Adaptive Slot-Filling for Turkish Natural Language Understanding

Ahmet Zahid Balcıoğlu [*]

April 30, 2022

Abstract

Slot Filling aims to extract the values holding certain attributes for a dialogue system. In this paper we propose a adaptive transfer-learning based slot filling model using BERT and conditional random fields. Our experiments show that the results of our model out perform previous models on Turkish slot filling. We also introduce and discuss the problem of stem suffix separation in NLU for agglutinative languages such as Turkish. We propose a solution based on wordpiece tokenizers used in transformer models such as BERT.

1 Introduction

Dialogue systems comprise an important part of human and machine interaction. They increasingly have a more prominent role in our society due to recent technologies such as chatbots and virtual assistants. In general a dialogue system has to process human input such as speech or handwriting into text, then analyse the contents of the text and extracts important information from the input, which is generally called natural language understanding (NLU). The information could be about the sentiment and intent of the user, or it could be about keywords in the sentence called entities or slots.[1] Finally, the system returns an appropriate response with respect to the users request.

An important distinction between the two main tasks of intent classification and slot filling is that the latter does not have a language agnostic output.[2] Problem of slot-filling is closely related to Named Entity Recognition (NER) in which the machine is required to learn from a given set of labelled examples the words which are named entities. Both problems are generally known to be expensive for data due to the amount of data required as well as time consuming nature of labelling [3]. However, slot-filling is more domain specific and slots could change from context to context as opposed to named entities which do not vary within or, even sometimes between languages. As a result slot-filling is a particularly data demanding problem.

[*]Yıldız Technical University, Department of Statistics, Istanbul, Turkey, e-mail: zahud.balcioglu@yildiz.edu.tr

In literature slot-filling has been treated as a sequence tagging problem using popular methods such as include maximum entropy Markov models [4], conditional random fields (CRFs) [5], and recurrent neural networks (RNNs) [6–8]. Deeper networks that involve a combination of these methods as well as recently developed technologies such as attention mechanisms [9] have also been proposed. Most notably, Bidirectional LSTMs used in conjunction with CRF models were among the most successful [10–12], CNN-LSTM models have seen a lot of success [12]. More recently, models that can adapt to more than one domain or language have seen more attention [8, 13]. Namely transformer models such as BERT [14], and DIET [15] which currently have the most promising results. Another particular strength of transformer models is their success in transfer learning. Another avenue of research in NLU is the advent of active-learning which allows users to train and label NLU machine learning models specific to their domains without the need of machine learning expertise.[16, 17]

One of the most important benchmark data sets for NLU is the ATIS data set [18], which consists of questions given to an air travel information system. Every sentence in the data set is labelled for the intent of the speaker. In addition words containing important information are labelled as slots. Since its inception in the 90s, considerable part of NLU literature focused on ATIS data set and it has been translated to more than six languages.[13] However, recently especially with the advent of active-learning that allow for extensive user interaction. It has become increasingly more important to produce successful results in user provided domain-specific data.

Turkish NLU problem has been appeared in a few studies involving multilingual BERT models using a translated version of ATIS.[19] However, there is only a single recent paper which is specific to Turkish, [20] which studies another translated version of ATIS data set using BERT. One important challenge for Turkish NLU which has to our knowledge not been addressed in the literature concerns the agglutinative nature of Turkish. In particular, due to a word having to take many suffixes it may difficult to identify the part of the word aught to be returned. Furthermore, the relevant part of the word may also change between sentences. For example the word for hot-dog in Turkish is ”sosisli”, the same word can also be used to mean ”with/has sausage” due to the ”-li” suffix. In the first case, the word ”sosisli” would be the slot, yet in the second case the root word ”sosis” is ought to be the slot instead. Because of these reasons it becomes important for a machine learning model to separate a word into meaningful root and suffixes and learn which suffixes are useful. We will call the above problem we have described as the suffix problem.

