Turkland'14

Istanbul Technical University
6-7 November 2014

TÜRKİYE BİLİŞİM VAKFI BİLGİSAYAR BİLİMLERİ VE MÜHENDİSLİĞİ DERGİSİ

TÜRKİYE BİLİŞİM VAKFI BİLGİSAYAR BİLİMLERİ VE MÜHENDİSLİĞİ DERGİSİ

Editor

Eşref ADALI

Organizing Committee

- Eşref ADALI (İTÜ)
- A. Cüneyd Tantuğ (İTÜ)
- Gözde Gül İşgüder (İTÜ)

Technical Committee

- Eşref ADALI (İTÜ) Chair
- A. Cüneyd Tantuğ (İTÜ)
- Gülşen Eryiğit (İTÜ)
- Banu Diri (Yıldız Technical University)
- Ş. Haluk Akalın (Hacettepe University)
- Tunga Güngör (Boğaziçi University)
- Arzucan Özgür (Boğaziçi University)
- Deniz Yüret (Koç University)
- Sharipbay Altynbek (L.N Gumilev Eurasia National University)
- Lenara Kubedinova (Crimean Engineering University)
- Belgin Aksu (Turkish Language Association)

Sponsor

İstanbul Technical University (İTÜ)

Location and Time

İstanbul Technical University, 6-7 November 2014

Eser Adı:TURKLANG14
ISBN No.:978-975-561-449-6
Basım: Özkaracan Matbaacılık-Bağcılar/İstanbul Tel.: 0212 630 64 73

Contents

The Impact of NLP on Turkish Sentiment Analysis

Ezgi Yıldırım
Istanbul Technical University
yildirimez@itu.edu.tr

Fatih Samet Çetin
Turkcell Global Bilgi
fatih.cetin@global-bilgi.com.tr

Gülşen Eryiğit
Istanbul Technical University
gulsenc@itu.edu.tr

Tanel Temel
Turkcell Global Bilgi
tanel.temel @global-bilgi.com.tr

ABSTRACT

Sentiment analysis on English texts is a highly popular and well-studied topic. On the other hand, the research in this field for morphologically rich languages is still in its infancy. Turkish is an agglutinative language with a very rich morphological structure. For the first time in the literature, this paper investigates and reports the impact of the natural language preprocessing layers on the sentiment analysis of Turkish social media texts. The experiments show that the sentiment analysis performance may be improved by nearly 5 percentage points yielding a success ratio of 78.83% on the used data set.

1 Introduction

Sentiment analysis has become a very popular research area because of needs to track and manage population tendency. Many companies today work on this area in order to meet customer expectations and demands. Social microblogging platforms (e.g. Twitter and Facebook) offer an opportunity to get huge amount of easily accessible and processable data. Users of micro-blogging platforms write about their personal lives, their own opinions about political cases, economic changes, companies and their products.

With the emergence of social media platforms, the sentiment analysis studies are shifted from document level analysis [4, 18, 19] towards sentence or phrase level analysis [14, 22, 12, 23, 21]. Recent years showed that syntactic and/or semantic analysis outperforms baseline sentiment analysis methods in many areas such as aspect-based and comparative opinion mining [9, 13, 2]. In order to reach this level of analysis, many other natural language preprocessing stages are required; i.e. tokenization, normalization, parts-of-speech tagging etc...

As in all other natural language processing (NLP) problems, the most widely studied language for sentiment analysis is English. However, studies for morphologically rich languages are not mature yet. Abdul-Mageed et al. [1] used a supervised, two-stage classification approach employing morphological, dialectal, genre specific features besides basic ones for a morphologically rich language, Arabic. Jang and Shin [10] proposes an approach for agglutinative languages and test their method on Korean short movie reviews and news articles. Wiegand et al. [20] investigate the impact of negation in sentiment analysis of German.

In the literature, it has been shown several times that Turkish, due to its highly inflectional and derivational structure, poses many different problems for different NLP tasks when compared to morphologically poor languages. By this property, previous NLP research on Turkish language pioneered the

studies for many similar languages. On the other hand, sentiment analysis studies for Turkish are very preliminary; although there exist a couple of studies on sentiment classification of movie reviews, political news, fairytales [17, 11, 3, 15], there exist very few studies on sentiment analysis of social media posts [5,6].

With the emergence of new tools dealing with automatic language processing of social media texts [8], it is now becoming possible to integrate them into higher level applications; i.e. sentiment analysis in our case. But, the following issues still reside as open questions:

1. the impacts of each NLP layers on sentiment analysis.
2. information (e.g. stems, main POS tags, inflectional features) to use from the outputs of beneficial layers.

In this paper, for the first time in the literature, we investigate and report the impact of the preprocessing layers (namely, tokenization, normalization, morphological analysis and disambiguation) on the sentiment analysis of Turkish social media texts. In order to show the maximum sentiment analysis performance to be achieved with flawless NLP tools, we used a hand-annotated sentiment corpus with gold-standard linguistic features.

2 Turkish

Turkish is an agglutinative language where each stem may be inflected by multiple suffixes. Every new suffix concatenation may change the meaning of the word or redefine its syntactic role within the sentence. This feature of Turkish yields to relatively long words (having higher number of characters when compared to other languages). As an ordinary example of this situation, the Turkish word "yapabilirmişcesine" can be translated as "as if he/she is able to do" into English. In addition, the example shows that the same English statement is expressed by a lesser word count (smaller mean sentence length) in the Turkish side. Therefore, semantic analysis of Turkish social media texts is more risky to be defeated by the erroneous writings within this informal domain. The various problems observed in the Turkish Tweets are presented in detail in [16]; these are mainly the missing vowels, diacritics, usage of emoticons, slang words, emo-style writings, spoken accents and high occurrence of spelling errors. The lower word count within a sentence leads to strict dependencies between words in Turkish and the only one single misspelled word can ruin the understandability of the whole sentence. This indicates the importance of normalization preprocessing stage for Turkish differently from English.

POS tagging task for other languages is performed in two stages for Turkish: morphological analysis and morphological disambiguation. Morphological analysis of a single word can produce several possible analyses regardless of the context in sentence. However, only one of them is correct in its context. The correct analysis can be selected by morphological disambiguation process on the morphological analysis results. Linguistic information about the word and possible relations with other words in the sentence can be extracted from the correct analysis.

3 The Used Data Set

For this study, we collected a twitter Turkish sentiment corpus mainly from the telecommunication domain. The data is retrieved from the Twitter API by querying a predetermined list of keywords. The time frame of the collected data was between May, 10th of 2012 and July, 7th of 2013. We refined the corpus from non-Turkish tweets through a language specifier based on a "Language Detection Library for Java"[1]. For the manual annotation of our corpus, we used TURKSENT [7] - a sentiment annotation tool

[1] It is available on https://code.google.com/p/language-detection/

which allows us to annotate the corpus on the following layers: general and target based sentiment, text normalization, morphology and syntax. For this study, we used only the general sentiment, the normalization and the morphological annotation layers of the tool.

Since the sentiment annotations depend on subjective decisions of the human annotators, we applied an inter-annotator agreement filter to increase the confidence level of our sentiment annotations. Our final dataset consists of 12790 tweets manually normalized, morphologically analyzed and classified between 3 sentiments (3541 positive, 4249 negative and 5000 neutral) agreed by two human annotators.

4 Feature Extraction Methods

In this study, we treat the sentiment detection of a tweet as a multi-class classification problem. We used support vector machines (SVM) in order to classify the tweets into one of the three classes (positive, negative, and neutral). When we extract unigrams from all collected data without preprocessing and feature filtering, we get 97472 unique features. This amount of features is extremely huge for machine learning algorithms, because more features ends up with more training time and more resources. In addition to time and resource constraints, irrelevant features may also ruin the steady nature of the trained model. Since feature extraction is an indispensable stage of machine learning algorithms, we applied an extraction method utilizing Inverse Document Frequency (IDF). While Term Frequency is easier and simpler than IDF calculation, it is not convenient if there are lots of recurring parts of texts which are the case for our study. Tweets are treated as single documents while calculating the document frequencies in IDF. After the calculation of IDF values of all unigram features, we filter them according to our proposed filtering algorithm MinClosestTh given below.

MinClosestTh. A small IDF value indicates a characteristic feature for a given class. But, in order for a feature to be discriminative between different classes, the difference between its IDF values should be bigger than a given threshold. In other words, a feature having similar IDF values for two classes does not help for the discrimination of these classes. For example, a stopword or a keyword which is used to retrieve data from Twitter API will have similar small IDF values for all classes. In the light of these observations, after testing with several feature extraction methods[1], we found that MinClosestTh performed the best. In this approach (Equation 1), we find the difference between the smallest and the second smallest IDF[2] value for a feature among all classes. The features, falling outside of this threshold are removed from the feature set.

$$|minIDF - medianIDF| > threshold \quad (1)$$

Figure 1 shows the histogram of |*minIDF* – *medianIDF*| difference distributions. One should notice that a determined threshold value will also determine the number of features to be used in the experiments; all the words entering to the bins greater than the threshold value will be included into the feature set. In order to select a good threshold value for further experiments, we investigate the sentiment analysis performance with different threshold values (0.1, 0.15, 0.3, 0.5 and 0.8). These are given in small line chart in Figure 1. As seen from the figure, the maximum f-measure is achieved at 0.15. F-measure is not the only metric to select the optimum threshold since the total feature count should also be considered. For example, the number of features in the feature set is 18536 when the threshold is chosen 0.1 and 3685 when 0.15. Although the difference between f-measures is

[1] Due to space constraints, we only provide here our best model.

[2] Since we have only three classes, the second smallest IDF is represented as medianIDF in Equation 1.

not dramatic, the lesser number of features is preferable. We selected 0.15 for further experiments since as seen from the figure, the performance drops consistently without having any important difference in feature counts.

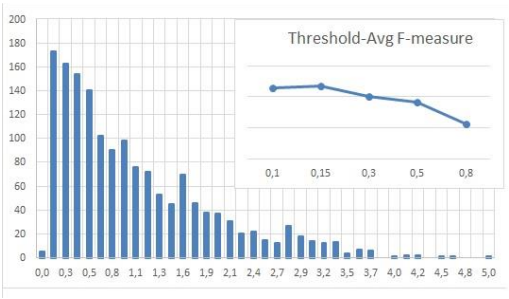


Figure 1. minIDF - medianIDF Histogram and Related Performance

5 Natural Language Processing Layers

Turkish is an agglutinative language and stems can be transformed theoretically to unlimited number of variations with derivational affixes. Moreover, all these different variations of a word may not make a difference on sentiment classification of tweets. Therefore, we want to polarize features which have the similar impact on sentiment to the same pole, and make explicit the difference between poles. We applied mainly three different NLP preprocessing layers (explained in previous sections) to transform features from original versions to the desired representations. Below we give the information extracted from the output of these layers.

Normalization. We used the normalized forms of the words before extracting the features. For instance, "tşkkrler" is normalized as "teşekkürler" (thanks).

Stemming. Stems of words have more general coverage than surface forms. To match different surface forms of a word into one simple token, we used stemming by deleting all inflectional groups and tags from its correct morphological analysis. For instance, "uzmanlar" (specialists), "uzmanlığı" (his/her specialty), "uzmanlık" (specialty) are derived from the same stem "uzman" (specialist). All three forms are turned into their stem "uzman".

Negation. As stated in [20], the detection of negation needs extra treatment in morphologically rich languages where the negation may be realized within the word with an affixation rather than a separate individual word. The case holds very frequently for Turkish, that's why our motivation in this section is to model the negation for sentiment analysis.

Negative indicators -such as the inflectional tags at the output of morphological analysis: "+Neg", "+WithoutHavingDoneSo" (like in use of regardless of, or without stopping)- have a power to turn meaning of words into opposite. For instance, "çekmiyor" (meaning "there is no signal" for the the telco domain) has a morphological analysis such as "çek+Verb+Neg+Prog1+A3sg" where the stem "çek" translated literally as to pull into English. If a feature will be extracted from this word we represent it as "çek+Neg". In addition, negation word, "değil" (means to not in English), has the same negative effect on preceding words. We put negation tag if a word contains negative indicators, or has "değil" as its successor. For instance, "iyi değil" (not good) is represented as "iyi+Neg". Furthermore, we added negation tag to the adjective if its successor is a negative verb. "Net göremiyorum." (I can't see clearly.) is transformed to "Net+Neg gör+Neg". When a word achieved double negation tag because of these conditions, we removed all the negation tags belonging to this word. For example, "sessiz değil" (not silent - "siz" suffix matches with less, like use in noiseless.) converted to "ses", not to "ses+Neg+Neg".

Using adjectives. We performed extra effort for adjectives in this research, because of the general belief that adjectives have a direct

Model#	Model	Name	Avg.	F-measure	Accuracy	Feature#
1	no_normalization	–	no_preprocessing	73.38	73.72	78025
2	normalization	78.05	78.28	39788	78.28	39788
3	normalization-stem	78.35	78.63	17855	78.63	17855
4	normalization-stem-neg	78.83	79.09	18493	79.09	18493
5	normalization-stem-neg-adj	77.93	78.27	23613	78.27	23613

Table 1. Sentiment Analysis Experiments Results

impact on sentiment analysis in comparison with other word types. We added adjectives to the feature set without exposure them to filtering by feature extraction methods defined previously. Even if we applied any of the other NLP preprocessing methods on adjectives just like any other word types, we also used surface form of adjectives as an additional feature instead of using only preprocessed versions. For example, we represent the adjective "tatsız" (tasteless) with two different features, "tat+Neg" (taste+Neg) and "tatsız".

6 Experiments and Discussions

In all of our experiments, we used SVM with linear kernel. In order to increase the confidence level of sentiment analysis, we applied 10-fold-cross-validation. The results are presented in terms of macro average of all iterations in Table 1.

We tested with 5 different NLP preprocessing models where each of them is the addition of a new processing layer on top of the previous one.

The first line of the table (no_normalization – no_preprocessing) presents our baseline model. This test is performed on the original version of the data set, in other words without applying any preprocessing during the selection of the feature set. The further experiments are evaluated according to their preceding experiments, and the performance improvement of the best model is reported with respect to this baseline.

Table 1 shows that the normalization stage (Model #2) contributes to the sentiment analysis, and increases the overall success by about 5 percentage points. On the other hand, although the addition of the stemming (Model #3) results in a slight improvement on top of Model #2, this improvement is not proven to be statistically significant according to McNemar's test. Despite this, Model #3 is considered very valuable since the total number of features is almost reduced by 50% (39788→17855). As a result, the lesser number of features provide us the ability to train our classifier by using less time and less resource as we mentioned in Section 4. This yields the possibility of adding more valuable training data to our machine learning algorithm, especially for active learning experiments.

Our final two experiments (Model #4 & Model #5) deal with the addition of some morphological features into sentiment analysis (detailed in Section 5). Although with the addition of negation (Model #4), we observed a slight improvement in the results, this improvement is again not statistically significant whereas it also increases the total number of selected features. A similar case holds for Model #5 again with no statistical significance, but this time with a small decrease.

As the conclusion, in this study, we showed that normalization is an indispensable stage for sentiment analysis whereas stemming is also very valuable for further studies (e.g. active learning). However, our tested model for the addition of morphological information into the system does not seem well-fitted for this domain. Nevertheless, we may not conclude that the morphological information such as

negation has no impact on sentiment analysis. We rather sense that we need to make further research on the inclusion of morphological features such as using them as separate features instead of the approach defined in here (the concatenation: Stem+Neg).

7 Conclusion and Future Work

Feature extraction methods provide us to decrease training time of classifiers, and also they have a positive impact on sentiment analysis success rate. We achieved higher sentiment analysis success rate with less number of features. In addition, we showed how the normalization improves the sentiment analysis on Turkish social media posts. With the normalization preprocessing, we increased the success rate of sentiment analysis from 73.38% to 78.05%, which is the 6.36% relative improvement. By the addition of morphological features we saw a slight improvement from 78.05% to 78.83% which is not statistically significant according to McNemar. However, stemming, which is the first morphological feature that we applied, is dramatically reduced the number of features as an advantage of ability to train models with more data. For our future studies, we will work on developing automatic NLP tools to make use of morphological information. Thereby, we want to build an environment for further linguistic analysis, such as syntax and semantics. We expect to increase sentiment analysis success by such deep analyzes of language.

8 Acknowledgments

This work is accomplished as part of a TUBITAK-TEYDEB (The Scientific and Technological Research Council of Turkey – Technology and Innovation Funding Programs Directorate) project (grant number: 3120605) in “Turkcell Global Bilgi” Information Technology Department. The authors want to thank Ozan Can Arkan for his valuable support during system development.

9 References

[1] **Muhammad Abdul-Mageed, Mona Diab, and Sandra Kübler.** 2014. Samar: Subjectivity and sentiment analysis for Arabic social media. Computer Speech & Language, 28(1):20–37.

[2] **Alexandra Balahur, Rada Mihalcea, and Andrés Montoyo.** 2014. Computational approaches to subjectivity and sentiment analysis: Present and envisaged methods and applications. Computer Speech & Language, 28(1):1–6.

[3] **Zeynep Boynukalin.** 2012. Emotion analysis of Turkish texts by using machine learning methods. Ms.

[4] **Rebecca F Bruce and Janyce M Wiebe.** 1999. Recognizing subjectivity: a case study in manual tagging. Natural Language Engineering, 5(2):187–205.

[5] **Mahmut Çetin and M Fatih Amasyali.** 2013. Active learning for Turkish sentiment analysis. In Innovations in Intelligent Systems and Applications (INISTA), 2013 IEEE International Symposium on, pages 1–4. IEEE.

[6] **Mahmut Çetin and M Fatih Amasyali.** 2013. Supervised and traditional term weighting methods for sentiment analysis. In Signal Processing and Communications Applications Conference (SIU), 2013 21st, pages 1–4. IEEE.

[7] **Gülşen Eryiğit, Fatih Samet Çetin, Meltem Yanik, Tanel Temel, and Ilyas Çiçekli.** 2013. Turksent: A sentiment annotation tool for social media. In Proceedings of the 7th Linguistic Annotation Workshop and Interoperability with Discourse, pages 131–134, Sofia, Bulgaria, August. Association for Computational Linguistics.

[8] **Gülşen Eryiğit.** 2014. ITU Turkish NLP web service. In Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Gothenburg, Sweden, April. Association for Computational Linguistics.

[9] **Minqing Hu and Bing Liu.** 2004. Mining and summarizing customer reviews. In Proceedings of the Tenth ACM SIGKDD International

Conference on Knowledge Discovery and Data Mining, KDD '04, pages 168–177, New York, NY, USA. ACM.

[10] **Hayeon Jang and Hyopil Shin.** 2010. Language specific sentiment analysis in morphologically rich languages. In Proceedings of the 23rd International Conference on Computational Linguistics: Posters, COLING '10, pages 498–506, Stroudsburg, PA, USA. Association for Computational Linguistics.

[11] **Mesut Kaya, Guven Fidan, and I Hakkı Toroslu.** 2013. Transfer learning using twitter data for improving sentiment classification of Turkish political news. In Information Sciences and Systems 2013, pages 139–148. Springer.

[12] **Soo-Min Kim and Eduard Hovy.** 2004. Determining the sentiment of opinions. In Proceedings of the 20th international conference on Computational Linguistics, page 1367. Association for Computational Linguistics.

[13] **Bing Liu.** 2012. Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies, 5(1):1–167.

[14] **Satoshi Morinaga, Kenji Yamanishi, Kenji Tateishi, and Toshikazu Fukushima.** 2002. Mining product reputations on the web. In Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining, pages 341–349. ACM.

[15] **Sadi Evren Seker and Khaled Al-naami.** 2013. Sentimental analysis on Turkish blogs via ensemble classifier. In Proceedings Of The 2013 International Conference On Data Mining. DMIN.

[16] **Dilara Torunoğlu and Gülşen Eryiğit.** 2014. A cascaded approach for social media text normalization of Turkish. In 5th Workshop on Language Analysis for Social Media (LASM) at EACL, Gothenburg, Sweden, April. Association for Computational Linguistics.

[17] **A Gural Vural, B Barla Cambazoglu, Pinar Senkul, and Z Ozge Tokgoz.** 2013. A framework for sentiment analysis in Turkish: Application to polarity detection of movie reviews in Turkish. In Computer and Information Sciences III, pages 437–445. Springer.

[18] **Janyce M Wiebe, Rebecca F Bruce, and Thomas P O'Hara.** 1999. Development and use of a gold standard data set for subjectivity classifications. In Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics, pages 246–253. Association for Computational Linguistics.

[19] **Janyce Wiebe.** 2000. Learning subjective adjectives from corpora. In AAAI/IAAI, pages 735–740.

[20] **Michael Wiegand, Alexandra Balahur, Benjamin Roth, Dietrich Klakow, and Andrés Montoyo.** 2010. A survey on the role of negation in sentiment analysis. In Proceedings of the Workshop on Negation and Speculation in Natural Language Processing, NeSp-NLP'10, pages 60–68, Stroudsburg, PA, USA. Association for Computational Linguistics.

[21] **Theresa Wilson, Janyce Wiebe, and Paul Hoffmann.** 2005. Recognizing contextual polarity in phrase level sentiment analysis. In Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing, HLT '05, pages 347–354, Stroudsburg, PA, USA. Association for Computational Linguistics.

[22] **Jeonghee Yi, Tetsuya Nasukawa, Razvan Bunescu, and Wayne Niblack.** 2003. Sentiment analyzer: Extracting sentiments about a given topic using natural language processing techniques. In Data Mining, 2003. ICDM 2003. Third IEEE International Conference on, pages 427–434. IEEE.

[23] **Hong Yu and Vasileios Hatzivassiloglou.** 2003. Towards answering opinion questions: Separating facts from opinions and identifying the polarity of opinion sentences. In Proceedings of the 2003 conference on Empirical methods in natural language processing, pages 129–136. Association for Computational Linguistics.

A RULE BASED NOUN PHRASE CHUNKER FOR TURKISH

Kübra Adalı
Dep. Of Computer Eng.
Istanbul Technical University
Istanbul, Turkey
kubraadali@itu.edu.tr;

Yrd. Doç.Dr.A. Cüneyd Tantuğ
Dep. Of Computer Eng.
Istanbul Technical University
Istanbul, Turkey
tantug@itu.edu.tr;

ABSTRACT

In this paper, we presented a noun phrase chunker for Turkish as an agglutinative language. For finding noun phrases in Turkish sentences, we propose a rule based model which includes preprocessing part and a unit that applies the local grammatical rules to the output of the dependency parser. To the best of our knowledge, our model gives the first results on noun phrase chunking of Turkish sentences that is expected to find not only the basic noun phrase sentences but also the complex noun phrases including the relative clauses. We believe that on that sense, our model will be a good reference for future studies in this domain. We tested our model both on manually annotated data and the output version of the dependency parser. Our model gives the results with annotated data for full match 66.15\% and the partial match 76.79\% (for F1 results). Using output of the dependency parser, the results are 47.91\% and 60.75\% for F1 results accordingly (for F1 results).

1 Introduction

As the conclusion of the wide usage of the internet and social media and the data on the web which is getting bigger day by day, the applications that are used to summarize huge amount of data such as information retrieval, text summarization, text categorization, information extraction become more popular and considerable for analysis of the web data. These applications needs meaningful groups of words so that they can analyze huge amounts of data without wasting effort for unnecessary details. Chunking is accepted as shallow parsing that parses a sentence into meaningful word groups (Ramshaw and Marcus, 1995). Additionally, it is used as preprocessing stage of dependency parsing and does not deal with as many details as full parsers. For this reason, chunkers become an important part of the applications that are motivated to summarize huge amount of data because they gives the word groups that gives the considerable information about the meaning of the sentence or gives the limited search spaces for a specific word or word group which represents a topic or an issue.

For machine translation, chunking of sentences gives the chance of making alignments in smaller search space in a sentence, increases the percentage of the word groups for alignment of the words from the source language to target language by finding phrases. For that reason, chunkers are used as a preprocessing stage in a machine translation system. In named entity recognition, especially noun phrase chunkers can be a very useful and substantial stage for named entity recognition.

After noun phrase chunker find the noun phrases in a sentence, it would be much more easier to find the named entities in noun phrases rather than in the whole sentence and in most of cases, the noun phrase becomes directly a named entity itself. (Sassano and Utsuro, 2000)

Although chunking is basicly accepted as dividing sentences into meaningful groups of words, the types of chunks are determining factor for the purpose of use of the chunker.

Noun phrase chunkers are used very commonly because they find chunks which contain words that organize around a noun and nouns are main types of words that gives the information about the topic or the meaning of the sentence. Noun phrases are defined as the basic word groups that qualifies a main word whose type is a noun in the literature, but we enlarge the area of the definiton of noun phrases because the scope of description of the noun phrases needs to be include more complex structures of word groups according to Turkish grammar. To give an example for Turkish, "büyük kırmızı bir elma" (a big red apple) is a standard basic noun phrase, but "ağaçtan düşen elma" (the apple which fell from the tree) is accepted by our study as a noun phrase although it is a relative clause and can not be a basic noun phrase.

In this paper, we propose a model that finds noun phrases in a Turkish sentence. Our model uses two stages: first part that is used preprocessing unit that gives the relations from the dependency parser and in the second stage, grammatical rules that uses the relations which are given by the Turkish dependency parser.

Although it is a rule based model and has language dependence, it is the first study for Turkish according to the scope of definition or complexity of noun phrases that are expected to find by our model. The first results of noun phrase chunker for Turkish 47.91% (for full match) and 60.75% (for partial match) as F1 results. We hope that our model and these results will be a reference for next studies for more complex noun phrase chunkers.

The remaining of the paper is structured as follows: Section 2 presents the related work, Section 3 explains the definition of a noun phrase in literature and the definition of noun phrases and the scope of the noun phrases in Turkish. Section 4 discusses our proposed model and Section 5 explains our experiments and results. The conclusion is given in Section.

2 Related Work

Noun phrase chunking has been done for many different types of languages by using many different methods. To start with English, Church (1988) used a stochastic model for noun phrase chunking. To the best of our knowledge, this is the first study about noun phrase chunking for English. Chen et al. (1993) deals with the noun phrase chunking problem on English by applying bigram language model as a statistical method. Cardie and Pierce (Cardie and Pierce, 1998) works on a system based on machine learning of the patterns that are composed of specific and common used sequences of part of speech tags on English. This is a hybrid model that includes rule based and statistical approach.

Another study about fusional languages is on German which Kermes et al. (2002) has built a rule based system that is composed of rules according to German grammatical structure. Another noun phrase chunker system for German is done by Atterer and Schlangen (Atterer and Schlangen,2009). This study uses an incremental structure.

On the other hand, Singh et al. (2005) reports the results of a chunker system for not only noun phrases but also the other types of phrases. The system works on Hindu by using HMMs. Vuckovic et al. (2008) uses a rule based model for noun phrase chunking of Croatian sentences based on morphological and syntactic structure of Croatian as a Slavic language. This study also works on the other types of phrases addditional to noun phrases. Dhanalakshmi et al. (2009) is an important study for Turkish because it possess noun phrase chunking problem on Tamil which is agglutinative language like Turkish. Three machine learning techniques that are CRFs, SVMs and memory based learning are used and compared. The winner method is CRFs for 9 different types of phrases including noun

phrases. Another chunking system which uses a machine learning technique is the system of Radziszewski and Piasecki (Radziszewski and Piasecki, 2010). They uses decision trees to deal with the noun phrase chunking problem on Polish which is a Slavic language. To train for the decision trees some tree patterns are used as features. Asahara et al. (2003) uses support vector machines for word segmentation using the groups of characters on Chinese. It can be accepted as a chunking study because each character can represent a word in Chinese.

The work done so far for Turkish are the work of Kutlu (2010) for noun phrase chunking and Akın and El-Kahlout (El-Kahlout and Akin, 2013) for chunking of constituents of Turkish sentences. The noun phrase scope of the study of Kutlu (2010) is different from the scope of our study (see Section 3 for further discussions), the application of the work is not accessible for public use, and Kutlu (2010) didn't give the details of the rules to apply the similar version of this. For the second work, also (El-Kahlout and Akin, 2013) only finds the constituents in a sentence and can not say if a chunk is a noun phrase or not. For these reasons, it is impossible to give the comparison between two later systems and our proposed model and the first study that we could find and which finds not only basic noun phrases but also complex noun phrases of a Turkish sentence is our study.

3 Noun Phrase Chunking

This section gives the details about the noun phrases in Turkish as an agglutinative language, the scope and complexity of the noun phrases that our work tries to find by supporting with examples.

3.1 Turkish

Noun phrases in Turkish has a main rule that the head word must be a noun. This rule is valid for all types of the noun phrases.

As an agglutinative language, the words in Turkish has the possibility of turn into an infinite number of different words by using the iterative sequences of inflectional and derivational suffixes. So as for all types of words, theoretically, it is a possibility that a noun is derivated from any types of words and this makes impossible to find the surface of the noun is derived from a noun or a verb. Morphological analyzer can not be enough in some cases to find this detail to decide the word can be a head of a noun phrase or not.

In Turkish sentences, the words that have a relation can have a long distance so this causes noun phrases become very long or a disambiguation problem about which noun is the head word of the noun phrase, which word defines which head word or belongs to which noun phrase. Additional to them, it is an another disambiguation problem to detect the phrases which seems to be a noun phrase but it composes a verb phrase.

3.2 Noun Phrases in Turkish

In Turkish, the noun phrases can be categorized as four main groups that are:
sanctus est Lorem ipsum dolor sit amet

1. Some examples can be as "Fatma'nın kitabı" (Fatma's book), "Kapının anahtarı" (key of the door), "Araba tekerleği" (wheel of car) etc. In Turkish, there are four different subtypes of this type of noun phrases according to suffixes that the head word and the assisting words of the noun phrase have.(Hengirmen, 2002)
2. Basic noun phrases in which the head word is connected by assisting words with the modification relation. To give some examples, "beyaz elbise" (white dress),"kırmızı gül" (red rose),"küçük ve zeki bir çocuk" (a small and clever child),"eski, şirin, küçük, boş, ve mavi bir ev" (a blue, old, sweet, small, empty house) etc.

3. The hybrid of the last two types of noun phrases. In hybrid type, only head part or only assisting part of the phrase can be a word or another type of phrase that is latter first or second type, or both of them can be a word or another type of phrase that is latter first or second type. Both two parts can be in three states (a word, a first type phrase and a second type phrase) so there are totally 9 subtypes of the hybrid type.
   The examples that belongs to these subtypes are: “Sema’nın güzel kızı” (the beautiful daughter of Sema), “Küçük çocuğun bisikleti” (the bicycle of small child), “Kırmızı başlıklı kızın yaşlı anneannesi” (the old grandmother of the girl with red hat), “Büyük kapılı evin küçük yeşil bakımlı bahçesinin ağaçları” (the trees of the small, green, well-kept garden of the house with big door)
4. Noun phrases whose head word is related with relative clauses. In this type, the head word must be a noun and if it is a derived noun, it shouldn’t be derived from a verb. The assisting part of the noun phrase is represented by a relative clause. This type is most complex and long type of the noun phrases in Turkish. The examples that are from real texts of news in Turkish are as follows:
   The examples that are from real texts of news in Turkish are as follows:
   - “Başbakan Kostas Simitis başkanlığında düzenlenen güvenlik toplantısı” (a security meeting that is headed by Prime Minister Costas Simitis)
   - “Dünyanın dört bir yanından televizyon izleyicilerinin tanık olduğu muhteşem gösteri” (a spectacular performance which is witnessed by television viewers around the world.)
   - “1996 yılında Atlanta’ da altın madalya alan Yunan rüzgar sörfcüsü Nikos Kaklamanakis” (windsurfer Nikos Kaklamanakis who is a gold medallist in Atlanta in 1996.)

