TurkishDelightNLP: A Neural Turkish NLP Toolkit

Hüseyin Aleçakır
Afiniti
Istanbul
huseyinalecakir@gmail.com

Necva Bölücü
Computer Engineering
Hacettepe University
Ankara
necvaa@gmail.com

Burcu Can
RGCL
University of Wolverhampton
Wolverhampton
b.can@wlv.ac.uk

Abstract

We introduce a neural Turkish NLP toolkit called TurkishDelightNLP that performs computational linguistic analyses from morphological level to semantic level that involves tasks such as stemming, morphological segmentation, morphological tagging, part-of-speech tagging, dependency parsing, and semantic parsing, as well as high-level NLP tasks such as named entity recognition. We publicly share the open-source Turkish NLP toolkit through a web interface that allows an input text to be analysed in real-time, as well as the open source implementation of the components provided in the toolkit, an API, and several annotated datasets such as word similarity test set to evaluate word embeddings and UCCA-based semantic annotation in Turkish. This will be the first open-source Turkish NLP toolkit that involves a range of NLP tasks in all levels. We believe that it will be useful for other researchers in Turkish NLP and will be also beneficial for other high-level NLP tasks in Turkish.

1 Introduction

Turkish is one of the low-resource languages with a rich morphology. Although still limited, there has been an increasing interest in Turkish NLP in the last decade. Being a morphologically productive language is the main drawback of the Turkish NLP research. Current deep learning models are notoriously data-hungry. When it comes to morphologically productive languages, the data requirement substantially increases compared to other morphologically poor languages. This is due to the number of different word forms that can be generated via inflection and/or derivation. Although current word embedding models such as BERT (Devlin et al., 2019) rely on tokenization that considers sub-word tokens rather than word tokens, the recent research (Haley, 2020) still shows that the performance of such models degrades with novel words.

We introduce a new neural Turkish NLP toolkit that involves the following linguistic and NLP tasks in Turkish: Stemming, morphological segmentation, morphological tagging, part-of-speech tagging, dependency parsing, semantic parsing, and named entity recognition. Morphological segmentation, morphological tagging, part-of-speech tagging, and dependency parsing are learned jointly using a multi-task learning approach. Most of the previous work on Turkish morphology and syntax considers morphological and syntactic tasks as independent problems. However, syntax is strongly defined by morphology and vice versa, especially in agglutinative languages. Therefore, in this study, we benefit from the mutual interaction between morphological and syntactic layers in the language.

All components apart from semantic parsing model are built on LSTMs that are capable of learning especially long distance relations. The models also utilise a Bahdanau (Bahdanau et al., 2015) attention mechanism in various layers for an efficient learning of the valuable contextual information within the sentence/word. Moreover, we investigate cross-level information flow between the layers by incorporating information between words in different time steps. As distinct from the other components, semantic parsing model adopts an encoder-decoder model where the encoder is based on self-attention mechanism (Vaswani et al., 2017).

TurkishDelightNLP is available at http://rgcl.wlv.ac.uk/TurkishNLP/ and the source codes of all components are also publicly available, which are specified in each section below. We also provide API to allow users to process their data using HTTP requests[1]. In addition to the NLP toolkit, we provide few datasets; that are an UCCA-based semantic annotation for Turkish,

[1] The REST API for the TurkishDelightNLP toolkit is available at https://github.com/halecakir/turkish-delight-nlp-api.

Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: System Demonstrations, pages 17 - 26
July 10-15, 2022 ©2022 Association for Computational Linguistics

a Turkish stemming training set based on METU-Sabanci Turkish Treebank (Oflazer et al., 2003b), and a word similarity test set along with the human judgements for assessing morphologically rich Turkish word embeddings.

2 Related Work and Tools on Turkish NLP

Despite being a low-resource language, Turkish has been one of the actively studied languages among other low-resource languages especially in the last decade. Numerous models have been recently released for Turkish. However, most of them were not released publicly available, and they were not shared as tools that facilitate generating a result in real time. The earlier studies on Turkish morphology include **morphological analyzers** such as the two-level description of Turkish morphology (Oflazer, 1993), the stochastic morphological analyzer based on finite state transducers (Sak et al., 2009), paradigmatic approaches (Can and Manandhar, 2009, 2012, 2018), and few other open source analyzers such as Zemberek (Akın and Akın, 2007), TRmorph (Çöltekin, 2010), and the syntactically expressive morphological analyzer by Ozturel et al. (2019). The earlier studies also involve **dependency parsers** such as the probabilistic and deterministic dependency parser by Eryiğit et al. (2008), the two-phase statistical parser based on Conditional Random Fields (CRFs) by Durgar El-Kahlout et al. (2014), and the recent neural parser by Tuç and Can (2020). There are a couple of Turkish **stemmers** introduced such as the probabilistic stemmer by Dincer and Karaoğlan (2003), and the finite state machine-based Govde-Turk by Yücebas and Tintin (2017); a few **part-of-speech taggers** were also proposed such as the Hidden Markov Model-based PoS tagger by Dinçer et al. (2008), the deterministic tagger using the two-level morphological description by Oflazer and Kuruoz (1994), and unsupervised Bayesian approaches (Bölücü and Can, 2019, 2021). The first **semantic parsing** annotation for Turkish (Azin and Eryiğit, 2019) has been presented for Abstract Meaning Representation (AMR) (Flanigan et al., 2014) and there is not any other semantic parser introduced for Turkish yet, to our knowledge.