In section 2, we go over the proposed Adaptive BERT-CRF model, define the stemming problem we mentioned above. In section 3, we explain the choices made for the data set and go over our experimental setup and model evaluation. In the last section, we discuss model results and propose directions for future research.

2 Methodology

2.1 Problem Definition

We define the stemming problem for Turkish NLU as following: Due to the aglutinative language of Turkish labelling entire words as slots may lead to ambiguity and confusion between slots. As a means to remove any ambiguity it is necessary to label parts of a word as the slot.

One of the most well-known methods which could be used to solve the suffix problem is stemming. We can simplify the problem in most cases to the question of whether the entire word or just the word root is needed for the slot. From this assumption, we can tokenize a sentence into a set word stem - suffix pairs and analyze the problem from this new state of tokenization. There are finite state transducer(FST) models and other probabilistic tools already existent for Turkish which can help in stemming.[21] Another relatively recent tool, which to our knowledge has not been used for stemming is sub-word tokenizers which are used in the training of models like BERT.[22] The advantage of BERT over the FST model stemmer models is the already prepared word vectors for BERT.[23] For FST on the other hand, one would need to train a new word vector model. Another disadvantage of FST models is their relatively poor performance on named entities.

Another problem which NLU models face is the data requirements for training deep-learning models. The requirement of gathering and labelling data for every specific task is too resource intensive. However, especially transformer models that have seen large amounts of data in pre-training have been utilized to address this problem. Especially in the recent papers in which multilingual slot-filling has been studied, the data set for Turkish in particular was no larger than 700 sentences. [13]

2.2 Model Description

Our main objective is a BERT transfer learning model which can distinguish labelled stem-suffix pairs in learning and also adapt to domain specific problems. For our data set we assume that for each slot either the entire word or just the word root is labelled. Our proposed model is as follows, first we use a BERT model that we have fine-tuned on a part of speech tagging (POS) problem. After fine-tuning the weights on the BERT model is frozen and the results are passed on to a CRF layer which trains on the actual data set. The predictions of the model is then checked by a FST model in order to ensure that each predicted word is either a full word or is split into correct stem-suffix pairs. For any incorrect stem-suffix pairings we predict the entire word instead. Usage of the FST model is optional as it is not trained; however, we find that it increases the success of model predictions.

One of the main reasons for our model choice is that especially for active-learning it is incredibly important to have a model that has a low training time and can produce reasonable results in a low data setting. Since the BERT model is already fine-tuned on a general sequence tagging problem such as POS tagging, the additional CRF layer acts as a domain specification layer. This helps our model success in a low data setting, while the relatively fast training times for CRFs help in reducing training times.

2.3 Stemming and Wordpiece tokenization

One use key idea for our model is the use of a wordpiece tokenizer with a similar purpose to a stemmer. With the help of labelling we may distinguish any ambiguity and specify the important part of a slot. In the case of Turkish, the proposed method also can with separating declensional suffixes. As an example in the sentence "**İstanbul'dan Ankara'ya** uçakla gideceğim." (I will go **to Istanbul from Ankara** by plane.) 'İstanbul' is in the ablative case and 'Ankara' is in the dative case. However, in the slot-filling problem one is interested in the words "İstanbul" and "Ankara" themselves. Hence in a given data set in order to specify, we would have the slots İstanbul and Ankara labelled yet the suffix "-dan" and "-ya" would have no label. This of course would be impossible to do without a word-piece tokenizer allowing us to separate words into multiple meaningful pieces and a fine-tuned BERT model which can produce word embeddings for stems and suffixes.

A key problem, which have thus far we have not mentioned is that the definition of a successful stemming for wordpiece tokenizers is not well-defined. Having a well-defined definition is a crucial step for successful stemming. There are a few important problems which need to be taken into consideration. Most agglutinative languages have suffixes that can change word stems, the suffixes may be identified by the tokenizer and seperate the word into much smaller pieces. There are also rare words which may not appear in the training data of the tokenizer. Additionally, some words may have parts which look like suffixes without etymological connection, these may again seperate the word into much smaller pieces. Therefore it would be unreasonable to expect a wordpiece tokenizer to output just the stem-suffix pairs we want. However, it is often the case that wordpiece tokenizer outputs can be regrouped into stem and suffix categories. We will base our definition on this idea, to illustrate see the following example.