As can be observed from the examples, the assisting part of the noun phrase can be so long also a phrase or a big sentence so it causes the growing of the complexity of the noun phrase chunking. The main rule that the head word must be a noun is valid for the multi word expressions whose last word is a verb can not accepted as a noun phrase.

4 Proposed Model

The work which is done so far in order to chunk noun phrases is based on two main different approaches that are mainly rule based systems and the systems in which machine learning methods such as CRFs, SVM, TBL etc. are used.(Section 2). Our model belongs to first group of approaches. Turkish is an agglutinative language. with rich, highly inflectional and derivational morphology, and complex relations between words in a sentence. Additional to the basic types of noun phrases, the scope of our study contains more complex and longer noun phrases as seen on the examples in the previous section. For these reasons, a rule based system needs to get detailed morphological analysis and relations between words in order to develop and apply the rules that finds accurate bounds of noun phrases in a sentence.

We propose a two stage model (Figure 1)

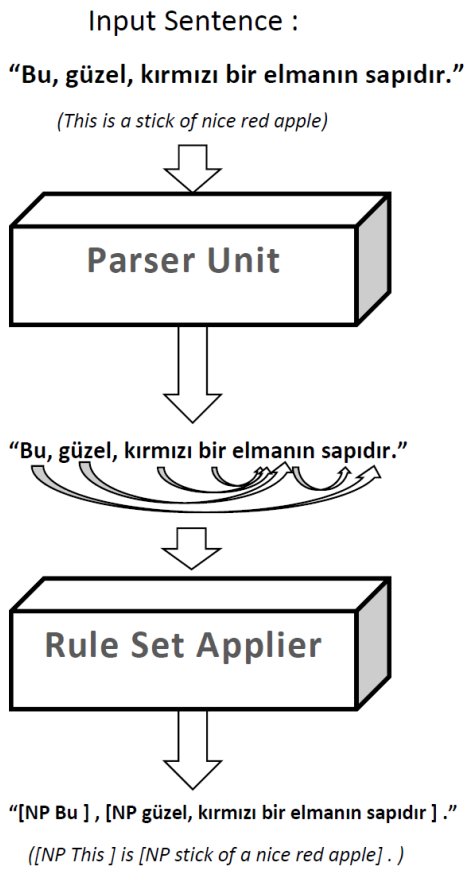


Figure-1. Proposed Model

which has basically two components.

1. A parser unit
2. A rule set applier

Our model uses four different labels for chunking noun phrases:
B : means that the word is the first word of the noun phrase.
I : means that the word is one of intermediate words of the noun phrase.
H : means that the word is head word of the noun phrase.
O : means that the word is not in a noun phrase.

4.1 Parser Unit

This stage is the part that we get the dependency relations of words. We employed dependency parser of Eryiğit et al. (2008) in order to obtain the relations to apply rules. The dependency parser needs morphological analysis of words with tags at the beginning and end of the sentence in a particular format so that this part contains two main substages:

1. Preprocessing
2. Parsing

In the preprocessing stage, there are two steps for a sentence as an input to the system. In the first step, we use the modified version (xxx, 2014) of two-level morphological analyzer (S¸ahin et al.,2013). After this step, we use morphological disambiguator by (xxx, 2014) and make the sentence in the available format for the dependency parser secondly. This preprocessing stage not only prepares input data for dependency parser but also serves the morphological information directly for rules. In the parsing stage, we use dependency parser of Eryiğit et al. (2008). It takes the input data which is prepared in the first stage, and gives the relation types and relation numbers in the same format.

It is shown by the Table 1 that an example sentence “Bu, güzel, kırmızı bir elmanın sapıdır.” which has 9 tokens tokenized, morphologically analyzed and disambiguated in the preprocessing stage and parsed in the second stage. At the end of the parse unit, the output that we use as the parse information of the sentence is shown on the Table-1. As seen on the Table-1 each word is represented by a word number and each relation number states the word has the dependency relation to the word which is the owner of the row. For example, the first word “bu” is related to the word “sapıdır” whose number is 8 and the relation type is subject.

Word Num.	Surf. Form	Lemma Form	Coarse P.Tag	Fine P.Tag	Feats	Rel. Num.	Rel. Type
1	Bu	bu	Pron	DemonsP	A3sg‖*Pnon*‖*Nom*	8	SUBJECT
2	,	,	Punc	Punc	_	3	notconnected
3	güzel	güzel	Adj	Adj	_	7	MODIFIER
4	,	,	Punc	Punc	_	5	notconnected
5	kırmızı	kırmızı	Adj	Adj	_	7	MODIFIER
6	bir	bir	Adj	Num	_	7	DETERMINER
7	elmanın	elma	Noun	Noun	A3sg‖*Pnon*‖*Gen*	8	POSSESOR
8	sapıdır	sap	Verb	Zero		9	SENTENCE
9	.	.	Punc	Punc	Pres‖*A3sg*‖*Cop*	0	ROOT

Table-1. The output from the dependency parser of the sentence that Figure 1 shows.

```
Procedure:
Initialize chunkbin
Initialize chunk
for i = 1 to Number_of_Words_in_Sentence
   Set currentWord = i. Word
   Set rel = relation type of i. Word
   Set nRel = next relation of i. Word
   Set nWord = next related word of i. Word
   Set chunk = null
   if  i. word doesn't has any incoming relations then
       add i to chunk
       while (nRel is not (sentence or root)) or (nWord is noun)
         if nRel is (modifier or determiner or possessor)
             add word number of relatedWord to chunk
         else if rel is (modifier or determiner or possessor)
             add word number of relatedWord to chunk
         else if rel is not (subject or object)
              if relatedWord is noun
                 add word number of relatedWord to chunk
          end
          Set currentWord = relatedWord
          Set rel = relation type of currentWord
          Set nrelation = next relation of currentWord
          Set nWord = next related word of currentWord
       end while
       add chunk to chunkbin
   end
end
for each (chunk as c in chunkbin)
   if c is a subset of any chunk in chunkbin then
       Remove c from chunkbin
   end
end for each
```

Figure-2. The pseudo code for the algorithm of rule based system.

There are totally 26 types of relations that is defined for the dependency parser and the number of possible relation word for each word is equal to the number of words in a sentence. So the complexity for the rule set for each word in a sentence is 26xn (n is the number of words in a sentence.).

4.2 Set Of Rules

The morphological analysis and dependency relations of sentence that is taken from the Parse Unit are transferred to the Set of Rules stage. This stage which is based on a chunking algorithm gives the noun phrases annotated in the sentence as output. The pseudo code of the algorithm is given as in the Figure 2.

The algorithm visits each word in a sentence and firstly checks if the word has an incoming relation from any other words in the sentence in order to decide to start a chunk. If the word has an incoming link, the algorithm passes to the next word and start the rest of the procedure again for the next word, if the word does not, the algorithm starts to a new chunk and puts the word into the chunk. After that, the algorithm checks the related word recursively if it should be put into the chunk until the related word reaches the sentence relation. It puts the related word to the chunk and goes on at three states:

1. the next relation type is one of modifier, determiner, or possessor: this is for the noun phrases that contains relative clauses.
2. the relation type is one of modifier, determiner, or possessor.
3. the relation type is not one of (subject or object) and related word is noun.

After the algorithm reaches the sentence relation for each chunk, the chunk is put to the chunk bin. When the walk of the algorithm on the words of the sentence completed, the chunks that is subset of an another chunk in the sentence are filtered from chunk bin as the last step of the algorithm. In the example at Figure 1, the relations of the sentence that can be seen on the Table-1 is given to the algorithm and gets 2 noun phrases that are "Bu" (This) and "güzel, kırmızı bir elmanın sapı" (stick of a nice, red apple).

5 Experimental Setup And Results

We will focus on the results of experiments of our proposed model after we give the details about our datasets and evulation metrics that we used in experiments in this section.

5.1 Datasets and Evaluation

As it is shown in the model (Section 4), our model uses dependency parser so we need parsed data with manual annotation as gold standard data for the Set of Rules stage of our model or in order to identify the effect of the dependency parser to the score. For this reason, we collected a test and development set from Metu-Sabancı Treebank by (Eryigit et al., 2011),(Atalay et al., 2003) and (Oflazer et al., 2003) and we manually annotated the noun phrases of the set.

The set has 500 sentences, 1252 noun phrases, 5293 tokens, and 6333 relations in total. We divide the set into two parts that are equal number of sentences. We used the first part as development set which has 250 sentences, 557 noun phrases, 2250 tokens, 2745 relations and the second part as test set which has 250 sentences, 695 noun phrases, 3043 tokens, 3588 relations in total. We use two types of F1 scores first of which is F1 score over the noun phrases that matches fully all of the words of the gold standard noun phrases (F1Fullmatch Equation 1) and the other F1 score is calculated by the average(F1Pmatch Equation 2) of the F1 scores of partial match scores(pm Equation 3). The second metric is also used by Kutlu (2010).

$$F1_{Fullmatch} = \frac{\#\ of\ fullmatched\ noun\ phrases}{\#\ of\ noun\ phrases} \quad (1)$$

$$F1_{P_{match}} = \frac{\#\ of\ total\ pm\ of\ noun\ phrases}{\#\ of\ noun\ phrases} \quad (2)$$

$$pm = \frac{\#\ of\ corr\ match\ words\ of\ noun\ phrases}{\#\ of\ words} \quad (3)$$

5.2 Experiments

Since we work on a rule-based system, we need to the rule for start point of a chunk. For this aim, we calculate the distribution of numbers of types of relations onto the types of combination that consists of the chunk label of current word, and the chunk label of related word. For example, for the phrase “beyaz çiçek” (white flower), the chunk label of “beyaz” (white) is B, and the other chunk label is H and the type of the combination of chunk labels which belongs to the relation from the word “beyaz” (white) to the second word is ”BH”. From the chunk labels B,I,H,O , 16 different types of combinations can exist(BB,BI,BH,BO,IB,II,IH,IO etc...). The numbers of distribution of 26 types of relations to the 16 types of chunk label combinations tells that the start point of a chunk is any word that does not has an incoming relation in a sentence can be a chunk because the total number of combinations BB,IB,HB,OB is 0. Additional to it, more than 90% of the numbers of the relation types (modifier, determiner, possessor) states inside a chunk (has numbers BI, IH, II, BH chunk label groups).The third main idea which comes from the analysis is that more than 90% of the numbers of the relation types (subject, object) states outside a chunk or goes out of a chunk (has numbers HH, BO, IO, HO, OO chunk label groups). The last result that 100% of the numbers of the relation types (sentence, root) 6 tells that the end of the constituent should be the end of the opened chunk.

After we complete the algorithm by writing rules that are constructed by using the analysis of numbers of the chunk label combinations according to relation types, we evaluated our model both with the data which is parsed by manual annotation and the same data but parsed by automatically by the dependency parser.

System	F1 Score F. Match	F1 Score P.Match
Model+Parse By Manual An.	66.15	76.79
Model+Parse By Dep.Parser	47.91	60.75

Table-2 Results of The Proposed Model

As seen on the (Table 2), the quality of parsing directly effects the scores of our model. Our scores are 47.91% for full match and 60.75% for partial match. That means that our noun phrase chunker can find almost half of the noun phrases or the cuts the scope of the related topic in half in an NLP system and increases the performance of the system.

6 Conclusion And Future Work

Our study was a rule based model for chunking noun phrases in Turkish sentences which can assist many types of applications such as information extraction, text summarization, machine translation... etc.

We obtained F1 scores 47.91% for full match and 60.75% for partial match. This is the first study that does the chunking of noun phrases that are as complex as they have relative clauses to the best of our knowledge although a study of noun phrase chunker for Turkish has been done as an agglutinative language so it would be irrational to compare our model and the previous study. Since our model was a rule based model, the development of the model becomes impossible at the level of the state that the system start to develop rules that are for rare, single, and specific conditions. So, as the first future plan, we want to apply a machine learning method for noun phrase chunking or different machine learning methods to detect the machine learning method that gives the best results for Turkish.

The second future plan about noun phrase chunking is to use language models instead of using dependency relations because using language models would make the chunking time less and gave good results for fusional language such as English. The other four future plans for this study is to do the experiments how our study effects the scores of mainly NLP tools that does named entity recognition, dependency parsing, detecting of multi word expressions, sentiment analysis.

7 References

[1] Masayuki Asahara, Chooi Ling Goh, Xiaojie Wang, and Yuji Matsumoto. 2003. Combining segmenter and chunker for chinese word segmentation. In Pro- ceedings of the Second SIGHAN Workshop on Chinese Language Processing - Volume 17, SIGHAN'03, pages 144–147, Stroudsburg, PA, USA. Association for Computational Linguistics.

[2] Nart B. Atalay, Kemal Oflazer, Bilge Say, and Informatics Inst. 2003. The annotation process in the turkish treebank. In Proc. of the 4th Intern. EACL Workshop on Linguistically Interpreteted Corpora (LINC).

[3] Michaela Atterer and David Schlangen. 2009.Rubisc: A robust unification-based incremental semantic chunker. In Proceedings of the 2Nd Workshop on Semantic Representation of Spoken Language, SRSL'09, pages 66–73, Stroudsburg, PA, USA. Association for Computational Linguistics.

[4] Claire Cardie and David Pierce. 1998. Errordriven pruning of treebank grammars for base noun phrase identification. In Proceedings of the 17th International Conference on Computational Linguistics - Volume 1, COLING'98, pages 218–224, Strouds- burg, PA, USA. Association for Computational Linguistics.

[5] Kenneth Ward Church. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In Proceedings of the Second Conference on Applied Natural Language Processing, ANLC'88, pages 136–143, Stroudsburg, PA, USA. Association for Computational Linguistics.

[6] V. Dhanalakshmi, P. Padmavathy, M. Anand

Kumar, K. P. Soman, and S. Rajendran. 2009. Chunker for tamil. In ARTCom, pages 436–438. IEEE Computer Society.

[7] Ilknur Durgar El-Kahlout and Ahmet Afsin Akin.2013. Turkish constituent chunking with morphological and contextual features. In CICLing (1), pages 270–281.

[8] Gülşen Eryiğit, Tugay Ilbay, and Ozan Arkan Can.2011. Multiword expressions in statistical dependency parsing. In Proceedings of the Second Work- shop on Statistical Parsing of Morphologically Rich Languages (IWPT - 12th International Conference on Parsing Technologies), pages 45–55, Dublin, Ireland, October. Association for Computational Linguistics.

[9] Gülşen Eryiğit. 2014. ITU Turkish NLP web service.In Proceedings of the Demonstrations at the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL), Gothenburg, Sweden, April. Association for Computational Linguistics.

[10] Gülşen Eryiğit, Joakim Nivre, and Kemal Oflazer. 2008. Dependency parsing of turkish. Computational Linguistics, 34(3):357–389.

[11] Mehmet Hengirmen. 2002. Tamlamalar. In Türkçe Dilbilgisi, pages 118–142. Engin.

[12] Kuang hua Chen and Hsin-Hsi Chen. 1993. A probabilistic chunker. In In: Proceedings of ROCLING VI, pages 99–117.

[13] Hannah Kermes and Stefan Evert. 2002. Yac – a recursive chunker for unrestricted german text. In Rodriguez M G, Araujo C P (eds), Proceedings of the Third International Conference on Language Resources and Evaluation, Las, pages 1805–1812.

[14] Mücahit Kutlu. 2010. Noun phrase chunker for Turkish using dependency parser.

[15] Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani Tür , and Gökhan Tür. 2003. Building a turkish treebank.

[16] Adam Radziszewski and Maciej Piasecki. 2010. A preliminary Noun Phrase Chunker for Polish. In Intelligent Information Systems, pages 169–180. Springer.

[17] Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. CoRR, cmp-lg/9505040.

[18] Muhammet Şahin, Umut Sulubacak, and Gülşen Eryiğit. 2013. Redefinition of Turkish morphology using flag diacritics. In Proceedings of The Tenth Symposium on Natural Language Processing (SNLP-2013), Phuket, Thailand, October.

[19] Manabu Sassano and Takehito Utsuro. 2000. Named entity chunking techniques in supervised learning for japanese named entity recognition.

[20] Akshay Singh, Sushma Bendre, and Rajeev Sangal.2005. Hmm based chunker for hindi. In In the Proceedings of International Joint Conference on NLP.

[21] Kristina Vuckovic, Marko Tadic, and Zdravko Dovedan. 2008. Rule-based chunker for croatian.In LREC. European Language Resources Association.

[22] İnternet Sistemleri Konsorsiyomu, www.isc.org/solutions/survey, Erişim tarihi: Kasım 2011.

[23] xxx. 2014. Morphological processing of turkish. xxx, x(x), x.

TOWARDS AUTOMATIC SPEECH RECOGNITION FOR THE TATAR LANGUAGE

A.F. Khusainov
Institute of Applied Semiotics, Tatarstan Academy of Sciences
Kazan (Volga region) Federal University, Kazan, Russia

khusainov.aidar@gmail.com

Dz. Sh. Suleymanov
Institute of Applied Semiotics, Tatarstan Academy of Sciences
Kazan (Volga region) Federal University, Kazan, Russia

dvdt.slt@gmail.com

ABSTRACT

In this paper we describe an approach to create automatic speech recognition systems for the Tatar language. We developed speech analysis platform to work with under-resourced languages and used this tool to create baseline speech recognition system. Additionally, some changes have been made to this language-independent system to take into account specific Tatar morphological structure. The resulting adapted system showed 75% accuracy on testing audio records.

1 Introduction

Using speech as a tool for manipulating electronic devices is becoming more and more common. This fact can be proved by lots of desktop and web-based services that provide functionality of automatic dictation, voice search, etc. Nevertheless, while these kinds of systems successfully work for main world languages such as English, French, Spanish, there are many languages for which speech analysis systems are not so developed.

According to Ethnologue project's statistics, more than 7100 languages are spoken in the world [1]. The significant part of these languages suffers from absence of speech services on their native languages, therefore people have to learn and use other languages in order to communicate with modern information technologies.

In this paper, we aimed to develop a platform that can be used for building baseline language-independent speech analysis systems and to use this platform to create specific speech recognition system for the Tatar language.

The structure of the rest of this paper is as follows: in Section 2 we give overview of proposed platform, including the description of its features and language-independent tools. In Section 3 we describe the aspects of using proposed platform to build the Tatar speech recognition system. Finally, Section 4 deals with experimental results achieved for continuous speech recognition task.

2 The architecture of the platform

Speech analysis systems differ by their final goal (speech recognition, speaker identification, etc.), by the language they built for and especially by the conditions under which they work properly and can be successfully used. Nevertheless, most speech analysis systems use several common blocks and similar tools. According to that fact proposed platform are built consisting of two main elements: modules (which allow re-using standard parts of algorithms) and projects (which consist of modules and focused on solving specific analysis problem).

Each module deals with some subtask and can be repeatedly used without code duplicating. In order to provide possibility to enhance quality of model's work without losing any information about its settings and relations with other modules platform provides simple version control system. In addition, it can be used to compare different realizations of some algorithm by running it two times choosing different versions of module.

Speech analysis systems not only use several common subsystems like feature calculating, but also use information from other speech analysis systems. For instance, continuous speech recognition system can use information from speaker identification system in order to increase effectiveness of its work. To implement this possibility into platform each module has list of input and output parameters. Parameter's value can be equal to a simple value or can be a reference to other module's parameter.

In addition to universal mechanism of version control and possibility to exchange information between modules platform provides several tools to ease and automate common steps of speech analysis system's creation:

1. "Acoustic features" – allows user to define phoneme and character alphabets of language and to formulate main grapheme-to-phoneme rules.
2. "Text analysis" – provides functionality of automatic phoneme transcribing (based on rules constructed in "Acoustic features" tool) and statistical analysis of the result transcription (2- and 3-gramm calculation, plotting histogram, etc.). Allows constructing text corpus with associated transcription file.
3. "Recording" – automate basic operations of constructing speech corpus, contains special visual form for saving information about speakers (age, gender, mother tongue, dialects, noise conditions), helps with creating distribution of sentences between speakers and recording corpus based on this distribution.
4. "Acoustic models" – this module allows to create acoustic models based on Gaussian mixtures models.
5. "Grammar" – automate process of creating named group of words and allows to create file which contains grammar rules for specified recognition task.
6. "Speech recognition" – execute decoding procedures according to acoustic models and given task grammar.

Developed modules are language-independent, so they can be easily configured to work with specific language. Together these modules form the skeleton of the baseline speech recognition system for any language, Fig. 1. As can be seen in Fig. 1 first five modules do the initial work of building language, pronunciation, acoustic models. These models are used by "Speech recognition" module in order to analyze input speech utterance and calculate recognition accuracy.

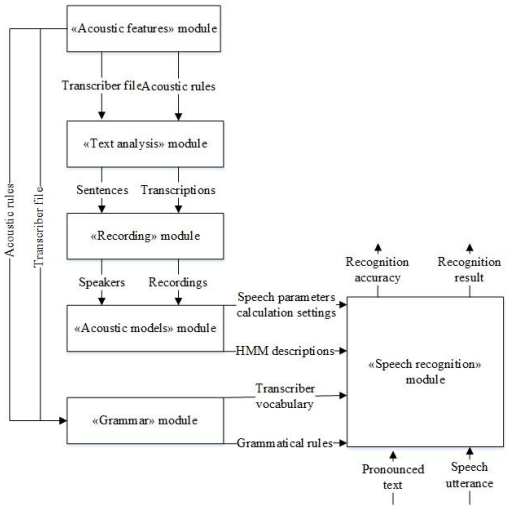


Fig. 1. Baseline speech recognition system structure

3 Continuous speech recognition system for the Tatar language

The Tatar language can be referred to under-resourced languages due to the low-level of developed information technologies and absence of well-designed text and speech corpora. At the same time, there are more than 8 million Tatar-speaking people in the world. Therefore, there is a great demand for speech technologies adapted to work with Tatar language.

To satisfy this demand and to show the potential of using the proposed platform we developed two speech recognition system for the Tatar language.

The first application is baseline speech recognizer built based on proposed analysis tools (for example, grapheme-to-phoneme conversion tool, acoustic modeling and training/decoding tools) that are encapsulated by 6 modules. Each module has been properly set up and used to create initial data for the Tatar language. These data used to build necessary acoustic, pronunciation and language models.

The second application is adapted recognition system that takes into account specific morphological features of the Tatar language. Changes have been primarily made to pronunciation and language models, details presented in Section 3.5.

3.1 Acoustic features of the Tatar Language

Obviously, acoustic features of specific language are the basic information for all types of recognition systems. These features can be described as consisting of character and phoneme alphabets and rules of conversion from character to phoneme representations. This information will be used by the next steps of analysis. The main result of this stage is automatic phoneme transcribing tool.

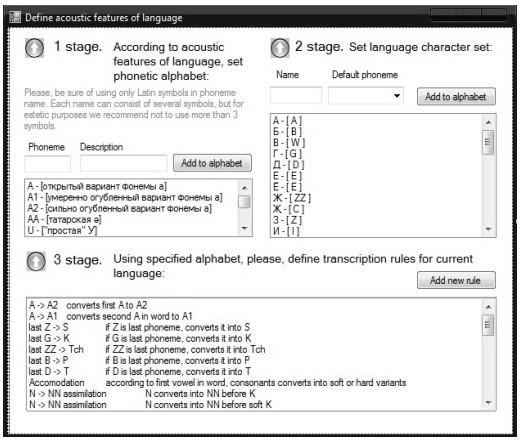


Fig. 1. "Acoustic features" module

"Acoustic features" module used to automate and provide necessary visual forms and formal kind of acoustic rules representation, Fig. 2. Module consists of three main parts, each specialized to work with character alphabet, phoneme alphabet and acoustic rules respectively. As a result, for the Tatar language we have used 39 characters alphabet (Russian alphabet plus 6 specific Tatar characters: Ә-ә, Ө-ө, Ү-ү, Җ-ж, Ң-ң, Һ-h), 56 phonemes (43 consonants and 13 vowels) and 37 rules of grapheme-to-phoneme conversion [2].

3.2 Text corpus and language model

In order to build phonetically rich and balanced speech corpus, we have to create text corpus with similar features. Therefore, we used automatic phoneme transcription subsystem and statistical analysis of resulting transcriptions in "Text analysis" module, which is shown in Fig. 3.

Based on the mentioned tools we have created text corpus, which consists of separate parts differentiating by text source types: news, literature, separate words, spontaneous spoken sentences. Total amount of sentences is 776,

number of words – 6913; all chosen phonemes are presented in sentences' transcriptions.

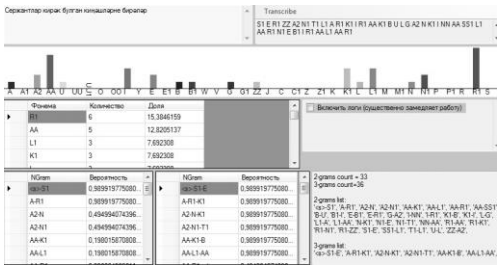


Fig. 3 "Text analysis" module

Based on collected text data language model can be constructed. We apply 3-gram language model into our speech recognition system. This model based on assumption that probability of each word depends only on previous two words, so this probability can be approximately estimated via statistical analysis of huge sequence of words.

Estimated probabilities will be used at the decoding stage to help recognition system to predict right sequence of words.

3.3 Speech corpus and acoustic models

Building multi-speaker speech corpus for the Tatar language is currently in progress. Currently it contains voices of 251 speakers, average age – 18.2. Each of speakers read the set of 36 sentences from the text corpus: 13 sentences from literature part, 7 – from news part, 15 – from words part, and 1 – from spontaneous part. At the same time, each sentence from literature has been read by 20 different speakers, from news and words – by 10 speakers, from spontaneous part – by one speaker. The total number of sentences in corpus is equal to 8638. The result features of currently available speech corpus are shown in Table 1.

Table 1. Features of multi-speaker speech corpora for the Tatar language

Parameter	Value
Number of files	8638
Total duration	8:14:24
Number of files in training subcorpus	8125
Duration of training subcorpus	7:48:12
Number of files in testing subcorpus	513
Duration of testing subcorpus	0:26:12

Corpus contains additional information about speakers (gender, age, mother tongue) and expert's score of speakers' proficiency in Tatar.

Automatic phoneme alignment approach realization has been built in "Acoustic module". This module allows to create acoustic models using two different types of input data: speech records from corpus and corresponding texts. Based on this data 57 acoustic models (56 – for phonemes, 1 – for silence model) were trained by "Acoustic module" using the HTK toolkit [3]. Models are 3-state left-right Gaussian mixture models. Number of Gaussians in mixtures varied from 1 to 170, the best phoneme recognition accuracy was showed by the models with 31 Gaussians in each mixture.

3.4 Pronunciation model

For evaluating the quality of the developed system, we used task grammar that allow speakers to pronounce every possible word sequence. Vocabulary for this task is medium-size (1135 words) and consists of words that occur in the test subcorpus, so, we have simulated rather compact task domain.

The last step in preparing data for the decoding stage is creating pronunciation model. This kind of model is a bridge between phonemes and words level of the recognition system. Each word has to be represented by sequence

of appropriate phonemes, this will make possible to solve the inverse task of decoding words from sequence of phonemes. Phoneme transcription of all words have been defined using developed grapheme-to-phoneme tool.

3.5 Adapted speech recognition system

The second application differs with the approach used to build language and pronunciation models. The idea is practically the same: we have to estimate statistics of 3-grams and to build phoneme transcriptions for all elements. The difference is that these elements are not whole words but sub-words units. The Tatar language is agglutinative language (words are constructed by concatenating of several morphemes) with rich morphology. Using sub-words units is profitably for this kind of languages, because it helps to reduce the number of units in vocabulary, but at the same time to widen the amount of covered words [4].

This approach called particle-based and requires implementing additional morpheme level into recognition system. Considering this fact, the process of building adapted language model is as follows:

- All words in existing text corpus are divided into morphemes.
- Last morphemes of each word are provided with additional '#' sign that means 'the end of the word'.
- Statistical 3-gram model are built for morphemes and '#' sign.

The pronunciation model also needs to be changed, because not words but morphemes have to be constructed from phonemes. This leads to the multiple transcription model, because some morphemes can be pronounced differently depending on context in concrete word.

4 Experimental results

We used the test part of the speech corpus for the purpose of continuous speech recognition experiments. Overall, the speech recognition systems have shown good accuracy rates near 70 percent.

As can be seen in Table 3, the adapted system outperformed baseline in both correctness and accuracy coefficients; that proves the fact that adding morphological level helps to build models and execute recognition in more accurate manner.

Table 2. Continuous speech recognition results

Parameter	Baseline system	Adapted system
Correctness	77%	83%
Accuracy	67%	75%
Number of words in all sentences	3368	3368
Substitution errors	735	533
Deletion errors	50	39
Insertion errors	316	269

4 References

[1] Lewis, M. Paul, Gary F. Simons, Charles D. Fennig (eds.). "Ethnologue: Languages of the World", Dallas, Texas: SIL International, 2013.

[2] Khusainov A.F. "Automatic phoneme recognition system for the Tatar language". In: The 1st International Conference "TurkLang", Astana, 2013, pp 211–217.

[3] Young S., Kershaw D., Odell J., Ollason D., Valtchev V., Woodland Ph. The HTK Book [Electronic resource]. URL: http://nesl.ee.ucla.edu/projects/ibadge/docs/ASR/htk/htkbook.pdf.

[4] Kurimo M, Puurula A., Arisoy E., Alumae T., Saraclar M.. "Unlimited vocabulary speech recognition for agglutinative languages". In: HLT-NAACL, NY, USA, 2006, pp 487–494.

A Mobile Assistant for Turkish

Gökhan Çelikkaya *Gülşen Eryiğit*
Dep. of Computer Eng. Istanbul Technical University Istanbul, Turkey
gcelikkaya@itu.edu.tr; gulsen.cebiroglu@itu.edu.tr;

ABSTRACT

In this paper we present a design and an implementation of a mobile assistant application that understands and meets users' requests in Turkish language. The application is able to understand requests for a set of phone operations such as calling a contact or sending an email and requests for a set of information services such as map, weather, and traffic. The understanding of user queries relies on existing research on natural language processing for Turkish and a hybrid approach of rule-based and statistical classification methods. Our performance tests revealed high accuracy results for the operations that our application supports.

