Turk J Elec Eng & Comp Sci
() : –

Mention detection in Turkish coreference resolution

Şeniz DEMİR[1]*, **Hanifi İbrahim AKDAĞ**[1]
[1]Department of Computer Engineering, MEF University, Istanbul, Turkey

ORCIDs:
First AUTHOR: https://orcid.org/0000-0003-4897-4616
Second AUTHOR: https://orcid.org/0009-0009-0829-1440

Received: .2023 • **Accepted/Published Online:** .202 • **Final Version:** ..202

Abstract:

A crucial step in understanding natural language is detecting mentions that refer to real-world entities in a text and correctly identifying their boundaries. Mention detection is commonly considered as a preprocessing step in coreference resolution which is shown to be helpful in several language processing applications such as machine translation and text summarization. Despite recent efforts on Turkish coreference resolution, no standalone neural solution to mention detection has been proposed yet. In this article, we present two models designed for detecting Turkish mentions by using feed-forward neural networks. Both models extract all spans up to a fixed length from input text as candidates and classify them as mentions or not mentions. The models differ in terms of how candidate text spans are represented. The first model represents a span by focusing on its first and last words whereas the representation also covers the preceding and proceeding words of a span in the second model. Mention span representations are formed by using contextual embeddings, part-of-speech embeddings, and named-entity embeddings of words in interest where contextual embeddings are obtained from pre-trained Turkish language models. In our evaluation studies, we not only assess the impact of mention representation strategies on system performance but also demonstrate the usability of different pre-trained language models in resolution task. We argue that our work provides useful insights to the existing literature and the first step in understanding the effectiveness of neural architectures in Turkish mention detection.

Key words: Coreference resolution, mention detection, neural network, language model, Turkish

1. Introduction

Coreference resolution addresses the identification of all mentions in a text that refer to the same real-world entity (e.g., a person or a location) [1]. Resolving coreferent entities can be handled by detecting text spans that constitute entity mentions and grouping (clustering) these mentions into coreference chains based on the entities that they refer to. The task has been commonly formulated in one of three forms [2]: i) the mention-pair model where the goal is to determine whether two mentions are coreferent or not, ii) the mention-ranking model that aims to select the correct antecedent of a mention from a set of candidates, and iii) the entity-mention model with the goal of determining whether a mention is referring to an entity represented by a partially formed mention cluster or not. This challenging task plays a crucial role in text understanding and has been widely studied in various language processing applications such as machine translation [3], question answering [4], and information extraction [5].

*Correspondence: demirse@mef.edu.tr

Entity mentions have different lifespans in the text where some mentions appear only once (*singleton mentions*) and some mentions are repeated further in the text after their introduction (*coreferent mentions*) [6]. A considerable attention has been devoted to differentiating singleton mentions from coreferent mentions and identifying their boundaries in the text [7]. The literature has showed that filtering out singleton mentions significantly reduces the search space and hence improves the performance of downstream coreference clustering [8]. It is also important to note that entity mentions that spread across a text might take varying forms such as a noun phrase, a named entity, or a pronoun.

Many traditional coreference resolution studies have utilized rule-based, statistical-based, or deep learning-based methodologies. In rule-based methods [9], a number of hand-crafted rules that rely on linguistic features (e.g., synonymity relations and part-of-speech tags of words) and external knowledge resources are used [10]. The adaptation of such solutions to new languages and domains is not straightforward and requires tremendous manual effort. On the other hand, statistical-based methods use learning methodologies (e.g., decision tree and support vector machine) while resolving coreferent entities and require large-scale training data [11–13]. Mention detection is often defined as a sequence labeling or classification problem and clustering predictions are performed by assigning scores to mention pairs that indicate whether the mentions are coreferent or not. Deep learning-based methods eliminate the need of feature engineering by learning required features and underlying relations between entity mentions directly from the text [14, 15]. With recent advancements in neural architectures (e.g., transformers), the widespread availability of large-scale language models, and the representational power of word embeddings, these approaches significantly improve over previous state-of-the-art results in coreference resolution.

The literature has followed two main approaches to integrate the main components of a resolution system. In pipeline approaches, mention detection and mention clustering components are developed separately and combined in a pipeline [16, 17]. The major drawback of these approaches is that the performance of mention detection has a significant impact on clustering performance due to error cascading (e.g., incorrect identification or missing of entity mentions) [18]. On the other hand, mention detection and clustering tasks are jointly performed in end-to-end approaches which often benefit from deep learning architectures [19–21]. These architectures enable all spans extracted from a text to be identified as candidate mentions and the top-scoring candidates to be considered for clustering. However, developing the best-performing architecture is still an open research issue with much room for improvement especially in domains with limited data.