In the word "Gözlükçülerde" (in opticians') the word stem is "Gözlükçü" (Optician). The word "Gozlukcu" itself is derived from the words "göz,(eye), and "gözlük" (eyeglasses). The stem "Gözlükçü" also took two suffixes "-ler" and "-de" which are for the plural and locative cases. For successful stemming, we are interested in separating "Gözlükçülerde" into the stem-suffix pair ("Gözlükçü", "-lerde"). Naturally, a wordpiece tokenizer may not output this pair but might give us a longer sequence which can be reorganized into the pairs we expect. Therefore, we define a successful stemming by a wordpiece tokenizer as the division of a word into a not necessarily meaningful sequences of strings that can be reduced to a stem-suffix pair. To illustrate, consider again the word "Gözlükçülerde" the following are examples of successful tokenization, where # denotes a token:

Successful Tokenization	**Unsuccessful Tokenization**
Goz, #luk, #cu, #ler, #de	Goz, #luk, #culer, #de
G, #oz, #luk, #cu, #ler, #de	Gozlukc, #uler, #de
Gozlukcu, #ler, #de	Gozlu, #kculer, #de
Gozluk, #cu, #ler, #de	Gozluk, #culer, #de
Go, #zluk, #cu, #ler, #de	Gozluk, #culerde
Gozlukcu, #lerde	
Gozluk, #cu, #lerde	

Table 1: Examples of successful and unsuccessful tokenizations of the word "Gözlükçülerde" based on our definition.

3 Experiments

In this section we will share the results of the experiments we have conducted. A fundamental assumption of our model is that wordpiece tokenization should consistently separate a word into stem-suffix pairs. In order to test this hypothesis, we have used a 58770 word dictionary provided by [24]. In this dictionary words along with their stems, origins and part of speech tags are given. We have compared the stemming success of subword tokenizers to traditional FST stemmers.

In order to test the success of our proposed method we have gathered two different data set from different domains. The data was provided and labelled by a domain expert. Before any data collection, we have shown experts examples of translated sentences from the ATIS data set, in order to show good example sentences. In all the experiments we have asked the expert to either label the entire word or the word stem to the best of their knowledge. We did not do any stemming corrections post-labelling.

There are a few reasons for our choice in the data sets. The first one is that there is no data set existent for Turkish addressing the stemming problem. In fact to our knowledge this is the first time stemming problem is considered in the Turkish NLU literature. Also, so far there is a lack of an established benchmark data set. Currently, most studies have been done on a translated version of the ATIS data set. However, none of the data is publicly available. Moreover, as translations were done independently and without cooperation, it is not possible to guarantee any consistency with the results. Above all, the main purpose of our work is to create a model capable of learning in a relatively low data setting which can then be generalised via active-learning.

Our data set consists of two separate sets of examples, one is on the form of a set from exchanges between a customer and a phone operator for a bank. The other one consists of sentences from an intercity bus terminal service. There are 180 sentences in each data set respectively, making a total of 360 sentences. In both data sets, only the word roots are labelled as entities while suffixes are marked separately.

Example Sentences	**Labelling**
Faturalarımı ödemek istiyorum.	[Fatura][larımı] <*paymentType,O*> ödemek istiyorum.
Bu Beyoğlu'nda herhangi bir banka var mı özellikle Vakıfbank.	Bu [Beyoğlu]['nda]<*Location,O*> herhangi bir banka var mı özellikle [Vakıfbank]<*Bank*>.
125 tl para çekmek istiyorum.	[125]<*Amount*> [tl]<*Money*> para çekmek istiyorum.

Table 2: Example sentences from the data set. O represents suffixes that are not part of the slot.