1 Introduction

Today people try to find easier and simpler ways to interact with their mobile devices such as smart phones and tablet computers to have their work done and access to information. In this regard, voice-controlled mobile assistant applications became very popular recently. Ever since SIRI [1]became a success and several alternative applications have been released to the application markets. Nevertheless, most of those applications support only English and provide services mostly for English-speaking countries.

In the last years, successful voice recognition [1], and natural language processing [2], [3], [4], [5], [6], [7], [8] components have been released for Turkish. However, such components have not been integrated with any high tier applications.

In this study, we present our ongoing effort on developing a mobile assistant application and its associated services that understands Turkish input and accomplish users' requests with high accuracy. The application is able to understand various user intents including but not limited to calling and texting to a contact, accessing to weather and traffic information. We explain our system architecture and our methodology of understanding user requests, in which we leverage and improve existing research and tools for Turkish NLP such as morphological analyzer, morphological disambiguator, named entity recognizer, and dependency parser.

Understanding user requests is a multi-step process in our system; the user query is processed with NLP tools, and then the query is mapped to one of the supported operations through a hybrid approach of rule-based and statistical classification. In the following step, the parameters such as contact name or search term -if any- from the query are extracted so that the mobile device can handle the requested operation or an external web service can be queried with correct parameters in order to fetch the requested information. We present and discuss the performance of the classification and parameter extraction methods that we have tested using a set of real user queries.

[1] http://www.apple.com/ios/siri/

Table 2: Performance of parameter extraction in terms of accuracy

Domain	Call	SMS	E-mail	Web Search	Start Apps	Map	Weather	Exch.	Overall
Acc. (%)	72.86	81.22	88.67	37.50	74.51	81.48	72.38	36.36	69.25

The rest of this paper is organized as follows: Section 2 gives brief information about related work, Section 3 describes the design of our system, Section 4 explains the process and methodology of understanding users' requests in the Turkish language within the context of our application, and finally, Section 5 concludes and discusses future work.

2 Related Work

As it is the case for most of the language technology studies, the mobile assistant technology also started firstly for English. There are various patents [9], [10], publications [11] and applications for English; e.g. Siri (Apple) and Google Now (Google) are the most advanced and the most used voice-controlled personal mobile assistant applications in the world today. Since Turkish is a morphologically rich language, its morphology and syntax are very different and arguably more complex compared to English. Thus, most of the time it is not convenient to use the same methodologies and tools that are proposed and available for English.

To the best of our knowledge, there do not exist any voice-controlled assistant application that supports the Turkish language by leveraging advanced natural language processing techniques and services to understand users' true intent with high accuracy. Our survey on the existing Turkish virtual assistant applications, of which there are a few in total, revealed that they rely on rule-based approaches and use very simple natural language processing components or even none and focusing more on visual user experience. For such reasons, the ability of those applications to process queries in natural language and perform the requested actions is not strong or comprehensive. Turkcell Mobil Assistant[2] is one of the most well-known Turkish mobile assistant application. However, it can reply questions only in a specific pattern. For instance, while it can understand the question "hava nasıl?" (how is the weather?) and returns the weather forecast in the current location, it fails to reply with the correct result to the question "NewYork'da hava nasıl?" (how is the weather in NewYork?) where the user actually wants to query the weather in a different location. In this work, we show that utilizing the recent research and tools on Turkish NLP will facilitate the development of a mobile assistant application with a high level of performance. By this way, we expect our project to become a pioneer in its field of practice.

3 System Design

In this section we describe the design of our system which comprises a mobile Android client application and a server-side service. The mobile application (Figure 1) is responsible to convey user requests to server and display the service results or perform the requested operation on behalf of the user such as calling a contact person or starting an application. The server processes a user query to understand his/her true intent, then if required delegates the request to the third party web services, and finally, compiles the results and send it back to the client. It is beyond the scope of this paper to discuss the runtime performance of the server-side implementation; though, it is worth mentioning that the query understanding (natural language processing and query classification) takes about half a second on average on a commodity hardware.

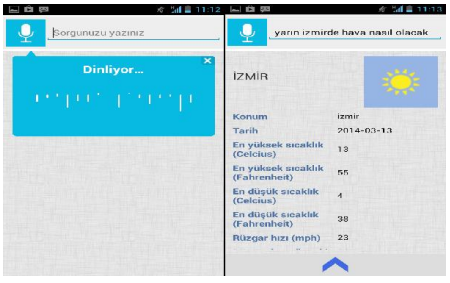


Figure 1: Screen shots of the mobile application: recognizing speech input (left), displaying results for a weather query (right).

[2] http://www.turkcell.com.tr/servisler/turkcell-mobil-asistan

We use Google's built-in speech recognizer to convert users' speech queries into text, which we have tested and found efficient for Turkish speech recognition (we have considered accent variations of Turkish as well). The speech recognizer has a limit of 160 characters per speech request. Besides, it provides a set of confidence levels along with the text results. We consider the text with the highest confidence level as the correct mapping of what user has said, if the confidence level of this text is greater than 0.7 (max. being 1.0), otherwise, the application presents a list of text suggestions to the user ordered in the descending order of their confidence levels. The selected query text is further sent to the server along with user context which includes the user id, the current location and time. The communication between client and server is based on REST principles to make the server-side more scalable as no user session being stored on the server between requests. Both the request and response messages are encoded with JSON format on our messaging protocol.

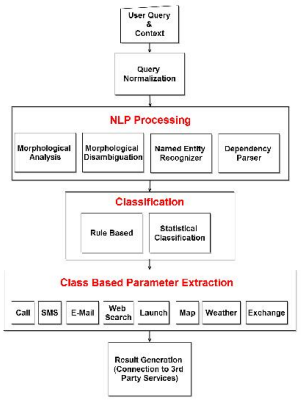


Figure 2:The flow of operations for understanding user requests.

4 Understanding User Requests

In this section, we explain the process and methodology of understanding users' requests in Turkish. Currently our application supports nine types of requests:

- calling a contact,
- sending sms to a contact,
- sending email to a contact,
- searching terms on Web (we rely on Google search),
- starting installed apps on the phone,
- getting route directions or searching a point of intereston the map (Google Maps), exchanging between currencies (Google Currency Service),
- accessing the weather forecast (World Weather Online)
- accessing to traffic information (Yandex Traffic Service)

Figure 2 presents the flow of operations for understanding user requests. Upon receiving a user's query and context, the query is normalized (curated) through the steps of abbreviation expansion, capitalization of proper nouns, and verb tense correction [8]. In the second phase, the normalized query is processed via a Turkish NLP pipeline available as a SaaS [12] consisting of a morphological analyzer [7], a morphological disambiguator, a named entity recognizer [6], and a dependency parser [5]. The outputs of natural language processing components are used both in the classification and parameter extraction phases. In the classification phase, the processed query is mapped to one of the supported operations (i.e., domains). Our classification methodology is a hybrid approach of rule-based and statistical classification. After a query is mapped to a domain, the parameters[3] are extracted from the query, so that the phone can perform the requested operation or related web services can be called with correct parameters.

4.1 Query Normalization

As speech queries are in natural language, they often include grammar and pronunciation errors. Moreover, the used speech recognizer contributes to such errors by for instance separating suffixes from their stem despite the fact that Turkish is an agglutinative language and by outputting mistakenly the first letter of proper nouns with lower case letters (e.g. "gökhan a" instead of the correct form "Gökhan'a"). Consequently, the performance of the NLP tools we use degrades due to these erroneous spellings. In order to alleviate this problem, we needed to make extra normalization effort in addition to the previous work [8]

[3] Since we do not extract any parameter for the traffic service, Figure 2 shows only 8 domains in the parameter extraction phase

normalizes its input by the following stages: letter case transformation, replacement rules and lexicon lookup, proper noun detection, deasciification, vowel restoration, accent normalization. During our experiments, we observed that the most important normalization layers for our case were 1) The capitalization of Proper Nouns and 2) Accent Normalization which fixes verb tense errors occurring under spoken accents. The normalizer understands the tense of a given verb and replaces it with its correct tense form. For example "gelcem" (informal way of saying "I will come") is turned into "geleceğim" (I will come) which is the proper form. In addition to verb inflections, incorrect spelled question words are corrected as well. For instance, "gidiyozmu?" is corrected into "gidiyor muyuz?" (are we going?). In this work, we have extended the normalizer with a support for handling separated suffixes issue which is special to our used speech recognizer. We have compiled a Turkish suffix list, which we use to merge separated suffixes with the preceding word. If the word is a proper noun, we merge the noun and its suffix with an apostrophe and then capitalize the proper noun. For instance; "ahmet e" (Ahmet is a proper noun) becomes "Ahmet'e".

4.2 Query Normalization

The Named Entity Recognizer (NER) tool that we use in this work is based on the work of Şeker and Eryiğit, [6]. It utilizes a statistical modeling method, namely Conditional Random Fields (CRFs) [13], which is a framework for building probabilistic models to segment and label sequences of input samples. The original NER tool is designed for general purpose, hence we needed to adapt it for our domain of concern. To this end, we have added 750 tagged queries and commands in to the existing data set. We have also added currency and time gazetteers, and then divided all the gazetteers into two categories: 1) Base Gazetteers: First Name, Last Name, Location, Money, Time and 2) Generator Gazetteers: Person, Location, Organization. Similarly to the original article, to retrain the NER tool, we considered the following features; whole word, word stem, POS tag or word, noun state, proper noun, inflectional groups, lower-upper case, sentence begin, and existence in gazetteers. Besides, the extended NER tool is able to tag PERSON, LOCATION, ORGANIZATION, TIME, MONEY, and PERCENTAGE entities in a given sentence. This tool is crucial for our system, since the output of the NER tool is directly used in the classification and extraction phases presented in the following sections, respectively.

4.3 Classification

In this section, we explain our query classification methodology.

1) ***Rule Based Classification***: In this work, we have developed a rule engine to directly create and modify sets of classification rules. The engine enables to define rules containing regular expressions, and POS and NER tags together. Each rule expression follows the order of a pattern string to be matched against the processed user query, the domain name, and a value to define an order of priority for the evaluation sequence. The following expression is an example of a rule of Call domain.

^(Lütfen)?(<Person>) arayabilir misin)?,CALL,1
(Can you)? (please)? call (<Person>)?, CALL,1

2) ***Statistical Classification***: As it is impractical to define rules that would cover all possible kinds of queries for a domain, we use a statistical classifier for the queries that do not match any rules. In our current application setting, we use a Support Vector Machine (SVM) classifier [14], a decision

Table 1: Performance of classification methods

Algoritma	Doğruluk	Precision	Recall	Fskoru	ROC Alanı	TP oranı	FP oranı
Decision Tree 0.25	85,90%	0,883	0,859	0,861	0,956	0,859	0,025
Decision Tree 0.5	85,70%	0,878	0,857	0,860	0,957	0,857	0,025
Decision Tree Unprunned	85,40%	0,870	0,854	0,858	0,959	0,854	0,025
Logistic Regression	94,10%	0,942	0,941	0,941	0,991	0,941	0,007
NaiveBayes Multinominal	92,60%	0,929	0,926	0,922	0,986	0,926	0,011
NaiveBayes	92,20%	0,924	0,922	0,922	0,982	0,922	0,012
Bayes Net	82,50%	0,841	0,825	0,830	0,959	0,825	0,029
IB1	87,90%	0,883	0,879	0,878	0,970	0,879	0,016
SVM	95,80%	0,959	0,958	0,958	0,989	0,958	0,007

based on our performance evaluation that we present below.

a) Data Model: Our training and test data consists of 1000 queries or commands (100+ queries per domain) collected from real users. For each query, first, we remove the stop words. Then, each remaining word in the query is stemmed, and replaced with a NER tag if it matches any. As a consequence, the same type of entities are represented with the same feature to improve the classification performance. Finally, we tag the resulting query with its corresponding domain type. Below is an example of this process.
Original Query: yarın istanbul da yağmur yağacak mı (will it rain tomorrow in istanbul).
Processed Query: <time> <location> yağmur yağ, Weather (rain <time> <location>, Weather)
To represent the data as numerical values, we use the bag-of-words (bag-of-n-grams) model, in which a query is represented as the multi set of its n-gram (contiguous sequence of n items from the query). This model is widely considered in document classification methods, where the frequency of occurrence of each n-gram is used as a feature for training a classifier.

b) Classification Methods: In this work, we have evaluated the performance of various classification methods using our query data set. These methods include Decision Tree (with 0.25, 0.5 and unpruned as confidence factors), Logistic Regression, NaiveBayes (Multinominal), Bayesian Network, K-Nearest-Neighbor, and Support Vector Machine. Since the scale of our data is small, we have used 10-fold cross validation to compare the performance of the classification methods. We repeated the tests ten times and we report the average results. We only present the results in which we use 1-gram model, which yields better results compared to that of when we use 2-gram and 3-gram models.

Table 1 presents the performance of the classification methods in terms of accuracy, precision, recall, F-Score, ROC area, true and false positive rate metrics. The SVM and the Bayesian Network achieves the best and worst results, respectively. The SVM outperforms the other methods except for the metric of ROC area. We observe that the overall results are very high as by nature our data set comprises relatively short queries and commands. Moreover, we also found out that the performance of the classifiers for the Web Search and Start Application related queries are relatively low. We attribute this to the lack of NER support for application names and to the fact that one can search on web with very broad search terms.

4.4 Parameter Extraction

For each domain, we have devised a complex individual parameter extraction method. Table 3 presents the parameters that we can extract from queries for each domain. The explanation of all extraction methods is beyond the scope of this paper, though, it is worth mentioning that the extraction methods leverage user context, NER and POS tags, the dependency relations, and the existence of pre-defined keyword patterns. As an example, Figure 3 illustrates the parameter extraction method for the Map domain.

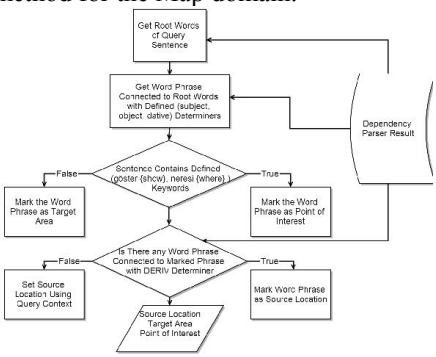


Figure 3: Parameter extraction process of the Map domain.

Table 3:The parameters that can be extracted for each domain

Domain	Parameters
Call	receiver name, phone number
Sms	body text, receiver name, phone number
Email	body text, receiver name, email address
Web Sear	query text, web site
Start Apps	application name
Map	departure, destination, point of interest
Weather	location, time
Exchange	from/to currency pair
Traffic	location

Table 2 shows the accuracy results of our parameter extraction methods that we run on our 1000 queries data set. We define accuracy as the ratio of true parameter extractions to

the total number of parameters. We have attained high accuracy results for our parameter extraction methods, with an average score of 88%, except the methods of Call and Web Search, which yields around 70% accuracy.

1) Mapping Parameters to the Entities in the Phone: The Turkish alphabet, which is a variant of the Latin alphabet, consists of 29 letters, six of which (C-Ç, G-Ğ, I-İ, O-Ö, S-Ş, U-Ü) have two variants for the phonetic requirements of the language. This poses a challenge in matching names in queries to the names on the contact list of a phone, as users tend to save contact names using dot-less characters while speech-to-text service converts person names to their original form (e.g., saving Özgür as Ozgur – a Turkish person name). Moreover, users may mention only the first or last name of a contact or partial name for an application when they ask to call someone or start an application. To solve the exact matching issues mentioned above, in the mobile application we use the Monge-Elkan approximate text string matching algorithm [15], which measures the similarity between two strings each of which may comprise one or more words. In case the highest similarity score is below a certain threshold, we let the user choose among a few results corresponding to the strings with the highest similarity scores.

5 Conclusion and Future Work

In this paper we have presented our ongoing work on developing a mobile assistant application that understands Turkish language. We explained our system architecture and our methodology of understanding user queries, in which we leverage and improve existing research and tools for Turkish NLP. We also presented the performance of query classification and parameter extraction methods that we use in this work.

As future work, we plan to support a dialog-based interaction between users and the application and make the application personalized considering the context and previous queries of the users. We also plan to extend our services for instance with support for factual questions, and extend the performance tests with a broader set of real user queries.

Acknowledgements

This work is supported by the research project SANTEZ (Grant: 0073-STZ.2013-1). The authors want to thank Ö.Ozan Sönmez for helpful discussion.

6 References

[1] H. Sak, M. Saraclar, and T. Gungor, "Morphology-based and sub-word language modeling for turkish speech recognition," in Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on. IEEE, 2010, pp. 5402–5405.

[2] J. Nivre, J. Hall, J. Nilsson, G. Eryig˙it, and S. Marinov, "Labeled pseudo-projective dependency parsing with support vector machines," in Proceedings of the Tenth Conference on Computational Natural Language Learning. Association for Computational Linguistics, 2006, pp. 221–225.

[3] S. Buchholz and E. Marsi, "Conll-x shared task on multilingual de- pendency parsing," in Proceedings of the Tenth Conference on Compu- tational Natural Language Learning. Association for Computational Linguistics, 2006, pp. 149–164.

[4] K. Oflazer and İ. Kuruöz, "Tagging and morphological disambiguation of turkish text," in Proceedings of the fourth conference on Applied natural language processing. Association for Computational Linguistics, 1994, pp. 144–149.

[5] G. A. Şeker and G. Eryiğit, "Initial explorations on using CRFs for Turkish named entity recognition," in Proceedings of COLING 2012, Mumbai, India, 8-15 December 2012.

[6] M. Şahin, U. Sulubacak, and G. Eryiğit, "Redefinition of Turkish morphology using flag diacritics," in Proceedings of The Tenth Symposium on Natural Language Processing (SNLP-2013), Phuket, Thailand, October 2013.

[7] J. C. Hawkins, W. B. Rees, D. O. Chyi, and R. Y. Haitani, Interface for processing of an alternate symbol in a computer device. Google Patents, dec " 13" 2005, uS Patent 6,975,304.

[8] P. Sawhney, P. King, H. W. Ham, and L. Wang, Apparatus and method for mobile personal assistant. Google Patents, aug " 11" 2011, uS Patent App. 13/207,781.

[9] A.NeusteinandJ.A.Markowitz,MobileSpeechandAdvanced Natural Language Solutions. Springer, 2013.

[10] G. Eryiğit, "ITU Turkish NLP web service," in Proceedings of the Demonstrations at EACL 2014 (EACL). Gothenburg, Sweden: Association for Computational Linguistics, April 2014.

[11] J.Nivre,J.Hall,J.Nilsson,A.Chanev,G.Eryiğit,S.Kübler,S.Marinov, and E. Marsi, "Maltparser: A language-independent system for data-driven dependency parsing," Natural Language Engineering Journal, vol. 13, no. 2, pp. 99–135, 2007.

[12] G. Eryiğit, J. Nivre, and K. Oflazer, "Dependency parsing of Turkish," Computational Linguistics, vol. 34, no. 3, pp. 357–389, 2008.

[13] D. Torunoğlu and G. Eryiğit, "A cascaded approach for social media text normalization of Turkish," in 5th Workshop on Language Analysis for Social Media (LASM) at EACL. Gothenburg, Sweden: Association for Computational Linguistics, April 2014.

[14] C. Cortes and V. Vapnik, "Support-vector networks," Mach. Learn., vol. 20, no. 3, pp. 273–297, Sep. 1995. [Online]. Available: http://dx.doi.org/10.1023/A:1022627411411

[15] A. Monge and C. Elkan, "The field matching problem: Algorithms and applications," in In Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, 1996, pp. 267–270

SEMANTIC ANNOTATION OF TATAR VERBS FOR LINGUISTIC APPLICATIONS

Alfiia Galieva
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, Kazan Federal University
Kazan, Russia
amgalieva@gmail.com

Ayrat Gatiatullin
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, Kazan Federal University
Kazan, Russia
agat1972@mail.ru

Olga Nevzorova
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, Kazan Federal University
Kazan, Russia
onevzoro@gmail.com;

Dilyara Yakubova
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences
Kazan Federal University
Kazan, Russia
suleymanovad@gmail.com

ABSTRACT

The paper discusses the problem of meta-language for linguistic applications and proposes a tag set for semantic annotation of verbs for Tatar National Corpus. The approach is based on data from explanatory dictionaries of Tatar and Russian languages, bilingual Russian-Tatar dictionaries and Russian National Corpus.

1 Introduction

The development of meta-language for semantic annotation for linguistic applications and corpora is one of actual problems in applied linguistics. Since there is no common semantic theory, semantic tags given to words and word combinations denote different semantic classes which specify the word meanings. Usually, morphological annotation, which gives basic lexical and grammatical classes of words, is used as a foundation of semantic annotation for vocabulary.

Grammatical annotation uses a fixed set of grammatical classes. The number of semantic attributes depends on generalization level: the more abstract attributes are, the less their number is; the more concrete semantic attributes are, the larger their number is. There are some problems in semantic describing of vocabulary such as the absence of distinct boundary between taxons, the necessity to process very large sets of attributes and semantic features, the complexity of delineation of semantic components in lexical unit and the inability to unambiguously define their features. As a general rule, separate semes do not have special formal indices, so it is difficult, if not impossible, to locate them in the language and to describe them in comprehensive and consistent way. The matching of meanings is a complicated problem, as the question of whether vocabulary, beyond the boundaries of separate groups, such as relationship terms, is systemic or not still open [1], and the meanings of words are very individualized.

Semantic annotation scheme assumes the existence of set of tags, their meanings and rules for application of tags to units of text or vocabulary. At present, there are no standards for semantic annotation creation and there is no semantic annotation in most of the developed corpora of Turkic languages (Tatar [2], Crimean Tatar [3], Turkish [4], Kazakh [5], Bashkir [6], Tuvinian [7], Yakut [8]).

Quantitative and qualitative features of sets of tags, used in thesauri, electronic corpora and lexicographic databases, are varying. It is

obvious, that the larger set of tags is, the more comprehensive analysis of linguistic material can be performed. On the other side, there are some advantages in simple encodings – they are more error-prone, more consistent during the process of annotation and require less handwork. So, it is important to work out the system with balance between available level of detail and simplicity for developers and-users.

Explanatory dictionaries of the Tatar language, bilingual Russian-Tatar dictionaries, thesauri of Russian language and data from Russian National Corpus were used during the development of classification.

There were no integral description of the Tatar language lexical system as a complex hierarchical network of units of different layers and types, so at present there are no ideographic dictionaries of the Tatar language. Therefore, not only general principles of representation of Tatar verbs in corpus annotations have to be developed, but the Tatar vocabulary has to be classified or real content of lexical-semantic groups has to be extracted from raw alphabetical word-list.

The problem of ideographic classification creation (the extraction of thesaurus from semantically unordered alphabetical word-list) comes as applied one, but the process of its solution leads to necessity of general theoretical analysis for systems in the vocabulary, to questions of language nomination and to necessity of revisiting some aspects of field theory of linguistics and vocabulary's structural features and properties [9].

Thesaurus is a specific object which allows an ability to research the systemic properties of language, various relational features, different significative and logical-semantic relationships and relationships of given lexeme to others.

2 Features of semantic annotation of Tatar verbs

The following basic principles of arrangement of lexical data (these principles are used in creation of ideographic dictionaries) were used during the development of semantic annotation of verbs in the Tatar language: system principle, hierarchy principle, variability principle, overlapping of word classes principle.

The ability to consider the overlapping of word classes, when lexeme is described by different independent tags (examples of such lexemes are given in Table 1), is an important advantage of corpora annotation, which is difficult to implement in printed ideographic dictionaries.

Table 1. A fragment of semantic annotation of lexemes

Tatar	English	Tags
aldau	*deceive*	t:speech, t:behav
zarlanu	*resent*	t:speech, t:psych:emot
yavu	*fall (on atmospheric precipitates)*	t:move, t:nat
gırlau	*snore*	t:sond, t:phys

The development of classification is connected to extraction of different lexical-semantic groups (LSG) of verbs, e.g. in well-known research of Levin [10], 57 basic semantic classes of verbs for English language are distinguished. These evaluations are considerably lower for Turkish languages – there are, at average, 10 lexical-semantic classes of verbs.
F. Ganeev [11] distributes verbs of the Tatar language into following 11 LSG:

1. Movement verbs;
2. Action verbs;
3. Process verbs;

4. State verbs;
5. Relationship verbs;
6. Behavior verbs;
7. Sound verbs;
8. Speech verbs;
9. Thought process verbs;
10. Perception verbs;
11. Imitative verbs.

M. Orazov [12] distinguishes the following LSG for Kazakh language:
1. Action verbs;
2. Movement verbs;
3. Relationship verbs;
4. Subjective evaluation verbs;
5. Nature related verbs;
6. Emotional verbs;
7. Sense-describing verbs;
8. Verbs with meaning of creation and appearance;
9. Thought process verbs;
10. Speech verbs;
11. Sound verbs;
12. State verbs.

The semantic annotation solutions from the Russian National Corpus (RNC) were used during the development of semantic annotation for the Tatar Corpus with s for lexical and word-derivational systems of the Tatar language. For example, the following tags were adopted from RNC:

t:move — movement (Example: *cabu (Tat) 'to run*');

t:move:body — change of position of body or a body part (Example: *utıru (Tat)* 'to sit,to sit down');

t:put — object placement (for example: *töyäw (Tat)* 'to load up', *quyu (Tat)* 'to put smth. on/in');
t:impact — physical influence (for example: *sugu (Tat) '*to hit');

t:impact:creat — object creation (for example: *tözü (Tat)* 'to build')

t:impact:destr — object destruction (for example: *yandıru (Tat)* 'to burn smth. down').

3 Basic and additional tags

The system principle assumes the reuse of the same tags for different grammar classes with common meanings. There are some differences in semantic annotation of different parts of speech with common meanings in RNC. For example, during the semantic annotation of nouns, tag t:temper – temperature (Example: cold, chill, heating) is used, but the same tag is not used in semantic annotation of verbs. The verb 'to heat' is only annotated with t:changest [13] – only change of feature is specified.

It is assumed, that in many cases t:chagest tag (state or feature change) can be further clarified with parameter describing the change (if corresponding tags describing LSG, which may or may not belong to the same part of speech, exist), as in:

T:changest:size – change of size (for example: *zurayu (Tat)* 'to grow');

T:changest:form – change of shape (for example: *yäncü (Tat)* 'to flatten', *tügäräkläw (Tat)* 'to make round');

T:changest:color – change of color (for example: *sargayu (Tat)* 'to become yellow');

T:changest:humq – change of human's mood (for example: *yavızlanu (Tat)* 'to become exasperated').

Any tag, which used as main tag describing nouns or adjectives, can be used as clarifying in course of the verb annotation.

It is assumed, that common designation, if possible, should be used when annotating part of speech in lexicographic base of the Tatar

language. In Turkic languages there is a special grammar category between nouns and verbs – verbal nouns. The verbal noun describes an action (state or process) in most generalized form (without respect to mood and tense) and has certain grammar features of the verb (aspect, voice, raritivity forms) and the noun (case, plurality, possession) [14]. As such, there are little formal differences between nouns and verbs, and it is a reason for supporting the maximum possible commonality in semantic feature systems for nouns and verbs. For example, in modern grammar dictionaries of the Tatar language many verbs ending in *–u* are tagged as noun and verb at the same time.

Another feature of the Tatar language is a presence of many verbs describing physical influence, for their description tags, missing in RLC, are used, for example:

T:impact:tool – instrumental influence (for example: *boraulau (Tat)* 'drilling', *pıcaqlau (Tat)* 'to cut with knife', *ütükläw (Tat)* 'to iron').

Possessive domain (t:poss) in the Tatar language is clarified using tags describing possession relationship, e.g.:

T:poss:acquire – acquiring (for example: *tabu (Tat)* 'to find', *qorallanu (Tat)* 'to arm with smth.');

T:poss:deprive – depriving (for example: *yugaltu (Tat)* 'to loss smth.', *qoralsızlandıru (Tat)* 'to disarm').
There is a relationship domain in the Tatar language (t:relat) with following types of relations:

T:relat:interp – interpersonal relations (for example: *hörmätläw (Tat)* 'to respect');

T:relat:social –social relations (for example: *čiŋüü (Tat)* 'to win', *yaqlau (Tat)* 'to defend').

The following tags, which used to describe semantic in different part of speech, can be used for clarifying the semantics of derived verbs:

T:poss:acquire, pt:part & pc:plant (for example: *botaqlanu (Tat)* 'to branch'), - here pt:part & pc:plant are related to parts of plants (for example: *yafraq (Tat)* 'leaf', *sabaq (Tat)* 'stem).

In the Tatar language the special tags are used for phase and auxiliary verbs:
Aux: phase – phase verbs (for example: *baślaw (Tat)* 'to begin');

Aux – auxiliary verbs (for example: *itü (Tat)* ' to do, to make').

Table 2. Example of semantic annotation of Tatar verbs

	Causation	**Taxonomy**
Sabaqlanu (Tat) 'to make stems'	ca:noncaus	t:poss:acquire, pt:part & pc:plant
Qaraltu (Tat) 'to darken'	ca:caus	t:changest:color
Käbäkhätlänü (Tat) 'to become sneaky'	ca:noncaus	t:changest: humq

4 Conclusion

The proposed system of semantic annotation of verbs can be used for various linguistic applications for the Tatar language. The work for development of semantic annotation tag system for the Tatar National Corpus is in progress. Currently 170 semantic tags are described, the resulting tag set is used in linguistic databases developed at Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, for example for

annotating of multilingual lexicographic databases.

The work is supported by the Russian Foundation for Humanities and the Government of the Republic of Tatarstan, (project # 14-14-16031 a(r)/2014).

5 References

[1] **RAHILINA, E. V., PLUNGAN, V.A.** On lexical-semantic typology // Verbs describing movement in water: Lexical typology / Edited by. T. A. Mysac, E. V. Rahilina. — M.: Indirk, 2007. - pp. 11-26. In Russian.

[2] Tatar Corpus. [Electronic resource]. URL: http://web-corpora.net/TatarCorpus/search/?interface_language=ru, August 2014.

[3] Crimean Tatar Corpus. [Electronic resource]. URL: http://korpus.juls.savba.sk/QIRIM/#id9, August 2014.

[4] Turkish Corpus. [Electronic resource]. URL: www.tnc.org.tr/index.php/en/, August 2014.

[5] Kazakh Corpus. [Electronic resource]. URL: http://kazcorpus.kz/klcweb/, August 2014.

[6] Bashkir Corpus. [Electronic resource]. URL: http://mfbl.ru/bashkorp/korpus, August 2014.

[7] Tuvinian Corpus. [Electronic resource]. URL: http://www.tuvancorpus.ru/, August 2014.

[8] Yakut Corpus. [Electronic resource]. URL: http://adictsakha.nsu.ru/corpora/corp/, August 2014.

[9] **KARAULOV, Y.N.** General and Russian ideography / U. N. Karaulov.—Moscow: Science, 1976.—355 p. In Russian.

[10] **LEVIN, B.** English Verb Classes and Alternations: a Preliminary Investigation. Chicago: University of Chicago Press, 1993.