Despite a substantial amount of research devoted to high-resource languages, there is a limited number of mention detection and clustering approaches developed for less-resourced languages [22–24] including Turkish. The majority of Turkish studies have tackled the pronoun resolution problem [25] and utilized traditional statistical-based methods including naive bayes, support vector machine, and k-nearest neighbour [26, 27]. The first work on Turkish that broadly address coreference resolution has followed the mention-pair approach and used decision tree and support vector machine classifiers fed by linguistic features [28]. Recently, deep learning-based Turkish coreference resolution studies which followed the mention-ranking approach have been introduced [29, 30]. The first of these studies [29] has experimented with two different models to cluster entity mentions by assuming that the mention detection task is completed in advance. The first neural model captures fine-grained linguistic features of entity mentions as input whereas the second model uses a neural architecture with embeddings learned from large-scale language models as input. The study has reported the impact of various pre-trained Turkish language models on mention clustering. The second study [30] has applied a similar end-to-end model to the dataset used in the first study after being extended with the incorporation of dropped

pronouns. The only work that addressed Turkish mention detection is a rule-based system which marks all noun phrases, pronouns, named entities, and capitalized common nouns as entity mentions [31].

This article presents the first deep learning-based study on mention detection in Turkish. Our two-step approach first automatically extracts all possible candidate mentions from a text by limiting the number of words that a candidate can span. These candidates are then passed through a feed-forward neural network in order to be classified as mentions or not mentions. For this purpose, two neural models with different mention span representation strategies are explored. In the first model, a candidate span is represented with its first and last words whereas the words that respectively precedes and follows the candidate are also considered in the second model. Both models represent candidate mentions by combining word embeddings (contextual representations of words) with the embeddings that reflect part-of-speech (POS) tags and named-entity information of words in focus. In the experiments, we assess the performance of our mention detection models and measure the impact of Turkish language models on mention representation. Moreover, we compare our model performances with the performance of a transformer-based sequence labeling (token classification) model.

The rest of the article is organized as follows. Section 2 discusses related work on mention detection. Section 3 introduces the neural models and mention span representation strategies used in the study. Section 4 describes the dataset used in experiments and the language models along with the neural model parameters. Section 5 presents the experimental results and Section 6 concludes the article with future work.

2. Related Work

Mention detection research has been evolved from rule-based [32, 33] and machine learning-based [34, 35] approaches to recent studies that heavily depend on neural network methodologies. By using deep learning architectures, the researchers have demonstrated that the semantic and syntactic features of input text can be automatically learned to identify singleton and coreferent mentions that appear in the context. Some previous research have proposed standalone mention detection approaches whereas most studies have integrated a mention detector in their end-to-end coreference resolution systems [19, 36, 37].

Recurrent neural networks (RNN) and in particular Long short-term memory networks (LSTM) have been frequently used by prior mention detection studies. One of the earlier studies [38] defined the problem as a sequence labeling task and benefited from different variations of an RNN architecture (e.g., with ELMAN and JORDAN methods). All words in a sentence were converted into embeddings which are formed by concatenating word embeddings (trained from a large corpus) with a binary vector containing different features (e.g., capitalization and trigger words). The robustness of proposed architectures was measured across different domains in the evaluations. Sequence labeling was also the main objective of a later research study [39] which managed nested mentions as well (i.e., a single word is tagged with multiple labels). In the study, a sequence-to-sequence model encoded words of a sentence using a Bi-directional LSTM (Bi-LSTM) network and the decoder initialized with the last hidden state of the encoder was implemented as an LSTM network. Pointer networks, an extension of RNNs, were also used in mention detection studies. These networks were shown to be effective in addressing detection problems of overlapped and singleton mentions in multiple languages [40]. A more recent study also targeted nested mention problem by using a StackLSTM architecture [41]. Sentences with overlapping mentions were converted into forests and a shift-reduce parser based system was used to learn the constructs of these forests. Processed nested mentions were kept in a stack which was encoded via a Stack-LSTM. The words in a sentence were represented as a concatenation of three embeddings that correspond to word embeddings, POS tag embeddings, and character-level embeddings learned by a Bi-LSTM network. The

Table 1: Standalone neural mention detection systems for other languages.

