Dependency Parsing Based Treebank for Kyrgyz Language

İbrahim BENLİ[1*], Bakyt SHARSHEMBAEV[2]

[1]Department of Computer Engineering, Faculty of Engineering, Kyrgyzstan-Turkey Manas University, Bishkek, Kyrgyzstan

[2] Department of Computer Engineering, Faculty of Engineering, Kyrgyzstan-Turkey Manas University, Bishkek, Kyrgyzstan

[1] ibrahimbenli@hotmail.com, [2]bakyt.sharshembaev@manas.edu.kg

Abstract

UD Kyrgyz-KTMU is the first dependency parsing based treebank in Kyrgyz language prepared for participation in the Universal Dependencies project. The overall purpose of the study is to create a model for developing a natural language processing infrastructure for the Kyrgyz language. Dependency parsing identifies syntactic relationships between words in a sentence to extract its grammatical structure. Recent years have witnessed notable advancements in syntactic parsing of natural language, predominantly employing data-driven or grammatical methods. Kyrgyz is characterized by free word order, facilitating its analysis through dependency parsing examples. We used machine learning-based UDPipe as a POS tagger, lemmatizer, and dependency parser to train a sample Kyrgyz language model on CoNLL-U formatted data. We used three methods to perform dependency parsing evaluation. As stated in the results of our study, Unlabeled Attachment Score (UAS) and Labeled Attachment Score (LAS) scores were close to each other when compared with newly created treebanks of fewer than 20,000 words.

Keywords: Dependency parsing, Kyrgyz language, natural language processing, treebanks, universal dependencies.

1. Introduction

UD Kyrgyz-KTMU represents the inaugural dependency parsing-based treebank established for the Kyrgyz language, specifically designed for integration into the Universal Dependencies project. The primary objective of this research endeavor is to lay the foundation for constructing a robust natural language processing framework tailored to the nuances of Kyrgyz linguistic structures. Universal Dependencies[1] (UD) is a framework for coherent explanation of grammar (parts of speech, morphological features, and syntactic dependencies) in different human languages. UD is an open source community effort in which more than 300 contributors have created nearly 200 treebanks in over 100 languages.

UD is a project that develops cross-language coherent treebank descriptions for many languages in order to facilitate multilingual parser development, cross-language learning, and research and parsing from a language typology perspective.

The annotation scheme is based on developments in the Universal Stanford Dependencies [1], Google Universal Part-of-Speech tag set [2] and the study of format syntactic tag sets [3].

Dependency Parsing is a process that identifies syntactic dependencies between words in a sentence, aiming to extract the grammatical structure of the sentence. Figure 1 illustrates an example sentence definition based on dependency parsing in English.

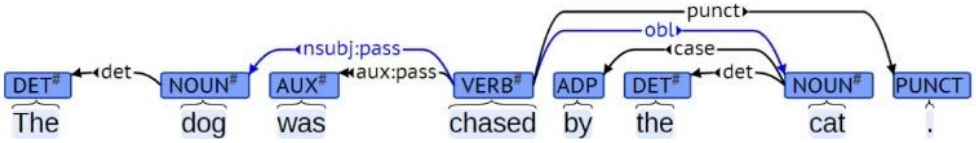

Figure 1 : An example sentence definition based on dependency parsing in English

It is estimated that the Turkic languages are spoken by more than 170 million speakers [4]. However, most studies in the field of Natural Language Processing (NLP) for Turkic languages have been conducted for Turkish [5]. The Kyrgyz language, which we conducted in the Dependency Parsing study, belongs to the Turkic languages in origin. In terms of syntactic structure and morphology, it has similarities with other Turkic languages (Kazakh, Uzbek, Tatar, Turkish) [6].

The Kyrgyz language also has a relatively free word order in terms of syntactic features, which is constrained by elements of speech and knowledge structures like Turkish [7]. For example, a simple sentence "Aydana saw Nurbek" in English can be expressed in Kyrgyz with the same meaning, using six different word orders.

a. *Айдана Нурбек-ти көр-дү.* (SOV 48%)
“Aydana saw Nurbek”
b. *Нурбек-ти Айдана көр-дү.* (OSV 8%)
c. *Айдана көр-дү Нурбек-ти.* (SVO 25%)
d. *Нурбек-ти көр-дү Айдана.* (OVS 13%)

[1] https://universaldependencies.org

e. *көр-дү Айдана Нурбек-ти*. (VSO 6%)
f. *көр-дү Нурбек-ти Айдана*. (VOS <1%) adapted by Hoffman [8].

The relative frequencies of these different word sequences are determined and presented alongside each sample. The most commonly used word order used in simple transitive sentences in Kyrgyz is Subject-Object-Verb. But all six permutations of a transitive sentence are grammatically correct. This variation of word order within a sentence is called local mixing. Slobin and Bever found that 52% of the transitive sentences were not in the general SOV word order in their study using 500 spontaneous speech expressions [9]. Therefore, free word order is a consideration for modeling speeches in the Kyrgyz native language.

Example Sentence in Kyrgyz: "Кыз китепти окуштурду." (The girl read the book.)

SOV : "Кыз китепти окуштурду." (The girl read the book.)

Dependencies:

- nsubj(окуштурду, кыз) (subject: girl)
- obj(окуштурду, китепти) (object: book)
- root(ROOT, окуштурду) (root: read)

OSV : "Китепти кыз окуштурду."