As seen, most of the linguistic analysis tasks on Turkish are based on either statistical or deterministic approaches. Currently, the Turkish NLP research focuses more on NLP applications such as **named entity recognition** (Güneş and Tantuğ, 2018; Güngör et al., 2019; Eşref and Can, 2019), text classification (Tokgoz et al., 2021), sentiment analysis (Gezici et al., 2019; Demirci et al., 2019), offensive language identification (Ozdemir and Yeniterzi, 2020), text summarisation (Ertam and Aydin, 2021), text normalisation (Göker and Can, 2018) with especially the availability of the large pretrained neural word embeddings in almost any language.

Most of the NLP tools in Turkish were released before the deep learning era and they still have not been replaced by the neural network approaches and the researchers in the field still use the old-fashioned statistical and deterministic models for morphological or syntactic processing. We aim to fill this gap with our Turkish NLP toolkit by introducing better performing neural-based methods for Turkish linguistic analysis and NLP. The most similar one to our toolkit is ITU NLP Toolkit (Eryiğit, 2014) that also involves a wide range of NLP tools such as normalization, spell correction, morphological analysis, dependency parsing, and named entity recognition. However, all of their models are independent from each other and they are built on either deterministic or statistical machine learning algorithms. Our toolkit deviates from theirs by adopting neural models and analysing morphology and syntax jointly by considering the interaction between them. Moreover, their toolkit does not involve any semantic parsing as ours.

3 About Turkish

Turkish is an agglutinating language with a rich morphology. The morphological rules are quite regular in Turkish that define the order of the morphemes in a word, as well as the morphophonemic processes such as consonant mutation and vowel harmony, which lead the suffix and the final consonant and vowel in a word to be harmonised with each other mutually. Therefore, a morpheme can have tens of different surface forms in Turkish, which are allomorphs of the same morpheme. In Turkish, syntactic information is encoded in inflectional morphemes. For example, the word ‘ *yapabileceğim’* (in English, ‘ I will be able to do’) involves the following inflectional morphemes that each correspond to a syntactic role: *‘ -abil’* (’be able’), *‘ -eceğ’* (‘ will’), and *‘ -im’* (‘ I’).

In this paper, we propose to process every word considering its left and right context through a cross-level information from morphological segments up to dependencies in a moving window, so that morphological information of the contextual words help to analyse the PoS tags, and the PoS information of the contextual words help to analyse the dependency relations in a sentence.

4 A Neural Turkish NLP Toolkit

The introduced toolkit involves different components that are all described thoroughly below.

4.1 Stemmer

The stemmer is built on an encoder-decoder model that employs a bidirectional LSTM (Can, 2019). The model has two versions, one without an attention mechanism considering all characters with equal probability and another version with Bahdanau attention (Bahdanau et al., 2015) over characters of a given word in both directions to learn character-based contextual information. The model is trained on a dataset with 17025 word types along with their stems obtained from Metu-Sabanci Treebank (Oflazer et al., 2003b). Both the model that is implemented in DyNet (Neubig et al., 2017) and the dataset are publicly available[2]. The accuracy of the stemmer is 85% and comparable to that of Zemberek (Akın and Akın, 2007), and outperforms the other open-source Turkish stemmers (Zafer, 2015).

4.2 Joint Morphology and Syntax Model

A multi-task learning model is proposed for joint learning of morphology and syntax (Can et al., 2022). The model is built upon a multi-layer LSTM structure where each layer contributes to the overall loss in a joint learning framework and the errors from all layers backpropagate from top layer to the bottom. LSTM structure has been preferred both due its low size data requirement compared to transformers and the flexibility of processing sequential information by controlling the vertical information flow between the layers. The model is trained on IMST Turkish Treebank (Sulubacak et al., 2016). The model involves 4 layers where each of them adopts a bidirectional LSTM that is specialised in either morphology or syntax. The layers are dedicated for morphological segmentation, morphological tagging, part-of-speech tagging, and dependency parsing. The order of the layers has been designed based on the direction of the information flow and the size of the units (from smaller to bigger). A separate component for morph2vec (Üstün et al., 2018) that is used to pretrain the morpheme embeddings is also involved.

The joint model is trained and evaluated on UD Turkish Treebank, which is called IMST Treebank (Sulubacak and It, 2018) and it is a re-annotated version of the METU-Sabanci Treebank (Oflazer et al., 2003a). For the pre-trained word embeddings, we use pre-trained 200-dimensional word embeddings trained on Boun Web Corpus (Sak et al., 2008) provided by CoNLL 2018 Shared Task. The overall architecture of the joint model is given in Figure 1, where each coloured component belongs to a different level of processing that starts from morphological segmentation till dependency parsing. Each level is built on LSTMs that sequentially process every unit (i.e. character, morpheme, word, or syntactic information) in a given sentence by utilising the contextual information as well (see Section 4.2.6 for the details of the cross-level information flow). An example analysis is also provided in Figure 2 and Figure 3. All layers are described in detail below. The open-source implementation in DyNet (Neubig et al., 2017) is publicly available[3].

