\mathbf{t}) = \prod_{i=1}^{n} Pr(w_i|t_i) = \prod_{i=1}^{n} \frac{\alpha + c(w_i, t_i)}{\alpha|V| + c(t_i)} \quad (4)$$

where $c(w_i, t_i)$ is the number of occurrences of word w_i with tag t_i. $|V|$ is the unique word number. Using above transition and emission scores, we could apply Viterbi decoding to find the best path of analysis.

In practice, there are several drawbacks of above approach when applying it on disambiguation task directly: i) if we consider each full analysis as a tag, the unique number of tag become 19,236 (observed in our corpus), then the number of parameters of transition probability will be $19,236^2$, and the number of parameters of emission probability become even more. ii) breaking down analysis into small subtags can definitely decrease the number of tags and it is tractable, but it has an effect on model performance. iii) first-order HMM not able to capture the long-term dependency information.

4. Voted Perceptron-based Disambiguation

In this section, we describe the voted perceptron-based approach for disambiguation. A major advantage of this approach is that it allows us to use arbitrary features and extracts features from both words and the candidate analyses.

4.1. Feature Vectors

In order to generalize the morphological analyses and to train the perceptron algorithm, we use a set of features to generate feature vector as representation of the analyses using global feature function $\Phi(\cdot)$. Table 3 summarizes the feature categories. Let $\phi(\cdot)$ function be the local feature function which is indicator function, it maps the input to an d-dimentional feature vectors. For example, if the template only contains these two: 1) w_0/w_{-1}; 2)$POS_{-1}/POS_0/POS_1$. w_i denotes for word in the position i-th, and POS_i is part-of-speech. $\phi(\cdot)$ extracts the current and previous word with previous, current and next word POS tag as local features through this template at each step to make the disambiguation. The global feature representation is the sum of all local features for input sequence:

$$\Phi(\cdot) = \sum_{n} \phi(\cdot) \quad (5)$$

4.2. Parameter Estimation

To estimate the parameters of the model, we apply the perceptron training algorithms (Collins, 2002) shown in Figure 1. $z_i \in z$ is a predicted path and $\Phi(\cdot)$ is global feature function that generate features. As we can see, it increases the parameter values for features which are extracted from correct morphological analysis's sequence, and down weighting parameter values for features extracted in the non-correct morphological analysis's sequence. The final analyses path is decoded through the Viterbi algorithm.

Data: Training examples (w_i, t_i).
Result: Parameters a.
Initialization: set parameters $a = 0$;
for $e \leftarrow 1$ **to** $Epoch$ **do**
  for $i \leftarrow 1$ **to** n **do**
    Calculate $z_i = argmax_{z \in GEN(x_i)} \Phi(x_i, z) \cdot a$
    if $z_i \neq t_i$ **then**
      $a = a + \Phi(x_i, t_i) - \Phi(x_i, z_i)$
    end
  end
end

Algorithm 1: Voted-Perceptron algorithm.

5. Experiments and Results

5.1. Corpus Construction

One of the aims of this work is to create a manually annotated morphological disambiguation data set as the database for future further research. As known that the task of annotating data is a time-consuming and tedious work. In order to assist the annotation process and to improve the correctness of the annotation, we build an annotation tool with user-friendly interface. Figure 1 shows a screenshot of an annotation process. The annotation process is not trivial and slow, the annotator should annotate every single word appeared in the document. We can briefly illustrate the annotation process as follows: click a word, then the corresponding candidate analyses will show up for annotation; the annotator not only considers the context of that word but also consider the previous/future words' morphological tags to make the decision.

We randomly selected 110 documents from the general news media[1] as the data source for annotation. The annotations have been executed manually by native speaker of Kazakh. The proposed approaches were evaluated on the new morphological disambiguation data set. The corpus consists of 15,466 tokens, and 90% is used as the training set, 10% for the test set. Table 4 shows the statistics about the data set.

	#token	OOV (%)	Avg.
Train	13,849	-	2.97
Test	1,617	27.58	2.93

Table 4: Statistics of the corpus. *OOV* - out-of-vocabulary rate. *Avg.* - the average number of analysis per token.