Study	*Architecture*	*Objective*	*Input Representation*	*Language*	*F-1*
[38]	RNN	Sequence Labeling	Word embedding, Feature embedding	English, Dutch	0.899 0.835
[40]	Pointer Network	Sequence Labeling	Word+POS embedding	English, Korean	0.797 0.801
[41]	Stack-LSTM	Mention Sequence Generation	Word embedding, POS embedding, Character embedding (Bi-LSTM)	English	0.753
[44]	FFN	Classification	i) Word embedding, Char embedding (Bi-LSTM) ii) Word embedding (Bi-LSTM) iii) Deep bidirectional transformers	English	0.888
[46]	FFN	Classification	Word embedding, POS embedding, NER embedding, Feature embeddings	English	0.752
[47]	MLP	Classification	Word embedding (Recursive autoencoder)	English	0.827
[48]	Fully connected network	Classification	Word embedding, Feature embedding (CNN)	Hindi	0.670
[49]	FFN	Classification	Word embedding, Character embedding (FOFE encoding)	English, Chinese, Spanish	0.909 0.753 0.756

use of Stack-LSTM and shift-reduce parser was later seen in other coreference studies [42].

A pioneer end-to-end coreference resolution work [19], despite not proposing a standalone mention detector, applied an exhaustive search method to identify all possible text spans and assign each span a score indicating its probability of being a mention. The words in a span were first represented with an embedding composed of context-independent pre-trained word embeddings and character embeddings learned by a convolutional neural network (CNN). These embeddings were then fed to a Bi-LSTM network in order to obtain contextual word encodings. Finally, mention span representations were formed by concatenating contextual embeddings of the first, last, and the head word of text spans (learned via an attention mechanism). These mention representations were scored by using a feed-forward neural network. Later research has improved upon this scoring approach in several directions [43].

Recently, three different architectures for standalone mention detection have been introduced [44]. The first architecture utilized a modified version of the mention detection approach used in [19]. The only difference was the use of sigmoid entropy as the loss function. The second architecture encoded sentences using a Bi-LSTM network fed by ELMO embeddings. Two separate feed-forward networks were applied to Bi-LSTM output to obtain distinct representations of the start and end of candidate mention spans and a biaffine classifier was deployed to obtain candidate scores. The third architecture utilized pre-trained BERT language model to encode words in text spans up to a fixed length. The representations of the first and last words of a span were concatenated to obtain mention span representations. Candidate mention scoring was performed via a feed-forward network. The second architecture was later adapted to Arabic mention detection by using BERT

model embeddings rather than ELMO embeddings [20]. The use of BERT language model in mention span encoding was also explored in the detection component of other end-to-end coreference resolution systems [45].

Another mention span representation for English mention detection has been proposed in [46]. Each mention span was represented as a combination of five embedddings that capture different features of the span. The embeddings of the first, last, previous, next, and head word of a mention span were concatenated to form its semantic representation. Context-independent pre-trained GloVe embeddings were used during this encoding. This semantic representation was augmented with POS tag embeddings, named-entity embeddings, Recasen's features embeddings, and embeddings of additional features (e.g., is_pleonastic and mention type). The mention span representations were used as input in a feed-forward network finalized with a softmax classifier.

Table 1 presents some standalone mention and singleton [47, 48] detection systems designed for other languages. The systems were often tested on multiple datases or languages and the highest reported F-1 score varied between 0.670 and 0.909. Our work is inspired by the exhaustive search methodology [19, 44] and melds the strengths of different models that utilize feed-forward network architecture for mention detection [44, 46, 49]. Our work differs from previous work in three ways. First, two new mention span representations that heavily depend on transformer-based language models are utilized. Second, the effectiveness of different tokenization strategies in detecting mentions is explored for a morphologically-rich language. Third, the performance of a classification model is compared with the performance of a transformer-based sequence labeling model for Turkish mention detection task for the first time in the literature.

3. System Architecture

Our mention detection approach consists of two steps. In the first step, text spans (a sequence of consecutive words) are extracted from a given text as candidate mentions and these text spans are identified as mentions or not in the second step. For the first task, we follow a straightforward approach used by previous research [19, 44] and extract all possible spans up to a pre-determined length (n) from input text. For instance, some candidate spans (n=3) that might be extracted from the sentence "Hayatın başlangıcıdır ve daha büyük bir ev olan dünyaya hazırlanma alanıdır ." are "Hayatın", "Hayatın başlangıcıdır", "Hayatın başlangıcıdır ve", "başlangıcıdır", and "başlangıcıdır ve". We define the second task as a binary classification problem. For this purpose, we train a feed-forward neural network with two hidden layers whose architecture is shown in Figure 1. The Gaussian error linear unit activation function (GeLU) is used for the hidden layer and the sigmoid function is applied as the final activation function for binary classification. The network takes text span representations as input and determines whether the span is a mention or not.