[11] **GANEEV, F.A.** *Semantic classes for verbs in Tatar language*. - Kazan: IALI, 1984. pp.75-84. In Russian.

[12] **ORAZOV, M.** Kazakh verb semantics (an experience in semantic classification). Alma-Ata, 1983, 56p. In Russian.

[13] Russian Language Corpus. Semantics // http://www.ruscorpora.ru/corpora-sem.html, August 2014.

[14] Tatar grammar in 3 vol.. Kazan: TBP, 1993. Vol. 2. Morphology. — 398 p. In Tatar.

Using Morphosemantic Information in Construction of a Pilot Lexical Semantic Resource for Turkish

Gözde Gül Şahin *Eşref Adalı*
Department of Computer Engineering
Istanbul Technical University
Istanbul, 34469, Turkey
isguderg@itu.edu.tr; adali@itu.edu.tr;

ABSTRACT

Morphological units carry vast amount of semantic information for languages with rich inflectional and derivational morphology. In this paper we show how morphosemantic information available for morphologically rich languages can be used to reduce manual effort in creating semantic resources like PropBank and VerbNet; to increase performance of word sense disambiguation, semantic role labeling and related tasks. We test the consistency of these features in a pilot study for Turkish and show that; 1) Case markers are related with semantic roles and 2)Morphemes that change the valency of the verb follow a predictable pattern.

1 Introduction

In recent years considerable amount of research has been performed on extracting semantic information from sentences. Revealing such information is usually achieved by identifying the complements (arguments) of a predicate and assigning meaningful labels to them. Each label represents the argument's relation to its predicate and is referred to as a semantic role and this task is named as semantic role labeling (SRL). There exists some comprehensive semantically interpreted corpora such as FrameNet and PropBank. These corpora, annotated with semantic roles, help researchers to specify SRL as a task, furthermore are used as training and test data for supervised machine learning methods [1]. These resources differ in type of semantic roles they use and type of additional information they provide.

FrameNet (FN) is a semantic network, built around the theory of semantic frames. This theory describes a type of event, relation, or entity with their participants which are called frame elements (FEs). All predicates in same semantic frame share one set of FEs. A sample sentence annotated with FrameNet, VerbNet and PropBank conventions respectively, is given in Ex.1. The predicate ”buy” belongs to ”Commerce buy”, more generally ”Commercial transaction” frame of FrameNet which contains ”Buyer”, ”Goods” as core frame elements and ”Seller” as a non-core frame element as in Ex. 1. FN also provides connections between semantic frames like inheritance, hierarchy and causativity. For example the frame ”Commerce buy” is connected to ”Importing” and ”Shopping” frames with ”used by” relation. Contrary to FN, VerbNet (VN) is a hierarchical verb lexicon, that contains categories of verbs based on Levin Verb classification.[2].

The predicate ”buy” is contained in ”get-13.5.1” class of VN, among with the verbs ”pick”, ”reserve” and ”book”. Members of same verb class share same set of semantic

roles, referred to as thematic roles. In addition to thematic roles, verb classes are defined with different possible syntaxes for each class. One possible syntax for the class ”get-13.5.1” is given in the second line of Ex. 1. Unlike FrameNet and VerbNet, PropBank (PB) [3] does not make use of a reference ontology like semantic frames or verb classes. Instead semantic roles are numbered from Arg0 to Arg5 for the core arguments.

[Jess]$_{\text{Buyer-Agent-Arg0}}$ bought [a coat]$_{\text{Goods-Theme-Arg1}}$ from [Abby]$_{\text{Seller-Source-Arg2}}$
Syntax: Agent V Theme {From} Source
Ex. 1

There doesn’t exist a VerbNet, PropBank or a similiar semantically interpretable resource for Turkish (except for WordNet [4]. Also, the only available morphologically and syntactically annotated treebank corpus: METU-Sabanci Dependency Treebank [5,6,7] has only about 5600 sentences, which has presumably a low coverage of Turkish verbs. VerbNet defines possible syntaxes for each class of verbs. However, due to free word order and excessive case marking system, syntactic information is already encoded with case markers
in Turkish. Thus the structure of VerbNet does not fit well to the Turkish language. PropBank simplifies semantic roles, but defines neither relations between verbs nor all possible syntaxes for each verb. Moreover only Arg0 and Arg1 are associated with a specific semantic content, which reduces the consistency among labeled arguments. Due to lack of a large-scale treebank corpus, building a high coverage PropBank is currently not possible for Turkish. FrameNet defines richer relations between verbs, but the frame elements are extremely fine-grained and building such a comprehensive resource requires a great amount of manual work for which human resources are not currently available for Turkish.

In this paper, we discuss how the semantic information supplied by morphemes, named as morphosemantics, can be included in the construction of semantic resources for languages with less resources and rich morphologies, like Turkish. We try to show that we can decrease manual effort for building such banks and increase consistency and connectivity of the resource by exploiting derivational morphology of verbs; eliminate mapping costs by associating syntactic information with semantic roles and increase the performance of SRL and word sense disambiguation by directly using morphosemantic information supplied with inflectional morphology. Then, we perform a pilot study to build a lexical semantic resource that contains syntactic information as well as semantic information that is defined by semantic roles both in VerbNet and PropBank fashion, by exploiting morphological properties of Turkish language.

2 Morphosemantic Features

In morphologically rich languages, the meaning of a word is strongly determined by the morphemes that are attached to it. Some of these morphemes always add a predefined meaning while some differ, depending on the language. However, only regular features can be used for NLP tasks that require automatic semantic interpretation. Here, we determine two multilingual morphosemantic features: case markers and verb valency changing morphemes and analyze the regularity and usability of these features for Turkish.

2.1 Declension and Case Marking

Declension is a term used to express the inflection of nouns, pronouns, adjectives and articles for gender, number and case. It occurs in many languages such as Arabic, Basque,

Sanskrit, Finnish, Hungarian, Latin, Russian and Turkish. Even though the languages differ, the same case markers are used to express similiar meanings with some variation. Relation between semantic roles and case markers can assist researchers in solving some of the challenging problems in natural language processing. In languages where case markers exist, these

- can be used as features for Semantic Role Labeling,
- can supply priori information for disambiguating word senses,
- can be used in language generation as such: Once the predicate and the sense is determined, the
- arguments can directly be inflected with the case markers associated with their roles.

2.2 Valency Changing Morphemes

The valency of a verb can be defined as the verb's ability to govern a particular number of arguments of a particular type. "In Turkish, verb stems govern relatively stable valency patterns or prototypical argument frames" as stated by [8]. Consider the root verb giy (to wear). One can derive new verbs from the root giy (to wear) such as giy-in (to get dressed), giy-dir (to dress someone) and giy-il (to be worn). These verbs are referred to as verb stems and these special suffixes are referred to as valency changing morphemes. By modeling the semantic role transformation from verb root to verb stem, we can automatically identify argument configuration of a new verb stem given the correct morphological analysis. By doing so, framing only the verb roots can guarantee to have frames of all verb stems derived from that root. This quickens the process of building a semantic resource, as well as automatizing and reducing the human error. In this section we present a pilot study for some available valencies in Turkish language. For the sake of simplicity, instead of thematic roles, argument labeling in the PropBank fashion is used.

Reflexive

The reflexive suffix triggers the suppression of one of the arguments. In Fig. 1, observed argument shift is given.

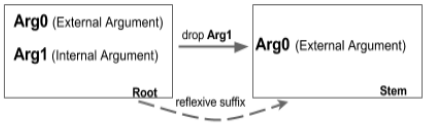


Figure 1: Argument transformation caused by reflexive suffix.

Reciprocal

Reciprocal verbs express actions done by more than one subject. The action may be done together or against each other. Reciprocal verbs may have a plural agent or two or more singular co-agents conjoined where one of them marked with COM case as shown in Fig 2. In both cases, the suppression of one of the arguments of the root verb is triggered. We have observed that the supressed argument may be in different roles (patient, theme, stimulus, experiencer, co-patient), but usually appears as Arg1 and rarely as Arg2.

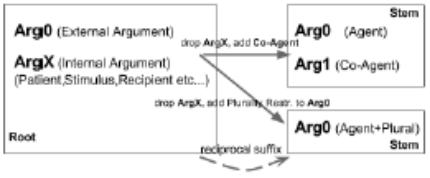


Figure 2: Argument transformation caused by reciprocal suffix.

Causative

Causative category is the most common valence-changing category among Bybee's [9]

world-wide sample of 50 languages. Contrary to other morphemes, causative morpheme introduces of a new argument called causer to the valence pattern. In most of the languages, only intranstive verbs are causitivized. In this case, as shown in Fig. 3 the causee becomes the patient of the causation event. In other words, the central argument of the root verb, (Arg0 if exists, otherwise Arg1), is marked with ACC case and becomes an internal argument (usually Arg1) of the new causative verb. Some languages can have causatives from transitive verbs too, however the role and the mark of the causee may differ across languages. For the languages where the causee becomes an indirect object, like Turkish and Georgian, the central argument, Arg0 of the root verb, when transformed into a verb stem, receives the DAT case marker and serves as an indirect object (usually as Arg2), while Arg1 serves again as Arg1. This pattern for transitive verbs is given in Fig. 3.

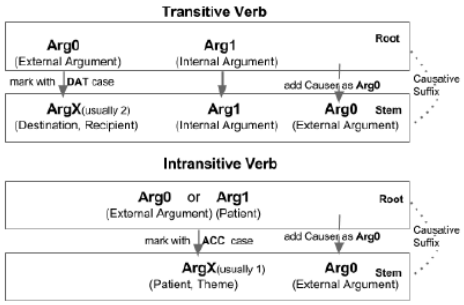


Figure 3: Argument transformation caused by causative suffix.

3 Methodology

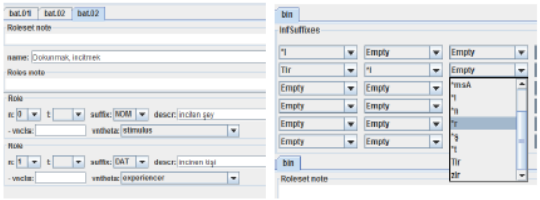
We have performed a feasibility study for using morphosemantic features in building a lexical semantic resource for Turkish. As discussed in Section 3.2, we assume we can automatically frame a verb (e.g sakla-n(reflexive)) that is derived with a regular valency changing morpheme (e.g. n), if the argument configuration of the root verb (e.g. sakla) is known. Hence, we have only framed root verbs. We have framed 233 root verbs and 452 verb senses. We have calculated the total number of valence changing morphemes as 425. This means 425 verbs can be automatically framed by applying the valency patterns to 233 root verbs. In this analysis we have only considered one sense of the verb since there may be cases where valency changing morpheme can not be applied to another sense of the verb. This can not be automatically determined. Moreover, a verb stem may have multiple senses. In that case automatically extracted argument transformation may be wrong, because the verb stem may have a completely different meaning.

Turkish is not among rich languages by means of computational resources as discussed before. Turkish Language Association (TDK) is a trustworthy source for lexical datasets and dictionaries. To run this pilot study, we have used the list of Turkish root verbs provided by TDK and the TNC corpus4. The interface built for searching the TNC corpus gives the possibility to see all sentences that were built with the verb the user is searching for [10]. The senses of the verbs and case marking of their arguments are decided by manually investigating the sentences appear in search results of the TNC corpus. Then, the arguments of the predicates are labeled with VerbNet thematic roles and PropBank argument numbers, by checking the English equivalent of Turkish verb sense. This process is repeated for all verb senses.

For framing purposes, we have adjusted an already available open source software, cornerstone [11]. To supply case marking information of the argument, a drop down menu containing six possible case markers in Turkish is added as shown in Fig 4a. Finally, another drop down menu that contains all possible suffixes that a Turkish verb can have is added, shown in Fig 4b. Theoretically, the

number of possible derivations may be infinite for some Turkish verbs, due to its rich generative property.

However, practically the average number of inflectional groups in a word is less than two. TDK provides a lexicon for widely used verb stems derived from root verbs by a valency changing morpheme. To avoid framing a nonexisting verb, we have used a simple interface shown in Fig 4b to enter only the stems given by TDK. An example with the Turkish verb ”bin” (to ride) is given in Fig 4b. The first line defines that one can generate a stem ”bin-il” (to be ridden by someone) from the root ”bin” by using the suffix ”l”. Similiarly, second line illustrates a two layer derivational morphology, which can be interpreted as producing two verbs: ”bin-dir” (cause someone to ride something) and ”bindir-il” (to be caused by someone to ride something).



(a) Case marker info given in suffix list (b) Verb derivational info as a drop down menu

Figure 4: Cornerstone Software Adjusted for Turkish

4 Experiments and Results

In Table 1, number of co-occurences of each thematic role with each case marker are given. Since in PropBank only Arg0 and Arg1 have a certain semantic interpretation, we have used VerbNet thematic roles in our analysis. Some roles look highly related with a case marker, while some look arbitrary. Results can be interpreted in two ways: 1) If the semantic roles are known and case marker information is needed, Agent will be marked with NOM, Destination with DAT, Source with ABL and Recipient with DAT case with more than 0.98 probability, furthermore Patient and Theme can be restricted to NOM or ACC cases; 2) If case markers are known and semantic role information is needed, only restrictions and prior probabilities can be provided. Highest probabilities occur with COM-instrument, LOC-location, DAT-destination, ACC-Theme and NOM-Agent pairs. We have applied our proposed argument transformation on verbs with different valencies, and compared the argument configurations of the roots and stems.

	NOM	ACC	DAT	LOC	ABL	COM	Total	Explanation
Agent	318	0	1	0	0	0	319	Human or an animate subject that controls or initiates the action.
Patient	36	34	0	0	0	0	70	Participants that undergoe a state of change.
Theme	101	117	14	0	7	1	240	Participants in a location or experience a change of location
Beneficiary	1	2	5	0	0	0	8	Entity that benefits negatively or positively from the action.
Location	0	0	2	6	0	0	8	Place or path
Destination	1	0	66	0	0	0	67	End point or direction towards which the motion is directed.
Source	0	0	0	0	29	0	29	Start point of the motion.
Experiencer	13	5	4	0	0	0	22	Usually used for subjects of verbs of perception or psychology.
Stimulus	8	2	4	0	2	0	16	Objects that cause some response from Experiencer.
Instrument	0	0	0	0	0	10	10	Objects that come in contact with an object and cause a change.
Recipient	0	1	13	0	0	0	14	Animate or organization target of transfer.
Time	1	0	2	2	0	0	5	Time.
Topic	0	1	3	0	2	0	6	Theme of communication verbs.
Total	479	162	114	8	40	11	814	

Table 1. Results of Semantic Roles – Case Marking

	#Intransitive	#Transitive	#Hold	#!Hold	Total
Reflexive	0	20	20	0	20
Reciprocal	8	18	26	0	26
Causative	26	11	37	0	37

Table 2. Results of argument transformation

In Table 2, rows represent the valency changes applied to verb root, where Intransitive column contains the number of intransitive verbs that the pattern is applied to, and Transitive similiarly. The #Hold column shows the number of root verbs for which the proposed patterns hold, and #!Hold shows the number of times the pattern can not be observed. Reflexive pattern can only be applied to transitive verbs, while others can be applied to both. Experiments are done for reflexive, reciprocal and causative forms. Our preliminary results on a small set of root verbs show that proposed argument transformation can be seen as a regular transformation.

6 Acknowledgements

We thank Gülşen Eryiğit for insightful comments and suggestions that helped us improve this work.

7 References

[1] **Ana-Maria Giuglea and Alessandro Moschitti.** 2006. Semantic Role Labeling via FrameNet, VerbNet and PropBank. Proceedings of the 21st International Conference on Computational Linguistics, pp. 929-936. 2006.

[2] **K. Schuler** 2006. VerbNet: A Broad-Coverage, Comprehensive Verb Lexicon PhD diss., University of Pennsylvania

[3] **Martha Palmer, P Kingsbury, and D Gildea**. 2005. The Proposition Bank: An Annotated Corpus of Semantic Roles. Computational Linguistics, 31(1):71–106

[4] **Orhan Bilgin, Ozlem Çetinoğlu and Kemal Oflazer.** 2004. Building a wordnet for Turkish. Romanian Journal of Information Science and Technology, 7.1-2 (2004): 163-172.

[5] **Gülşen Eryiğit, Tugay Ilbay, Ozan A. Can. 2011.** Multiword Expressions in Statistical Dependency Parsing. In Proceedings of the Workshop on Statistical Parsing of Morphologically-Rich Languages SPRML at IWPT, Dublin.

[6] **Kemal Oflazer, Bilge Say, Dilek Z. Hakkani-Tür, Gökhan Tür**. 2003. Building a Turkish Treebank. Invited chapter in Building and Exploiting Syntactically annotated Corpora, Anne Abeille Editor, Kluwer Academic Publishers

[7] **Nart B. Atalay, Kemal Oflazer, Bilge Say. 2003**. The Annotation Process in the Turkish Treebank. In Proceedings of the EACL Workshop on Linguistically Interpreted Corpora - LINC, Budapest, Hungary

[8] **Geoffrey Haig.** 1998. Relative Constructions in Turkish. Otto Harrassowitz Verlag., ISBN 3447040041, (1998)

[9] **Joan L. Bybee**. 1985. Morphology: A Study of the Relation between Meaning and Form. Typological Studies in Language 9 Amsterdam, Philadelphia: Benjami

[10] **Yeşim Aksan, Mustafa Aksan** 2012. Construction of the Turkish National Corpus (TNC). (LREC 2012). Istanbul.

[11] **Jinho D. Choi, Claire Bonial, and Martha Palmer.** 2010. Propbank Frameset Annotation Guidelines Using a Dedicated Editor, Cornerstone. (LREC 10), Valletta, Malta

FORMAL MODEL OF ADJECTIVE IN THE KAZAKH LANGUAGE

A.Mukanova *B.Yergesh*
asel_ms@bk.ru *b.yergesh@gmail.com*

A. Sharipbay *G.Bekmanova*
sharalt@mail.ru *gulmira-r@yandex.kz*

L.N Gumilyov Eurasian National University

ABSTRACT

This paper explains how semantic hypergraphs are used to construct ontological models of morphological rules in the Kazakh language. The nodes within these graphs represent semantic features (morphological concepts) and the edges within represent the relationships between these features. Word forms within the hypergraph structure are described in trees which are converted into linear parenthesis notation; the trees and the linear parenthesis notations correspond to each other. Linear parenthesis notations are the formal models of morphological rules and the software implementation of the linear parenthesis notation allows for the automation of the synthesis of the various morphological word form analyses of the Kazakh language.

Introduction

Agglutinative languages (lat. Agglutinatio — combine, stick) are languages that have a system in which the dominant type of inflection is the agglutination ("sticking") of different formants; these can be either a prefix or a suffix and have only one meaning [1].

The Kazakh language is part of the Turkic group of languages; this language group can be classified as an agglutinative language. Words in the Kazakh language contain many word inflections; inflections are formed by adding suffixes and endings to words. Suffixes and endings are attached in a strict sequence and words in the Kazakh language vary in number, case, and person. A possessive form in Kazakh exists as it does in the English language [2-3].

Currently, ontology is a powerful and widely used tool which is used to model the relationship between objects of different subject fields. It is acceptable to classify ontology based on the degree of dependence on the task or application area, the model of ontological knowledge representation and expressiveness as well as other parameters [4]. Applied ontologies describe concepts which depend on both the task and the subject field of ontology.

Applied ontology is based on the general principles of ontology building, using semantic hypergraphs as a model for the representation of knowledge. This formalism will determine ontology O as triplet (V, R, K) where V is a set of concepts of the subject field (hypergraph nodes), R is a set of relationships between these concepts (hypergraph and edges), and K is a set of the names of concepts and relationships in the given subject field.

The semantic hypergraph language is a formal means of the representation of knowledge in which it is possible to implement classifying, functional, situational, and structural networks and scenarios, depending on the relationship types. This language is an extension of semantic networks where N-ary relations are represented naturally; these relations not only

allow for the specification of the objects' attributes but also permit a representation of their structural, "holistic" descriptions [5].

There are some papers on the use of semantic hypergraph [6-7]. Zhen L, Jiang Z. [7] describes the semantic hypergraph model as a 'hyper-graph based semantic network' (Hy-SN), which can represent more complex semantic relationships and which have a more efficient data structure for storing knowledge in repositories.

In [8-9] the hypergraph H (V, E) is defined by the pair (V, E), where V is the set of vertices $V = \{v_i\}$, $i \in I = \{1, 2, \ldots, n\}$, and E is set of edges $E = \{e_j\}$, $j \in J = \{1, 2, \ldots, m\}$; each edge is a subset of V. Vertex v and edge e is described as an incident if v $\in$ e. For v $\in$ V by d(v) denotes the number of edges incident to a vertex v; d(v) is called the degree of a vertex v. Degree of edge e, the number of vertices incident to this edge, is denoted by r(e).

Use of the ontology model for the representation of morphological rules allows for the translation of the morphological model on an almost one to one basis within the object-oriented data model. Where classes are the part of speech of the Kazakh language and the objects refer to their semantic categories, for example, qualitative type, relative type and degrees of comparison of adjectives.

Use of the ontology model for the representation of morphological rules part of speech allows describing complete morphological model with their relationships. Use semantic hyper graph for the representation of morphological rules part of speech and structure (frame) for the representation the concept. This representation allows translating to the object-oriented data model, where semantic hypergraph vertices are classes.

The purpose of this research is the automated generation of word forms and new words in the Kazakh language as well as the morphological analysis of the Kazakh language.

The research problem consists of the difficulties of formalizing of any natural language.

The authors believe that the problem of formalization of the Kazakh language is handled well through the proposed model below.

In this paper we describe an adjective in Material and Method section. The formal model of a noun is described in [10].

2 Material and Method

The semantic features of the initial forms of adjective (Adj) are qualitative (qual), relative (rel) and comparison (comparison); the sign determines the trajectory of the inflection of the adjective. Adjective in the Kazakh language conjugate (pers_end) and variesy for case (cases), as well as numbers (number) and have a possessive form (poss_end), comparison degree.

We used the ontology editor Protege [11] to build an ontology. It is a free and open source ontology editor and framework for building knowledge bases and is being developed at Stanford University in collaboration with the University of Manchester. Figure 1 shows the ontological model of adjective with its semantic features.

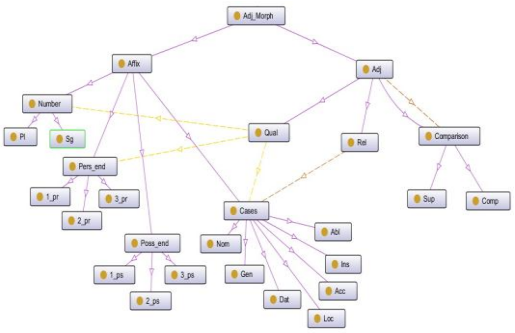


Figure1. Ontological model of adjective

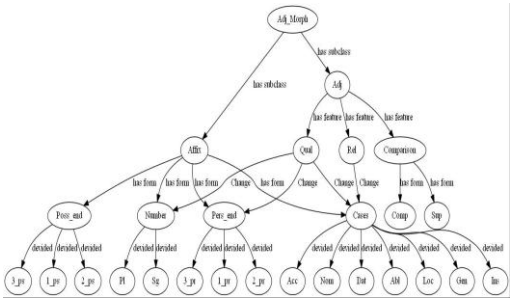


Figure2 . Visualization of adjective as a graph

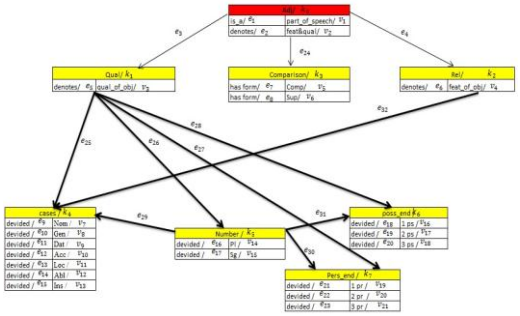


Figure3 . Graphical representation of ontology using semantic hypergraph

Table 1 describes the concepts and relationships used in the ontology

ID	Notation	Description
k_0	Adj	Adjective
v_1	Part_of_ speech	Part_of_ speech
v_2	Feat&qual	Quality and feature
k_1	Qual	Qualitative
v_3	Qual_of_obj	Quality of object
k_2	Rel	Relative
v_4	Feat_of_obj	Feature of object
k_3	Comparison	Comparison
v_5	Comp	Comparative
v_6	Sup	Superlative
k_4	Cases	Cases
v_7	Nom	Nominative case
v_8	Gen	Genitive case
v_9	Dat	Direction-dative case
v_{10}	Acc	Accusative case
v_{11}	Loc	Locative case
v_{12}	Abl	Ablative case
v_{13}	Ins	Instrumental case
k_5	Number	Number
v_{14}	Pl	Plural
v_{15}	Sg	Singular
k_6	Poss_end	Possessive endings
v_{16}	1 ps	1 personal
v_{17}	2 ps	2 personal
v_{18}	3 ps	3 personal
k_7	Pers_end	Personal endings
v_{19}	1 pr	1 personal
v_{20}	2 pr	2 personal
v_{21}	3 pr	3 personal
e_1	is_a	is a
e_2	denotes	Denotes
e_3, e_4, e_{24}	has_feature	has feature
e_5, e_6	has	Has
e_7, e_8	has form	has form
$e_9 - e_{23}$	devided	Devided
$e_{27} - e_{28}$ e_{32}	change	Change
e_{27}, e_{30}, e_{31}	add	Add

Hyper-arcs will be called as semantic arcs for separating semantic hypergraphs from other

types of graphs; it will also be assumed that the set of vertices of the semantic hypergraph includes set of classes $K = \{k_a\}$, where $a \in A = \{0, 1, 2, 3..., n\}$ each of which will consist of set of instances of the class [12]. Thus, vertex-class can be represented by triple:

$$k_a = \{V_a, E_a, S_a\},$$

where V_a - set of class properties, E_a - set of semantic arcs incident to class, S_a - set of instance of class.

The adjective vertex-classes:

$$k_0 = \{\{v_1, v_2\}, \{e_1, e_2\}, S_0\}$$

$$k_1 = \{\{v_3\}, \{e_5\}, S_1\}$$

$$k_2 = \{\{v_4\}, \{e_6\}, S_{2`}\}$$

$$k_3 = \{\{v_5, v_6\}, \{e_7, e_8\}, S_{3`}\}$$

$$k_4 = \{\{v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}\},$$

$$\{e_9, e_{10}, e_{11}, e_{12}, e_{13}, e_{14}, e_{15}\}, S_4\}$$

$$k_5 = \{\{v_{14}, v_{15}\}, \{e_{16}, e_{17}\}, S_{5`}\}$$

$$k_6 = \{\{v_{16}, v_{17}, v_{18}\}, \{e_{18}, e_{19}, e_{20}\}, S_6\}$$

$$k_7 = \{\{v_{19}, v_{20}, v_{21}\}, \{e_{21}, e_{22}, e_{23}\}, S_7\}$$

We can represent the adjective's morphological model with the semantic hypergraph model:

Hypergraph H(V,E), where

$$V = K = \{k_a\}, E = \{e_a\}$$

$$V = \{k_0, k_1, k_2, k_3, k_4, k_5, k_6, k_7\}$$

$$E = \{e_3 = \{k_0, k_1\}, e_4 = \{k_0, k_2\},$$

$$e_{24} = \{k_0, k_3\}, e_{25} = \{k_1, k_4\},$$

$$e_{26} = \{k_1, k_5\}, e_{27} = \{k_1, k_6\},$$

$$e_{28} = \{k_1, k_7\},$$

$$e_{29} = \{k_5, k_4\}, e_{30} = \{k_5, k_7\},$$

$$e_{31} = \{k_5, k_6\}, e_{32} = \{k_2, k_4\}\}$$

3 Results and Discussion

We have the base of initial forms containing 40,000 words with semantic features. Here 5,000 words are adjective. From the above described semantic hypergraph we can obtain formal rules using the parenthesis notation. The number of formal rules for adjective are 2,000. Through the use of these formal rules 65,000 word forms of the adjective are generated; it is also possible to generate adjective from other parts of speech.

As an example the inflection of the adjective word "akyldy" (in english "clever") includes all word forms of this adjective and their morphological information, which in abbreviated notation contains information on which case of the adjective, and which person is an action and whether it belongs to one or another person. An example shows the inflection of the adjective "akyldy" in cases.

Example. Inflection of the adjective "akyldy"

S=akyldy

$$k_0 = \{\{v_1, v_2\}, \{e_1, e_2\}, \text{akyldy}\}$$

$$e_4 = \{\text{akyldy}, \text{Rel}\}$$

$$e_{32} = \{\text{Rel}, \text{Cases}\}$$

$$k_4 = \{\{v_7, v_8, v_9, v_{10}, v_{11}, v_{12}, v_{13}\},$$

$$\{e_9, e_{10}, e_{11}, e_{12}, e_{13}, e_{14}, e_{15}\}, S_4\}$$

{akyldy(ақылды), akyldynyn(ақылдының), akyldyga(ақылдыға), akyldyny(ақылдыны), akyldyda(ақылдыда), akyldydan(ақылдыдан), akyldymen(ақылдымен)}

On the basis of these rules the morphological analyzer for the Kazakh language was created. It can be used to create spell checking technology of the Kazakh language and can be a cornerstone for translators, semantic search engines, speech technologies, etc.

Many methods of formalizing the morphological rules of a natural language do not allow the description of the semantic properties of words. This paper elaborates on the possibility of using semantic hypergraphs

as a tool in order to formalize the morphological rules of any natural language based on the semantic features of words. Although this paper uses the Kazakh language to illustrate this concept the semantic hypergraph can be applied to any natural language.

Earlier results were obtained using a semantic neural network. 2.8 million word forms were generated from 40,000 initial word forms; these results were approved in [13]. The application of the semantic hypergraph allowed an increase of the number of word forms to 400,000 units. This was achieved by a complete description of the semantic features of words, which utilized the expressive power of the semantic hypergraph.

In the future we plan to apply this proposed method towards other Turkic languages.

4 Conclusion

The construction of ontological models of the morphological rules of Kazakh language allowed for the creation of formal rules of inflection and word formation for each part of speech. Software implementation of these rules made it possible to automatically generate more than 3.2 million word forms (dictionary entries) from 40,000 initial word forms with marked semantic features

5 References

[1] **EİFRİNG H., THEİL R. (**online), Linguistics for Students of Asian and African Languages. Available at: http://www.uio.no/studier/emner/hf/ikos/EXFAC03-AAS/h05/larestoff/linguistics/ (accessed 19/08/2014)

[2] ***KAZAKH GRAMMAR***. Phonetics, word formation, morphology, syntax, Astana, 2002. In Kazakh.

[3] ***BATAYEVA Z.,*** Colloquial Kazakh, Routledge, 2012.