Dependencies:

- obj(окуштурду, китепти) (object: book)
- nsubj(окуштурду, кыз) (subject: girl)
- root(ROOT, окуштурду) (root: read)

VOS : "Окуштурду китепти кыз."

Dependencies:

- root(ROOT, окуштурду) (root: read)
- obj(окуштурду, китепти) (object: book)
- nsubj(окуштурду, кыз) (subject: girl)

The main challenge with free word order in Kyrgyz lies in disambiguating dependencies based solely on word order. In the examples above, the role of each word (subject, object, verb) changes depending on its position within the sentence. Dependency parsers trained on languages with rigid word order (like English) may struggle to accurately parse sentences with free word order due to increased ambiguity. Annotators must rely on additional linguistic features (such as case markings, agreement patterns, and semantic context) to accurately annotate and parse sentences with flexible word order in Kyrgyz.

One critical consideration is the suitability of various syntactic representations for different language types. Most studies in English linguistics have employed constituency-based examples influenced by data from the Penn Treebank [10].

Consistent tagging of words in training data is crucial for accurate model creation. In this study, we utilized data in CoNLL-U format to train the model. In the CoNLL-U format, word dependencies within sentences are defined under the "DEPREL" (Dependency Relations) title. This format allows for the indication of the word order in sentences, which words qualify each other, and their interdependencies.

Hierarchical trees serve as a graphical method to display dependencies. Figure 2 illustrates examples of hierarchical tree structures in both Kyrgyz and English. We utilized hierarchical trees to verify dependencies.

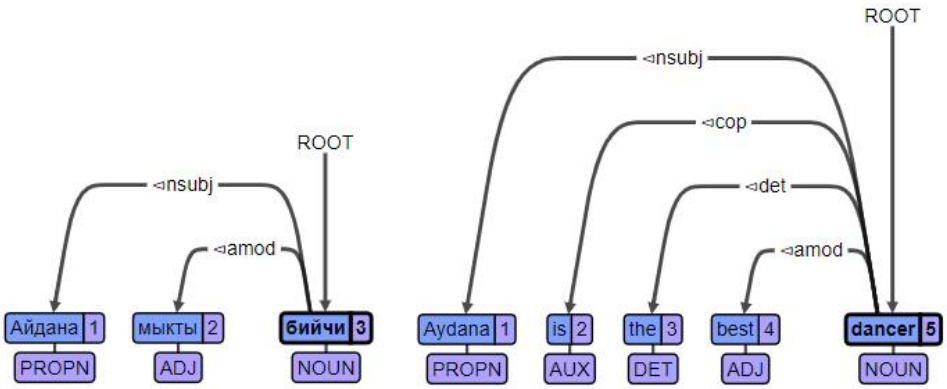

Figure 2 : Hierarchical Tree Structure Example in Kyrgyz and English

In these examples, a manual dependency annotation tool called “UD Annotatrix”, which supports the editing of POS tags and dependency relations with a drag and drop interface, is used to obtain the graphical tree [11].

2. Related Works

Dependency parsing based treebank for more than 100 languages is available at UD official web page. There is no treebank in the Kyrgyz language but UD v2.11 includes some Turkic treebanks. There is one UD treebank in Kazakh with 10K words [12], Uyghur with 40K words[13] and Tatar with 2K words[14]. There is also a treebank called UD Old Turkish Tonqq, which consists of 20 sentences and 158 words [15]. Turkish, which has 9 different UD treebanks, has 736K words. Other than Turkish, resources for Turkic languages are insufficient in UD v2.12.

Sulubacak et al. worked with 57K words with the Turkish-IMST Treebank they developed and reached 75.3% results for labeled attachment score (LAS) and 83.7% for unlabeled attachment score (UAS) [16]. When we examine several treebanks with smaller than 20K words, Lynn et al. obtained the results of LAS 63.3% and UAS 73.3% with Irish Treebank [17], Tyers et al. reached LAS 64.9% and UAS 77.0% with Kazakh-KTB Treebank [12] and Ishola et al. attained LAS 64.9% and UAS 71.8% with Yoruba Treebank [18]. Arnardóttir et al. conducted a study on Icelandic, achieving LAS and UAS scores of 55.29% and 63.03%, respectively, utilizing UDPipe [19]. In contrast to prior investigations, Özateş et al. achieved LAS of 77.65% and UAS of 82.58% employing the BERT model in their analysis of an Ottoman Turkish tree bank comprising 100 sentences [20]. Blaschke et al. employed various NLP techniques, including Stanza, BERT, and UDPipe, to analyze a Multi-Dialectal Bavarian UD Treebank (GSD). Among these, the UDPipe models consistently achieved the highest scores across all metrics. Specifically, the most effective

model was the UDPipe version trained on GSD, demonstrating LAS and UAS scores of 65.79% and 79.60%, respectively [21]. Table 1 shows the LAS and UAS results for some sample languages from CoNLL 2017 Shared Task [22].

Table 1: Results of CoNLL 2017 Shared Task

Language	UAS	LAS
Ancient_Greek	66.91	61.65
Bulgarian	91.86	87.56
Danish	83.82	81.13
English-LinES	83.36	80.51
French-Sequoia	88.11	86.66
Korean	68.10	62.06
Russian	83.73	80.84
Vietnamese	69.63	66.22

3. Method

3.1 Data Set

We selected 770 sentences to train and develop the Kyrgyz UDPipe model, incorporating a diverse range of sources. Specifically, 600 sentences were sourced from Kyrgyz news sites, while 170 sentences were drawn from Kyrgyz stories and novels. The average sentence length within our dataset is 9 words, totaling 6449 words and 7458 tokens. To extract data from news websites, we utilized the Python BeautifulSoup Library (Hajba, 2018) . The UD Kyrgyz-KTMU treebank is published in UD v2.12, and all utilized data are publicly accessible.