4 Results and Conclusion

For stemming experiment using the dictionary provided by [24], we have compared the success of FST stemmers, which currently have the most successful stemming for Turkish, with wordpiece tokenizer provided by [25]. Out of the total 58770 word, FST stemmers have achieved an accuracy score of 79.8% while the wordpiece tokenizer had an accuracy score of 75.4%. A contingency table of result comparisons can be seen below. Based on these results, we think that Bert tokenizers are successful enough to be considered a stemming alternative.

Before starting our experiments we prepared the data set with some preliminary steps. The goal of these steps was to increase feature quality and make the sentences compatible with Bert. First we removed all punctuation marks then we turned all letters to lowercase. Before training we have randomly selected 80% of the data set for training and used the remaining 20% for testing. All results reported come from the test set.

$F1$-score is the most commonly used metric for token classification. Many of the papers on the slot-filling problem for ATIS data set also used $F1$-score to report their results [2, 20]. Therefore, we employed $F1$-score as the evaluation metric for our models, with the formula given below

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}.$$

As it can be seen in table 3, our model has a relatively more successful results compared to the most recent Turkish slot-filling studies. Furthermore, across all labelled entities our model boasts a 95% accuracy rate of correct stemming. One possible reason for the increased success relative to 75.4% reported may be the expert system using FST stemmer increasing the success. Another possible explanation is that our data set had a relatively simple set of suffixes that are easy to differentiate which resulted in a higher success rate.

Model	$F1$ Score
STIL [2]	0.8523
Adaptive BERT-CRF	**0.913**
BERT [20]	0.872
Krishnan et al. [19]	0.9051
Xu et al.[13]	0.8604

Table 3: Reported weighted average $F1$ scores for Turkish NLU slot-filling

Due to the lack of publicly available slot-filling data sets for Turkish, we can not give reliable comparative results. However, based on the data we have our model outperforms state-of-the art models in slot-filling. Another challenge our model had was the ability work well in a low data setting, which is important for our model to be viable in user input oriented active-learning setting. As gathering and labelling data for slot filling is an expensive process, being able to leave the judgement of model success and data acquisition to domain experts gains importance. Building models that can train fast and score well on a small data set is an important step in that direction.

Although, we have stated some promising results for stemming with wordpiece tokenizers. There are still avenues for further progress. Notably, wordpiece tokenizers behave poorly with proper nouns and city names. Acronyms are also problematic. Especially since the our definition of successful stemming based on stem-suffix pairs do not apply well to the concept of acronyms. Thus, searching for a better suited tokenizer for stemming may be the most gainful avenue of research.

References

[1] N. Gupta, G. Tur, D. Hakkani-Tur, S. Bangalore, G. Riccardi, and M. Gilbert, "The at amp;t spoken language understanding system," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 14, no. 1, pp. 213–222, 2006. DOI: 10.1109/TSA.2005.854085.

[2] J. G. M. FitzGerald, *Stil – simultaneous slot filling, translation, intent classification, and language identification: Initial results using mbart on multiatis++*, 2020. arXiv: 2010.00760 [cs.CL].

[3] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer, *Neural architectures for named entity recognition*, 2016. arXiv: 1603.01360 [cs.CL].

[4] A. McCallum, D. Freitag, and F. C. Pereira, "Maximum entropy markov models for information extraction and segmentation.," vol. 17, no. 2000, pp. 591–598, 2000.

[5] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[6] G. Mesnil, Y. Dauphin, K. Yao, Y. Bengio, L. Deng, D. Hakkani-Tur, X. He, L. Heck, G. Tur, D. Yu, *et al.*, "Using recurrent neural networks for slot filling in spoken language understanding," *IEEE/ACM Transactions on Audio, Speech, and Language Processing*, vol. 23, no. 3, pp. 530–539, 2014.

[7] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," pp. 3104–3112, 2014.

[8] B. Liu and I. Lane, "Multi-domain adversarial learning for slot filling in spoken language understanding," *arXiv preprint arXiv:1711.11310*, 2017.