[4] ***GRUBER, T.R.,*** Toward Principles for the Design Of Ontologies Used for Knowledge Sharing, International Journal Human-Computer Studies, 1995, Vol. 43, Issues 5-6, p. 907-928.

[5] ***KHAKHALİN, G.,*** Applied Ontology in the language of hypergraphs, Proceedings of IInd All–Russian Conference "Knowledge - Ontology - Theory" (KONT-09), 2009, p. 223-231. In Russian.

[6] ***RUİTİNG LİAN, BEN GOERTZEL, SHUJİNG KE, JADE O'NEİLL, KEYVAN SADEGHİ, SİMON SHİU, DİNGJİE WANG, OLİVER WATKİNS, GİNO YU,*** Syntax-Semantic Mapping for General Intelligence: Language Comprehension as Hypergraph Homomorphism, Language Generation as Constraint Satisfaction, Artificial General Intelligence. Lecture Notes in Computer Science , 2012,Volume 7716, p. 158-167.

[7] ***ZHEN, L., JİANG, Z.***, Hy-SN: Hyper-graph based semantic network, Knowledge-Based Systems, 2010, Vol 23, Issue 8, p. 809-816.

[8] ***BRETTO A.,*** Hypergraph Theory, Springer International Publishing Switzerland, 2013.

[9] ***BERGE, C.C.***, Graphs and Hypergraphs, Elsevier Science Ltd., 1985.

[10] ***BANU YERGESH, ASSEL MUKANOVA, ALTYNBEK SHARİPBAY, GULMİRA BEKMANOVA, AND BİBİGUL RAZAKHOVA,*** Semantic Hyper-graph Based Representation of Nouns in the Kazakh Language. Computación y Sistemas Vol. 18, No. 3, 2014 pp. 627–635, ISSN 1405-5546, DOI: 10.13053/CyS-18-3-2041

[11] ***PROTÉGÉ.*** Available at: http://protege.stanford.edu (accessed 19/08/2014)

[12] ***POTCHİNSKII I.***, Formal representation of semantic hypergraphs and their operations, 2012. Available at: http://rgu-penza.ru/mni/content/files/2012_Pochinskii.pdf

[13] ***SHARİPBAEV A.A. , BEKMANOVA G.T., BURİBAYEVA A.K., YERGESH B.Z., MUKANOVA A.S., KALİYEV A.K.***, Semantic neural network model of morphological rules of the agglutinative languages, 6th International Conference on Soft Computing and Intelligent Systems, and 13th International Symposium on Advanced Intelligence Systems, SCIS/ISIS, 2012, p. 1094-1099.

MULTIFUNCTIONAL MODEL OF MORPHEMES IN THE TURKIC GROUP LANGUAGES (ON THE EXAMPLE OF THE KAZAKH AND TATAR LANGUAGES)

D.Sh. Suleymanov
Scientific Research Institute of Applied Semiotics, Tatarstan Academy of Sciences Sciences
dvdt.slt@gmail.com,

A.R. Gatiatullin
Scientific Research Institute of Applied Semiotics, Tatarstan Academy of Sciences
ayratrg@antat.ru

A.B. Almenova
Scientific Research Institute of Applied Semiotics, Tatarstan Academy of
almen_akmaral-baijan@mail.ru

ABSTRACT

This article describes a multifunctional computer model of the Turkic affixal morphemes. This model is a hierarchical system of characteristics of morphemes belonging to different language levels: phonological, morphological, syntactic and semantic, and it requires a certain structure and unification in the description of characteristics of morphemes. It is a kind of "inventory" base of the language that can be used for different purposes; in particular, to perform automated comparative analysis of the properties of the Turkic languages, and to develop different linguoprocessors working with Turkic languages. Here, we describe the elements of the multifunctional computer model with examples on the Tatar and Kazakh languages.

Keywords: multifunctional model, affixal morpheme, Tatar language, Kazakh language.

1 Introduction

One of the problems in creating of linguoprocessors for Turkic languages is a deficit in structured data that would describe the properties of the Turkic language units. Obviously, the presence of such databases can accelerate the process of development of applied computer systems working with languages of the Turkic family, such as multilingual search Turkic corpora, machine translation systems for Turkic languages and others. Also, the presence of such models and database software, implemented as Web-interface, will be an effective assistant to turkologists in conducting different comparative studies.

A comparative analysis among Turkic languages with maximum automation of these processes requires conceptual models that would appropriately and adequately describe language units both structurally and functionally.

Another role of this model is to contribute to the process of unification of morphological categories, terms and tags, which are of particular importance for the representation of linguistic information. The analysis of the current situation shows that in the development of linguoprocessors working with Turkic languages and, in particular, in the Turkic corpus linguistics, despite the genetic, structural and typological commonness of Turkic languages, there is still lack of general principles and approaches to linguistic annotation of texts, to the system of tags for morphemes and morphological categories. In the future this may lead to difficulties in conducting comparative researches, as well as in the development of Turkic parallel corpora,

multilingual text processing systems and in resolving of other theoretical and applied problems. We suggest a multifunctional model that will help in the process of the comparative study of morphemes and categories, and also will serve as a unified information system on Turkic morphemes, available in the Internet space.

In the proposed multifunctional model, the morphemes of each of the Turkic languages are described both at the morphological level and at the level of other linguistic phenomena, such as phonology, syntax, semantics, and at the joint of language levels, such as the morpho-semantic and morpho-syntactic levels. The morpho-syntactic level studies the auxiliary particles and postpositions, which in some Turkic languages can be written as a single word, while in others should be written separately. The morpho-semantic level studies compound morphemes, such as mAgAe in the Tatar language, and their counterparts in other Turkic languages. To describe the semantic aspect of the multifunctional model, the authors develop a special unit in the form of situational frames [1].

Currently, the authors are developing a toolkit for filling of the proposed multifunctional computer model. At the same time, we are filling it on the example of the Tatar and Kazakh languages units. The basic elements of the model of morphemes for the Tatar language are represented in the work of the authors [2]. This article reveals the new results concerning the description of the semantic aspect of the model and mechanisms of expansion of the model as a unified framework for the description of language units of all the Turkic languages.

2 Model structure

The structure of this model is shown in table 1:

Table 1. Model structure

	Tatar	Kazakh	Turkish
Functio-nal aspect	Properties A[1].Tat	Properties A[1].Kaz	Properties A[1].Tur
Morpho-nological	Properties A[2].Tat	Properties A[2].Kaz	Properties A[2].Tur
Morpho-logical	Properties A[3].Tat	Properties A[3].Kaz	Properties A[3].Tur
Syntactic aspect	Properties A[4].Tat	Properties A[4].Kaz	Properties A[4].Tur
Semantic aspect	Properties A[5].Tat	Properties A[5].Kaz	Properties A[5].Tur

Every aspect is divided into sub-parameters, and those, in turn, can be further subdivided into sub-parameters, etc. [2].

In this article, we consider the fragments of the morphonological, functional and semantic aspects.

3 Morphonological aspect

The morphonological aspect is presented as a table of allomorphs. It describes all possible allomorphs of a morpheme, which constitute a superficial representation of the deep description of a corresponding morpheme in some context according to the phonological rules.

The rules are described through a set of production rules of the following aspect: A → B, where A is the context-condition of use of this allomorph, and B is the allomorph itself.
The context consists of the morphological and phonological components: A = A1 & A2

The morphological component determines what morpheme is on the left in the wordform, and the phonological one regulates what vowels and consonants are on the left in the wordform.

For example, in the Tatar language the morpheme -nIkI has 2 allomorphs: -nıkı, -neke. The selection of the necessary allomorph is determined by the vowel on the left (whether it is hard or soft), whereas in Kazakh there are 3 allomorphs: -niki, -diki, -tiki, and their choice is determined directly by the leftmost character.

4 Functional aspect

One of the parameters of the functional aspect is that of the coherence of the morpheme.

The coherence of the morpheme indicates whether the morpheme is free (analytical) or coherent (synthetic). This parameter is important for Turkic languages, because the morpheme can be written as a single word in one of the Turkic languages and separately in some other.

For example, the interrogative morpheme in the Tatar language is written as one word, whereas in the Kazakh language it is written separately. At the same time, in the Kazakh language it also has all the phonological variants depending on the context:
Tatar: -mI *urmanmı '(is it) a forest ?'*
Kazakh: MA *orman ba '(is it) a forest?'*

According to this criterion, a morpheme can be classified as an affix if it is coherent, or as an auxiliary part of speech if it is free.

Let us see an opposite example. The Tatar language has a postposition *belən* that is written separately from the main word, whereas in the Kazakh language the counterpart of this postposition is the morpheme *-Ben* with the allomorphs -ben, -men, -pen.

For example:
abıj belən 'with the brother'
aƶamen 'with the brother'

5 Semantic aspect

To describe the meaning of the linguistic units in the model, the authors propose to use relative-situational frames (RSF). RSF present the implementation of a typical situation, consisting of the name of the situation and a number of slots, which are the roles of the constructive elements of this situation. The slots of the frame are filled by language constructs called syntaxemes. The structure of syntaxemes is represented by morphemes.

RSF has the following representation:
SituationSi
Role1: SintaxemI1;
Role2: Sintaxem I2;
…
Role 3: Sintaxem IN;
End_Situation

The choice of a particular situation determines the choice of a particular type of RSF, which is called the base frame, with its corresponding slots-roles filled with some concrete values-syntaxemes.

For example, the basic RSF for the situation that expresses the action according to the change of state has the following representation:

Situation 7.2: action_statel
Object: Sintaxem 5;
Old_state: Sintaxem121;
New_state: Sintaxem119, 120;
Time: Sintaxem78;
Period: Sintaxem97;
Manner: Sintaxem99;
End_Situation

The authors conducted a classification of semantic contexts depending on types of relations that participate in the formation of the deep meaning of a given context. On the basis of this classification we obtained a system

consisting of 60 basic relational-situational frames.

Let us see an example of a syntaxeme.
Syntaxeme 118 as the value of a slot.

Direction:

1. Number of syntaxeme: 118
2. Main word: LSG (‘physical objects’)
    a. Syntax type: AF
    b. Morphological type: N
    c. Morphemic structure: -GA; -LAR-GA; -Im-GA; -Iŋ-GA; -sI-GA; -IbIz-GA; -IgIz-GA; -LAr-Im-GA; -LAr- Iŋ-GA; -LAr-IbIz-GA; -LAr-IgIz-GA
    d. The analytical form: taba
3. Main word: LSG (‘move’)
    a. Morphological type: V
    b. Morphological structure: *
4. Dependent word: -
    a. Morphological type: -
    b. Morphological structure: -
5. Meaning
    a. Type of situation: action_local
    b. The role of the syntaxeme: direction

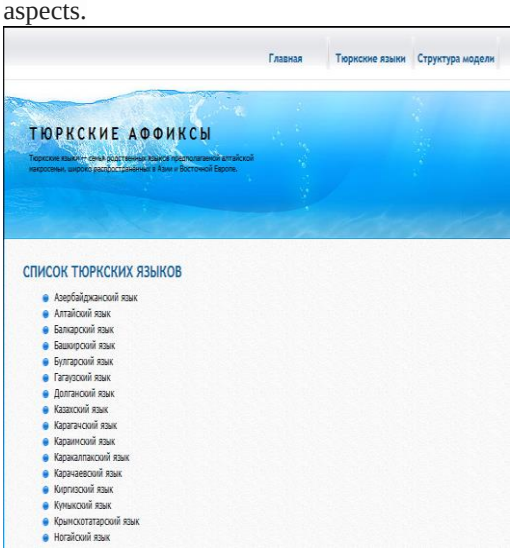
6 Programme description with pictures

The programme that is developed on the basis of the multifunctional model consists of the server and client parts. The server part is a relational database, and the client part is implemented as a Web-interface.

Let us consider the implementation of the interface elements in the form of a Web application. The programme is designed for different modes of use: the expert and the guest. It is expected that the experts will be granted access to the database for editing. The system administrator grants these rights after their registration. Other users can work with the programme as guests to make queries to the database for retrieval of the necessary information and to run the application programmes that work with the database.

The programme for working with the model is designed as multilingual, therefore on the first page of the website there is a list of languages of the Turkic group (Pic.1.). Thus, by selecting a language from this list, the user will be able to work with a single language or view the comparative information about all the Turkic languages.

The user can work with a specific aspect of the model. For this purpose, the main menu contains the option ‘Model Structure’. When chosen, it opens the list of aspects and sub-aspects.

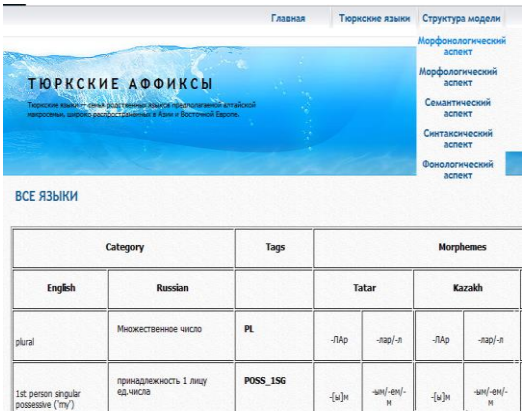


Pic.-1. “Home page” interface

As an example, let us see the interface of one the aspects – the morphonological one (Pic.2).

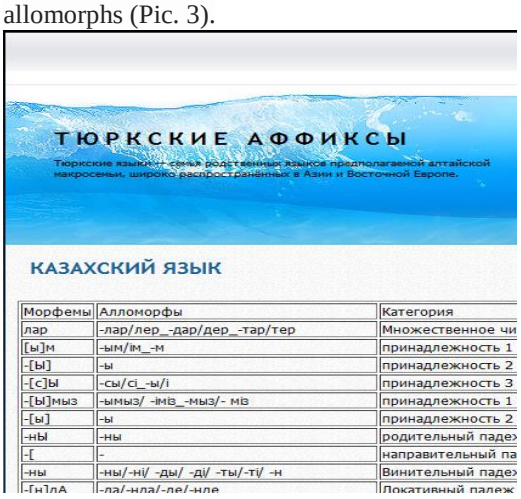
As can be seen in Pic.2, the table contains the information about the morphological categories, their designations in the form of grammatical tags, names of morphemes and lists of their allomorphs. In this picture a table

fragment with the information about the Tatar and Kazakh morphemes can be observed.



Pic.-2. Multilingual table of allomorphs

Morphonological information for each of the Turkic languages can be viewed separately. In this case, it has a detailed representation with an indication of the contexts of the use of allomorphs (Pic. 3).



Pic.-3. The morphonological aspect “The Kazakh language”.

It should be pointed out that the development of the programme is at an early stage and only a small number of functions are already implemented. The work on the programme is conducted simultaneously with the filling of the database model.

7 Conclusion

Due to the fact that the multifunctional computer model of morphemes described in this article is an open model, when being filled with concrete morphemes it can be supplemented by new aspects and new sub-parameters that characterize this morpheme. It is obvious that such characteristics as the adequacy of the model to the real linguistic phenomena or the completeness of the description of specific morphemes can be evaluated only on the basis of the stability of the model and its correct functioning in solving practical tasks on the basis of this model.

The model of a Turkic morpheme that is described in this article reflects the structure of morphemes, their purpose and their manifestation in the text. Apparently, it is a natural informative, educational and scientific base, as well as a database for building of different natural language processors.

8 References

[1] **SULEYMANOV, D.SH., GATIATULLIN, A.R.** *Strukturno-funktsionalnaya kompyuternaya model tatarskikh morfem* [Structural-functional computer model of the Tatar morphemes]. Kazan: FEN, 2003. 220 p. (rus)

[2] **SULEYMANOV, D.SH., GATIATULLIN, A.R.** *Napolneniye semanticheskikh slotov relyatsionno-situatsionnoqo freyma na primere tatarskikh sintaksem* [Filling of semantic slots of the relational-situational frame on the example of the Tatar syntaxemes] // Collection of works of the conference “Open semantic technologies of designing intelligent systems” (OSTIS-2014). Minsk: BSUIR, 2014. P. 178. (rus)

TOWARDS A DATA-DRIVEN MORPHOLOGICAL ANALYSIS OF KAZAKH LANGUAGE

Olzhas Makhambetov, Aibek Makazhanov, Zhandos Yessenbayev, Islam Sabyrgaliyev, and Anuar Sharafudinov

Nazarbayev University Research and Innovation System, 53 Kabanbay batyr ave., Astana, Kazakhstan

{omakhambetov, aibek.makazhanov, zhyessenbayev, islam.sabyrgaliyev, anuar.sharaphudinov}@nu.edu.kz

ABSTRACT

We propose a method for complete morphological analysis of Kazakh language that accounts for both inflectional and derivational morphology. Our method is data-driven and does not require manually generated rules, which makes it convenient for analyzing agglutinative languages. The intuition behind our approach is to label morphemes with so called transition labels, i.e. labels that encode grammatical functions of morphemes as transitions between corresponding POS, and use transitivity to ease the analysis. We evaluate our method on a fair-sized sample of real data and report encouraging results.

1 Introduction

Morphological analysis (MA) is one of the crucial steps in automated processing of any language, and in the case of agglutinative languages (ALs) it is hard to overestimate its importance. Agglutination causes words to acquire complex meanings, effectively transforming them into whole phrases. Consider a Kazakh word [*bolmaghandyqtan*] which translates into English phrases [*because something/someone is/was absent*] or [*because something does/did not go certain way*]. MA of the word reveals the underlying phrase: [bol-ma-ghan-dyq-tan] → [exist-NEG-PTCP-NOM-ADV] → [exist - (exist not) - (non existing) - (nonexistence) - (due to nonexistence)]. Obviously parsing and translating ALs require MA to deal with such cases. Even POS-tagging in ALs benefits from leveraging morphological information [1, 2]. Traditionally the MA problem has been approached by building finite state transducers (FST) [3–5] based on a formal description of the morphology of a language. FST-based approaches require a set of morphological and phonological rules to generate analyses that are both grammatically and orthographically correct. Although there are open source tools that effectively implement transducers [6, 7], certain language-specific morphotactics still need to be implemented. Whilst acknowledging the efficiency and descriptive power of transducers, in the present exploratory study, we focus on a pure data-driven approach, that can be later used as a baseline method or, indeed, as a lightweight morphological analyzer. In this respect, it should be noted that in our approach we do not consider certain language-specific issues. Namely, as we will show later, our method does not account for compound words and certain phonological rules. We plan to address these issues in the future.

We divide the MA problem into the problems of (i) morphological segmentation and (ii) ranking. The respective challenges are: (i)

pruning potentially huge lists of candidate segmentations, while trying to keep the correct ones (precision-recall trade-off) and (ii) employing an effective ranking strategy. To address the first challenge we label every morpheme with its respective POS-to-POS transition label (inflectional morphemes (e.g., plural endings, etc.) are converted to a transition of POS to itself). As we will show later this allowed us to achieve a data coverage of around 97% (i.e. a correct analysis was found for 97% of test words) and maintain a decent precision-recall trade-off. To rank candidate segmentations we use a Hidden Markov Model (HMM) and a Markov chain model, assuming mutual independence of roots and paradigms, and dependence of consequent morphemes within paradigms. Evaluating the models in terms of precision- and recall-at-*k*, we show that, simple as it is, our approach achieves encouraging performance.

The reminder of the paper is organized as follows. In the next section we review some of the existing work on morphological analysis of morphologically-rich languages in general, and Kazakh language in particular. In Section 3 we thoroughly describe the underlying methodology of our approach. Section 4 presents our experiments and discusses the results. In Section 5 we conclude the study and discuss the future work.

2 Related Work

Statistical approaches to the MA problem have been successfully applied in the past. In a work presented by Hakkani-Tur et al. [8] the distribution of morphological analyses for Turkish is modeled using n-gram models that formulate certain morphosyntactic features, which differ by morphotactical relation of inflectional groups (IGs) within the word and the final IGs of previous words. For Czech language, Hajič et al. [9] combined a rule-based system with a statistical model based on HMM, using these approaches sequentially. Chrupała et al. [10] cast the problem into a classification task, training two maximum entropy classifiers that provide probability distributions over analyses and word-lemma pairs. The authors use a language independent set of features, and show that their system performs well, achieving respective accuracies of 97%, 94%, and 82% for morphologically-rich languages, such as Romanian, Spanish, and Polish.

Along with supervised methods several unsupervised approaches were proposed [11, 12]. In a work by Creutz and Lagus [11] words are initially segmented using a baseline algorithm, which is based on a recursive minimum description length (MDL) model. Then, initial segmentations are reanalyzed by more advanced models formulated in a maximum a posteriori probability, a maximum likelihood or an MDL framework. The authors refer to this collection of models as the Morfessor. A slightly modified version of the Morfessor was presented by Kohonen et al. [12], who implemented a semi-supervised extension to the baseline algorithm.

Recently there have been attempts to develop formal methods for morphological analysis of Kazakh. While Sharipbayev et al. [13] addressed the problem of Kazakh word forms generation for all inflectional parts of speech, employing semantic neural networks[1] [14], a number of works [15–18] resorted to finite state approaches. Kairakbay et al. [18] present a formal description of the Kazakh nominal paradigm, and Zafer et al. [16] provide a rather vague description of a two-level Kazakh morphology. Both works, however, do not report any significant results. Kessikbayeva et al. [15] also resort to a finite state morphology, and provide a thorough description of the nominal and verb paradigms, and formalize some of the derivational rules. Using the Xerox finite state toolkit [19] the authors conduct experiments on a set of 2000 randomly chosen analyses and report an overall data coverage of 96% (precision was not reported). Finally, Makazhanov et al. [20] address the problem in a context of spelling correction. The authors

[1] Unfortunately the authors do not provide any information on the results of their experiments.

formalize nominal and verb paradigms and develop an error tolerant FSA, reporting 83% general accuracy on a dataset of 1700 word-error pairs.

Our work differs from the aforementioned works on Kazakh morphology in that (i) it considers both inflections and derivatives; (ii) it needs no manual rule generation; (iii) it was evaluated on the largest data set available for Kazakh.

3 Methodology

We divided the task of morphological analysis into two major components: (i) segmenting input words into morpheme sequences; (ii) finding the most probable sequence of morpheme-tag pairs (analysis).

In the absence of a transducer segmenting an input word becomes challenging. A naive approach is to try labeling all possible letter sequences in a given word. This is, however, computationally prohibitive and we want to do better than that. The first thing that comes to mind is to use a morpheme dictionary acquired from a labeled data, and search for matches in a given word. However, simple matching does not account for a natural morpheme order that exists in the language. One could parse all the morpheme sequences and infer this order, eventually ending up building a sort of a state machine. This approach, however, has a potential of missing correct analyses where a certain morpheme sequence occur that had not been seen in a training set.

To account for such omissions, we convert all morpheme labels into POS transitions, i.e. for a given analysis [bol-ma-ghan-dar] → [exist-NEG-PTCP-NOM.PL], we construct the following representation: [bol_R_VB-ma_VB_VB-ghan_VB_PTCP-dar_PTCP_PTCP][2].

Now, suppose, that in a training set we have seen both morphemes ghan-PTCP and dar-NOM.PL, but we have not observed them in a sequence, i.e. a pattern [ghandar] never occurred. Suppose, also, that we have seen a sequence of respective allomorphs [gen-NOM-der-NOM.PL], or in a transitive notation: [gen_VB_PTCP-der_PTCP_PTCP]. A method that works with conventional morpheme labels fails to segment this pattern, because a [ghan-PTCP-dar-NOM.PL] sequence had not occurred. However, due to the fact that we have seen transitional labels VB_PTCP and PTCP_PTCP, the transition-based method constructs a link, and successfully segments the pattern.

The segmentation module is developed using recursive function that tries to segment a word, from left to right, into substrings, which are elements of dictionary of morpheme transitions. The process stops when a left substring (prefix) matches a known root or its character length is equal to one. Once we acquire all segmentations we convert transitions back to conventional morpheme tags used in a given language.

To select the most probable segmentation we have conducted ranking experiments using two models. The first approach is based on Markov chains, where the probability of a sequence of morphemes is computed on morpheme bigrams using a chain rule (i.e. the current morpheme depends only on the previous one):

$$P(W) \propto P(r\_t) \prod_{i=1}^{n} P(m\_t_i | m\_t_{i-1})$$

where *r_t* is a POS-tag-labeled root of a given word and *m_t* is a grammatically-labeled morpheme. We estimate morpheme bigram probabilities using Maximum Likelihood Estimation (MLE). To account for a data sparseness problem we assign a portion of the probability mass to unseen cases by employing the Laplace smoothing:

$$P(m\_t_i | m\_t_{i-1}) \approx \frac{N(m\_t_i, m\_t_{i-1}) + \alpha}{N(m\_t_{i-1}) + \alpha |V|}$$

where $N(m\_t_i, m\_t_{i-1})$ is the count of a given morpheme bigram, $|V|$ denotes the cardinality of a set of unique morphemes, and smoothing parameter $\alpha = 0.1$ (estimated empirically). We have to note that while computing the

[2] Notice that a plural ending [dar-NOM.PL] is converted to a [PTCP_PTCP] transition, i.e. inflectional morphemes are replaced by transitions of POS to themselves.

probability of the first morpheme that immediately follows the root, we assume that it depends on the POS of the root. The probability of a root is estimated in the following manner:

$$P(r\_t) \approx \frac{N(r\_t) + \alpha}{N + \alpha|W|}$$

where $N(r\_t)$ is the count of a given POS-tag-labeled root and N is the total number of all words in the training set. In order to prioritize segmentations with vocabulary roots we heavily penalize segmentations containing OOV roots. As in the previous case, parameter α is estimated empirically to be equal to 0.1. The described model will be referred to as a simple Markov chain (SMC).

In the second approach we model a distribution of segmentations using HMM, and try to maximize the posterior probability, $P(T|W)$:

$$P(T|W) \approx \frac{P(T)P(W|T)}{P(W)}$$

where $P(T)$ denotes a probability of a morpho-tag sequence, and $P(W|T)$ denotes a probability of a word given a tag sequence T. The denominator $P(W)$ remains constant for all segmentations, and thus can be dropped.

We compute $P(T)$ using a chain rule:

$$P(T) = \prod_{i=1}^{n} P(t_i|t_{i-1})$$

the probabilities of morpho-tag bigrams are estimated in the following manner:

$$P(t_i|t_{i-1}) \approx \frac{N(t_i, t_{i-1}) + \beta}{N(t_{i-1}) + \beta|V|}$$

where $N(t_i,t_{i-1})$ denotes the count of a given bigram, $\beta = 0.9$ (estimated empirically) and $|V|$ is the cardinality of a set of all unique morpho-tags. We compute $P(W|T)$ as follows:

$$P(W|T) \approx \frac{N(m_i, t_i) + \beta}{N(t_i) + \beta|W|}$$

where $N(m_i,t_i)$ is the count of a given tagged morpheme (not just a morpho-tag, but also a surface form), and $|W|$ denotes the cardinality of a set of such all unique tagged morphemes.

	train, Δ-per fold	**overall**
# roots	22 980	24 255
# mrphs, unigr-s	1 623	1 655
# mrphs, innfl.	332	338
# mrphs, deriv.	1 291	1 317

Table 1. Per fold and overall characteristics of the data set

As it can be seen, unlike the previous model, this one is more abstract, and operates mostly with morpeme tags (except for calculation of $P(W|T)$), leaving out the actual surface forms of morphemes. Hereinafter this model will be referred to as HMM.

As we mentioned in the introduction, in the present study certain language-specific aspects of the MA were not addressed. First, we do not perform analysis of compounds, i.e. in multiple-root words we do not locate every single root and analyze them in isolation. Instead we collapse all roots and possible intermediate paradigms into a single root and consider a paradigm attached to the last root only. For instance, for a word [*ulkendi-kishili*] our method provides the following analysis: [ulkendi-kishi-li]→[big-small-ADJ], while the correct analysis is [ulken-di-kishi-li]→[big-ADJ-small-ADJ]. Second, in our analyses we do not recover roots or morphemes distorted due to the phonetics of the language. For instance, while a correct analysis for a word [*zhughystyghy*] is [zhuq-ystyq- y]→[infect-NOM-NOM-POSS.3SG], our method returns [zhu*gh*-ys-ty*gh*-y]→[infect-NOM-NOM-POSS.3SG], i.e. the root [zhuq] and a morpheme [tyq] remain distorted as [zhugh] and [tygh] respectively.

4 Experiments

We evaluate our models in terms of precision- and recall at-*k* on an annotated subset of Kazakh Language Corpus [21]. The data set consists of 610 867 word-tokens (78 704 unique). We perform a standard 10-fold cross-validation and report averages and standard deviations per fold.

Table 1 shows the characteristics of the data set as per training fold and overall data.

Morpheme stats counts include allomorphs. As it can be seen, in our data set there are 1 317 derivatives, almost four times as much as inflectional inflections. To the best of our knowledge, for Kazakh language, it is the largest number of derivational morphemes ever considered.

Precision-at-*k* is calculated as a ratio of correct analyses found at top-*k* positions to the total number of correctly analyzed tokens in a fold. Recall-at-*k* is calculated as a ratio of correct analyses found at top-*k* positions to the total number of all tokens in a fold.

Table 2 contains the results of the performance of the SMC model. As it can be seen 73% of all correct analyses were placed at the first position of the ranking lists, and, in terms of recall, in 71% of the cases correct analyses appeared at the first rank. There is a steady growth with increase in *k*, and for $k = 5$ the model achieves 90% precision and 87% recall. In general we observe close values for precision and recall for every *k*. Overall, in 97% of the cases (per fold) a correct analyses was provided.

Table 3 contains the results of the performance of the HMM model. We can see that this model performs slightly lower dragging behind SMC for about 5% (for $k = 1$) in both precision and recall. We believe that this happens because, in contrast to our initial intuition, by ignoring surface forms of morphemes HMM loses some important information rather than resolving ambiguous allomorphic cases. In terms of precision-recall trade-off we observe a trend similar to that of SMC.

When we analyzed the cases where our models failed to put a correct analysis in top-5, we found that a lot of such low ranked cases were due to context related errors. We have performed initial experiments with a context-sensitive model, which utilizes POS information of a preceding root and achieved a top-1 precision of 79% on a 95-to-5% train/test data split. These initial results suggest that incorporating context information may help to boost the accuracy of the method.

k	**precision at-*k***	**recall at-*k***
1	73.2±0.37	70.9±1.06
2	85.3±0.46	83.2±1.05
3	88.8±0.48	86.3±1.20
4	90.0±0.44	87.8±1.21
5	90.6±0.45	87.7±1.13

Table 2. Precision- and recall-*k* for SMC average± standard deviation per fold

k	**precision at-*k***	**recall at-*k***
1	68.3±0.46	66.2±0.73
2	81.6±0.50	79.0±1.07
3	86.0±0.48	83.7±1.22
4	88.5±0.49	85.7±0.91
5	89.7±0.46	86.8±1.15

Table 3. Precision- and recall-*k* for HMM average± standard deviation per fold

5 Conclusion and Future Work

We have developed a data-driven method for morphological analysis of Kazakh language that accounts for both inflectional and derivational morphology. The method does not require formalization, in that all the rules are induced directly from labeled data in the form POS-to-POS morpheme transitions. Our experiments suggest that these transition-based morphotactics help in pruning many false patterns while keeping correct analyses as candidate segmentations. We believe that the same technique could be used in the analysis of other agglutinative languages, as all that it requires is labeled data in a given language. We evaluated our method in terms of top-*k* precision using Kazakh as a reference language. The best of our models achieved 90% performance in terms of precision-at-*k*. The analysis of generated segmentations revealed that most of errors occurred due to context insensitivity of our method. We have already started experiments on incorporation of context information, and achieved encouraging initial results. Our future work will be directed at development of a context-sensitive extension of the method. In addition, we will make necessary adjustments to the method to

facilitate compound-sensitive and phonetically-correct analyses.