[1] https://www.inform.kz/kz

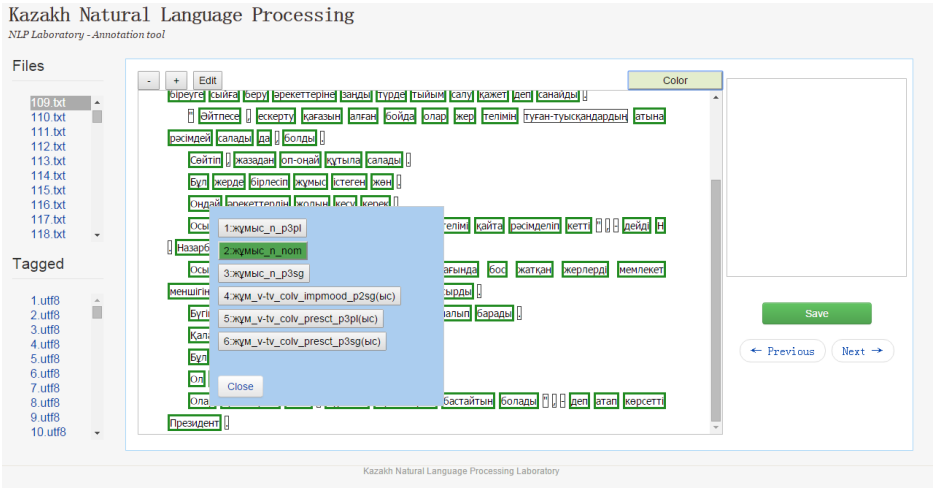

Figure 1: Annotation tool for manually annotating the morphological disambiguation corpus.

5.2. Model Setup

There are 19,236 analysis[2] can be observed in our current corpus, which results in that the number of model's parameters be very large and sparse when training the HMM. In practice, the memory overflow error was raised when we directly use the full analyses as labels. It worth to note that the number of analyses increases with the increase of the corpus volume because each root can produce hundred or thousand new words in Kazakh. To reduce the tag number, we tried to break-down the analysis into smaller units excluding the roots. Because there are 4,050 roots in the corpus, and if we only take the last one tag of the morpheme chain with root as an analysis, the number is still large about 11,167.

Models	Type of tags	#tag
M-1	the last 1 tag of MC	57
M-2	the last 2 tags of MC	241
M-3	the last 3 tags of MC	639
M-4	full morpheme chain	839
M-5	POS + the last 1 tag of MC	216
M-6	POS + the last 2 tags of MC	773
M-7	POS + the last 3 tags of MC	1,453
M-8	POS + full morpheme chain	1,699

Table 5: Various HMM-based models with their tag numbers for disambiguation after breaking down the analysis into a small number of morphological units.

Table 5 provides the summary statistics for HMM model's variations (denoted from M-1 to M-8), which are trained with different morphological units. For instance, model M-1 uses the last single tag of morpheme chain as label to train, and the total number of labels is 57. Models denoted with M-5 to M-8 include the POS combined with thee morpheme tags. We found the max-length and the average length of the morpheme chain in the corpus are 7 and 3.07, respectively. The idea behind such a model setup is to assess how that breaking-down way affects model's performance and to evaluate what is the optimal length of morpheme chain for disambiguation. The voted-perceptron model trained with features presented in Table 3 with suitable feature template.

[2]Each morphological analysis consists of root, POS, and morpheme chain.

5.3. Results

We report the accuracy results for the overall (all tokens), known tokens, and unknown tokens. 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). Unless stated otherwise we refer to the overall accuracy when comparing model performances.

Table 6 shows the results of HMM models with different tag sets. The top half of the Table 6 shows that the performances are relatively low when the tag from the morpheme chain was used only. The accuracy of models (M-1, M-2, M-3) use the last 1, 2 and 3 tags of morpheme chain are 72.04%, 78.72% and 78.47% respectively. Model M-4 uses full morpheme chain and gives 78.60%. To investigate this further, we take the last 4 tags of morpheme chain as labels and found that the result is 78.6% same with M-4's.