4.2.1 Morpheme-based Word Embeddings: morph2vec

Morph2vec (Üstün et al., 2018) is a morpheme-based word embedding model that learns word embeddings as a weighted sum of word embeddings each of which are obtained from a particular morphological segmentation of a word. It is assumed that the correct morphological segmentation of a word is not known apriori; therefore, each potential morphological segmentation of a word is predicted before training the model. Each morphological segmentation is fed into a bidirectional LSTM with each LSTM unit being fed with a morpheme embedding that is randomly initialised. So each LSTM generates a word embedding for that particular morphological segmentation. Finally, Bahdanau attention mechanism (Bahdanau et al., 2015) is employed to learn the weight of each segmentation-specific word embedding. The morpheme-based embeddings give a better Spearman correlation with the human judgements in word similarity tasks compared to both char2vec (Cao and Rei, 2016) and fasttext (Bojanowski et al.,

[2] https://github.com/burcu-can/Stemmer

[3] `https://github.com/halecakir/JointParser`

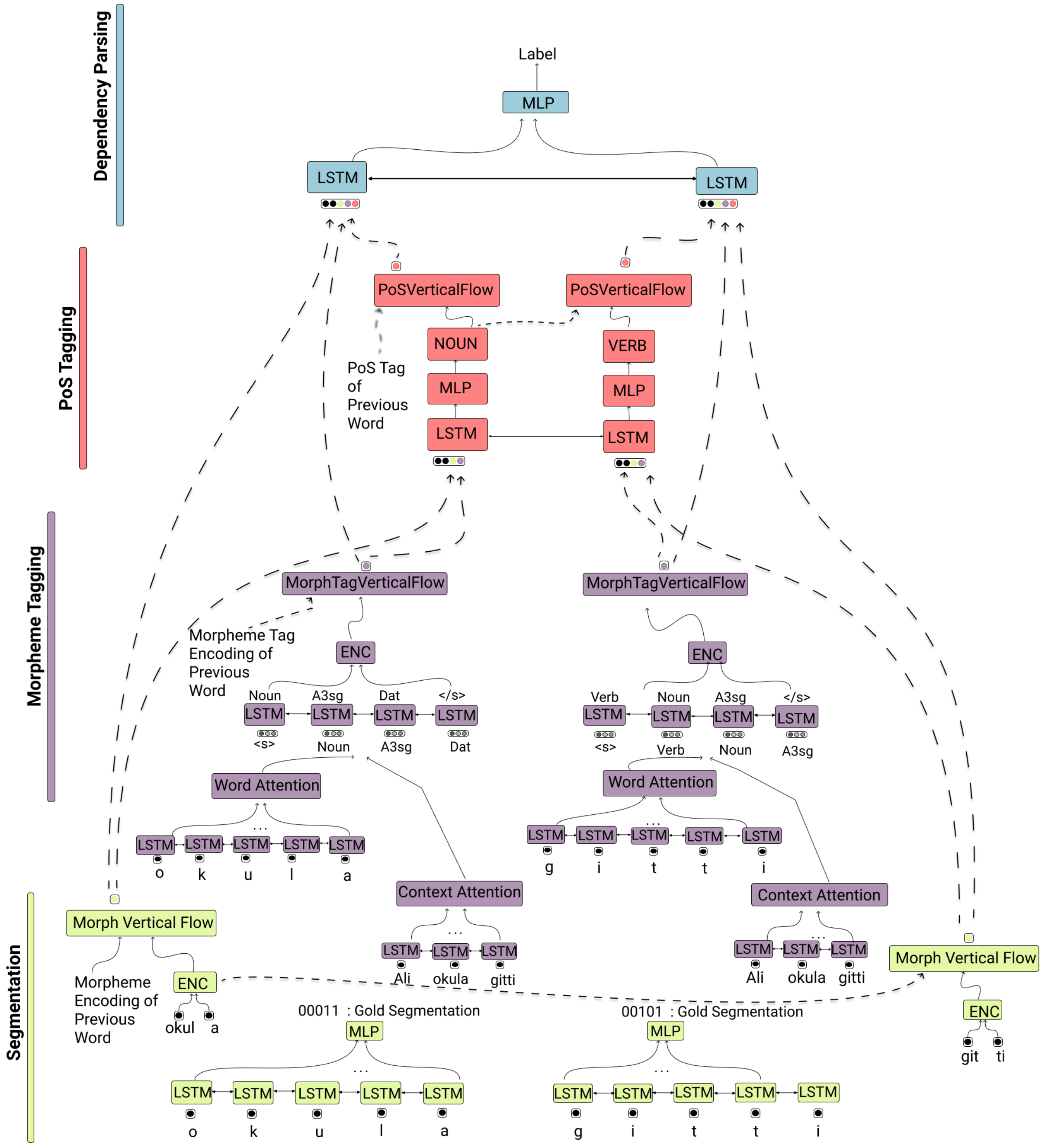


Figure 1: The layers of the proposed joint learning framework. The sentence “Ali okula gitti.” (“Ali went to school”) is processed from morphology up to dependencies (Can et al., 2022).