6 References

[1] D. Elworthy, "Tagset design and inflected languages," in In EACL SIGDAT workshop iFrom Texts to Tags: Issues in Multilingual Language Analysis, 1995, pp. 1–10.

[2] J. Hana and A. Feldman, "A positional tag set for Russian," Proceedings of LREC-10. Malta, 2010.

[3] K. Koskenniemi, "A general computational model for word-form recognition and production," in Proceedings of the 10th international conference on Computational linguistics. ACL, 1984, pp. 178–181.

[4] K. Oflazer and C. Güzey, "Spelling correction in agglutinative languages." in ANLP, 1994, pp. 194–195.

[5] H. Sak, T. Güngor, and M. Saraçlar, "A stochastic finite-state morphological parser for Turkish," in Proceedings of the ACL-IJCNLP 2009 Conference. Stroudsburg, PA, USA: ACL, 2009, pp. 273–276.

[6] M. Hulden, "Foma: a finite-state compiler and library." in EACL (Demos), A. Lascarides, C. Gardent, and J. Nivre, Eds. ACL, 2009, pp. 29–32.

[7] K. Linden, M. Silfverberg, E. Axelson, S. Hardwick, and T. Pirinen, HFST-Framework for Compiling and Applying Morphologies, ser. Communications in Computer and Information Science, 2011, vol. Vol. 100, pp. 67–85.

[8] D. Z. Hakkani-Tur, K. Oflazer, and G. Tur, "Statistical morphological disambiguation for agglutinative languages." Computers and the Humanities, vol. 36, no. 4, pp. 381–410, 2002.

[9] J. Hajič, P. Krbec, P. Pavel Květoň, K. Oliva, and V. Petkevič, "Serial combination of rules and statistics: A case study in czech tagging," in Proceedings of the 39th Annual Meeting on ACL. Stroudsburg, PA, USA: ACL, 2001, pp. 268–275.

[10] G. D. Grzegorz Chrupała and J. van Genabith, "Learning morphology with morfette," in Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08), Marrakech, Morocco: ELRA, may 2008.

[11] M. Creutz and K. Lagus, "Unsupervised models for morpheme segmentation and morphology learning," ACM Transactions on Speech and Language Processing (TSLP), vol. 4, no. 1, p. 3, 2007.

[12] O. Kohonen, S. Virpioja, L. Lepp¨anen, and K. Lagus, "Semi-supervised extensions to morfessor baseline," in Proceedings of the Morpho Challenge 2010 Workshop. Espoo, Finland: Aalto University, September 2010.

[13] A. Sharipbayev, G. Bekmanova, B. Ergesh, A. Buribayeva, and M. K. Karabalayeva, "Intellectual morphological analyzer based on semantic networks," in Proceedings of the OSTIS-2012, 2012, pp. 397–400.

[14] D. E. Shuklin, "The structure of a semantic neural network extracting the meaning from a text," Cybernetics and Sys. Anal., vol. 37, no. 2, pp. 182–186, Mar. 2001.

[15] G. Kessikbayeva and I. Cicekli, "Rule based morphological analyzer of Kazakh language," in Proceedings of the 2014 Joint Meeting of SIGMORPHON and SIGFSM. Baltimore, Maryland: ACL, June 2014, pp. 46–54.

[16] H. R. Zafer, B. Tilki, A. Kurt, and M. Kara, "Two-level description of Kazakh morphology," in Proceedings of the 1st International Conference on Foreign Language Teaching and Applied Linguistics (FLTAL11), Sarajevo, May 2011.

[17] G. Altenbek and W. Xiao-long, "Kazakh segmentation system of inflectional affixes," in CIPS-SIGHAN, 2010, pp. 183–190.

[18] B. M. Kairakbay and D. L. Zaurbekov, "Finite state approach to the Kazakh nominal paradigm," in Proceedings of the 11th International Conference on Finite State Methods and Natural Language Processing. St Andrews, Scotland: ACL, July 2013, pp. 108–112.

[19] A. Ranta, "A multilingual natural-language interface to regular expressions," in Proceedings of the International Workshop on Finite State Methods in Natural Language Processing, ser. FSMNLP '09. Stroudsburg, PA, USA: ACL, 1998, pp. 79–90.

[20] A. Makazhanov, O. Makhambetov, I. Sabyrgaliyev, and Z. Yessenbayev, "Spelling correction for kazakh," in Proceedings of the 2014 CICLing. Kathmandu, Nepal: Springer Berlin Heidelberg, 2014, pp. 533–541.

[21] O. Makhambetov, A. Makazhanov, Z. Yessenbayev, B. Matkarimov, I. Sabyrgaliyev, and A. Sharafudinov, "Assembling the kazakh language corpus," in EMNLP. Seattle, Washington, USA: ACL, October 2013, pp. 1022–1031.

THE ADVANTAGE OF INTERPHONEME PROCESSING AT DIPHONE RECOGNITION OF KAZAKH WORDS

Aigerim Buribayeva *Altynbek Sharipbay*
L.N. Gumilyov Eurasian National University
buribayeva@mail.tu; *sharalt@mail.ru;*

ABSTRACT

This paper presents a method of interphoneme processing at diphone recognition of Kazakh words. Authors made experiment to test how impact interphoneme processing to recognition accuracy. The experiment results show that recognizable word best differs from the other word on the DTW-Distance after interphoneme processing than without it. The results can be used in the construction of recognition system of single words.

1 Introduction

Automatic recognition of natural language verbal speech is one of important areas of development of artificial intelligence and computer science as a whole, as results in this area will allow to solve the problem of development of man's efficient voice response means with the help of computer. A principal opportunity for transition from formal languages-mediators between man and machine to natural language in verbal form as universal means of expression of man's ideas and wishes has appeared with development of modern voice technologies. Voice input has a number of advantages such as naturalness, promptness, input's notional accuracy, user's hands and vision freedom, possibility of control and processing in extreme conditions.

Specialists from several scientific areas research the problem of speech recognition for more than 50 years. Methods and algorithms which are used are separated into four big classes:

- Methods of discriminant analysis based on Bayesian Discrimination [1];
- Hidden Markov Model [2];
- Artificial neural networks [3];
- Dynamic programming - dynamic time warping (DTW) [4];

It should be noted a number of benefits sought by the development of speech recognition systems:

- Continuous speech - feature that allows users to speak naturally (continuous), not pausing between words (discrete speech input).
- Large dictionaries - the ability to process a large Word Count general and special categories of technical and subject areas of knowledge to increase the capacity and effectiveness of voice recognition systems.
- Independence from the speaker - the system's ability to recognize words without personal computer settings by repeating the same speech.

The most frequently and successfully for recognition of continuous speech using Hidden Markov Model (HMM) [5, 6] or Artificial Neural Networks [6, 7]. Different base units: phonemes, allophones, diphones and triphones, etc. selected for speech recognition. Dynamic time algorithms (DTW) still effective to recognize single words [8].

We chose the word recognition technology based on the collected diphone database, because single words recognize is more

accurately [9]. The system does not recognize the diphones separately, it synthesizes of these the words' etalons, and then recognize whole words by the algorithm DTW. The advantage of the system is that to add a new word there is no need to train the system voicing the word, but rather enter a word in text form. Automatic generation of words' etalons of diphones will make a step towards large dictionaries, and speaker-independent systems can be achieved by averaging the etalons.

2 Materials and Methods

According to [10], the authors have decided to formulate its present viewpoint in the following thesis: «One of the possible keys to speech recognition lies in interphoneme transitions.»
Analysis of the situation, we can start with the following simple experiment. Using any known program for working with sound, for example «Sound Forge», write any two words, and then cut out, fixed (middle) part of their component sounds. Reproducing the resulting audio signals, we can at the hearing to determine what the words sound. On the contrary, by cutting interphoneme transitions, and leaving the stationary part of the sound, we found it difficult to distinguish by ear, for example, words «шана» и «сана».
So, it was a program which allows using the diphone database, automatically generate etalons given dictionary of words and keep them DTW-recognition. we have described in detail the construction of such a system in [10]. Etalons of words recognized by the dictionary formed from the etalons of diphones, which database of approximately one and a half thousand created for each speaker in advance [9]. Creation of such a database in the future eliminates the need to create any etalons by voice. we mean that the corresponding diphone interphoneme transition within a word, the site in standard lengths: 3 windows in the 368 samples to the left of the label between the sounds and 3 of the same window to the right of the same label. Etalon of diphone - set of 6 appropriate vectors. In addition, we use a section of 3 windows at the beginning of words and site in 3 windows at the end of words, conditionally call them respectively the initial and final middiphone speech (the transition from silence to speech and vice versa). All vectors in etalons diphones, play the role of the code vectors and form a codebook B. All etalons diphones are numbered, numbered and all the code vectors. Every word of the dictionary automatically transcribed, transcription construct string names diphones. Each of them is replaced by the corresponding diphone's etalon. The resulting string vectors forms a word's etalons.
We apply for recognition has already become a classic algorithm T. Vintsyuk known as algorithm DTW. We use the feature vector related to the relative frequencies of the lengths of complete oscillations in the speech segments in 368 samples [9].
The recognition process is constructed as follows. Recognize words automatically segmented and then subjected to interphoneme processing: removed stationary components of the sounds and left only diphones around labels between sounds (interphoneme transitions). And only then word gets recognition.
It is known that the DTW-word recognition with etalons built of diphones, perhaps as a signal in which the stationary part of the deleted sounds (interphoneme processing) and for the original signal.
In this regard, we decided to check how interphoneme processing affects to recognition accuracy.

3 Experiments

We made experiment to test how impact interphoneme processing to recognition accuracy. We chose 100 of Kazakh word for recognition. Only one female speaker

participated in the experiment because our system recognizes a specific speaker. First, we recognize the words in the mode "No interphoneme processing." After the same words were recognized in the mode "with interphoneme processing". The experiment was done in a regular university's classroom without noise isolation.

4 Results

The results of the experiment are shown in the following table:

Table 1 - Result of recognition of the word "Қазақ" ("Kazakh")

Word	**DTW-distance without** *interphoneme processing*	**DTW-distance with** *interphoneme processing*
Қазақ (Kazakh)	15,88	10,32
Намыс (pride)	23,84	20,74
Бетеге (fescue)	30,77	26,35
Өнер (talent)	25,57	25,55
Араша (pull apart)	24,83	23,37

The table shows the result of the recognition of the word "Kazakh". The second column shows DTW-distance of 5 words at the most immediate recognition in the "No interphoneme processing." In the third column shows DTW - distance of 5 words at the most immediate recognition in the "interphoneme with treatment."

As you can see, in the first case relation of two first distances in column is:

$$\frac{23,84}{15,88} \approx 1,50 .$$

In the second case it is equal to

$$\frac{20,74}{10,32} \approx 2,00 .$$

Analogous result is visible in all our other experiments.

Recognizable word best differs from the other word on the DTW-Distance after interphoneme processing than without it. Conclusion: the recorded speech signal is advisable to expose interphoneme at diphone recognition.

5 Acknowledgments

The presented work is supported by "Automation of Recognition and Generation of the Kazakh Language Written and Oral Speech" Project implemented under the budget program 120 "Grant Financing of Scientific Researches", specificity 149 "Other Services and Works", by Priority 3. Information and Telecommunication Technologies.

6 References

[1] Raut, C.K., Bayesian discriminative adaptation for speech recognition, Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on Eng. Dept., Cambridge Univ., Cambridge , 19-24 April 2009, Page(s): 4361 – 4364

[2] Lawrence, R. Rabiner (February 1989). "A tutorial on Hidden Markov Models and selected applications in speech recognition". Proceedings of

the IEEE 77 (2): 257–286. doi:10.1109/5.18626.

[3] Al-Alaoui, M.A., Al-Kanj, L., Azar, J., and Yaacoub, E., Speech Recognition using Artificial Neural Networks and Hidden Markov Models, IEEE MULTIDISCIPLINARY ENGINEERING EDUCATION MAGAZINE, VOL. 3, NO. 3, SEPTEMBER 2008

[4] Винцюк, Т.К. Анализ, распознавание и интерпретация речевых сигналов. Киев, Наук. думка, **1987.**

[5] Najkar, N., Razzazi, F., Sameti, H. An evolutionary decoding method for HMM-based continuous speech recognition systems using particle swarm optimizationб Pattern Anal Applic, DOI 10.1007/s10044-012-0313-7

[6] Frikha, M., Ben Hamida, A. A Comparitive Survey of ANN and Hybrid HMM/ANN Architectures for Robust Speech Recognition American Journal of Intelligent Systems 2012 2(1): 1-8 DOI: 10.5923/j.ajis.20120201.01

[7] Hosom, J.P., Cole, R., and Fanty, M. Speech Recognition Using Neural Networks at the Center for Spoken Language Understanding. //Center for Spoken Language Understanding, Oregon Graduate Institute of Science and Technology, July 1999.

[8] Dev Dhingra, S., Nijhawan, G., Pandit, P., Isolated Digit Recognition Using MFCC AND DTW, International Journal on Advanced Electrical and Electronics Engineering, (IJAEEE), ISSN (Print): 2278-8948, Volume-1, Issue-1, 2012, pp 59-64

[9] Шелепов, В.Ю., Ниценко А., Дорохина, Г.В., Карабалаева, М.Х., Бурибаева, А.К. О распознавании речи на основе межфонемных переходов. Вестник. Астана: Евразийский национальный университет им. Л.Н.Гумилева, 2012. – Специальный выпуск.–С.436-440

[10] Бурибаева, А.К. Распознавание казахских слов на основе дифонной базы, Труды Международной конференции "Компьютерная обработка тюркских языков" (Turklang-2013), С. 230-239.

GRAMMATICAL DISAMBIGUATION IN THE TATAR LANGUAGE CORPUS

Bulat Khakimov
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, Kazan Federal University, Kazan, Russia
khakeem@yandex.ru

Rinat Gilmullin
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, Kazan Federal University, Kazan, Russia
rinatgilmullin@gmail.com

Ramil Gataullin
Research Institute of Applied Semiotics of the Tatarstan Academy of Sciences, Kazan Federal University, Kazan, Russia
ramil.gata@gmail.com

ABSTRACT

This article concerns the issues of corpus-oriented study of the most frequent types of grammatical homonymy in the Tatar language and the possiblities for automation of the disambiguation process in the corpus. The authors determine the relevance of alternative parses generated in the process of automatic morphological analysis in terms of real linguistic ambiguity. This work presents a variant of classification of frequent homoforms and methods for their disambiguation, and it estimates the potential impact on the corpus.

Keywords: linguistic corpus, Tatar language, grammatical homonymy, homoform, disambiguation

1 Introduction

The problem of grammatical ambiguity and its resolution is one of the most pressing problems in modern computer and corpus linguistics [1]. “Tugan Tel” Tatar National Corpus, that was developed in the “Applied semiotics” Research Institute of the Tatarstan Academy of Sciences [2], employs the system of automatic morphological annotation on the basis of our own morphological analyzer [3]. In order to adequately reflect the specifics of the Tatar language, a morphological standard of the corpus was developed [4]. Research on specification and improvement of the metalanguage for the description of a Tatar wordform is currently carried out [5]. The general conception of the corpus is presented in [6]. To implement the grammatical disambiguation in the Tatar National Corpus, developers have conducted a study of contextual constraints of different types of grammatical homonyms, involving statistical corpus data, and suggest the methods of automatic grammatical disambiguation for the Tatar language.

2 Statistical Characteristics of the Corpus

At the initial stage of work we obtained the statistical data on the frequency of wordforms with alternative parses, presented in Table 1, from the database of texts of the Tatar National Corpus [2]. The total volume of the corpus is 21,940,452 word usages, the proportion of Word usages with alternative parses is 25.75%.

№	Alternative parses	Amount	Proportion in the corpus
1	Wordforms with alternative	5650820	25,75%

	parses		
	of which:		
2	2 parses	4282108	19,51%
3	3 parses	1045392	4,76%
4	4 parses	296547	1,35%
5	5 and more parses	26773	0,12%
6	Wordforms with alternative parses in the sample	21940452	100%

Table-1. Some statistical charasteristics of Corpus

To identify the most frequent types of homonymy in the corpus and to assess their relevance in terms of real language homonymy, a sample of 500 most frequent combinations of alternative parses was created. On its basis 150 types with two parsing options were selected for further analysis, because this parsing type is presented in the corpus in the biggest proportion.

3 Relevance Evaluation of Types of Homonyms

In the first phase of work, irrelevant combinations of homonyms were identified. In such combinations alternative parses often appear because of the errors of the morpholoical analyzer, that is due to the redundancy in the stem set or in the model of inflection. Some cases are caused by incorrect morphonological rules of the analyzer; correction of these rules also allows to exclude the cases of ambiguity belonging to the specified types.

The cases conditioned by the disuse of one of the parsing options present special interest. We refer to such cases as irrelevant, because the potential wordforms, which are automatically generated during the work of the morphological analyzer, are not represented in the actual speech use. A corresponding set of wordforms was experimentally determined for them.

The suggested measures on the exclusion of irrelevant types of homonymy have reduced the number of homonymous parses in the corpus by about 8.5% (2.1% of the total volume of texts in the corpus).

4 Most Frequent Types of Grammatical Homonymy

For the most frequent linguistically relevant types of homonyms we have made a classification, which groups separate automatically determined subtypes. The following frequent types of homonyms were singled out:

1. Noun vs Pronoun
2. Verb vs Noun/Adjective
3. Pronoun vs Numeral
4. Noun vs Adjective
5. Postposition vs Noun/Numeral
6. Noun vs Adverb
7. Adjective vs Noun with attributive affix
8. Noun/Adjective vs Noun with possessive affix
9. Adjective vs Noun in aditive case
10. Adjective vs Verb
11. Verb vs Verb
12. Adjective vs Adverb
13. Pronoun vs Pronoun in locative-temporal case
14. Noun vs Adjective with affix -chA
15. Pronoun vs Noun

All types except type 1, 3, 5, 6, 9 and 15, are represented by a set of regularly formed wordforms, which possess a certain number of grammatical features. Contextual disambiguation rules for these types are conditioned by these characteristics and the characteristics of the disambiguating context.

Type 1 is represented by a single frequent word *ul* (‘he/son’). Different context principles work for each of the part-of-speech alternatives.

Type 3 is also represented by only one frequent word *ber*, which is used both in the meaning of the numeral 'one' and in the function of the indefinite pronoun, that is close to the function of the indefinite article. Each part-of-speech alternative has its own context patterns.

Type 5 includes four subtypes. Each of them is represented by one word – postposition: *öçen* ('for'), *turında* ('about') and *buyınça* ('on'), or pronoun: *tege* ('that'). Each of these words has a homonym, which is a noun in a definite form. Grammatical characteristics of homonymous words and syntactic functions of the respective postpositions define context rules for this type.

Types 6 and 9 are represented by the lexemes *bik* ('very/bolt') and *başka* ('other/head+ DIR'), respectively.

Type 15 is also an example of one wordform homonymy; it is represented by the word *bez* ('we/awl').

The total number of all types of word usages is 1624839. The proportion in the corpus sample is 7,4% (21940452 word usages). The proportion among the homonymous parses is 28,7% (5650820 in the indicated corpus sample).

This variant of classification does not include another special case of verb forms homonymy, which is related to the multifunctionality of voice affixes. Thus, a statistical study of corpus data has shown that the total number of such cases of homonymy in the analyzed sample of texts is 408346 word usages (1.8% of the total volume of texts and 7.2% of all the alternative parses). The most frequent subtype among them is the V - V + REFL subtype, where one and the same verbal form can stand both for a separate lexeme, which is included on its own in the stem set, and the voice form of another lexeme. For example, *ezlänergä, totınırga, yaşerenergä, seltänergä, ağulanırğa, alınırğa*. Disambiguation of this type is not a trivial task, and in many cases requires consideration of not only morpho-syntactic, but also semantic characteristics of the disambiguating context.

5 Context Rules for Automatic Grammatical Disambiguation

In order to make use of classical methods of grammatical disambiguation based on context rules, we classified the types of homonyms, of which homoforms represent the biggest part. The full classification of types of homonyms (analysis of the full range of types) is an extremely time-consuming and pragmatically unreasonable task, as the Tatar language belongs to the agglutinative languages, where the number of morphemes that can be attached to the stem is theoretically unlimited. For example, in the above mentioned corpus of Tatar texts, which includes more than 21 million Word usages, there are more that 7000 types of homoforms.

On the other hand, the use of classical statistical methods is complicated by the sparseness of data and the lack of a standard annotated disambiguated corpus. Thus, the use of each of these methods is not sufficiently effective.

One possible solution to this problem is described in [7]. The method was used for disambiguation of texts on the Turkish language, where the number of wordforms with multiple parsing options, reaches 40%. According to the results of this work, the accuracy of the method for the Turkish language reached 96% (with an accuracy of classical statistical methods of 91%). Typological and genetic proximity of the Turkish and the Tatar language suggests that this method is able to show good results for the Tatar language.

As well as in the Tatar language, in the Turkish language the number of possible types of homonymy is not limited, which in turn leads to failure when using classical statistical methods due to the sparseness of data. To avoid this, instead of searching for the contextual constraints for each type of homoforms, the algorithm searches for contextual constraints for each morpheme, the

number of which is limited, in contrast to the number of types of homoforms: 126 morphemes for the Turkish language [1] and 120 morphemes for the Tatar language [4]. It is obvious that this approach significantly reduces data sparseness.

According to this method, training data is collected for each morpheme from the sample of wordforms, which contain the given morpheme at least in one of the possible morphological parses. The received data are classified as "positive" or "negative", depending on whether the morpheme is included into the contextually suitable paradigm. On the basis of these data and using a special algorithm, the grammatical disambiguation rules are trained [1].

In order to predict a suitable parsing option of an unfamiliar wordform, the morphological analyzer firstly analyzes the wordforms to the greatest possible extend by all possible paradigms. Next, on the basis of rules, for each morpheme a certain probability of its presence or absence in the given wordform in the given context is defined. The final result is calculated taking into account the accuracy of each rule, and ultimately the most likely parse is selected [1]. A distinguishing characteristic of this model and the learning algorithm (GPA algorithm) is their high resistance to irrelevant and redundant features.

The problem of the lack of a fully annotated disambiguated corpus of the Tatar language, which would be used as training data, can be partially solved by choosing for analysis not the homoforms with a certain morpheme, but on the contrary, the wordforms with the given morpheme and a single parsing option. This will allow to identify the contextual constraints directly for the morpheme. However, this approach does not cover the entire set of morphemes (e.g., the morphemes, for which there have not been found worforms with a single parsing option). In such cases, contextual rules are designed manually or after a complete annotation of the model fragment of the corpus.

6 Software Modules for Context Rules Developement

As part of this research, we have developed a software tool designed to create, edit and test the database of context rules for the tasks of automatic grammatical disambiguation in the Tatar language [8].

This module can be used both separately (for this, contextual disambiguation rules should be designed for all types of homonyms), and in combination with the probabilistic and statistical methods. The second part of the toolkit "LangRuleBase-PMM module" [8] uses this database of context rules for grammatical disambiguation in texts. This kind of toolkit, which takes into account the particularities of the Tatar language, was developed fort he first time. It is aimed at assisting the research work of a philologist.

To facilitate the annotation process of the Tatar language corpus (including manual disambiguation), as well as to provide convenient access to the statistical data of the corpus, we developed a web application that makes the work with corpus texts more convenient and flexible for statistical research. This software module, in addition to the possibility of expanding the corpus and morphological annotation, supports the option of manual grammatical disambiguation.

7 Conclusion

Formal context-oriented classification of homoforms and development of context rules for grammatical disambiguation using experimental statistical data in the Tatar language have been carried out for the first time. Linguistic resources and software modules developed on the basis of the classification and context rules allow to

perform disambiguation in the Tatar National Corpus and other applications. Estimated cumulative effect in the case of disambiguation of the identified frequent types of homonymy in the Tatar language corpus can be up to 50%. Our future research will be focused, on the one hand, on the study of disambiguating contexts and the development of contextual disambiguation rules and, on the other hand, on the analysis of statistical regularities in the field of polysemy at different language levels and the search for effective approaches to disambiguation taking into account the particular characteristics of the Tatar language.

8 Acknowledgements

The work is supported by the Russian Foundation of Basic Research and the Government of the Republic of Tatarstan, (project # 12-07-97015)

9 References

[1] Yuret D., Ture F. *Learning Morphological Disambiguation Rules for Turkish*. Proceedings of the Human Language Technology Conference of the North American Chapter of the ACL. NewYork, 2006. Pp. 328–334.

[2] Galieva A.M., Khakimov B.E., Gatiatullin A.R. *Metayazyk opisaniya struktury tatarskoy slovoformy dlya korpusnoy grammaticheskoy annotatsii* [Metalanguage of description of a Tatar wordform for corpus-based grammatical annotation]. Proceedings of the Kazan University, Ser. Human sciences, 2013. V. 155, b. 5. Pp. 287-296. (rus)

[3] Nevzorova O.A., Zinkina Yu.V., Pyatkin N.V. *Razresheniye funktsionalnoy omonimii v russkom yazyke na osnove kontekstnykh pravil* [Resolution of functional homonymy in the Russian language based on context rules]. Proceedings of "Dialog'2005" International Conference. Moscow: Nauka, 2005. Pp. 198-202. (rus)

[4] Suleymanov D.Sh., Gilmullin R.A. *Dvukhurovnevoye opisaniye morfologii tatarskogo yazyka* [Two-level description of the Tatar language morphology]. Proceedings of "Language semantics and image of the world" International Scientific Conference. Kazan: Ed. Kazan State University, 1997. Vol 2. Pp. 65-67. (rus)

[5] Suleymanov D.Sh., Gilmullin R.A., Gataullin R.R. *Programmnyy instrumentariy dlya razresheniya morfologicheskoy mnogoznachnosti v tatarskom yazyke* [Software toolkit for morphologic disambiguation in the Tatar language]. Proceedings of OSTIS-2014 IV International scientific and technical conference. Minsk, 2014. Pp. 503-508. (rus)

[6] Suleymanov D.Sh., Khakimov B.E., Gilmullin R.A. *Korpus tatarskogo yazyka: kontseptualnyye i lingvisticheskiye aspekty* [Tatar language corpus: conceptual and linguistic aspects]. Bulletin of Tatar State Humanitarian Pedagogical University. 2011. № 4 (26). Pp.211-216. (rus)

[7] "Tugan Tel" Tatar National Corpus. – URL: http://web-corpora.net/TatarCorpus/search/?interface_language=ru.

[8] Khakimov B.E., Gilmullin R.A. *K razrabotke morfologicheskogo standarta dlya sistem avtomaticheskoy obrabotki tekstov na tatarskom yazyke* [Notes on the development of a morphological standard for automatic text processing systems in the Tatar language]. System analysis and semiotic modeling: Proceedings of all-Russia conference with international participation (SASM-2011). Kazan, 2011. PP. C. 209-214. (rus)

EXPLORING THE EFFECT OF BAG-OF-WORDS AND BAG-OF-BIGRAM FEATURES ON TURKISH WORD SENSE DISAMBIGUATION

Bahar İLGEN
Istanbul Technical University
ilgenb@itu.edu.tr

Eşref ADALI
Istanbul Technical University
adali@itu.edu.tr

ABSTRACT

Feature selection in Word Sense Disambiguation (WSD) is as important as the selection of algorithm to remove sense ambiguity. Bag-of-word (BoW) features comprise the information of neighbors around the ambiguous target word without considering any relation between words. In this study, we investigate the effect of BoW features and Bag-of-bigrams (BoB) on Turkish WSD and compare the results with the collocational features. The results suggest that BoW features yield better accuracy for all the cases. According to the comparison results, collocational features are more effective than both BoW and the BoB features on disambiguation of word senses.

Key words: *Word Sense Disambiguation, feature selection, supervised methods, bag-of-word features.*

1 Introduction

The determination of proper sense label is required in almost all applications of Natural Language Processing (NLP) area. Machine Translation (MT), Information Retrieval (IR), Information Extraction (IE), Semantic Annotation (SA) and Question Answering (QA) are some of the NLP branches that benefit from WSD. The performance of these applications depends on the performance of WSD unit.

The basic approaches for WSD comprise the supervised, unsupervised and knowledge based methods. The selection of the proper method can be considered the application and the resources available. The knowledge based methods primarily rely on resources such as dictionaries, ontologies and thesaurus. These methods do not need to use corpus evidence. On the other hand, unsupervised methods utilize external information and work on raw corpora. Supervised methods use sense annotated data to train from. Although supervised methods yield superior results, the number of annotated corpora are too few for the majority of the natural languages. As a result, unsupervised methods have gained attention recently, since the annotation scheme is expensive and labor intensive. There is also one group of approaches of semi-supervised (or minimally supervised) methods which utilize a small amount of sense annotated data and expand the annotated part iteratively.

WSD can also be classified considering two variants: (1) Lexical sample task, and (2) all-words task. The first approach focuses on the disambiguation of the previously selected words. Machine Learning (ML) methods are usually preferred to handle these tasks since both the words and senses are limited. The labeled portion of the dataset is used the train classifier. Then the unlabeled portion of samples can be labeled using classifier. On the

other hand, all-words approach disambiguates all the words in a running text.

Knowledge is the central component to remove sense ambiguity of the words. It may be lexical or learned world knowledge. Sense frequency, concept trees, selectional restrictions and the POS information are some of the examples of lexical knowledge category. Learned knowledge category refers the information such as "Indicative words", "syntactic features" and "domain specific knowledge"[1]. Unsupervised methods usually utilize lexical knowledge sources while supervised methods use world knowledge. But in practice different combinations of the knowledge can be used in WSD systems.

There are two important decisions to be considered for a WSD system: the selection of learning algorithm and the set of features to be used. ML techniques can be used to automatically acquire disambiguation knowledge of the corpus-based WSD. And the several resources such as sense labeled corpora, dictionaries and other linguistic resources can be used for a typical WSD system. Supervised methods can be grouped into categories considering the induction strategy they use. These methods comprise probabilistic models, similarity based methods, linear classifiers and Kernel based methods and the methods based on some properties (i.e., one sense per collocation/discourse, attribute redundancy, decision lists/trees, rule combination etc.).

WSD introduces additional difficulties comparing to POS tagging or syntax parsing since each word is associated with unique meaning. That means a complete training set requires huge number of examples. This case is also known as language sparsity problem. This language sparsity problem can be handled with the selection of proper features in training algorithms.