3.2. Related NLP Techniques

UDPipe[2] is a single C++ tool that includes a POS tagger, lemmatizer, and a dependency parser. UDPipe is released under the Mozilla Public License (MPL) and is available from the UDPipe homepage or directly using the fixed web address. Training codes are part of the UDPipe version.

Approximately 100 different natural languages can be analyzed using the software tool and service UDPipe, which analyzes text up to the dependency syntax level. Users specify the output format, input text files and desired features such as tokenization, segmentation, morphological analysis, lemmatization, POS tagging, dependency parsing.

The UDPipe software enables training on any language for which a CoNLL-U treebank is accessible, including all of the UD corpora. All trained models are stored in a single file. It is possible to train only a selected part of the dataset. In addition to the UDPipe binary,

[2] http://ufal.mff.cuni.cz/udpipe

libraries developed for many programming languages can also be used. Libraries are available in Java, Python, Perl, C#, R programming languages [23]. In this study, model training processes were carried out for the R language using the CRAN - Package udpipe library[3].

BERT (Bidirectional Encoder Representations from Transformers), which approaches NLP studies with a different perspective, was published by Google in 2019 [24]. BERT is a 12-layer deep neural network model that has been trained to understand language using self-supervised training. After the BERT architecture is trained, it also offers sentence representations and word-level representations. BERT is not comparable to word2vec, but both can be used to represent text data. BERT, unlike word2vec, is not a placement model. It is a language representation model that learns to calculate contextual word representations (embeddings), but also explicitly presents sentence representations.

Therefore, unlike the word2vec model, which is basically a recorded search of words and related vectors, tasks at both the token and sentence level can be performed using a pre-trained BERT model.

ELMo (Embeddings from Language Models) is a groundbreaking deep contextualized word representation model. ELMo is designed to capture complex linguistic features by leveraging the internal states of a deep bidirectional language model based on LSTM (Long Short-Term Memory) networks. Unlike traditional word embeddings that assign a fixed vector to each word, ELMo generates contextual embeddings that vary based on the surrounding context in which words appear within sentences. ELMo embeddings have demonstrated significant improvements in various natural language processing tasks, including sentiment analysis, named entity recognition, and question answering, by capturing syntactic and semantic nuances at different levels of linguistic abstraction [25].

OntoLex, also known as the Lexicon Ontology or OntoLex-Lemon, is a standard ontology developed for representing lexical resources and linguistic knowledge in the Semantic Web framework [26]. The OntoLex framework provides a formal model for describing lexical entries, word senses, and related linguistic information using RDF and OWL semantics [27]. One of the key strengths of OntoLex is its alignment with other ontological models and semantic standards, enabling seamless interoperability and integration of lexical resources across diverse computational systems [28]. Senuma and Aizawa constructed a dependency parsing model encompassing 10,000 words for Ainu, leveraging Ontolex [29].

The primary objective of our research was to develop a comprehensive dependency parsing-based treebank for the Kyrgyz language within the Universal Dependencies (UD) framework. Our focus was on creating a resource that facilitates morphosyntactic annotation and syntactic analysis, which are essential components of NLP tasks specific to Kyrgyz. UDPipe was selected for its effectiveness in processing natural language text, including part-of-speech tagging, lemmatization, and syntactic parsing. The tool offers pre-trained models specifically tailored for morphosyntactic annotation, which aligned closely with the tasks required for developing the Kyrgyz treebank.

[3] https://cran.r-project.org/web/packages/udpipe/index.html

3.2.1. UDPipe Tokenizer

In UD and CoNLL-U files, text is structured into multiple levels. A document consists of paragraphs containing (possibly partial) sentences composed of sequences of tokens. Therefore, the original text can be reconstructed not as a simple sequence of words, but as a sequence of tokens separated by appropriate spaces.

Sentence segmentation and tokenization are jointly performed using a single-layer, bidirectional GRU (Gated Recurrent Units) network that predicts, for each character, whether it marks the end of a sentence, the end of a token, or neither. Spaces are generally excluded from tokens, allowing the network to accurately predict sentence and token boundaries [22].

By using the *SpaceAfter=No* feature, which indicates that a given token was not followed by a space separator in the original text, the CoNLL-U format, which is used by UD treebanks, enables reconstruction of the original pre-tokenized text [23].

3.2.2. UDPipe Tagger

UDPipe generates several triples (UPOS, XPOS, FEATS) for each word based on its last four characters, and an average perceptron tagger with a fixed set of features eliminates the ambiguity of the generated tags. UDPipe generates (lemma rule, UPOS) pairs; where the lemma rule creates a lemma from a word by removing some prefixes and suffixes and adding the new prefix and suffix. To generate the correct lemma rules, the predictor generates the results not only by the last four characters of a word, but also by using the word prefix. Again, disambiguation is performed by an average perceptron tagger [22].