2017). The source code in DyNet is publicly available[4], and the datasets for syntactic analogy and word similarity along with the human judgement scores are also publicly available[5].

Morph2vec is pre-trained on METU-Sabanci Turkish Treebank (Oflazer et al., 2003b) before training the joint morphology and syntax model. Therefore, pretrained morpheme embeddings are used during joint learning.

4.2.2 Morphological Segmentation

The lowest layer of the joint model performs morphological segmentation through a bidirectional

[4] https://github.com/burcu-can/morph2vec_dynet

[5] https://nlp.cs.hacettepe.edu.tr/projects/morph2vec/

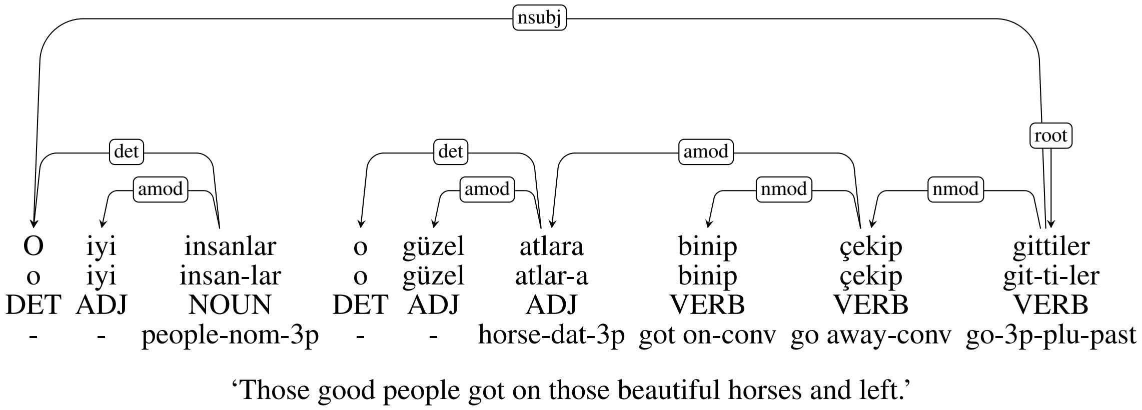

nsubj
det
amod
det
amod
amod
nmod
nmod
root
O iyi insanlar o güzel atlara binip çekip gittiler
o iyi insan-lar o güzel atlar-a binip çekip git-ti-ler
DET ADJ NOUN DET ADJ ADJ VERB VERB VERB
- - people-nom-3p - - horse-dat-3p got on-conv go away-conv go-3p-plu-past
'Those good people got on those beautiful horses and left.'

Figure 2: An example analysis of the toolkit for a sentence in Turkish. First line: The orthographic form. Second line: morphological segments. Third line: PoS tags. Fourth line: morphological features ('-' is for null). Dependencies in the article are arrowed (head to dependent) and labeled UD dependencies (de Marneffe et al., 2021).

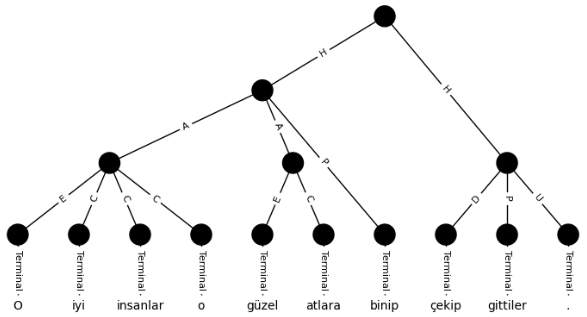


Figure 3: The UCCA-based semantic parse tree of the sentence, *O iyi insanlar o güzel atlara binip çekip gittiler.* (in English, *"Those good people got on those beautiful horses and left")*

LSTM that encodes each character of a given word with one hot encoding. The output at each time step is reduced to a single dimension using a multilayer perceptron (MLP) with sigmoid function to predict whether there is a morpheme boundary after that character or not. Each value above 0.6 refers to a morpheme boundary, and below means that there is not a morpheme boundary at that time step after the current character. Binary cross entropy is used for this layer that contributes to the overall loss of the joint model.

We obtain the gold segments from the rule-based morphological analyser Zemberek (Akın and Akın, 2007) to train the segmentation component since the IMST Treebank does not involve morphological segmentations but only morphological tags. Our joint model performs 98.97% of accuracy on morphological segmentation task[6].

4.2.3 Morphological Tagging

We adopt an encoder-decoder model for the morphological tagging layer. To encode the relevant contextual information both within the word and within the sentence, we use a character encoder and word encoder respectively. The character encoder processes the characters within the given word that will be analysed and the word encoder processes the contextual words in that sentence to better predict the morphological tagging of the given word in a particular context, which can also help to disambiguate the word in a particular context. Both of the encoders are built on bidirectional LSTMs and both of them adopt a Bahdanau attention (Bahdanau et al., 2015) to learn the weights over characters and words. The input to the decoder is the concatenation of the weighted outputs obtained from both character and context encoders. The decoder is also built on a bidirectional LSTM that generates morpheme tags using a softmax function at each time step. Our joint model performs 87.59% FEATS score on morphological tagging, and comparable to a recently introduced neural Turkish morphological tagger that performs 89.54% FEATS score (Dayanık et al., 2018).