In the scope of this study, we investigate the impact of bag-of-word and bag-of-bigram features on disambiguating senses. The rest of the paper is organized as follows. In section-2 related work has been summarized. Section-3 and Section-4 describe the dataset and features respectively. In Section-5, experimental results have been presented. Finally, Section-6 draws the conclusion.

2 Related Work

Feature selection has a critical importance in terms of correctly discriminating senses or categorizing them into proper labels. There are several studies to investigate the impact of feature selection strategies on WSD [2-7].

The impact of the features can be investigated by analyzing two aspects; feature type and the window size of the context. Selected features were classified as topical and local features in [8]. Topical features are usually extracted by checking the presence of keywords occurring anywhere in the sentence. The sentences around the ambiguous headword are taken as context. Local features comprise the information such as POS tagging, syntactic and semantic features for the neighbor words around headword.

In [9], main feature types have been grouped into local features, syntactic dependencies and global features. In total, six feature sets have been investigated including the bag-of-words, local collocations, bag-of-bigrams, syntactic dependencies, all features except bag-of-words and all features. They used different editions of Senseval [1] datasets in order to conduct experiments. The Lexical Sample data of the Senseval-2 has been used for parameter

[1] http://www.senseval.org

tuning. All-words and Lexical Sample datasets of Senseval-3 have been used for testing. It is reported that "all-features" set is the best single classifier for every method except one. It is also stated that local collocational features discriminate better than bag-of-word features for separate feature sets.

In[10], the impact of collocational features have been investigated on Turkish. The root forms and the POS information of the target word and its' neighbors have been used at encoding grammatical local lexical features. These features have been extracted from the text which is segmented into POS tagged units. The target word itself, the words within ±4 positions of the target word and the corresponding POS tags have been used in the study. Turkish Lexical Sample dataset (TLSD) have been used in the experiments. Figure-1 shows the sample window scope for the collocational features.

```
kriz: (Noun)(A3pl)(Pnon)(Nom)
sonra: (Noun)(Zero)(A3sg)(P3sg)(Loc)
büyük: (Adj)
şirket: (Noun)(A3pl)(Pnon)(Gen)
<HEAD-SENSE='baş'-SENSE_TDK_NO="2"…
baş: (Noun)(A3sg)(P3sg)(Loc)
</HEAD>
bulun: (Adj)(PresPart)
yönetici: (Noun)(A3pl)(Pnon)(Gen)
görev: (Noun)(A3sg)(Pnon)(Nom)
değişim: (Noun)(A3pl)(P3sg)(Nom)
```

Figure-1. Window scope for the colllocational features.

As being a member of agglutinative languages, Turkish is based on suffixation. And grammatical functions of the language are generated adding proper suffixes to the stems. As a result, number of POS features may be excessive. Because of the agglutinative property with inflectional and derivational suffixes in Turkish, two tools have been utilized. Firstly, a finite-state two level Turkish morphological analyzer has been used for morphological decomposition [11]. Then a disambiguation tool has been used since the output of the morphological analyzer is ambiguous [12].

3 Dataset

TLSD has been used in the experiments of this study. This dataset has been gathered to conduct our previous studies on Turkish WSD. TLSD comprises the highly ambiguous noun and verb samples of Turkish. These words were selected by considering the polysemous Turkish words in [13] and the polysemy degree of the words in Dictionary of Turkish Language Association (TLA) [14]. The results of our simple analysis on dictionary of TLA show that the average polysemy degree for Turkish is 3.53. The polysemy degrees of TLSD are calculated as 10.67 and 26.53 for noun and verb sets respectively. Both noun and verb groups in the dataset include 15 ambiguous words each of which has at least 100 samples. The samples have been gathered from Turkish websites on health, education, sports and news. We follow "one sense per sample principle" and each sample has only one sense of the ambiguous word. The ambiguous words in TLSD noun and verb sets are shown below (Table-1).

Table-1. Ambiguous noun and verb sets of TLSD.

Nouns:	*Açık, baskı, baş,derece, dünya,el, göz,hat, hava, kaynak, kök, kör, ocak, yaş, yüz*
Verbs:	*Aç, al, at, bak, çevir, çık, geç, gel, gir, gör, kal, ol, sür, ver, yap*

In the scope of the work, we also investigated the effective number of bag-of-word features and determined the most frequent content words as features. The most frequent 100, 75, 50 and 25 content words have been taken as features. We used vectors for the corresponding sizes and repeated the experiments (Figure-2). These vectors are initialized by assigning "0" to each cell. Then the values are incremented by "1" if the feature

exists in the lexical sample. Our findings suggest that the most frequent 75 and 100 content words yielded better accuracy for noun and verb sets respectively.

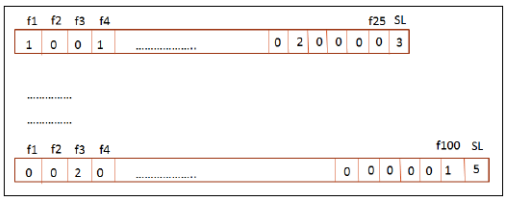


Figure-2. Bag-of-words feature vectors.

4 Features

4.1 Bag-of-word Features

Bag-of-word features comprise words around target word without considering their relations, grammar and even word order. Unordered set of words serve as features. The value of any feature is determined by counting the number of times that they occur for a given context. The context is fixed window with the ambiguous word as center of other words. For the experiments of this study, we followed the given steps:

- *Removing stopwords from samples*
- *Morphological analysis of dataset.*
- *Removing ambiguity after morphological analysis.*
- *Determination of features and encoding samples using them.*
- *Applying algorithms which we utilized for other features.*

4.2 Bag-of-bigram Features

We gathered bag-of-bigram features by following the similar steps with the bag-of-words features. After eliminating the stopwords, bigram words of the lexical samples have been exracted. Then we obtained most frequent bigram words. The only difference for BoB features is that we took the more features than the bag-of-word features since the features are more sparse. The number of features have been chosen considering the observation frequency of the bigrams and taken as approximately between 350 and 500 for each ambiguous word.

5 Experiments

After determining the number of effective bag-of-word features, we investigated the optimal window size to consider. We adjust the samples to take different values of ±n words (preceding and following words for values; ±30, ±15, ±10 and ±5) around target word. Our experiment on varying window sizes show that the best window size for noun and verb sets is 5. We kept this setting for the BoW features but took whole samples for the BoB features since the features are more sparse. Naïve Bayes, IBk, Support Vector Machines and tree based methods (J48 and FT algorithms) have been used in the experiments. Figure-3 shows the accuracy results for BoW and BoB features of Turkish nouns. Figure-4 displays the similar results for Turkish verb set. MFB represent the most frequent baseline.

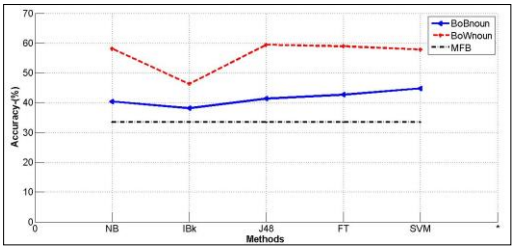


Figure-3. Accuracy(%) results of noun set.

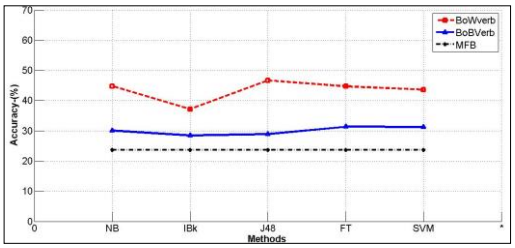


Figure-4. Accuracy(%) results of verb set

Table-2 and Table-3 summarizes the accuracy (%) results on TLSD for three feature sets. The results of the noun and verb sets are presented in Table-2 and Table-3 respectively. MFB values for noun and the verb sets are 33.47(%) and 27.60(%).

Table-2. Comparison of feature sets on Turkish nouns.

Feature	NB	IBk	J48	FT	SVM
BoW	58.1	46.3	59.4	58.9	57.8
BoB	40.4	38.1	41.3	42.6	44.8
Colloc	60.6	53.9	61.0	73.5	69.0

Table-3. Comparison of feature sets on Turkish verbs.

Feature	NB	IBk	J48	FT	SVM
BoW	44.8	37.2	46.7	44.7	43.6
BoB	30.1	28.4	28.9	31.3	31.2
Colloc	46.5	43.1	66.0	67.3	58.6

6 Conclusion

It is known that the features extracting from context words play important role on isolating senses. And there are many features to consider that can contribute the meaning of a given word. In this study, we investigated the impact of bag-of-word and bag-of-bigram features on Turkish WSD systems. Then we compare the results of these two groups with the results of collocational features. Our findings suggest that bag-of-word features yielded better results than bag-of-bigrams. The results also show that the collocational features are more efficient than both the bag-of-words and bag-of-bigram features. It is thought that the results of the bag-of-word and bag-of-bigram features can be improved by combining diverse set of features.

7 References

1. Zhou, X. and H. Han. *Survey of Word Sense Disambiguation Approaches*. in *FLAIRS Conference*. 2005.
2. Orhan, Z. and Z. Altan. *Effective Features for Disambiguation of Turkish Verbs*. in *IEC (Prague)*. 2005.
3. ORHAN, Z. and Z. Altan, *Determining Effective Features for Word Sense Disambiguation in Turkish.* IU-Journal of Electrical & Electronics Engineering, 2011. **5**(2): p. 1341-1352.
4. Agirre, E., O.L. de Lacalle, and D. Martınez. *Exploring feature spaces with svd and unlabeled data for Word Sense Disambiguation*. in *Proceedings of the Conference on Recent Advances on Natural Language Processing (RANLP'05)*. 2005.
5. Turdakov, D.Y., *Word sense disambiguation methods.* Programming and Computer Software, 2010. **36**(6): p. 309-326.
6. Suárez, A. and M. Palomar, *Feature selection analysis for maximum entropy-based wsd*, in *Computational Linguistics and Intelligent Text Processing*. 2002, Springer. p. 146-155.
7. Dang, H.T., et al. *Simple features for Chinese word sense disambiguation*. in *Proceedings of the 19th international conference on Computational linguistics-Volume 1*. 2002. Association for Computational Linguistics.
8. Dang, H.T. and M. Palmer. *Combining contextual features for word sense*

disambiguation. in *Proceedings of the ACL-02 workshop on Word sense disambiguation: recent successes and future directions-Volume 8*. 2002. Association for Computational Linguistics.

9. Agirre, E., O.L. de Lacalle, and D. Martínez. *Exploring feature set combinations for WSD*. in *Proc. of the SEPLN*. 2006.
10. Ilgen, B., E. Adali, and A. Tantug. *The impact of collocational features in Turkish Word Sense Disambiguation*. in *Intelligent Engineering Systems (INES), 2012 IEEE 16th International Conference on*. 2012. IEEE.
11. Oflazer, K., *Two-level description of Turkish morphology*. Literary and linguistic computing, 1994. **9**(2): p. 137-148.
12. Yuret, D. and F. Türe. *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*. 2006. Association for Computational Linguistics.
13. Göz, İ., *Yazılı türkçenin kelime sıklığı sözlüğü*. Vol. 823. 2003: Türk Dil Kurumu.
14. Sözlük, G.T., *Türk Dil Kurumu,[çevrimiçi]*. Elektronik adres: http://tdk. org. tr/tdksozluk/sozbul. ASP, 2005.

STRUCTURAL TRANSFER RULES FOR KAZAKH-TO-ENGLISH MACHINE TRANSLATION IN THE FREE/OPEN-SOURCE PLATFORM APERTIUM

Aida Sundetova *Aidana Karibayeva* *Ualsher Tukeyev*
Information Systems Department, Al-Farabi Kazakh National University, Almaty, Kazakhstan
sun27aida@gmail.com; aidana_karib@mail.ru; ualsher.tukeyev@gmail.com;

ABSTRACT

This paper describes process of building structural transfer rules for Kazakh-to-English machine translation system on free/open-source Apertium platform. Structural transfer rules are used for translating texts from Kazakh to English by couple of rules in three stages. This paper shows how sentences in Kazakh are transformed to English sentences, what types of phrases and attributes are used. Results are presented by comparing Apertium Kazakh—English system with other online translators.

1 Introduction

Nowadays developing machine translation from Kazakh language to English is very important and useful for people who want to understand texts in Kazakh and translate them. However, building translation system from a Turkic language, which has complex agglutinative morphology, faces some difficulties. For example, Kazakh morphology, as all Turkic language morphologies, is more complex than English morphology and very different from it. It is impossible to do translation form Kazakh to English by word-to-word. Because Kazakh is agglutinative language, words are done by adding morphemes with vowel harmony (synharmonism) [1]. English is an analytic language that conveys grammatical relationships without using complex inflectional morphemes like in Kazakh language. To be more precise, relationships are expressed by additional constructions with modal verbs or prepositions [2].

There are important differences in syntax between the Kazakh and English languages; for example, the order of constituents in sentences: subject–object–adverbial modifier–verb (in English it is: subject–verb–object–adverbial modifier). There are also important differences in translating verb tenses: Future Simple and Present Simple, Present Perfect and Past Perfect in Kazakh have the same translation, modal verbs are made by adding auxiliary verbs (I can play – Мен ойнай аламын) or using adjectives which mean "obligation": жөн ('should'), қажет ('necessary'), керек ('need') (I should go – Менің барғаным жөн) [3].

Kazakh language has no gender, so personal pronoun "Ол" could has three translations: he/she/it. By default, it is translated as "he", however, for special constructions as "Ол – қыз" (in English "She is girl") "Ол" is translated as "She".

By considering these features, we are developing machine translation from Kazakh to English based on the Apertium free/open-source machine translation platform (Forcada et al. 2011, http://www.apertium.org) [4]. Because, firstly, it already contains a rather complete Kazakh morphology (Salimzyanov et al. 2013), secondly, it includes an English monolingual dictionary which also contains morphological analysis [5]. Therefore for developing Kazakh–English system we need to

build bilingual dictionary and write couple of rules.

This paper contains 4 sections: Section 2 describes Apertium platform and its structure, Section 3 describes Kazakh–English structural transfer and Section 4 gives results of system by comparing with other systems.

2 Apertium platform and its modules

Apertium is a free/open source machine translation system. Apertium is a platform of machine translation which whose development started with financing from the governments of Spain and Catalonia at University of Alicante (Universitat d'Alacant) in 2005. Apertium is free software which is published by developers according to GNU GPL conditions.

Apertium was originally intended for translation between related languages. However this system has been expanded to translate texts between less similar language pairs. To create the new system of machine translation one needs develop linguistic data (dictionaries, rules) in accurately specified XML formats. This system uses finite state transducers for all of its lexical transformations, and hidden Markov models for part-of-speech tagging or word category disambiguation.

Apertium platform consisting of the modules (Figure 1):

– **Deformatter**. It separates the text to be translated from the formatting tags. Formatting tags are encapsulated as "superblanks" that are placed between words in such a way that the remaining modules see them as regular blanks.

– **Morphological analyser**. For each surface form (that is, for each lexical unit as it appears in the text), the morphological analyser generates one or more lexical forms composed of: lemma (dictionary or citation form), lexical category (or part-of-speech), and inflection information. The morphological analyser executes a finite-state transducer generated by compiling a morphological dictionary for the source language. Lexical units containing more than one word (multiword lexical units) are analyzed as a single lexical unit. Morphological analyser uses a finite state transducer based on two-level rules (in the case of Kazakh, `apertium-kaz.kaz.lexc`, `apertium-kaz.kaz.twol`). This module therefore separates lexemes and processes morphological analysis, and then returns possible lexical forms.

– **Part-of-speech (POS) tagger**. Apertium's POS tagger is based on a statistical model based on hidden Markov models which processes the result of the application of on constraint-grammar rules (Karlsson 2005), which are used to discard some analyses using simple rules (written in `apertium-kaz.kaz.rlx`) based on context. For example, consider the morphological analysis of word *қара*:

^қара/қара<adj>/қара<adj><advl>/қара<adj><subst><nom>/қара<v><tv><imp><p2><sg>/қара<adj>+e<cop><p3><pl>/қара<adj>+e<cop><p3><sg>/қара<adj><subst><nom>+e<cop><p3><pl>/қара<adj><subst><nom>+e<cop><p3><sg>

This word is ambiguous and has 6 meanings. Many surface forms are ambiguous, which means that these words have more than one POS and therefore more than one possible translation. After this module, all words have only one morphological analysis.

– **Lexical transfer**. This module uses a bilingual dictionary (`apertium-eng-kaz.eng-kaz.dix`) which has very simple structure [7]. The module reads each source-language lexical form and finds one or more corresponding target-language lexical forms. Multiword units are translated as a single word.

– **Lexical selection**. It uses rules that select for those lexical words having many translations, one of the translations in the target language according to context. All rules are written in file `apertium-eng-kaz.kaz-eng.lrx`.

– **Structural transfer.** This module identifies sequences of lexical forms (phrases or segments), which need syntactical

processing (handling of number, prepositions, etc.) to be translated. It uses files with rules, which specify the syntactic transformation as a cascaded process. Transfer rules, which transform lexical-form sequences into a new sequences for the target language, perform the work in this module. Structural transfer is the focus of this paper, and will be described in detail in section 3.

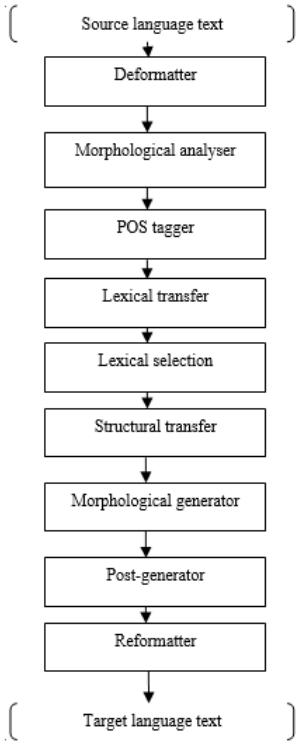


Figure-1. The Apertium machine translation pipeline

– **Morphological generator**. From the sequence of target-language lexical forms produced by the structural transfer, it generates a corresponding sequence of target language surface forms. The morphological generator executes a finite-state transducer generated by compiling a morphological dictionary for the target language.

– **Post-generator**. It takes care of some minor orthographical operations in the target language (for instance, it generates the English form *cannot* from *can* and *not*). This module is generated from file with rules which are very similar in format to dictionary files.

– **Reformatter**. It places format tags back into the text so that its format is preserved.

3 Structural transfer from Kazakh into English languages

The structural transfer module in Apertium does operations, which determined in transfer rules and can be like this: word reordering, adding some suffixes, removing unnecessary tags or attributes etc [8]. Structural transfer in Apertium system comprises two parts: *pattern* and *action*. "*Pattern"* defines the sequence to which the rule will be applied, whereas "*action"* consists of the actual operations needed to generate the corresponding sequence in the target language. Transfer in Apertium may be of two types. The first type is used in a similar languages and generates the sequence of lexical forms in the target language in a single step. The second type is the one used in our Kazakh–English system, and consists of three levels:

– "chunker" level (file `apertium-eng-kaz.kaz-eng.t1x`);

– "interchunk" level (file `apertium-eng-kaz.kaz-eng.t2x`);

– "postchunk" level (file `apertium-eng-kaz.kaz-eng.t3x`).The following sections describe the three levels of Kazakh-English structural transfer.

3.1 The Kazakh-English chunker

The chunker divides a sentence in *chunks* which may be seen as elementary sentence constituents such as noun phrases, verb phrases, etc. (see Table 1)

Table-1. Types of chunks

Patterns	**Meaning**
SN	Noun phrase
SV	Verb phrase
AdjP	Adjectival phrase
PP	Postpositional phrase

Some examples of noun- and verb-phrase chunks are given in the next tables (Table 2, Table 3):

Table-2. Noun-phrase chunks

Input pattern[1]	**Example**	**Output block**	**Translation**
n	бақша	SN{n}	garden
adj	әдемі	AdjP{adj}	beautiful
num	жеті	SN {num}	seven
adj n	әдемі бақша	SN{adj n}	beautiful garden
det n	менің бақшам	SN{det n}	my garden
num n	жеті бақша	SN{num n}	seven gardens
num adj n	жеті әдемі бақша	SN{num adj n}	seven beautiful gardens
det adj n	менің әдемі бақшам	SN {det adj n}	my beautiful garden
det num adj n	менің жеті әдемі бақшам	SN {det num adj n}	my seven beautiful gardens
n pr	үстел астында	PP {pr n}	under table
adj n pr	үлкен үстел үстінде	PP {pr adj n}	on big table
num n pr	бес үстел үстінде	PP {pr num n}	on five tables

Table-3. Verb-phrase chunks

Input pattern	**Example**	**Output block**[2]	**Translation**
v	ойна	SV{vblex}	play
v	ойнап отыр	SV{vbser vblex }	is playing
v	ойнаған	SV{vbhaver vblex}	has played
v	ойнамаған	SV {vbhaver adv vblex}	has not played
v	ойнар	SV {vaux vbhaver vblex}	will have played

1 Abbreviations: adj, adjective; n, noun; num, numeral; pr, postposition; det, determiner.

2 Abbreviations: vblex, lexical verb; vbser, verb 'to be'; vaux, auxiliary verb; vbhaver, verb 'to have'.

Take into account that the lexical forms have been translated in advance and that the remaining transfer modules work only on target-language lexical forms.

After these blocks (chunks) are created the interchunk module performs operations on these blocks, without modifying their contents. This module makes it possible to generate the correct target-language word order, to treat number and person, number agreement in verbs.

3.1.1 Translation of noun-phrases

We will illustrate the translation of noun-phrase with the example: *әдемі бақшаларда* ('in the beautiful gardens').

The chunker identifies this phrase as a noun-phrase (adj noun) and after that, it translates it into English by adding relevant tags. There may be such tags: number (plural form), cases (assign locative case).

In general, this phrase has the following attributes: number (singular or plural), cases, possessives. One of the main problems in translation noun-phrases is generating the English articles (*a, an, the*), which are absent in Kazakh. All nouns with nominative and accusitive cases are translated as noun-phrases:

- single noun: SN [қыз <n><nominative>] - SN [girl <n><nominative><sg>];

Also for structue like:

- adjective + noun: SN [әдемі<adj> үй<n><nom> - SN [beautiful<adj> house<n><sg>];
- numerals + noun (in accusative case): SN [жеті<num> бақша<n><accusative> - SN [seven<num> garden<n><plural>]. Rules for this phrase are not assigned to noun accusative case because in English translation it does not have any suffixes.

3.1.2 Translation of verb phrases

Translation of verb from Kazakh to English has specific difficulties. For instance, in

Kazakh the past tense might have two translations; for example, the sentence “Мен ойнағанмын” can be translated such as “I have played” or “I had played”, that is, the sentence can be translated as present perfect or past perfect. We decided to generate a present perfect translation, because in while developing in first steps it difficult to identify past perfect, which has to come before past simple and present perfect are more common in simple sentences. Below are shown examples of verb-phrases, which the system already translates (Table 4):

Table-4. Translation of verb phrases

Tense in Kazakh language	**Example**	**Tense in English**	**Translation**
Present (Ауыспалы осы шақ)	*Мен ойна+й+мын*	Present Simple	*I play*
Past (Жедел өткен шақ)	*Мен ойна+дым*	Past Simple	*I played*
Future (Болжалды келер шақ)	*Мен ойна+р+мын*	Future Perfect	*I will have played*
Present (Нақ осы шақ)	*Мен ойна+п жатыр+мын*	Present Continuous	*I am playing*

3.1.3 Translation of adjectival and postpositional phrases

Adjectival phrases do not have any attributes and are marked as “AdjP”, and are used for those cases in which adjectives are not part of a noun phrase.

Postpositional phrases are structures in which function words are found after the noun. For instance, the phrase «жеті әдемі бақшаның астында» translated as «under seven beautiful gardens». In a construction like this the compound postposition formed by the genitive ending “*-ның”* in “*бақшаның”* and the word “астында*”* are used to express the notion expressed in English with the function word “under”.

In this level of transfer rules are written 57 rules.

3.2 “Interchunk” level

As we can see from the other translation systems (try translate texts [9,10]), in target-language texts word order is incorrect. It means that reordering does not work well. When we write a chunker rule (`.t1x`), we aim at dividing the sentence in a sequence of patterns or chunks. After that, we take care of the order of these chunks by writing interchunk rules in file `apertium-eng-kaz.kaz-eng.t2x`, by writing appropriate reordering rules. For instance, in sentence “Біз кітапты оқимыз” pattern of pronoun “Біз” ('We') is “SN”, pattern of object “кітапты” ('book') is “SN-accusative” and pattern of verb “оқимыз” ('read') will be “SV”: “Біз кітапты оқимыз” - “We read book”(reordering “We book read”). So in Kazakh language verb stays at the end of sentence, although in English that can stay at the beginning or in the middle: “Мен[1] әдемі[2] бақшаны[3] көремін[4]” → “I[1] see[4] beautiful[2] garden[3]”. Rules of this level do next operations:

- build new sequence of chunks;
- adding prepositions by cases: “әдемі бақшаДА[locative]” - “**in** beautiful garden”;
- agreement. Agreement of words – subject and verb, adjective and noun, for example, for agreement between verb and subject are person and number agreement. For example: “*Бала ойнайды” – “Child plays”* (noun is third person and number is singular, so why noun should have morpheme of person).

The number of rules like this is some few, about ten rules, furthermore these rules will be extended.

4 Results

The current version of the system (revision №56387) can translate SN-, SV-. AdjP- and PP- phrases. We plan to extend the number of rules to improve translation quality. In the table below we compare some translation systems with examples that our system can translate [9, 10]. All available translations of sentences and phrases can be seen from tests (see [11]).

Table-5. Results of comparison

Phrase	Example	Apertium	Pragma 6	Sanasoft
SN	менің екі әдемі көйлегім	My two beautiful dresses	*me* two *әдемі* my the dress	My two beautiful *көйлегім*
SV	Мен студент емеспін	I am not student	I *not* student	I student *емеспін*
PP	Анау суреттер дің астында	under those pictures	Under those *by* pictures	under *that* pictures

5 Conclusion

We have described Kazakh—English machine translation system on Apertium platform and process of developing structural transfer rules. Many features in translating from Kazakh to English as assign cases, agreement, prepositions, etc. were solved. In the future this system will be considered the translation task of future transitional tense, the passive voice, degree adjective, interrogative sentence and other tasks will be observed.

Acknowledgements: the authors thank Mikel L. Forcada, Francis Tyers, Jonathan N. Washington and Ilnar Salimzyanov and other developers in the Apertium project for their help during the development of this system, authors would also like to express their gratitude to Mikel L. Forcada for advises in writing this paper.

6 References

[1] Агглютинативные языки (2012). Retrieved from http://ru.wikipedia.org/wiki/Агглютинативный_язык

[2] Аналитический язык (2013). Retrieved from http://ru.wikipedia.org/wiki/Аналитический_язык

[3] Печерских, Т.Ф., Амангельдина, Г.А. (2012) “Особенности перевода разносистемных языков (на примере английского и казахского языков)”, Молодой ученый. №3, 259–261

[4] Forcada, M.L., Ginestí-Rosell, M., Nordfalk, J., O'Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J.A. Sánchez-Martínez, F., Ramírez-Sánchez, G., Tyers, F.M. 2011. “Apertium: a free/open-source platform for rule-based machine translation”. Machine Translation 25(2)127-144.

[5] Salimzyanov, I., Washington, J.N., Tyers, F.M. “A free/open-source Kazakh-Tatar machine translation”. Proceedings of MT Summit XIV (Nice, France, 4–6 September 2013), accepted.

[6] Karlsson, F., Voutilainen, A., Heikkilä, J., Anttila, A. 1995. Constraint Grammar: A Language-Independent System for Parsing Unrestricted Text. Mouton de Gruyter, Berlin.

[7] Сундетова А.М., Кәрібаева А.С., Апертиум платформасындағы Ағылшын–Қазақ машиналық аудармашы үшін екітілді сөздікті құру. Материалы международной научно-практической конференции «Применение информационно-коммуникационных технологий в образовании и науке», посвященной 50-летию Департамента информационно-коммуникационных технологий и 40-летию кафедры «Информационные системы» КазНУ им. аль-Фараби. 22 ноября 2013г. – Алматы: Қазақ Университеті, 2013. – С.53-57.

[8] Sundetova A., M.L. Forcada, A. Shormakova, A.Aitkulova, Structural transfer rules for English-to-Kazakh machine translation in the free/open-source platform Apertium. Компьютерная обработка тюркских языков**.** Первая международная конференция: Труды. – Астана: ЕНУ им. Л.Н. Гумилева, 2013. – С. 317-326.

[9] Online-translator «Sanasoft»: http://www.sanasoft.kz/c/ru/node/47 (in Russian), http://www.sanasoft.kz/c/kk/node/53 (in Kazakh).

[10] Online-translator «Trident: http://www.translate.ua/us/on-line;

[11] Regression tests. http://wiki.apertium.org/wiki/English_and_Kazakh/Regression_tests

LEXICAL SELECTION IN MACHINE TRANSLATION OF RUSSIAN-TO-KAZAKH

D.Rakhimova, M.Abakan
Laboratory of Intelligent Information Systems, Institute of Mathematics and Mechanics,
al-Farabi Kazakh National University, Almaty, Kazakhstan
di.diva@mail.ru; mayerabak@gmail.com;

ABSTRACT

This article presents a method of resolving lexical ambiguity of words in an automatic text processing for different groups of natural languages that have not marked corpus. The proposed method is based on creating context vectors by which held the semantic analysis of the text. This method has been successfully applied in the machine translation from Russian into Kazakh. The practical results presented.

1 Introduction

Resolution of lexical ambiguity - is the establishment of the word meaning in some context [1]. For a human the process of eliminating ambiguity is largely subconscious and does not present any difficulties. Despite this, as a computational problem, the task of resolving lexical ambiguity is a difficult task. The resolution of lexical ambiguity is used to improve the accuracy of classification methods and clustering of texts, increasing the quality of machine translation, information retrieval and other applications.

The task of resolving lexical ambiguities (word sense disambiguation) occurred in 50-ies of the last century as a subtask of machine translation. Since then, researchers have proposed a great number of methods to solve this problem, but it remains more relevant today. The resolution of lexical ambiguity is one of the central tasks of text processing. To solve the problem it is necessary to identify possible meanings of words and the relationships between these meanings and the context in which words were used. At the moment, the main source of meanings are dictionaries and encyclopedias. Thesauri, semantic networks and other specialized structures are created by linguists to establish the relationships between the meanings of the words. However, creating such resources requires an enormous effort.

2 Overview of scientific works and approaches

The importance of task resolution of lexical ambiguity is difficult to overestimate. The electronic library ACL (The Association for Computational Linguistics) contains more than 700 articles on this topic [1]. Obviously, the solution of this problem is a prerequisite for a full understanding of natural language. As there is no recognized ways to determine, where the meaning of one word ends and another begins, it is problematic to formalize the task of eliminating the ambiguity.