The bottom half of the Table 6 shows the results of all models trained with POS+certain morpheme tags. It is apparent from this table that the models M-6 (POS + last 2 tags) and M-7 (POS + last 3 tags) give same accuracy 84.91% which is the best result among them. No significant differences were found between M-7 and M-8 and the latter one was trained with POS + full morpheme chain, and M-8 shows a little drop over M-7 in terms of unknown tokens accuracy. According to these results, we found that the HMM model achieves the best result when using POS+the last 2/3 morpheme tags. There is no significant improvement when increasing the length of the morpheme tags.

Table 7 presents the results for the voted-perceptron approach and it shows how each feature affects the model. A plus (+) sign before the feature name indicates that these feature combinations are added on top of the rest with suitable feature templates. It can be seen that the proposed model enhanced with the word context ($+w$) only gives 53.68% accuracy. When adding the root feature ($+r$) to

Models	Overall acc.	Known acc.	Unk. acc.
M-1	72.04	77.25	57.24
M-2	78.72	82.19	68.88
M-3	78.47	82.44	67.22
M-4	78.60	82.52	67.45
M-5	79.96	84.86	66.03
M-6	**84.91**	**89.54**	**71.73**
M-7	**84.91**	**89.54**	**71.73**
M-8	84.78	89.54	71.25

Table 6: Accuracy results of HMM-based models. *Unk. acc.* denotes for unknown tokens accuracy.

#	Models	Overall acc.	Known acc.	Unk. acc.
0	+w	53.68	55.51	48.45
1	+r	62.09	65.21	53.21
2	+pos	70.56	75	57.95
3	+mc	**89.11**	**92.39**	**78.81**
4	+ma	89.54	92.47	81.23
5	+#t	89.17	92.47	79.81
6	+wc	90.23	93.39	81.23
7	+ps	**90.53**	**93.39**	**82.42**

Table 7: Accuracy results of the voted-perceptron approach. $+mc$ indicates that the current feature with its feature combinations is added to the model with previous features $+w$, $+r$, $+pos$ accumulatively.

Models	Overall acc.	Known acc.	Unk. acc.
HMM	84.91	89.54	71.73
Voted-Perceptron	90.53	93.39	82.42
Improv.	**5.62**	**3.85**	**10.69**

Table 8: Comparison of the best results from HMM-based models and the voted-perceptron.

the model (trained with $+w$ and $+r$ features), the model performance can be improved to 62.09%, which means that the root feature contributes around 8% improvements. It is apparent that the pos feature ($+pos$) contributes 8.47%, and the model achieves 70.56% accuracy. As we expected, the morpheme chain features ($+mc$) contributes most to model performance. It gives 18.55% improvement over the accumulation of previous features and the model ends up with 89.11% overall accuracy. The feature of full morphological analysis ($+ma$) only gives a minor improvement. Other features like $+t$, $+wc$, and $+ps$ provide positive effect and finally the proposed approach achieves 90.53% overall accuracy.

Table 8 compares the best results obtained from the HMMs and the voted-perceptron. It is apparent from this table that the proposed approach outperforms than HMM by 5.62% overall accuracy, and 10.69% unknown tokens accuracy.

To compare the proposed approach with previous work, we take the two existing models as baselines: i) a statistical model proposed by Assylbekov et al. (2016)[3] and ii) a rule-based constrained grammar tool from the Apertium-kaz CG tagger[4]. These tools cannot applied to our data set directly[5], instead of converting the tools, we evaluate all models on their data set (Assylbekov et al., 2016), and the proposed voted-perceptron was trained on this data with the corresponding features. Since voted-perceptron is a purely statistical model, for the fair comparison, we use the baseline of Assylbekov et al. (2016) of their statistical model based on HMM not the combined model of HMM with CG.

Models	Overal acc.	Unk. acc.
Assylbekov et al. (2016)	84.55	80.90
Constrained Grammar (CG)	87	-
Voted Perceptron	**88.11**	**85.11**

Table 9: Comparison of the best results from HMM-based models and those of voted-perceptron.