3.2.3 UDPipe Dependency Parcing

The embeddings FORM, UPOS, FEATS, and DEPREL are used by the parser. All embeddings are initialized randomly, updated throughout training, and the form embeddings are precomputed with word2vec using the training data. UDPipe calculates as many network operations as possible in advance for input embeddings. However, to keep memory requirements and load times reasonable, UDPipe does this only for the 1000 most frequently used placements of each type. In UD not allow sentences with multiple roots. So UDPipe generate only one root node and use the root dependency relation only for this node[22].

3.3. Annotation Guidelines for Kyrgyz [4]

3.3.1. Tokenization and Word Segmentation

UD annotation is based on a lexicographical view of syntax, which means that there are dependency relationships between words. Therefore, morphological properties are encoded as properties of words, and there is no attempt to separate words into morphemes. In Kyrgyz, words are mostly separated by whitespace characters. According to typographical rules, many punctuation marks are attached to a neighboring word. These are normally tokenized as separate tokens (words), with the following exceptions:

- The period that marks an abbreviation is part of the abbreviation token: млн.

[4] https://universaldependencies.org/ky/index.html

- The hyphen that attaches a morphological suffix to a number is not a token separator: 200-ге
- There are a few instances of multi-word tokens that are segmented to individual syntactic words. 250’дөн

3.3.2. Morphology

Kyrgyz has a rich inflectional and derivational morphology. Nouns in Kyrgyz take a number of case endings that change based on vowel harmony. Question suffixes are written adjacent to the word in Kyrgyz language. –бы

There is no grammatical gender in Kyrgyz. The two values of the Number feature are ‘Sing’ and ‘Plur’. For NOUN, PROPN and ADJ, only the ‘Plur’ value is used if the plural suffix is present; the singular is unmarked and unannotated. Pronouns (PRON) have both values and they are treated as lexical, that is, the plural pronoun has its own lemma, distinct from the corresponding singular pronoun. Case has 7 possible values: ‘Nom’, ‘Gen’, ‘Dat’, ‘Acc’, ‘Loc’, ‘Abl’, ‘Ins’. It occurs with the nominal words, i.e., NOUN, PROPN, PRON, ADJ, NUM, as well as gerunds and participles (VERB, AUX).

Degree applies to adjectives (ADJ) and adverbs (ADV) and has only one value: ‘Cmp’. The basic (positive) form is unmarked and unannotated. Polarity applies to verbs (VERB, AUX) and has only one value: ‘Neg’. The basic (positive) form is unmarked and unannotated.

Finite verbs are normally annotated as the habitual Aspect (‘Perf’). Other values (‘Imp’, ‘Prog’) can be observed with infinitives and converbs. Finite verbs always have one of five values of Mood: ‘Ind’, ‘Imp’, ‘Opt’, ‘Pot’ or ‘Des’. The conditional mood (‘Cnd’) is only used with conditional converbs. Verbs in the indicative mood always have one of three values of Tense: ‘Past’, ‘Pres’, ‘Fut’. The future tense (‘Fut’) may occur with participles. The Evident feature (evidentially) distinguishes first-hand past tense (‘Fh’). There is one value of the Voice feature: ‘Pass’. The basic (active) form of the verb is unmarked and unannotated.

3.3.3. Syntax

The syntactic annotation in the UD schema involves establishing dependency relationships between words. The fundamental dependency representation, illustrated in Figure 3, constructs a tree structure where one word serves as the head of the sentence, depending on a conceptual ROOT, and all other words in the sentence are dependent on another word.

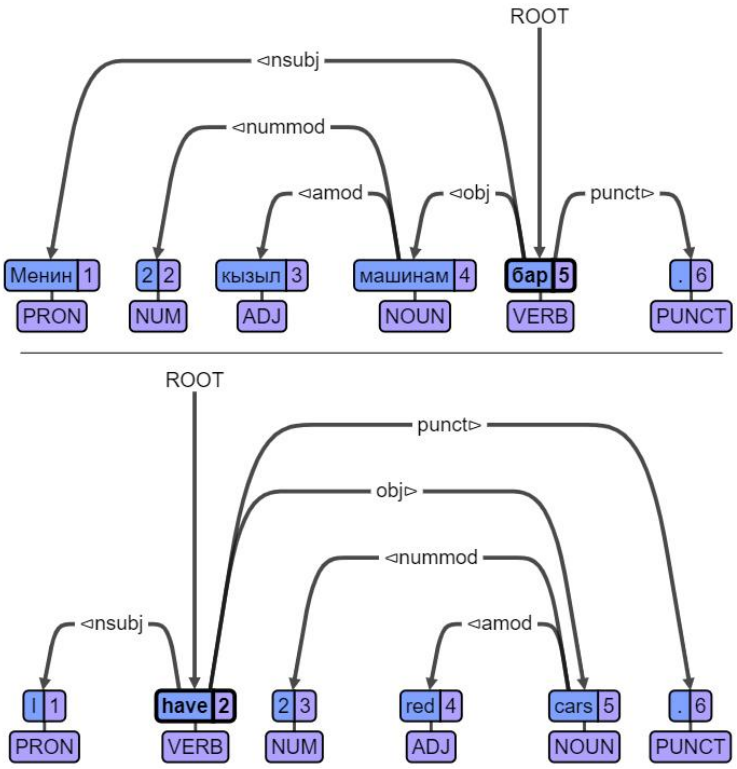

Figure 3: Dependency Representation of "*Менин 2 кызыл машинам бар (*I have 2 red cars*).*" clause in Kyrgyz and English.