We obtain a candidate span representation by concatenating the representations of words contained in the span. Each word is represented with three kinds of embeddings: word embedding (w_e), part-of-speech tag embedding (pos_e), and named-entity embedding (ne_e). The word embedding which is a contextualized representation of the word is used to capture its sequence-level semantics in the sentence. The part-of-speech tag embedding is a fixed-length vector that contains the POS tag information of the word. The vector represents eight POS tags, namely 'Noun', 'Verb', 'Adjective', 'Adverb', 'Pronoun', 'Proper Noun', 'Number', and 'Other' (i.e., all other tags). The named-entity embedding is also a fixed-length vector used to represent whether the word is a named entity or not. The vector features five different named entity types, namely 'Person', 'Location', 'Organization', 'OtherType', and 'None' (i.e., the word is not a named entity).

We explore two different span representation models in this work. In our first model (**FL_Model**), we only use the representations of the first (f_word) and last (l_word) words of a text span. However, in the second

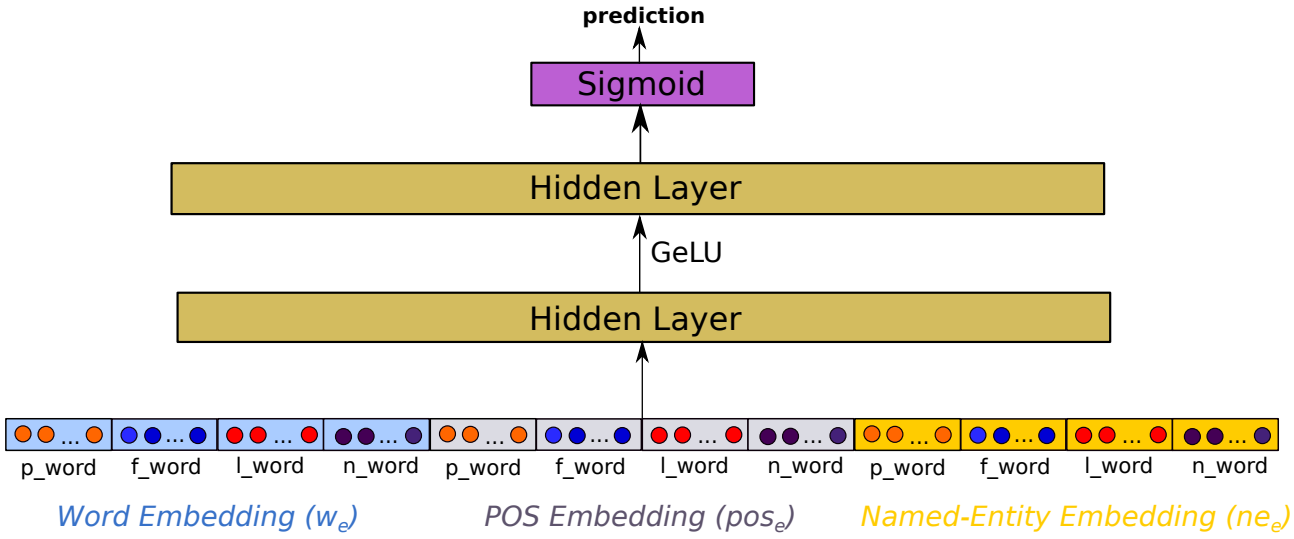

Figure 1: System architecture.

model (**PFLN_Model**), we also use the representations of the previous word (*p_word*) that precedes the span and the next word (*n_word*) that follows the span in the sentence. In cases where the text span starts with the first word in a sentence (x^1), the representation of the previous word consists of only 0's. Similarly, for a span that ends with the last word of a sentence (x^n), the next word is represented with a vector of 0's. In order to obtain the final span representation, we first concatenate the word embeddings, pos embeddings, and named-entity embeddings of all words in focus separately and then concatenate these representations as follows:

Sentence: $x^1, x^2, x^3, x^4, x^5, ...x^n$

Text Span: x^2, x^3, x^4

FL_Model Span Rep.: $w_e^2 \oplus w_e^4 \oplus pos_e^2 \oplus pos_e^4 \oplus ne_e^2 \oplus ne_e^4$