Next, we will consider existing approaches to the definition of values, context, and comparison methods of different approaches to the resolution of lexical ambiguity.

In 1993-94 [3] David Yarowsky made the observation and determined that the length of microcontext may vary depending on the type of ambiguity.

He suggested that to resolve local ambiguities 3-4 words of context are enough, while for the semantic ambiguities a

larger box, consisting of 20-50 words is needed. Thus, researchers still have not come to a consensus regarding the optimal length of microcontext. Additionally, for the resolution of lexical ambiguity in some works phrases and syntactic relations are used.

So D. Yarowsky [3] found that for the same combinations of two words, the likelihood that the relevant words in the same values ranges from 90-99%.

This observation is used in many modern heuristics works. So, one value for the phrase (one sense per collocation), i.e. the appropriate words in the same phrase must have the same meaning.

Thematic context researches appeared somewhat later than microcontext and for several years was actively discussed in the field of information retrieval [4]. Modern works mainly combine thematic and microcontext approaches.

William Gale and others [5] have improved the accuracy of their method from 86% to 90%, expanding the context of the 12 words in the target environment up to 100 words. In addition, they showed that the importance of words in context decreases with distance from the target word. In their works[6] they showed that in the same thematic contexts the meanings of the corresponding ambiguity words are the same (one sense per discourse).

There is also an approach based on learning on marked blocks. The success of this approach depends on the availability of large annotated collections of texts. Rapid progress in the automatic determination of the parts of speech and syntactic analysis has been made, in particular, due to the large markup enclosures, such as Penn Treebank [7]. Models that derived from the annotated corpus methods of machine learning show good performance in many problems in natural language processing.

It is possible to distinguish two dominant approaches from the set of all existing algorithms for solving problems of lexical ambiguity.

The first approach of lexical ambiguity resolution is based on external sources of knowledge (knowledge-based methods). This approach can be easily adapted to the documents obtained from any source and not tied to a specific language. **The second approach** is based on machine learning. Algorithms based on this approach show good results in comparison with the algorithms presented in the recent literature, however, they require the training on documents similar to the processed further. This is due to the problem of sparseness of language.

3 Description of the context vector method.

Methods based on external knowledge sources have several advantages, so they attract researchers` interests. These methods can be easily adapted to the documents obtained from any source, in contrast to methods based on learning, which is applied only to the words that are available in the marked case. Another important advantage of these methods is that they do not depend on the availability of tagged corpus and can be easily applied to any other languages.

In the current work, the solution of lexical ambiguity of words based on Bag of Words (BoW) model will be proposed. The Bag of words (BoW) model is one of the two methods of representing context feature vector [7] for supervised learning technology.

Another method is a method of vector of colocational features which represents the words left and right of the target word to determine its meaning. **Method context feature vectors** (CV) is an unordered set of a certain length, the most frequent context words generated by processing a certain body of text for the target words. Then for each sense of the target word forms a binary

vector CV. In this model, the text (e.g. a sentence or a document) is represented as a set (multiset) with his words, disregarding grammar and even word order, but keeping many in the form of vectors.

The task of disambiguation in text can be easily represented as a task multivalued mappings:

Let *X and Y* — an arbitrary set. A multivalued mapping from the set X into Y is called every display :

$$F:X\rightarrow\Omega(Y),$$

which we will call this mapping from X in Y. Where each input word $x_i \in T$ of text *T* should be attributed to one of the output values of the classes $m_j, i \in M_i$, where M_i — the set of meanings of the word x_i. F - representation function of multivalued mappings.

To obtain knowledge about the external sources we must have information about the elements of the text (grammar , syntax properties) and relationships between them. However, a full analysis of the text, you can replace the partial. To optimize the analysis of the text we will consider the word context that used only to describe and highlight a specific group of values. As result, we will build a set of meaning vectors of allocating a noun, verb and adjectives groups, for efficiency building a complete semantic mapping for each unit complex word

Lets consider the multivalued mapping for the case when the source language is Russian and the target language is Kazakh. Consider the class of ambiguous words, which are called **lexical homonyms**, i.e. sound and grammatical match different linguistic units, which are not semantically related to each other.

For example, the word "*коса*" - braid, braiding hair, in kazakh "*бұрым*" (hairstyles), and "*коса*" spit - subject to mowing grass, in kazakh "*орақ"* (agricultural tools);

the word "*лук",* onion as plant, in kazakh *"пияз"*or *"лук"* like weapon for throwing arrows , in kazakh "*садақ"*.

Unlike ambiguous words, lexical homonyms do not have subject-semantic relationship, i.e. they have no common semantic features by which you could judge the polisemantism words. In this work, will be considered this kind of multiple meanings of words and will be the method of resolving this issue .

Below is the segment tables of multivalued mappings (m -mappings) for ambiguous words (in this case homonyms)

$$X^m \rightarrow Y^m$$

where $X^m = \{a_k\}$, a_k- initial form of ambiguous words that have the k-th value.

Y^m- represented as a matrix consisting of elements CV , that are corresponding words in context for each a_k values.

$$Y_{ij}^m = b_{ij}\mu_{ij}$$

$$Y_{ij}^m = \{(b_{1j}\mu_{1j}), (b_{2j}\mu_{2j}), (b_{3j}\mu_{3j}),\}$$

where b_{ij} - elements of a particular group of CV, *i=1,3* (where b_1- verb group , b_2-noun group , b_3- adjective group), and for each element is given by the ratio of preference (relativity) μ_{ij} of given element in text, in the following range $0 \leq \mu_{ij} \leq 1$.

If after a full lexical and syntactic analysis of the sentence ambiguous words show up in the text, then on the basis of the approach is determined by the availability of appropriate CV words the context of a set of vectors of the multivalued mapping a_k. If such b_{ij} words of was found, then in accordance with its relativity to one or another meaning $\mathbf{a_k}$ meanings was selected.

Suppose that the word a_k (where k =1,2) have two different meaning values : a_1| a_2. If defined one or more elements from a_1 , the system output will give the required value of a_1.

In some situations, the preferred analysis and selection on the basis of the coefficient $\mu_{ij.}$, it happens when the text includes several

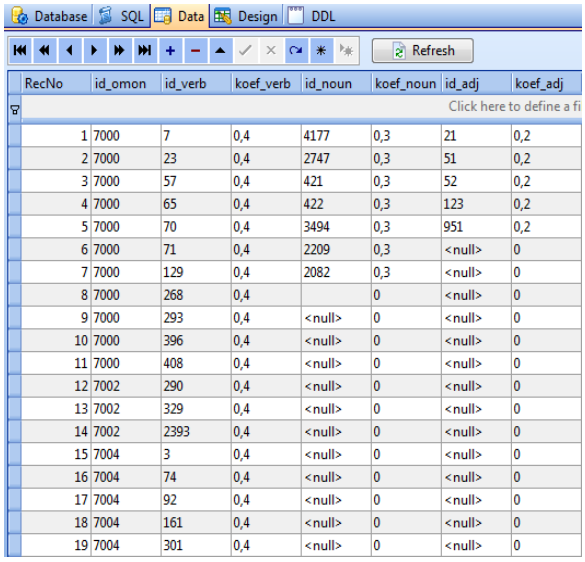
items CV different value.
Lets introduce the new notation $p(a_k)$- that is the number of b_{ij} in sentence for values a_k.
If in the sentence will be determined :
$p(a_1) > p(a_2)$ then will be determined value a_1.; If will be determined the same number of words that are elements of CV $p(a_1)=p(a_2)$, then the decision will be made on the basis of the analysis of the coefficients μ_{ij} , with the help of which will be determined by the weight of the preferred meaning of the word in the context of proposals .
Depending on the grammatical characteristics, communication and relationship between the words for each element b_{ij} we enter a specific value as s coefficient.
For example, the basic steps performed to a particular subject will reveal its essence, therefore, the verb group was assigned the largest value from relatively nominal and prepositional groups. Preference setting to be made on the basis of comparison of the sums of the coefficients of one or another value .

$$S_k=\sum p_i(b_j)*\mu_{ij}$$

where $p_i(b_j)$ -number of elements KV i-th group for values k , i=1,2,3 and j=1,..,n
Using the proposed method, the view function F multivalued mappings can be represented as a set of rules applied to the matrix elements of the context groups, and the coefficients of preference.

4 Practical results

In the implementation of the system of machine translation from Russian into Kazakh language found many difficulties in the description of grammatical rules and organization of data on different levels of the analyzer and generator. Taking into account the representation of the data and grammatical and semantic properties of different languages multivalued mapping was to present tabular data and their attributes [8]. The data for the ambiguous words were presented in m-mappings table, which is represented in Figure-1.



RecNo	id_omon	id_verb	koef_verb	id_noun	koef_noun	id_adj	koef_adj
							Click here to define a fi
1	7000	7	0,4	4177	0,3	21	0,2
2	7000	23	0,4	2747	0,3	51	0,2
3	7000	57	0,4	421	0,3	52	0,2
4	7000	65	0,4	422	0,3	123	0,2
5	7000	70	0,4	3494	0,3	951	0,2
6	7000	71	0,4	2209	0,3	<null>	0
7	7000	129	0,4	2082	0,3	<null>	0
8	7000	268	0,4		0	<null>	0
9	7000	293	0,4	<null>	0	<null>	0
10	7000	396	0,4	<null>	0	<null>	0
11	7000	408	0,4	<null>	0	<null>	0
12	7002	290	0,4	<null>	0	<null>	0
13	7002	329	0,4	<null>	0	<null>	0
14	7002	2393	0,4	<null>	0	<null>	0
15	7004	3	0,4	<null>	0	<null>	0
16	7004	74	0,4	<null>	0	<null>	0
17	7004	92	0,4	<null>	0	<null>	0
18	7004	161	0,4	<null>	0	<null>	0
19	7004	301	0,4	<null>	0	<null>	0

Figure-1. The segment of m-mappings table ambiguous words of the Russian language.

For quick search and filling facilities in m-mappings table presents only the id numbers of the meanings in database dictionary not whole words.
In id_omon column fills the numbers of fundamentals of polysemantic words from the main table of the word; in columns id_verb, id_noun, id_adj respectively filled with the id number of context-related words of certain groups (verb-verbs, noun -nouns, adj- adjectives and adverbs); in columns koef_verb, koef_noun and koef_adj respectively fills the coefficients of preference μ_{ij} for each group.
For example : multivalued word *"лук"* — onion "пияз" or weapon, bow *"садақ"*.

$X(a_k)::=Y(a_1)|Y(a_2)$

X(*лук*)::=Y(*пияз*)|Y(*садақ*)

where

$$Y(a_1)::=\begin{cases} b_1\ (садить, чистить, готовить, ...) \\ \quad (to\ plant, to\ clean,\ to\ cook, ...) \\ b_2\ (овощ, сад, огород, ...) \\ \quad (vegetable, garden, kitchen\ garden, ...) \\ b_3\ (большой, зеленый, сочный...) \\ \quad (big, green, juicy...) \end{cases}$$

$$Y(a_2)::=\begin{cases} b_1\ (стрелять, воевать, биться, ...) \\ \quad (to\ shoot, be\ at\ war, to\ fight, ...) \\ b_2\ (стрела, войн, стрельба, ...) \\ \quad (arrow, wars, firing, ...) \\ b_3\ (тугой, деревянный, удобный, ...) \\ \quad (taut, wooden, comfortable, ...) \end{cases}$$

Figure-2. Elements of CV for ambiguous words "лук"

Here given some examples :
*я купил **лук**.- I bought **onions/ bow**.*
*я купил зеленый **лук**.- I bought green **onions**.*
*я стрелял из **лука**.- I shot a **bow**.*
*я в саду стрелял из **лука**.- I in the garden shot a **bow**.*

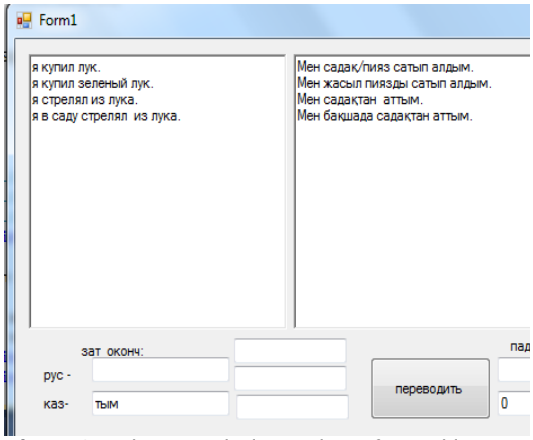


Figure-3. The practical results of machine translation of simple sentence with ambiguity words from Russian into Kazakh language.

In the example, the first sentence is limited in the context of information and in this case it is not clear which the meaning of the word *"лук"* the author means. Undefined b_{ij} – elements of CV. Therefore, the output was thrown wide range of values for post-editing by the user. In the following examples the various forms of relative context of the ambiguous words in the text was shown. In the last sentence defines two different value items CV. By the context of the action *"стрелял"* (*shot*) the proposal relates to the value of a_2, and place *"в саду" (in the garden)* the execution relates to the value of a_1.

In such conflicting situations the number and ratios of preference μ_{ij} of multiple meanings of words are considered.

For deciding the choice will be made the greatest value S_k.
$p_2(b_{ij})=1$ b_{1j} *(стрелял(shot))* $\mu_{1j}=0,4$;
$p_1(b_{ij})=1$ b_{2j} *(в саду(in the garden))* $\mu_{2j}=0,3$;

The proposed method of multivalued mappings and solving problems with multi-tasking words were applied to a simple sentence in the system of machine translation from Russian into Kazakh language. Practical implementation is done in the programming language C#, MS Visual Studio with DB SQLite Expert for 10,000 words units. The comparative analysis was done to test the resolution of the problems of ambiguity in the modern online translators (Sanasoft http://www.sanasoft.kz , Pragma6 http://translate.ua/ru/pragma-6x, Audaru http://audaru.kz) from Russian into Kazakh language. The results of the test show that considered machine translation systems do not determine the ambiguity of words and give one of the options of values to the output language, which often does not conform to the desired sense.

5 Conclusion

For the solution of problems regarding the resolution of lexical ambiguities there were tasked and solved the following tasks:
1. To determine the values for each word, related to the text;
2. To choose the most suitable value of meaning based on the context in which the word exists.
Most of the modern works are based on predefined values: lists of words found in dictionaries, translations into foreign languages, etc. The advantage of this method is an improvement of the good qualities of the classical approach based on external sources of knowledge through the application of the method of CV and multivalued mappings. In contrast to the method of the neighboring words and phrase structures, the method of CV handles all components of the sentence, and not just standing around ambiguous words. Due to this, semantically more complete analysis of the text comes out. Taking into account not only the number of elements of context, but the introduction of the preference factor for each individual item types of context vectors improved the quality of machine translation. This method can be successfully applied in various systems of automatic text processing and semantic search for a variety of natural languages.

Gratitude. This work is carried out under the grant of the Ministry of education and science of the Republic of Kazakhstan.

6 References

[1] Word Sense Disambiguation: Algorithms and Applications (Text, Speech and Language Technology), Ed. by **AGIRRE E., EDMONDS P. G.**.— 1 edition.— Springer, 2007.—November

[2] **YAROWSKY D**., One sense per collocation // HLT '93: Proceedings of the workshop on Human Language Technology.— Morristown, NJ, USA: Association for Computational Linguistics, 1993.— Pp. 266–271.

[3] **TURDAKOV D**., Recommender System Based on User-generated Content // Proceedings of the SYRCODIS 2007 Colloquium on Databases and Information Systems.— 2007.

[4] **GALE W. A., CHURCH K. W., YAROWSKY D.**, A method for disambiguating word senses in a large corpus. // Computers and the Humanitzes.— Vol. 26.— 1993.— Pp. 415–439

[5] **GALE W. A., CHURCH K. W., YAROWSKY D**. One sense per discourse // HLT '91: Proceedings of the workshop on Speech and Natural Language.— Morristown, NJ, USA: Association for Computational Linguistics, 1992.— Pp. 233–237

[6] **RICHENS R. H.**. Interlingual machine translation // Computer Journal.— Vol. 3.— 1958.— Pp. 144–147.

[7] **JURAFSKY D., MARTIN J. M**., Speech and Language Processing // second edition, Pearson Prentice Hall, New Jersey pp.640-644.

[8] **TUKEYEV U.A. , RAKHIMOVA D.R. et al.,** Development of morphological analysis and synthesis for machine translation from Russian into Kazakh using multivalued mapping tables. Computer processing of Turkic languages. First International Conference: Proceedings / Astana L.N.Gumilev ENU Publishing House, 2013, 182-191.(in Kazakh)

ITU Validation Set
for Metu-Sabancı Turkish Treebank

Gülşen ERYİĞİT
gulsen.cebiroglu@itu.edu.tr;

Tuğba PAMAY
pamay@itu.edu.tr;

ABSTRACT

This paper presents the ITU Turkish Dependency Validation Set firstly introduced in 2007 [36] in order to serve as the test set of the CoNLL-XI shared task (shared task of the Conference on Computational Natural Language Learning 2007 [28]). The dataset is available from http://web.itu.edu.tr/gulsenc/treebanks.html and is used by several academic studies so far.

1 Introduction

The Turkish Treebank [1], [20] created by the Middle East Technical University and Sabanci University is available to the researchers since 2003 and it is used by many researchers since then [2], [3], [4], [5], [6], [7], [8], [9], [10], [11], [12], [13], [14], [15], [16], [17], [18], [19], [22], [24], [25], [26], [27]. Although it has some inconsistencies and still continues to be updated with newer versions[1] it served very much in the recent years for the development of the research on dependency parsing of Turkish.

The Turkish treebank is composed of 5635 sentences and annotated with dependency structures.

The modest data size of the treebank has been mentioned in many studies [19], [4]. There is no need to say that the size should be increased for better research on the field,
but we should also state that the small size of the number of words (48K) of this treebank can be actually related to one of the features of the language itself. In the treebank, the average number of words in a sentence is 8.6 which is very low when compared to other languages. This is since in Turkish, the words are sometimes equivalent to a whole sentence in another language which is a result of its agglutinative structure. This property of the language makes look the treebank smaller than it is when compared to the other treebanks having similar number of sentences (refer to [19] for further analysis).

This paper presents the validation set prepared at Istanbul Technical University (ITU) for the Turkish Treebank. The same annotation scheme with the original treebank has been adapted and the sentences are annotated with dependency structures. The presented language resource "ITU Validation Set" which is firstly introduced and used in Conll-XI [27] has been used in many other studies so far. Some of which are [28], [29], [30], [31], [32], [33], [34], [35] . The remaining of the paper first presents the structure of the prepared dataset (Section 2), then its available data formats (Section 3) and finally its differences from the previous versions of the treebank (Section 4).

2 Validation Set

ITU Validation Set contains 300 sentences from 3 different genres (20% article, 20% novels and 60% short stories). The sentences are first analyzed with the morphological analyzer of Oflazer [21] and then multiple morphological analyses are manually disambiguated. The sentences are then manually annotated according to dependency structure. Two annotators worked during the preparation of the dataset. Since, most of the observed inconsistencies on the current treebank is due to the incoherence between

[1] The changes between the versions of the treebank have been explained in [15].

different annotators, during the preparation of the validation set the annotators were charged with different stages of the annotation process; the sentences are first morphologically disambiguated by one annotator then the second annotator double-checked the results of this disambiguation phase and annotated the dependencies simultaneously. We believe that this working style resulted in a viable validation set.

The dependency annotator used a special dependency type to emphasize the collocation structures. We then automatically combined these collocations[2] into single units and reindex the sentences by using scripts.

3 Data Formats

The validation set is available in two different data formats[3]. *XML Data Format* which is the Turkish treebank original data format and *Conll Data* format which is the data format used in the Conll-X (Shared task on on Multi-lingual Dependency Parsing) and Conll-XI (Multilingual Track of the shared task). Please refer to [23] and [4] for the details of these formats. Figure 1[4] and Figure 2 give the representation of the sentence " Her obje bir inceleme konusu olabilir. " *(Each object can be an investigation topic)* with these data formats.

4 Differences from the previous versions

The recent official version of the Turkish treebank is the version used in the Conll-X shared task [4]. This version is available as two subversions (one in XML and one in Conll format)
from the treebank website http://www.ii.metu.edu.tr/~corpus/corpus.html. There is one major difference between these two subversions. The data used in the Conll-X shared task (in Conll format) is actually a variant of the treebank in XML format; some conversions are made on punctuation structures in order to keep consistency between all languages[5]. In Conll-XI, the entire treebank will be used as the training data and the validation set introduced in this paper will be used as the test data.

The treebank which will be used this year differs from the previous year mainly in two points:

- Unlike to Conll-X, for Conll-XI shared task, no conversion is applied to the punctuation structures,
- All the dependencies emanating from and coming to the words with a special stem "değil"[6] have been re-annoted in order to keep consistency on the overall treebank.

Following the changes in the treebank, the validation set is also prepared according to the final structure of the treebank and differs from Conll-X Turkish data and the original treebank on the items listed below.

5 Conclusion

In this paper, a validation set of 300 sentences for the Turkish Treebank has been introduced. The data set has been prepared according to the same annotation style of the original treebank and is publicly available from http://web.itu.edu.tr/gulsenc/treebanks.html.

6 Acknowledgments

The author wants to thank to Prof. Kemal Oflazer for his valuable comments on the development of the validation set and Prof. Joakim Nivre for discussions on the Turkish Treebank.

[2] In the treebank, the words in a collocation have been combined into single units by putting an underscore "_" character in between.
[3] Actually, it is prepared in the original treebank XML format and then converted to Conll format.
[4] The fields "Lem" and "Morph", which are originally available in the treebank format but are empty in its current state, are removed from the figure because of the space limit.
[5] refer to http://nextens.uvt.nl/~conll/software.html# conversion for further discussion
[6] This is a special word which occurs under different part-of-speech categories (Verb and Conj). The annotation manner for this verb is modified in the new version of the treebank.

```
<W IX="1" IG="[(1,"her+Det")]" REL="[2,1,(DETERMINER)]">Her</W>
<W IX="2" IG="[(1,"obje+Noun+A3sg+Pnon+Nom")]" REL="[6,2,(SUBJECT)]">obje</W>
<W IX="3" IG="[(1,"bir+Det")]" REL="[4,1,(DETERMINER)]">bir</W>
<W IX="4" IG="[(1,"inceleme+Noun+A3sg+Pnon+Nom")]" REL="[5,1,(CLASSIFIER)]">inceleme</W>
<W IX="5" IG="[(1,"konu+Noun+A3sg+P3sg+Nom")]" REL="[6,2,(OBJECT)]">konusu</W>
<W IX="6" IG="[(1,"ol+Verb+Pos")(2,"Verb+Able+Aor+A3sg")]" REL="[7,1,(SENTENCE)]">olabilir</W>
<W IX="7" IG="[(1,".+Punc")]" REL="[,( )]">.</W>
```

Figure 1: XML Data Format

```
1 Her       her         Det     Det     _                 2       DETERMINER
2 obje      obje        Noun    Noun    A3sg|Pnon|Nom     7       SUBJECT
3 bir       bir         Det     Det     _                 4       DETERMINER
4 inceleme  inceleme    Noun    Noun    A3sg|Pnon|Nom     5       CLASSIFIER
5 konusu    konu        Noun    Noun    A3sg|P3sg|Nom     7       OBJECT
6 _         ol          Verb    Verb    Pos               7       DERIV
7 olabilir  _           Verb    Verb    Able|Aor|A3sg     8       SENTENCE
8 .         .           Punc    Punc    _                 0       ROOT
```

Figure 2 : Conll Data Format

7 References

[1] **Atalay, N. B., Oflazer, K.,** and **Say, B..** 2003. The annotation process in the turkish treebank. In *Proceedings of the EACL Workshop on Linguistically Interpreted Corpora*.

[2] **Attardi, G..** 2006. Experiments with a multilanguage non-projective dependency parser. In *Proceedings of CONLL-X*, pages 166-170, New York.

[3] **Bick, E..** 2006. LingPars, a Linguistically Inspired, Language-Independent Machine Learner for Dependency Treebanks. In *Proceedings of CONLL-X, pages 171-175, New York.*

[4] **Buchholz, S.,** and **Marsi, E..** 2006. Conll-X shared task on multilingual dependency parsing. In *Proceedings of CONLL-X*, pages 149-164, New York.

[5] **Çakıcı, R.,** and **Baldridge, J..** 2006. Projective and Non-Projective Turkish Parsing. In *Proceedings of the 5th International Treebanks and Linguistic Theories Conference*, pages 43-54, Prague.

[6] **Canisius, S., Bogers, T., Bosch van de, A., Geertzen, J.,** and **Tjong Kim Sang, E..** 2006. *Dependency Parsing by Inference over High-recall Dependency Predictions*, pages 176-180, New York.

[7] **Carreras, X., Surdeanu, M.,** and **Marquez, L..** 2006. Projective dependency parsing with perceptron, In *Proceedings of CONLL-X,* pages 181-185, New York

[8] **Chang, M.W., Do, Q.,** and **Roth, D..** 2006. A pipeline model for bottom-up dependency parsing. In *Proceedings of CONLL-X*, pages 186-190, New York.

[9] **Cheng, Y., Asahara, M.,** and **Matsumoto, Y..** 2006. Multi-lingual dependency parsing at NAIST. In *Proceedings of CONLL-X,* pages 191-195, New York.

[10] **Corston-Oliver, S.,** and **Aue, A..** 2006. Dependency parsing with reference to Slovene, Spanish and Swedish. In *Proceedings of CONLL-X,* pages 196-200, New York.

[11] **Dreyer, M., Smith, D. A.**, and **Smith, N. A..** 2006. Vine parsing and minimum risk reranking for speed and precision. In *Proceedings of CONLL-X,* pages 201-205, New York.

[12] **Eryiğit, G.,** and **Oflazer, K..** 2006. Statistical dependency parsing of Turkish. In *Proceedings of EACL'06*, pages 89-96, Trento.

[13] **Eryiğit, G., Adalı, E.,** and **Oflazer, K..** 2006a. Türkçe cümlelerin kural tabanlı bağlılık analizi (Rule-based dependency parsing of Turkish sentences). In *Proceedings of the 15th Turkish Symposium on Artificial Intelligence and Neural Networks*, pages 17-24, Muğla.

[14] **Eryiğit, G., Nivre, J.,** and **Oflazer, K..** 2006b. The incremental use of morphological information and lexicalization in data-driven dependency parsing. *Computer Processing of Oriental Languages, Beyond*

the Orient: The Research Challenges Ahead, Springer, LNAI 4285:498-507.

[15] **Eryiğit, G..** 2006. Türkçenin Bağlılık Ayrıştırması (Dependency Parsing of Turkish). Ph.D. thesis, Istanbul Technical University, Istanbul.

[16] **Johansson, R.** and **Nugues P..** 2006. Investigating multilingual dependency parsing. In *Proceedings of CONLL-X*, pages 206-210, New York

[17] **Liu, T., Ma, J., Zhu, H.,** and **Li S..** 2006. Dependency parsing based on dynamiz local optimization. In *Proceedings of CONLL-X,* pages 211-215, New York.

[18] **McDonald, R., Lerman, K.,** and **Pereira, F..** 2006. Multilingual dependency analysis with a two-stage discriminative parser. In *Proceedings of CONLL-X*, pages 216-220, New York.

[19] **Nivre, J., Hall, J.,Nilsson, J., Chanev, A., Eryiğit, G., Kübler, S., Marinov, S.,** and **Marsi, Erwin..** 2007. MaltParser: A language-independent system for data-driven dependency parsing. *Natural Language Engineering Journal, 13*(1):1-41.

[20] **Oflazer, K., Say, B., Hakkani-Tür D. Z.,** and **Tür, G..** 2003. Building a Turkish treebank. In A. Abeillé, editor, *Treebanks: Building and Using PArsed Corpora*, pages 261-277. Kluwer, London.

[21] **Oflazer, K.,** 1994. Two-level description of Turkish morphology. *Literary and Linguistic Computing, 9*(2):137-148.

[22] **Riedel, S., Çakıcı, R.,** and **Meza-Ru,z, I..** 2006. Multi-lingual dependency parsing with incremental integer linear programming. In *Proceedings of CONLL-X*, pages 226-230, New York.

[23] **Say, B..** 2004. Metu-sabancı turkish treebank user guide.

[24] **Schiehlen, M.,** and **Spranger, K..** 2006. Language independent probabilistic context-Free parsing bolstered by machine learning. In *Proceedings of CONLL-X*, pages 231-235, New York.

[25] **Shimizu, N..** 2006. Maximum spanning tree algorithm for non-projective labeled dependency parsing. In *Proceedings of CONLL-X*, pages 241-245, New York.

[26] **Wu, Y.C., Lee, Y.S.,** and **Yang, J.C.**. 2006. The exploriation of deterministic and efficient dependency parsing. In *Proceedings of CONLL-X*, pages 241-245, New York.

[27] **Yüret, D..** 2006. Dependency parsing as a classification problem. In *Proceedings of CONLL-X*, pages 246-250, New York.

[28] **Nivre, J., Hall, J., Kübler, S., McDonald, R., Nilsson, J., Riedel S.,** and **Yüret, D.** 2007. The CoNLL 2007 shared task on dependency parsing. In *Proceedings of the CoNLL Shared Task Session of EMNLP-CoNLL*, pages 915-932. Prague.

[29] **Meral, H. M., Sankur, B., Özsoy, A. S., Güngör, T.,** and **Sevinç, E.** 2009. Natural language watermaking via morphosyntactic alterations. Retrieved from DOI: 10.1016/j.csl.2008.04.001

[30] **Eryiğit, G., İlbay, T.,** and **Can, O. A.** 2011. Multiword expressions in statistical dependency parsing. In *Proceedings of the Second Workshop on Statistical Parsing of Morphologically Rich Languages(SPMRL)*, pages 45-55. Dublin, Ireland.

[31] **Eryiğit, G.**. The impact of automatic morphological analysis & disambiguation on dependency parsing of Turkish. 2012.

[32] **Çetinoğlu, Ö.,** and **Kuhn, J.** 2013. Towards joint morphological analysis and dependency parsing of Turkish. In *Proceedings of the Second International Conference on Dependency Linguistics(DepLing)*, pages 23-32. Prague.

[33] **Goenaga, I., Ezeiza, N.,** and **Gojenola, K.** 2013. Exploiting the Contribution of Morphological Information to Parsing: the BASQUE_TEAM system in the SPRML'2013 Shared Task.In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically Rich Languages*, pages 71-77. Seattle, Washington, USA.

[34] **Durgar El-Kahlout, İ., Akın, A.A.,** and **Yılmaz, E.** 2014. Initial explorations in two-phase Turkish dependency parsing by incorporating constituents. In *First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 82-89. Dublin, Ireland.

[35] **Çetinoğlu, Ö.** Turkish Treebank as a gold standard for morphological disambiguation and its influence on parsing.

[36] **Eryiğit, G**. 2007. ITU Validation Set for Metu-Sabancı Turkish Treebank.