A nominal subject (nsubj) refers to a noun phrase in the nominative case, without an adposition. In cases where a subordinate clause serves as the subject, it is labeled as 'csubj'. An object (obj) represents a noun phrase without an adposition, typically in the accusative case, although it can also appear in the nominative or dative case.

3.4. Conversion to CoNLL-U Format

The data selected for conversion to CoNLL-U format underwent manual annotation for the first 1000 words. Subsequently, a model was trained using the R language with this annotated training data. Using the trained model, the next 1000 words were automatically tagged and then validated manually. During preprocessing, sentences were kept in their original form without altering punctuation marks, capitalization, or special characters. Table 2 provides an overview of the conversion process to CoNLL-U format implemented in our work.

Table 2: CoNLL-U Conversion Process

LABEL	PROCESS
Lemmas	Automatically annotated, Manual correction done.
UPOS	Automatically annotated, Manual correction done.
XPOS	Automatically annotated, Manual correction done.

LABEL	PROCESS
Features	Automatically annotated, Manual correction done.
Relations	Automatically annotated, Manual correction done.

3.5. Part of Speech Tagging

The grammatical definition of the word (Subject, Verb, Adjective, Adverb etc.) in CoNLL-U format is defined under the UPOS heading. There are 17 definable universal POS tags in UD v2.0. 14 POS tags were used in the UD Kyrgyz-KTMU treebank. These tags mark the base segments of linguistic categories. Figure. 4 shows tokenization and annotation of "*Сенин балдарын мектепке кетишти (*Your children went to school*).*" clause in Kyrgyz and English.

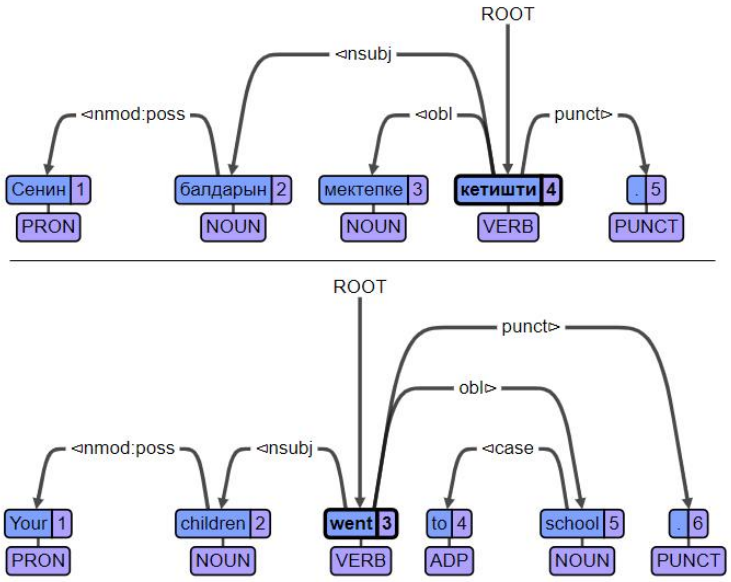

Figure 4: Tokenization and Annotation of "*Сенин балдарын мектепке кетишти (*Your children went to school*).*" Clause in Kyrgyz and English.

To distinguish additional lexical and grammatical features of words, definitions have been made under XPOS and Features. Table 3 shows the number of POS and tokens in the UD Kyrgyz-KTMU treebank.

Table 3: POS and Tokens

POS	Number of Tokens
ADJ	310
ADP	9
ADV	308

POS	Number of Tokens
AUX	55
CCONJ	205
DET	22
NOUN	2698
NUM	420
PRON	179
PROPN	670
PUNCT	1060
SCONJ	2
SYM	1
VERB	1512

3.6. Universal Dependency Relations

There are 37 universal syntactic relations in UD v2. UD v2 is a revised version of the relationships originally described in "Universal Stanford Dependencies: A cross-linguistic typology". [30]

Our study has 25 universal syntactic relations. Figure 3 and Figure 4 also shows examples of Dependency parsing. Table 4 lists the dependency parsing parameters and statistics of UD Kyrgyz-KTMU.

Table 4: UD parsing parameters and statistics of UD Kyrgyz-KTMU

Dependency Relation	Dependency Relation	Number of Tokens
acl	adjectival clause	173
advcl	adverbial clause modifier	214
advmod	adverbial modifier	270
amod	adjectival modifier	360
aux	auxiliary	34
case	case marking	75

Dependency Relation	Dependency Relation	Number of Tokens
cc	coordinating conjunction	108
ccomp	clausal complement	247
compound	compound	329
conj	conjunct	84
cop	copula	8
csubj	clausal subject	40
det	determiner	23
discourse	discourse element	1
fixed	fixed multiword expression	57
flat	flat multiword expression	15
mark	marker	38
nmod	nominal modifier	1509
nsubj	nominal subject	668
nummod	numeric modifier	272
obj	object	274
obl	oblique nominal	688
parataxis	parataxis	20
punct	punctuation	1043
root	root	781

3.7. Word2vec

Word2Vec is a word vector representation method based on Artificial Neural Networks developed by Mikolov et al[31]. In this method, firstly words are converted to vectors. The

distances between the word vectors are calculated and a connection is established between the words.

A model trained with a large text dataset generates a unique vector for each word in a high-dimensional space. These vectors exhibit a characteristic where words with similar meanings in the dataset are positioned close to each other in this space. Two methods commonly used to train the Word2vec model are Continuous Bag of Words (CBoW) and Skip-gram [32].