Table 9 shows the comparison results. It is can be seen that voted-perceptron model outperforms the HMM-based disambiguation and also beats the constrained grammar (CG), the rule-based disambiguaton tool.

5.4. Error Analysis

We categorize the errors of model output into three groups: root inconsistency, POS inconsistency and the morpheme chain inconsistency. Table 10 shows error percentages. It can be seen that in models M-1 to M-4 only trained with different length of morpheme chain, the root inconsistency error takes almost half of the total. The POS inconsistency error takes around 25%. After adding the POS to the models (M-5 to M-8), the root and POS inconsistency percentages decreased to around 44% and 20% respectively. It is apparent from this table that for the best HMM-based model, the root inconsistency error accounts for the large part of errors (44.85%). It is reasonable because these models did not include root as a label in training.

Models	root	POS	mc
M-1	48	21.23	30.75
M-2	53.48	25.29	21.22
M-3	52.29	26.43	21.26
M-4	52.31	26.58	21.09
M-5	37.96	16.66	45.37
M-6	44.26	19.67	36.06
M-7	**44.85**	**20.90**	**35.24**
M-8	44.30	21.13	34.55

Table 10: Percentage of root, POS and morpheme chain errors for HMM-based models.

Table 11 shows the error' percentages for voted perceptron.

[3]https://svn.code.sf.net/p/apertium/svn/branches/kaz-tagger/

[4]http://wiki.apertium.org/wiki/Apertium-kaz

[5]We used our new developed morphological analyzer to decompose the words, and it has some issue of inconsistency of the name of morphological tags with their analyzer.

It can be seen that the different features affect the model's output error percentage for voted perceptron. The error percentage of the final model for root, POS and morpheme chain inconsistency are 25.49%, 17.64% and 56.86%. It is apparent that a very large portion of error is accounted for morpheme chain inconsistency.

Models	root	POS	mc
+w	37.65	21.76	40.58
+r	11.58	28.87	59.54
+pos	11.34	8.19	80.46
+mc	30.11	18.75	51.13
+ma	30.17	22.48	47.33
+#t	25.14	24	50.85
+wc	27.21	18.98	53.79
+ps	25.49	17.64	56.86

Table 11: Percentage of root, POS and morpheme chain errors for voted perceptron.

Further analysing the output of the models, we found that the models tend to make error prediction for the possessive tags with 3-rd person and tags of tense. Because the possessive tags have the attribute of plural or singular, and these attributes can be determined only after the subject is captured. If there are many words between the subject and the current word with possessive tags or the subject is in hidden form, then the former involves the long-distance dependency problem, and the latter requires the model need certain semantic information of sentences that reflects the subject. Similarly, the corresponding tense tag in also involved to define the sentence tense before tagging the corresponding word with a tense tag. For example, in Kazakh, a verb surface word can have a future or current tense tag simultaneously, and the disambiguation can be done when the sentence tense is determined. As the HMM-based model is the first-order model, these errors cannot be solved. In voted-perceptron, we apply the [-2,-2] window to extract features, and can partially solve the long-distance problem.

6. Conclusion

In this paper, we represent an approach of voted perceptron for morphological disambiguation for the case of Kazakh language. The approach can use arbitrary features in training and testing and can also apply to other languages easily. A new manually annotated corpus for Kazakh morphological disambiguation is presented in this paper for the further research. Experimental results show that voted perceptron outperforms the frequently used HMM-based and the rule-based constrained grammar. One possible future work is to perform transfer learning by using the learned feature vector of this approach for the typologically similar languages. Solving the long-distance dependency problem of morphological disambiguation is the another prior future work.

Acknowledgement

This research has been conducted within the framework of the grant num. BR05236839 "Development of information technologies and systems for stimulation of personalities sustainable development as one of the bases of development of digital Kazakhstan".

7. Bibliographical References

Assylbekov, Z., Washington, J., Tyers, F., Nurkas, A., Sundetova, A., Karibayeva, A., Abduali, B., and Amirova, D. (2016). A free/open-source hybrid morphological disambiguation tool for kazakh. The First International Conference on Turkic Computational Linguistics, TurCLing 2016 ; Conference date: 02-04-2016 Through 08-04-2016.