PFLN_Model Span Rep.: $w_e^1 \oplus w_e^2 \oplus w_e^4 \oplus w_e^5 \oplus pos_e^1 \oplus pos_e^2 \oplus pos_e^4 \oplus pos_e^5 \oplus ne_e^1 \oplus ne_e^2 \oplus ne_e^4 \oplus ne_e^5$

4. Experimental Setup

4.1. Dataset

We used the Marmara Turkish Coreference Corpus [31] as our dataset which contains 33 documents from the METU-Sabancı Turkish Treebank Corpus [50]. Each document consists of 26 to 424 sentences and the sentences were manually annotated with mentions (between 17 and 359 mentions in each document). The largest text spans that refer to real-word entities were identified as mentions and annotated mentions do not overlap. Only coreferent mentions were tagged and singleton mentions were left unannotated. The dataset contains 5,170 mentions with the majority consisting of a single word (4,133 mentions). Multi-word mentions are formed by at least 2 words (652 mentions) and at most 24 words (2 mentions).

There is a high variety in part-of-speech tags of words that form the mentions. Figure 2 shows the POS tags of single-word and multi-word mentions (i.e., the tags of the first and last words) that appear at least 20 times in the dataset. The POS tags correspond to manually annotated tags of words in the treebank corpus.

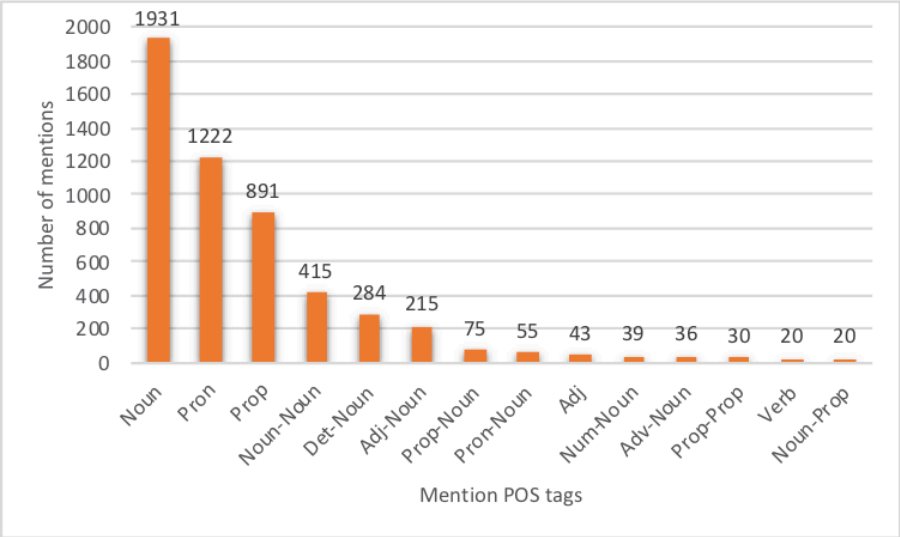

Figure 2: POS tag statistics of mentions.

As expected, single-word mentions often consist of a noun (Noun), pronoun (Pron), or proper noun (Prop). In total, 14 different POS tags were used to annotate the mentions including verb (Verb), adverb (Adv), determiner (Det), and number (Num).

In the dataset, coreferent mentions that refer to the same entity were grouped into 944 coreference chains. The majority of these chains contain 2 (365 chains) or 3 (179 chains) mentions as shown in Figure 3. The number of chains that contain more than 20 mentions is 45 with the longest chain having 66 mentions. The mentions of the same chain are often not interleaved by the mentions of other chains in the documents. On the other hand, at most 14 other mentions appear in between coreferent mentions of the same chain and in slightly more than half of the cases, coreferent mentions are interleaved by at most 3 other mentions. The following is a representative chain from the dataset that consists of three multi-word coreferent mentions.

- Ve hiç kimse benim eşyamla ilgili düşüncelerimi de merak etmiyordur kuşkusuz
  And of course, no one is wondering what I think about my stuff

- Ama bu düşünceleri neden buraya aktardığım, elbette merak edilecektir
  But, why I transfer these thoughts here, of course, will be wondered

- Bu satırlar, o düşüncelerin kağıda dökülmüşüdür
  These lines are those thoughts that are put on paper

4.2. Input Embedding

Last decade has witnessed the increasing use of contextualized (context-dependent) word representations generated by transformer-based language models and their contribution to system performances in several language processing tasks [51]. In morphologically-rich languages such as Turkish, the high number of possible word forms and the limit on the vocabulary size of a language model increase the importance of the tokenization