During the training process, pre-trained word vectors were utilized, which were trained on Common Crawl and Wikipedia using fastText [33]. Specifically, the model was trained using CBoW with position-weights in a 300-dimensional space, incorporating character n-grams of length 5, a window size of 5, and 10 negative samples.

3.8. Training Parameters

The UD Kyrgyz-KTMU treebank was trained using the R statistical programming language. The CRAN package 'udpipe' library enabled us to manage and execute tokenization, parts-of-speech tagging, lemmatization, and dependency parsing processes. The default parameters of the udpipe package were used to train the model. Tables 5, 6, and 7 present the specific training parameters used for the UD Kyrgyz-KTMU treebank.

Table 5. UD Kyrgyz-KTMU Training Parameters of Tokenizer

Parameter	Value
batch_size	100
dimension	24
dropout	0.1
early_stopping	1
epochs	100
initialization_range	0.1
learning_rate	0.005

Table 6. UD Kyrgyz-KTMU Training Parameters of Tagger

Parameter	Value
early stopping	0
iterations	20
prefix max	4
provide_lemma	1
suffix rules	8
use_lemma	1
use_xpostag	0

Table 7. UD Kyrgyz-KTMU Training Parameters of Parser

Parameter	Value
batch size	10
embedding_deprel	20
iterations	20
learning rate	0.0200
oracle	dynamic
structured interval	8
system	projective

4. Results and Discussion

UD Kyrgyz-KTMU represents the first Kyrgyz language dependency parsing treebank included in the Universal Dependencies project. During the initial evaluation of this treebank, challenges arose due to limited data, resulting in inconsistent results.

The UD community emphasizes the importance of large datasets to achieve maximum efficiency and accuracy. However, the community recognizes the value of publishing small datasets for new languages without imposing restrictions. For small treebanks, conducting tenfold cross-validation remains beneficial despite the official dataset split for comparing experimental results.

We employed three methods for Dependency Parsing evaluation, following the Universal Dependencies community's guidelines for developing and evaluating a new treebank with less than 20,000 words of data.

In the first method, the dataset was evenly split into 50% test data and 50% training data, each containing 335 sentences.

In the second method, the dataset was divided into 90% training data and 10% test data. This division resulted in 770 sentences consisting of 6,400 words, with 70 sentences allocated to the test set and 700 sentences to the training set.

In the third method, we utilized the K-Fold Cross Validation technique, a commonly employed approach in machine learning studies [34]. This method is particularly suitable for training datasets with fewer than 20,000 words.

For our dataset comprising 770 sentences, we implemented K-Fold Cross Validation by dividing it into 10 subsets, with 90% of each subset used as test data and the remaining 10% as training data. We computed the average evaluation results across all subsets to derive a consolidated outcome.

Table 8 presents the evaluation results obtained using these three methods.

Table 8. Evaluation results of UD Kyrgyz-KTMU

Method	Data set name	Dependency tagger and parser scores		
		UPOSTAG %	UAS %	LAS %
1	-	79.25	75.04	63.82
2	-	84.39	79.29	68.10
3	Set1	81.72	71.69	61.23
	Set2	79.33	73.27	60.93
	Set3	78.77	75.19	61.37
	Set4	81.67	77.10	64.53
	Set5	78.24	81.99	68.44
	Set6	84.16	80.44	70.68
	Set7	84.92	79.47	73.12
	Set8	83.85	77.78	68.63
	Set9	84.79	81.80	71.29
	Set10	85.07	78.09	67.70
	Average	82.18	77.60	66.65

The attachment scores are created to gauge the effectiveness of dependency parsing. The percentage of words with the proper heads or labels makes up the attachment score. UAS and LAS are the two different types of attachment scores. UAS (1) evaluates the accuracy of dependency parsing based solely on the main word, without considering the relation tag. LAS (2), on the other hand, assesses the percentage of words that correctly specify both the syntactic head and the associated dependency tag.

$$UAS = \frac{\text{number of tokens with correct heads}}{number\ of\ tokens} \quad (1)$$

$$LAS = \frac{\text{number of tokens with correct heads and labels}}{number\ of\ tokens} \quad (2)$$

In the first method, the UAS was 75.04% and the LAS was 63.82%. In the second method, the UAS was 79.29% and the LAS was 68.10%. The third method yielded an average UAS of 77.60% and an LAS of 66.65%. Among these methods, the second method (90% training - 10% test split) achieved superior results due to the larger amount of training data used. The higher scores observed in the second method are attributed to the increased volume of training data, which enhanced the model's performance.

Although our model's UAS (approximately 80%) and LAS (approximately 70%) fall slightly short for advanced NLP applications, the dataset provides valuable resources for

future studies on this previously understudied language, leveraging dependency parsing as a foundation. Additionally, sentences annotated with UD Kyrgyz-KTMU can be utilized in NLP research with minimal manual validation, reducing the need for full-scale annotation efforts.

5. Conclusion

When comparing the results of UD Kyrgyz-KTMU, achieving UAS of 79.29% and LAS of 68.10%, with small-sized treebanks such as Irish Treebank [17], Kazakh Treebank [12] and Yoruba Treebank [18], a close resemblance in UAS scores was observed. However, the LAS score for previously annotated data was comparatively lower than that of other treebanks. This disparity can be attributed to consistency issues encountered during the initial annotation process of the dependency parsing-based treebank developed for the Kyrgyz language. As the study progressed and the corpus expanded, discrepancies in labeling similar structures were identified and manually rectified.