[6] It should be noted that the test set is also obtained from Zemberek.

4.2.4 Part-of-Speech Tagging

The PoS tagging layer is built upon a bidirectional LSTM that is fed with the concatenation of word-level (i.e. word2vec embeddings), character-level (learned through a character BiLSTM for each word), morpheme-level (i.e. morph2vec), and morpheme tag encodings of each particular word in a sentence. Morpheme-level word embeddings are obtained from pretrained morph2vec as mentioned before. However, the other embeddings are all randomly initialised and learned during training. The output at each time step is passed through an MLP with softmax activation function to predict the PoS tag of the word at that time step. Our joint model performs exactly the same with the state-of-art PoS tagger by Che et al. (2018) with an accuracy of 94.78%.

4.2.5 Dependency Parsing

The dependency parsing is also built on BiLSTM that is fed with the same embeddings used in PoS tagging layer, and in addition, we concatenate the PoS encodings of the words that are obtained from the previous layer. PoS encodings are randomly initialised and learned during training. The arcs are scored by an MLP that involves a pointer network that predicts whether there is an arc between the given two words or not. Once the scores are predicted by the MLP, the projective trees are generated using Eisner's decoding algorithm (Eisner, 1996). Labels are analogously predicted using another MLP with a softmax function. Our joint model gives comparable results with the state-of-art dependency parsing results of Straka (2018) with 71% UAS and 63.92% LAS (whereas Straka (2018) achieves 72.25% UAS and 66.44% LAS).

4.2.6 Cross-Level Information Flow

The custom in such a multi-layered and multi-task models is to feed the obtained information regarding the current word from the previous layers to pass it over to the upper layers for the same word (Nguyen and Nguyen, 2021). However, to our knowledge, we are the first to analyse cross-level information flow between the layers by allowing information flow across different words in different layers. For this, we incorporate contextual information from the previous word to the current word in different layers, by adding contextual information obtained from morpheme tagging encoding and morpheme encoding of the previous word to POS and dependency layers of the current word. Similarly, we incorporate POS tagging encoding of the previous word into the dependency layer of the current word. This is shown in Figure 1 with PoSVerticalFlow, MorphTagVerticalFlow, and MorphVerticalFlow. The results show that such cross-level information flow improves the performance of the model especially in upper layers.

4.3 Semantic Parsing

We use the Universal Conceptual Cognitive Annotation (UCCA) (Abend and Rappoport, 2013) framework for semantic annotation, which is a cross-lingual semantic annotation framework. Since there is no Turkish UCCA dataset, the model is trained using a combination of English, German and French datasets (Hershcovich et al., 2017)[7], and the Turkish annotations are obtained in a zero-shot setting and manually revised. We annotated 50 sentences obtained from METU-Sabaci Turkish Treebank that is also publicly available[8].

We adopt an encoder/decoder model that tackles the semantic parsing task in the form of a chart-based constituency parsing (Bölücü and Can, 2021)[9]. Self-attention layers (Vaswani et al., 2017) are used in the encoder where the encoding is fed into an MLP with two fully-connected layers with ReLU activation function, and the CYK (Cocke-Younger-Kasami) algorithm (Chappelier and Rajman, 1998) is used within the decoder that generates the tree with the maximum score using the scores obtained from the encoder. Our Turkish UCCA-based semantic parser performs 81.11% F1 score on labeled evaluation and 90.24% F1 score on unlabeled evaluation in few shot learning.

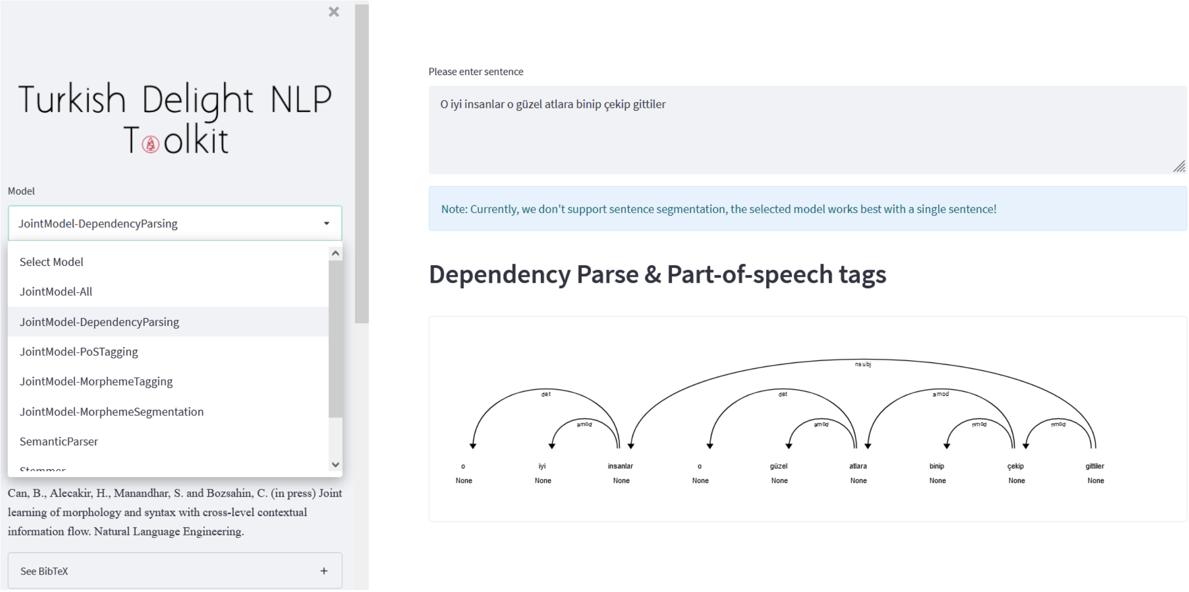
4.4 Named Entity Recognition (NER)