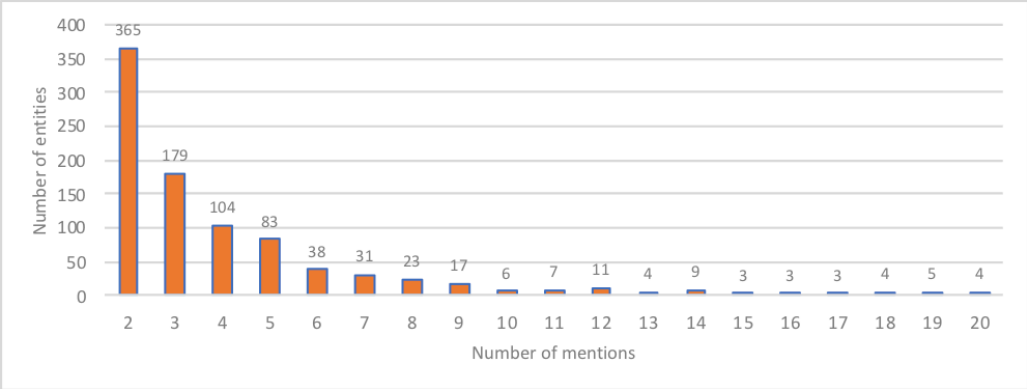

Figure 3: Coreference chain statistics.

method and the kind of embedding used in representing data. Moreover, our previous study showed that the use of different embeddings to represent mention spans plays an important role in capturing the contextualized similarity between Turkish coreferent mentions [52].

In order to obtain word embeddings (w_e) that are input to our models, we explored the use of several transformer-based language models trained using different tokenization methods for Turkish. We used publicly available Turkish language models and their associated tokenizers. In particular, we used the character-based model (**CM**) [53] with ByT5 tokenizer, the subword-based models [53] with Byte-Pair Encoding [54] (**SM_BPE**) and WordPiece [55] (**SM_WP**) tokenizers, and the word-based model (**WM**)[1] with WordPiece tokenizer. In all cases, we used the average of token embeddings that belong to a word as its final embedding.

We represented POS embedding (pos_e) and named-entity embedding (ne_e) by using one-hot encoded vectors. We utilized a Turkish morphological parser and disambiguator to obtain POS tags of words [56] and a Turkish named-entity recognizer [57] to identify named-entities in input text. Both tools are publicly available.

4.3. Parameter Settings

Single word and multi-word mentions with two or three words constitute 5030 out of 5170 mentions in the coreference corpus. To reduce the computational cost, we limit the length of extracted text spans to three words in our experiments. Nonetheless, the final number of candidate spans was huge and there was an imbalance between the number of mention spans (positive samples) and not-mention spans (negative samples). Therefore, we followed two different methods to determine the not-mention spans to be used in the experiments. In the first method, for each mention span, we randomly selected a fixed number of not-mention spans that are extracted from the same sentence (**Any_Corpus**). However, using the second methodology, we randomly selected a fixed number of not-mention spans from the same sentence that share at least one word in common with the mention span (**Common_Corpus**). The second corpus makes the detection task more challenging since not-mention spans share some commonality with mention spans but do not reflect their correct boundaries. During the evaluation study, we experimented with different number of not-mention spans (i.e., 3, 5, and 10) in both cases. The data was split into training, validation, and test portions using 80%-10%-10% scheme and the performances of FL_Model and PFLN_Model were tested on both corpora. Each model architecture was trained with different language models and tokenizers using 5 epochs. The word-based language model produced

[1]https://huggingface.co/dbmdz/bert-base-turkish-128k-cased

Table 2: Evaluation results of FL_Model on both corpora.

		Any_Corpus		*Common_Corpus*	
Embedding Combination	*Language Model*	F-Measure	Accuracy	F-Measure	Accuracy
Word ⊕ Pos ⊕ Ner	CM	0.913	0.938	0.839	0.873
	SM_BPE	0.931	0.950	0.840	0.874
	SM_WP	0.935	0.954	0.843	0.877
	WM	0.971	0.978	0.863	0.893
Word ⊕ Ner	CM	0.903	0.931	0.825	0.870
	SM_BPE	0.923	0.945	0.832	0.871
	SM_WP	0.925	0.946	0.840	0.875
	WM	0.972	0.979	0.868	0.897
Word ⊕ Pos	CM	0.903	0.931	0.814	0.863
	SM_BPE	0.927	0.947	0.830	0.868
	SM_WP	0.932	0.948	0.836	0.870
	WM	0.971	0.978	0.867	0.896
Word	CM	0.902	0.930	0.810	0.860
	SM_BPE	0.922	0.944	0.835	0.872
	SM_WP	0.929	0.950	0.836	0.873
	WM	0.970	0.977	0.865	0.896