Collins, M. (2002). Discriminative training methods for hidden markov models: Theory and experiments with perceptron algorithms. *EMNLP*.

Dayanik, E., Akyürek, E., and Yuret, D. (2018). Morphnet: A sequence-to-sequence model that combines morphological analysis and disambiguation. *CoRR*, abs/1805.07946.

Girdhar, R., Carreira, J., Doersch, C., and Zisserman, A. (2019). Video action transformer network. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June.

Kessikbayeva, G. and Cicekli, I. (2016). A rule based morphological analyzer and a morphological disambiguator for kazakh language. *Linguistics and Literature Studies*, 4:96–104, 01.

Labeau, M., Löser, K., and Allauzen, A. (2015). Non-lexical neural architecture for fine-grained POS tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 232–237, Lisbon, Portugal, September. Association for Computational Linguistics.

Ling, W., Dyer, C., Black, A. W., Trancoso, I., Fermandez, R., Amir, S., Marujo, L., and Luis, T. (2015). Finding function in form: Compositional character models for open vocabulary word representation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1520–1530, Lisbon, Portugal, September. Association for Computational Linguistics.

Ma, X. and Hovy, E. (2016). End-to-end sequence labeling via bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1064–1074, Berlin, Germany, August. Association for Computational Linguistics.

Makhambetov, O., Makazhanov, A., Sabyrgaliyev, I., and Yessenbayev, Z. (2015). Data-driven morphological analysis and disambiguation for kazakh. In Alexander Gelbukh, editor, *Computational Linguistics and Intelligent Text Processing*, pages 151–163, Cham. Springer International Publishing.

Malaviya, C., Gormley, M. R., and Neubig, G. (2018). Neural factor graph models for cross-lingual morphological tagging. In *ACL*.

Mamyrbayev, , Turdalyuly, M., Mekebayev, N., Mukhsina, K., Alimhan, K., BabaAli, B., Nabieva, G., Duisenbayeva, A., and Akhmetov, B. (2019). Continuous speech recognition of kazakh language. *ITM Web of Conferences*, 24:01012, 01.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'13, pages 3111–3119, USA. Curran Associates Inc.

Mueller, T., Schmid, H., and Schütze, H. (2013). Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332, Seattle, Washington, USA, October. Association for Computational Linguistics.

Müller, T. and Schütze, H. (2015). Robust morphological tagging with word representations. In *HLT-NAACL*.

Pang, B., Zha, K., Cao, H., Shi, C., and Lu, C. (2019). Deep rnn framework for visual sequential applications. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June.

Sutskever, I., Vinyals, O., and Le, Q. V. (2014). Sequence to sequence learning with neural networks. In Z. Ghahramani, et al., editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

Tkachenko, A. and Sirts, K. (2018). Modeling composite labels for neural morphological tagging. In *Proceedings of the 22nd Conference on Computational Natural Language Learning*, pages 368–379, Brussels, Belgium, October. Association for Computational Linguistics.

Tolegen, G., Toleu, A., Mamyrbayev, O., and Mussabayev, R. (2019). Neural named entity recognition for kazakh. In *Proceedings of the 20th International Conference on Computational Linguistics and Intelligent Text Processing*, CICLing. Springer Lecture Notes in Computer Science.

Toleu, A., Tolegen, G., and Makazhanov, A. (2017a). Character-aware neural morphological disambiguation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 666–671, Vancouver, Canada, July. Association for Computational Linguistics.

Toleu, A., Tolegen, G., and Makazhanov, A. (2017b). Character-based deep learning models for token and sentence segmentation. In *Proceedings of the 5th International Conference on Turkic Languages Processing (TurkLang 2017)*, Kazan, Tatarstan, Russian Federation, October.

Toleu, A., Tolegen, G., and Mussabayev, R. (2019). Keyvector: Unsupervised keyphrase extraction using weighted topic via semantic relatedness. volume 23, page 861–869. 10.