We use a BiLSTM-CRF model where each word is encoded through a BiLSTM and decoded with a CRF layer to learn the named entities in a given text (Kağan Akkaya and Can, 2021). We feed the BiLSTM with character-level (learned through a character-level BiLSTM), character n-gram-level (fasttext), morpheme-level (morph2vec), and word-level word embeddings (word2vec), as well as orthographic embeddings that are learned either with a CNN or BiLSTM by encoding alphabetic characters similar to that of Aguilar et al. (2017).

[7] https://github.com/UniversalConceptualCognitiveAnnotation

[8] https://github.com/necvabolucu/semantic-dataset

[9] https://github.com/necvabolucu/ucca-parser



Turkish Delight NLP Toolkit

Model

JointModel-DependencyParsing

Select Model
JointModel-All
JointModel-DependencyParsing
JointModel-PoSTagging
JointModel-MorphemeTagging
JointModel-MorphemeSegmentation
SemanticParser
Stemmer

Can, B., Alecakir, H., Manandhar, S. and Bozsahin, C. (in press) Joint learning of morphology and syntax with cross-level contextual information flow. Natural Language Engineering.

Please enter sentence

O iyi insanlar o güzel atlara binip çekip gittiler

Note: Currently, we don't support sentence segmentation, the selected model works best with a single sentence!

Dependency Parse & Part-of-speech tags

Figure 4: The user interface of the TurkishDelightNLP. The user selects a task from the dropdown menu on the left and populates an input sentence. The output is displayed on the right.

Since the particular target domain for NER in our study is noisy text especially obtained from social media, we use transfer learning to utilise any available information in a formal but possibly larger text to learn the named entities in an informal but usually a smaller text. Therefore, we adopt two CRF layers one of which is trained on the formal text (i.e. Turkish news corpus) and the other one is trained on an informal text (i.e. tweets) (Şeker and Eryiğit, 2017). Training is performed alternately between the two CRF layers which share the same BiLSTM layer. Our named entity recognition model outperforms the current state-of-art model on noisy text by Şeker and Eryiğit (2017) with 67.39% F1 score on DS-1 v4 (Şeker and Eryiğit, 2017). All source code and related material on NER are publicly available[10].

5 Web Interface

TurkishDelightNLP is a Streamlit application that provides a simple user interface for producing predictions for different tasks. We selected Streamlit since it is a low-code web framework that enables researchers to easily create a data-driven app. Streamlit has a relatively simple application programming interface and it is specifically designed for data science applications. In TurkishDelightNLP, in the backend, query and model are cached to avoid repeated calculations of the same input. Docker is used to increase portability and to be deployed in different operating systems and hardware platforms.

Figure 4 shows the user interface. In the left panel, there is a menu for the models. Whenever the user selects a model and populates a sentence, the result is displayed on the right panel.

We also provide a REST API that allows users to access the toolkit with HTTP requests. To be able to use the API, we provide an API token, so a user can access it from clients such as cURL and Postman. Moreover, with the help of Swagger and Redoc documentation, users can see how to consume API endpoints.

6 Conclusion and Future Work

We introduce a new Neural Turkish NLP toolkit that performs different levels of linguistic analysis from morphology to semantics, as well as other NLP applications such as NER. All source codes and relevant datasets are publicly available and we believe that this framework for Turkish NLP will be beneficial for other researchers in the area, and will eventually expedite the Turkish NLP research.

Acknowledgements

The joint model was funded by Scientific and Technological Research Council of Turkey (TUBITAK), project number EEEAG-115E464.

[10] https://github.com/emrekgn/turkish-ner

References

Omri Abend and Ari Rappoport. 2013. UCCA: A semantics-based grammatical annotation scheme. In *Proceedings of the 10th International Conference on Computational Semantics (IWCS 2013)–Long Papers*, pages 1–12.

Gustavo Aguilar, Suraj Maharjan, Adrian Pastor López Monroy, and Thamar Solorio. 2017. A multi-task approach for named entity recognition in social media data. In *Proceedings of the 3rd Workshop on Noisy User-generated Text*, pages 148–153.

Ahmet Afşın Akın and Mehmet Dündar Akın. 2007. Zemberek, an open source nlp framework for Turkic languages.

Zahra Azin and Gülşen Eryiğit. 2019. Towards Turkish Abstract Meaning Representation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 43–47, Florence, Italy. Association for Computational Linguistics.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015*.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching Word Vectors with Subword Information. *Transactions of the Association for Computational Linguistics*, 5:135–146.