word embeddings of length 768 whereas the character-based and subword-based models produced embeddings of length 512. The number of hidden units in feed-forward networks was set to 50. During training, the Adam optimizer with a learning rate of 0.00005 was used. The evaluation results were measured using f-measure and accuracy metrics and the average results of 10-fold cross validation were reported. The average training and testing times of the FL_Model was computed as 30.97 (12.7-81.86) seconds and 0.15 (0.07-0.38) seconds and those of the PFLN_Model was measured as 31.34 (12.63-76.75) seconds and 0.16 (0.08-0.41) seconds.

5. Results and Discussion

5.1. Model Experiments

We conducted several experiments to evaluate the performances of FL_Model and PFLN_Model on Any_Corpus and Common_Corpus. In the first set of experiments, we limit the number of not-mention spans selected for each mention to three. We also explored the impact of using different span representations in both models. In particular, we experimented with different contextual representations of words (language models) and combinations of embeddings that form final span representations. To our best knowledge, there is not any standalone mention detection system for Turkish that we can use to compare our model performances.

As shown in Table 2, the FL_Model achieves the highest performance by using word and named-entity embeddings to represent mention spans on both corpora where word embeddings are obtained from the word-based language model WM. The model receives the lowest performance scores once only word embeddings learned from the character-based language model CM are used. The scores also demonstrate that using word-based and character-based language models results in the highest and lowest scores for each embedding combination, respectively. The performance of using a subword-based language model shows the same pattern on both corpora where the model trained with byte-pair encoding achieves lower scores than the model with wordpiece encoding.

According to the results shown in Table 3, the highest performance of the PFLN_Model is achieved with different embedding combinations along with the word-based language model on our corpora. The concatenation

Table 3: Evaluation results of PFLN_Model on both corpora.

		Any_Corpus		*Common_Corpus*	
Embedding Combination	*Language Model*	F-Measure	Accuracy	F-Measure	Accuracy
Word ⊕ Pos ⊕ Ner	CM	0.935	0.953	0.820	0.865
	SM_BPE	0.949	0.963	0.842	0.877
	SM_WP	0.957	0.969	0.847	0.881
	WM	0.980	0.985	0.881	0.906
Word ⊕ Ner	CM	0.930	0.950	0.821	0.867
	SM_BPE	0.944	0.960	0.843	0.877
	SM_WP	0.953	0.966	0.847	0.881
	WM	0.980	0.985	0.880	0.905
Word ⊕ Pos	CM	0.930	0.950	0.814	0.862
	SM_BPE	0.946	0.961	0.839	0.874
	SM_WP	0.954	0.960	0.845	0.877
	WM	0.981	0.986	0.880	0.905
Word	CM	0.929	0.949	0.810	0.860
	SM_BPE	0.944	0.960	0.843	0.877
	SM_WP	0.954	0.966	0.848	0.882
	WM	0.980	0.985	0.878	0.904

of word and POS embeddings in span representations is found to be the best configuration of the PFLN_Model on Any_Corpus whereas the use of word, POS, and named-entity embeddings in span representations obtains highest evaluation scores on Common_Corpus. As also observed in the FL_Model, the use of word embeddings with character-based language model results in the lowest f-measure and accuracy scores. Moreover, the results of subword-based language models demonstrate that using wordpiece encoding enables the model to reach a higher performance once compared to byte-pair encoding. Lastly, the PFLN_Model behaves similarly to the FL_Model in that higher scores on Any_Corpus are obtained for all embedding combinations and language models in comparison to Common_Corpus.

Our evaluation results reveal that the PFLN_Model yields the highest performance on both corpora with different configurations according to f-measure and accuracy metrics. The use of word embeddings learned from word-based language model along with POS embeddings is observed to be the best configuration on Any_Corpus whereas the inclusion of all embeddings in mention span representations where word-based language model is used is found to be the best configuration on Common_Corpus. Moreover, the PFLN_Model performs better than the FL_Model on both corpora when word-based language model or subword-based language model with wordpiece encoding scheme is used. On contrary, the use of character-based language model on Common_Corpus results in the FL_Model to perform better than the PFLN_Model. As shown in Figure 4, the highest improvement of the PFLN_Model over the FL_Model on Any_Corpus is achieved with the configuration where word and named-entity embeddings learned from subword-based language model with wordpiece encoding are used together for representing mention spans. However, the biggest performance gain on Common_Corpus is observed when all embeddings learned from word-based language model are used as shown in Figure 5. We applied Wilcoxon signed-rank test to the predictions made by our models and the results were found to be statistically significant.