Additionally, the resemblance between our training and testing data, derived from limited sources, could contribute to the observed LAS score disparity.

Nivre (2008) emphasized that a dependency parsing-based treebank can achieve acceptable parsing accuracy when trained on a treeset of at least 1,500 sentences. Moving forward, it is recommended to employ a dataset comprising a minimum of 2,000 sentences to enhance the performance of future treebank implementations [35].

6. Future Works

We will focus on improving tagging operations and the consistency of tags in the corpus. Better results will be achieved when consistent annotation processes are complete.

Sulubacak et al. In the new version of the Turkish-IMST model used multiword expression annotations and obtained LAS 75.4% , UAS 83.8% results [36]. No major changes were observed in the results compared to the original version of the model. So we didn't use multiword expression annotations in the UD Kyrgyz-KTMU study. When the number of words is over 20.000, it is considered to identify multiword expressions and compare them with model's original form.

Since the syntax, affix and root structure are similar to other Turkic languages, it is aimed to carry out similar studies for these languages. We will endeavor to release the UD Kyrgyz-KTMU treebank in Universal Dependencies v2.14 with a minimum dataset of 20.000 words.

Currently, there are models in the Kyrgyz language developed with the BERT approach, the majority of which are automatic speech recognition studies[5]

In future studies, our existing data set will be expanded and the focus will be on developing a new model of the Kyrgyz language using the BERT algorithm.

References

[1] M.-C. De Marneffe, C. D. Manning, J. Nivre, and D. Zeman, “Universal Dependencies,” *Assoc. Comput. Linguist.*, vol. 47, pp. 255–308, 2021, doi: https://doi.org/10.1162/COLI_a_00402.

[5] https://huggingface.co/models?search=kyrgyz

[2] S. Petrov, D. Das, and R. Mcdonald, “A Universal Part-of-Speech Tagset,” in *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC’12)*, 2012, pp. 2089–2096.

[3] D. Zeman, “Reusable tagset conversion using tagset drivers,” *Proc. 6th Int. Conf. Lang. Resour. Eval. Lr. 2008*, pp. 213–218, 2008.

[4] A. Menz, “Speakers of Turkic languages: Numbers and countries. In: Eker, Süer & Çelik Şavk, Ülkü (eds.) Tehlikedeki Türk dilleri I = Endangered Turkic Languages I: Kuramsal ve genel yaklaşmalar = Theoretical and general approaches. Ankara, Astana: Uluslararası Türk.” Ankara-Astana, pp. 199–204, 2017.

[5] E. Kuriyozov, Y. Doval, and C. Gómez-Rodríguez, “Cross-lingual word embeddings for turkic languages,” *Lr. 2020 - 12th Int. Conf. Lang. Resour. Eval. Conf. Proc.*, pp. 4054–4062, 2020.

[6] T. K. Toktonaliev, “Ratio Of In Kyrgyz and Korean To The Altaic Language Theory,” *ИЗВЕСТИЯ ВУЗОВ КЫРГЫЗСТАНА*, vol. 1, pp. 204–207, 2018.

[7] İ. Maviş, S. Arslan, and Ö. Aydin, “Comprehension of word order in Turkish aphasia,” *Aphasiology*, vol. 34, no. 8, pp. 999–1015, 2020, doi: 10.1080/02687038.2019.1622646.

[8] B. Hoffman, “The computational analysis of the syntax and interpretation of" free" word order in Turkish,” *IRCS Tech. Reports Ser.*, no. June, p. 130, 1995.

[9] D. I. Slobin and T. G. Bever, “Children use canonical sentence schemas: A crosslinguistic study of word order and inflections,” *Cognition*, vol. 12, no. 3, pp. 229–265, 1982, doi: 10.1016/0010-0277(82)90033-6.

[10] M. P. Marcus, B. Santorini, and M. A. Marcinkiewicz, “Building a large annotated corpus of English: The Penn Treebank.,” *Comput. Linguist.*, vol. 19, no. 2, pp. 313–330, 1993, doi: 10.1162/coli.2010.36.1.36100.

[11] F. M. Tyers, M. Sheyanova, and J. N. Washington, “UD Annotatrix : An Annotation Tool For Universal Dependencies,” in *Proceedings of the 16th International Workshop on Treebank and Linguistics Theories*, 2017, pp. 10–17, [Online]. Available: https://ufal.mff.cuni.cz/tred/.

[12] F. M. Tyers and J. N. Washington, “Towards a Free/Open-source Universal-dependency Treebank for Kazakh,” in *3rd International Conference on Turkic Languages Processing, (TurkLang 2015)*, 2015, pp. 276–289.

[13] M. Eli, W. Mushajiang, T. Yibulayin, K. Abiderexiti, and Y. Liu, “Universal dependencies for {U}yghur,” *Proc. Third Int. Work. Worldw. Lang. Serv. Infrastruct. Second Work. Open Infrastructures Anal. Fram. Hum. Lang. Technol.*, pp. 44–50, 2016, [Online]. Available: https://aclanthology.org/W16-5206.

[14] C. Taguchi, S. Iwata, and T. Watanabe, “Universal Dependencies Treebank for Tatar: Incorporating Intra-Word Code-Switching Information,” *Proc. Work. Resour. Technol. Indig. Endanger. Lesser-resourced Lang. Eurasia within 13th Lang. Resour. Eval. Conf.*, no. June, pp. 95–104, 2022, [Online]. Available: https://aclanthology.org/2022.eurali-1.17.