[9] A. Gupta, J. Hewitt, and K. Kirchhoff, "Simple, fast, accurate intent classification and slot labeling for goal-oriented dialogue systems," *arXiv preprint arXiv:1903.08268*, 2019.

[10] A. S. Varghese, S. Sarang, V. Yadav, B. Karotra, and N. Gandhi, "Bidirectional lstm joint model for intent classification and named entity recognition in natural language understanding," *International Journal of Hybrid Intelligent Systems*, vol. 16, no. 1, pp. 13–23, 2020.

[11] X. Ma and E. Hovy, "End-to-end sequence labeling via bi-directional lstm-cnns-crf," *arXiv preprint arXiv:1603.01354*, 2016.

[12] M. Firdaus, A. Kumar, A. Ekbal, and P. Bhattacharyya, "A multi-task hierarchical approach for intent detection and slot filling," *Knowledge-Based Systems*, vol. 183, p. 104 846, 2019.

[13] W. Xu, B. Haider, and S. Mansour, "End-to-end slot alignment and recognition for cross-lingual nlu," *arXiv preprint arXiv:2004.14353*, 2020.

[14] Q. Chen, Z. Zhuo, and W. Wang, "Bert for joint intent classification and slot filling," *arXiv preprint arXiv:1902.10909*, 2019.

[15] T. Bunk, D. Varshneya, V. Vlasov, and A. Nichol, "Diet: Lightweight language understanding for dialogue systems," *arXiv preprint arXiv:2004.09936*, 2020.

[16] P. Simard, D. Chickering, A. Lakshmiratan, D. Charles, L. Bottou, C. G. J. Suarez, D. Grangier, S. Amershi, J. Verwey, and J. Suh, *Ice: Enabling non-experts to build models interactively for large-scale lopsided problems*, 2014. arXiv: 1409.4814 [cs.AI].

[17] J. D. Williams, E. Kamal, M. Ashour, H. Amr, J. Miller, and G. Zweig, "Fast and easy language understanding for dialog systems with microsoft language understanding intelligent service (luis)," in *Proceedings of the 16th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, 2015, pp. 159–161.

[18] C. T. Hemphill, J. J. Godfrey, and G. R. Doddington, "The ATIS spoken language systems pilot corpus," 1990. [Online]. Available: https://aclanthology.org/H90-1021.

[19] J. Krishnan, A. Anastasopoulos, H. Purohit, and H. Rangwala, "Multilingual code-switching for zero-shot cross-lingual intent prediction and slot filling," *arXiv preprint arXiv:2103.07792*, 2021.

[20] F. Şahinuç, V. Yücesoy, and A. Koç, "Intent classification and slot filling for turkish dialogue systems," in *2020 28th Signal Processing and Communications Applications Conference (SIU)*, IEEE, 2020, pp. 1–4.

[21] C. Cöltekin, "A freely available morphological analyzer for turkish.," in *LREC*, vol. 2, 2010, pp. 19–28.

[22] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, M. Norouzi, W. Macherey, M. Krikun, Y. Cao, Q. Gao, K. Macherey, J. Klingner, A. Shah, M. Johnson, X. Liu, Łukasz Kaiser, S. Gouws, Y. Kato, T. Kudo, H. Kazawa, K. Stevens, G. Kurian, N. Patil, W. Wang, C. Young, J. Smith, J. Riesa, A. Rudnick, O. Vinyals, G. Corrado, M. Hughes, and J. Dean, *Google's neural machine translation system: Bridging the gap between human and machine translation*, 2016. arXiv: 1609.08144 [cs.CL].

[23] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *Bert: Pre-training of deep bidirectional transformers for language understanding*, 2019. arXiv: 1810.04805 [cs.CL].

[24] E. M. BAYOL, *Trnlp*, https://github.com/brolin59/trnlp, 2020.

[25] S. Schweter, *Berturk - bert models for turkish*, version 1.0.0, Apr. 2020. DOI: 10.5281/zenodo.3770924. [Online]. Available: https://doi.org/10.5281/zenodo.3770924.