Necva Bölücü and Burcu Can. 2019. Unsupervised joint PoS tagging and stemming for agglutinative languages. *ACM Transactions on Asian Low-Resource Language Information Processing*, 18(3).

Necva Bölücü and Burcu Can. 2021. A cascaded unsupervised model for PoS tagging. *ACM Transactions on Asian Low-Resource Language Information Processing*, 20(1).

Necva Bölücü and Burcu Can. 2021. Self-attentive constituency parsing for UCCA-based semantic parsing. *CoRR*, 2110(621).

Burcu Can. 2019. Stemming Turkish words with lSTM networks. *Bilişim Teknolojileri Dergisi*, 12:183 – 193.

Burcu Can, Hüseyin Aleçakır, Suresh Manandhar, and Cem Bozşahin. 2022. Joint learning of morphology and syntax with cross-level contextual information flow. *Natural Language Engineering*, page 1–33.

Burcu Can and Suresh Manandhar. 2009. Clustering morphological paradigms using syntactic categories. In *Proceedings of the 10th Cross-Language Evaluation Forum Conference on Multilingual Information Access Evaluation: Text Retrieval Experiments*, CLEF'09, page 641–648. Springer-Verlag.

Burcu Can and Suresh Manandhar. 2012. Probabilistic hierarchical clustering of morphological paradigms. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 654–663, Avignon, France. Association for Computational Linguistics.

Burcu Can and Suresh Manandhar. 2018. Tree structured Dirichlet processes for hierarchical morphological segmentation. *Computational Linguistics*, 44(2):349–374.

Kris Cao and Marek Rei. 2016. A joint model for word embedding and word morphology. In *Proceedings of the 1st Workshop on Representation Learning for NLP*, pages 18–26, Berlin, Germany. Association for Computational Linguistics.

J-C Chappelier and Martin Rajman. 1998. A generalized CYK algorithm for parsing stochastic CFG. In *Proceedings of 1st Workshop on Tabulation in Parsing and Deduction (TAPD'98)*, CONF, pages 133–137.

Wanxiang Che, Yijia Liu, Yuxuan Wang, Bo Zheng, and Ting Liu. 2018. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64. Association for Computational Linguistics.

Çağrı Çöltekin. 2010. A freely available morphological analyzer for Turkish. In *Proceedings of the Seventh International Conference on Language Resources and Evaluation (LREC'10)*, Valletta, Malta. European Language Resources Association (ELRA).

Erenay Dayanık, Ekin Akyürek, and Deniz Yuret. 2018. MorphNet: A sequence-to-sequence model that combines morphological analysis and disambiguation. *CoRR*, abs/1805.07946.

Marie-Catherine de Marneffe, Christopher D. Manning, Joakim Nivre, and Daniel Zeman. 2021. Universal Dependencies. *Computational Linguistics*, 47(2):255–308.

Gözde Merve Demirci, Şeref Recep Keskin, and Gülüstan Doğan. 2019. Sentiment analysis in Turkish with deep learning. In *2019 IEEE International Conference on Big Data (Big Data)*, pages 2215–2221.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics.

Bekir Dincer and Bahar Karaoğlan. 2003. Stemming in agglutinative languages: A probabilistic stemmer for Turkish. In *Lecture Notes in Computer Science book series*, volume 2869, pages 244–251. Springer.

Bekir Taner Dinçer, Bahar Karaoglan, and Tarik Kisla. 2008. A suffix based part-of-speech tagger for Turkish. *Fifth International Conference on Information Technology: New Generations (itng 2008)*, pages 680–685.

İlknur Durgar El-Kahlout, Ahmet Afşın Akın, and Ertuğrul Yılmaz. 2014. Initial explorations in two-phase Turkish dependency parsing by incorporating constituents. In *Proceedings of the First Joint Workshop on Statistical Parsing of Morphologically Rich Languages and Syntactic Analysis of Non-Canonical Languages*, pages 82–89, Dublin, Ireland. Dublin City University.

Jason M. Eisner. 1996. Three new probabilistic models for dependency parsing: An exploration. In *Proceedings of COLING 1996 Volume 1: The 16th International Conference on Computational Linguistics*, pages 340–345.

Fatih Ertam and Galip Aydin. 2021. Abstractive text summarization using deep learning with a new Turkish summarization benchmark dataset. *Concurrency and Computation: Practice and Experience*.

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. Association for Computational Linguistics.

Gülşen Eryiğit, Joakim Nivre, and Kemal Oflazer. 2008. Dependency parsing of Turkish. *Computational Linguistics*, 34:627.

Yasin Eşref and Burcu Can. 2019. Using morpheme-level attention mechanism for Turkish sequence labelling. In *2019 27th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4.

Jeffrey Flanigan, Sam Thomson, Jaime Carbonell, Chris Dyer, and Noah A. Smith. 2014. A discriminative graph-based parser for the Abstract Meaning Representation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1426–1436, Baltimore, Maryland. Association for Computational Linguistics.

Bahar Gezici, Necva Bölücü, Ayça Tarhan, and Burcu Can. 2019. Neural sentiment analysis of user reviews to predict user ratings. In *2019 4th International Conference on Computer Science and Engineering (UBMK)*, pages 629–634.