[15] M. O. Derin and T. Harada, “Universal Dependencies for Old Turkish,” *UDW 2021 - 5th Work. Univers. Depend. Proc. - To be held as part Syntax. 2021*, pp. 129–141, 2021.

[16] U. Sulubacak, M. Gokirmak, F. Tyers, Ç. Çöltekin, J. Nivre, and G. Eryi\ugit, “Universal Dependencies for Turkish,” in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, 2016, pp. 3444–3454, [Online]. Available: https://aclanthology.org/C16-1325.

[17] T. Lynn, J. Foster, M. Dras, and E. U. Dhonnchadha, “Active Learning and the Irish Treebank,” *Proc. Australas. Lang. Technol. Assoc. Work. 2012*, pp. 23–32, 2012.

[18] O. Ishola and D. Zeman, “Yorùbá dependency treebank (YTB),” *Lr. 2020 - 12th Int. Conf. Lang. Resour. Eval. Conf. Proc.*, no. May, pp. 5178–5186, 2020.

[19] \THórunn Arnardóttir, H. Hafsteinsson, A. Jasonarson, A. Ingaon, and S. Steingr\’\imsson, “Evaluating a {U}niversal {D}ependencies Conversion Pipeline for {I}celandic,” *Proc. 24th Nord. Conf. Comput. Linguist.*, pp. 698–704, 2023, [Online]. Available:

https://aclanthology.org/2023.nodalida-1.69.

[20] Ş. B. Özateş, T. E. Tıraş, E. E. Genç, and E. F. Bilgin Taşdemir, "Dependency Annotation of Ottoman Turkish with Multilingual BERT," *LAW 2024 - 18th Linguist. Annot. Work. Co-located with EACL 2024 - Proc. Work.*, pp. 188–196, 2024.

[21] V. Blaschke, B. Kovačić, S. Peng, H. Schütze, and B. Plank, "MaiBaam: A Multi-Dialectal Bavarian Universal Dependency Treebank," *arXiv Prepr. arXiv2403.10293*, 2024.

[22] M. Straka and J. Straková, "Tokenizing, POS tagging, lemmatizing and parsing UD 2.0 with UDPipe," *CoNLL 2017 - SIGNLL Conf. Comput. Nat. Lang. Learn. Proc. CoNLL 2017 Shar. Task Multiling. Parsing from Raw Text to Univers. Depend.*, vol. 2, pp. 88–99, 2017, doi: 10.18653/v1/k17-3009.

[23] M. Straka, J. Hajič, and J. Straková, "UDPipe: Trainable pipeline for processing CoNLL-U files performing tokenization, morphological analysis, POS tagging and parsing," *Proc. 10th Int. Conf. Lang. Resour. Eval. Lr. 2016*, pp. 4290–4297, 2016.

[24] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, no. Mlm, pp. 4171–4186, 2019.

[25] M. Peters, M. Neumann, and M. Iyyer, "Deep contextualized word representations. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies," vol. Volume 1 (, pp. 2227–2237, 2018.

[26] P. Cimiano, E. Montiel-Ponsoda, P. Buitelaar, M. Espinoza, and A. Gómez-Pérez, "A note on ontology localization," *Appl. Ontol.*, vol. 5, pp. 127–137, 2010, doi: 10.3233/AO-2010-0075.

[27] J. P. McCrae, J. Bosque-Gil, J. Gracia, P. Buitelaar, and P. Cimiano, "The Ontolex-Lemon model: development and applications," in *Proceedings of eLex 2017 conference*, 2017, pp. 19–21.

[28] P. Buitelaar, P. Cimiano, J. McCrae, and M. Sintek, "LexInfo: A declarative model for the lexicon-ontology interface," *J. Web Semant.*, vol. 9, no. 1, pp. 29–51, 2011.

[29] H. Senuma and A. Aizawa, "Universal dependencies for AinU," *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, pp. 2354–2358, 2019.

[30] M. C. De Marneffe *et al.*, "Universal stanford dependencies: A cross-linguistic typology," *Proc. 9th Int. Conf. Lang. Resour. Eval. Lr. 2014*, pp. 4585–4592, 2014.

[31] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," *31st Int. Conf. Mach. Learn. ICML 2014*, vol. 4, pp. 2931–2939, 2014.

[32] M. Bilgin, "Kelime Vektörü Yöntemlerinin Model Oluşturma Sürelerinin Karşılaştırılması," *Bilişim Teknol. Derg.*, pp. 141–146, 2019, doi: 10.17671/gazibtd.472226.

[33] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, and A. Joulin, "Advances in pre-training distributed word representations," *Lr. 2018 - 11th Int. Conf. Lang. Resour. Eval.*, no. 1, pp. 52–55, 2019.

[34] D. Anguita, L. Ghelardoni, A. Ghio, L. Oneto, and S. Ridella, "The 'K' in K-fold cross validation," in *20th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN)*, 2012, pp. 441–446.

[35] J. Nivre, "Algorithms for deterministic incremental dependency parsing," *Comput. Linguist.*, vol. 34, no. 4, pp. 513–553, 2008, doi: 10.1162/coli.07-056-R1-07-027.

[36] U. Sulubacak and G. Eryig It, "Implementing universal dependency, morphology, and multiword expression annotation standards for Turkish language processing," *Turkish J. Electr. Eng. Comput. Sci.*, vol. 26, no. 3, pp. 1662–1672, 2018, doi: 10.3906/elk-1706-81.