Onur Güngör, Tunga Güngör, and Suzan üsküarlı. 2019. The effect of morphology in named entity recognition with sequence tagging. *Natural Language Engineering*, 25(1):147–169.

Sinan Göker and Burcu Can. 2018. Neural text normalization for Turkish social media. In *2018 3rd International Conference on Computer Science and Engineering (UBMK)*, pages 161–166.

Asim Güneş and A. Cüneyd Tantuğ. 2018. Turkish named entity recognition with deep learning. In *2018 26th Signal Processing and Communications Applications Conference (SIU)*, pages 1–4.

Coleman Haley. 2020. This is a BERT. Now there are several of them. Can they generalize to novel words? In *Proceedings of the Third BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 333–341, Online. Association for Computational Linguistics.

Daniel Hershcovich, Omri Abend, and Ari Rappoport. 2017. A Transition-Based Directed Acyclic Graph Parser for UCCA. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1127–1138.

Emre Kağan Akkaya and Burcu Can. 2021. Transfer learning for Turkish named entity recognition on noisy text. *Natural Language Engineering*, 27(1):35–64.

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

Linh The Nguyen and Dat Quoc Nguyen. 2021. Phonlp: A joint multi-task learning model for Vietnamese part-of-speech tagging, named entity recognition and dependency parsing. *CoRR*, abs/2101.01476.

Kemal Oflazer. 1993. Two-level description of Turkish morphology. In *Proceedings of the Sixth Conference on European Chapter of the Association for Computational Linguistics*, EACL '93, page 472, USA. Association for Computational Linguistics.

Kemal Oflazer and Ilker Kuruoz. 1994. Tagging and morphological disambiguation of Turkish text. In *Fourth Conference on Applied Natural Language Processing*, pages 144–149, Stuttgart, Germany. Association for Computational Linguistics.

Kemal Oflazer, Bilge Say, Dilek Zeynep Hakkani-Tür, and Gökhan Tür. 2003a. *Building a Turkish Treebank*, pages 261–277. Springer Netherlands, Dordrecht.

Kemal Oflazer, Bilge Say, Dilek Zeynep, and Gokhan Tur. 2003b. Building a turkish treebank. *Abeillé*.

Anil Ozdemir and Reyyan Yeniterzi. 2020. SU-NLP at SemEval-2020 task 12: Offensive language IdentifiCation in Turkish tweets. In *Proceedings of the*

Fourteenth Workshop on Semantic Evaluation, pages 2171–2176, Barcelona (online). International Committee for Computational Linguistics.

Adnan Ozturel, Tolga Kayadelen, and Isin Demirsahin. 2019. A syntactically expressive morphological analyzer for Turkish. In *Proceedings of the 14th International Conference on Finite-State Methods and Natural Language Processing*, pages 65–75, Dresden, Germany. Association for Computational Linguistics.

Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2008. Turkish language resources: Morphological parser, morphological disambiguator and web corpus. In *Advances in Natural Language Processing*, pages 417–427, Berlin, Heidelberg. Springer Berlin Heidelberg.

Haşim Sak, Tunga Güngör, and Murat Saraçlar. 2009. A stochastic finite-state morphological parser for Turkish. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, pages 273–276, Suntec, Singapore. Association for Computational Linguistics.

Gökhan Akın Şeker and Gülşen Eryiğit. 2017. Extending a CRF-based named entity recognition model for Turkish well formed text and user generated content 1. *Semantic Web*, 8(5):625–642.

Milan Straka. 2018. UDPipe 2.0 prototype at CoNLL 2018 UD shared task. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 197–207, Brussels, Belgium. Association for Computational Linguistics.

Umut Sulubacak, Memduh Gokirmak, Francis Tyers, Çağrı Çöltekin, Joakim Nivre, and Gülşen Eryiğit. 2016. Universal Dependencies for Turkish. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 3444–3454, Osaka, Japan. The COLING 2016 Organizing Committee.

Umut Sulubacak and G. It. 2018. Implementing universal dependency, morphology, and multiword expression annotation standards for turkish language processing. *Turkish Journal of Electrical Engineering and Computer Sciences*, 26:1662–1672.

Meltem Tokgoz, Fatmanur Turhan, Necva Bolucu, and Burcu Can. 2021. Tuning language representation models for classification of Turkish news. In *2021 International Symposium on Electrical, Electronics and Information Engineering*, ISEEIE 2021, page 402–407, New York, NY, USA. Association for Computing Machinery.

Salih Tuç and Burcu Can. 2020. Self attended stack pointer networks for learning long term dependencies. In *Proceedings of the 17th International Conference on Natural Language Processing*, pages 90–100. NLP Association of India.

Ahmet Üstün, Murathan Kurfalı, and Burcu Can. 2018. Characters or morphemes: How to represent words? In *Proceedings of The Third Workshop on Representation Learning for NLP*, pages 144–153, Melbourne, Australia. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Sait Can Yücebas and Rabia Tintin. 2017. Gövde-Türk: A Turkish stemming method. *2017 International Conference on Computer Science and Engineering (UBMK)*, pages 343–347.

Harun Resit Zafer. 2015. Resha stemmer. https://github.com/hrzafer/resha-turkish-stemmer/. [Online; accessed 6-Feb-2022].