The experiments also enabled us to compare the impact of augmenting word embeddings with POS and named-entity embeddings individually or together on performances. Here, we considered the model performance where only word embeddings are used as baseline. Our observations with FL_Model and PFLN_Model reveal

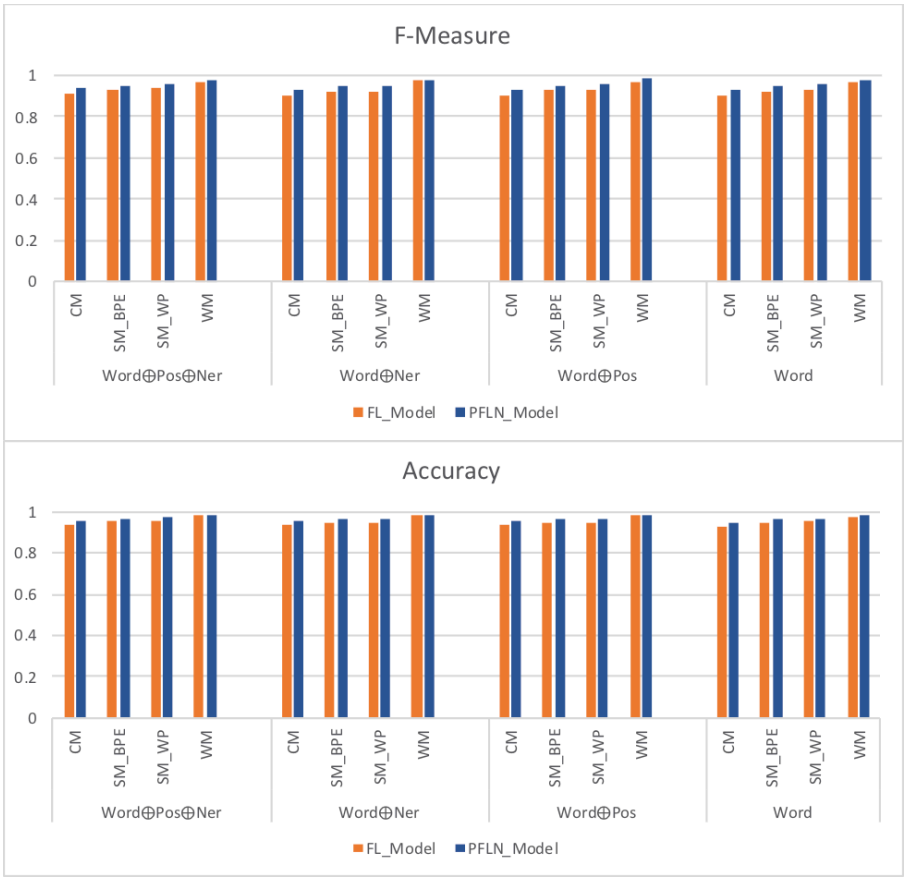

Figure 4: The comparison of FL_Model and PFLN_Model on Any_Corpus.

that incorporating both embeddings in mention span representation most of the time improves the performance more than the configurations where only one of them is used along with word embeddings. Additionally, in the FL_Model, incorporating only named-entity embeddings results in a higher performance gain as compared to the use of POS embeddings along with word embeddings on Common_Corpus. Not a clear pattern is observed on Any_Corpus for the FL_Model. Similarly on Common_Corpus, augmenting word embeddings with only named-entity embeddings in the PFLN_Model yields a higher increase once compared to augmenting with only POS embeddings. However, augmenting word embeddings with POS embedding performs better than augmenting them with only named-entity embeddings most of the time on Any_Corpus.

In the second set of experiments, we analyzed the effect of the number of not-mention spans used for training and testing the models. Here, we assessed model performances for the cases where 3, 5, or 10 not-mentions are randomly selected from Any_Corpus and Common_Corpus, respectively. In both models, we observe that increasing the number of not-mentions and hence obtaining an even more imbalanced data negatively affect f-measure scores on all corpora. On the other hand, relatively more imbalanced data results in higher accuracy scores. Figure 6 shows how the FL_Model reacts to the changes in the number of not-mention spans on Any_Corpus once all embeddings are used to represent mention spans. A similar behaviour is observed for both models with all configurations on all corpora. We argue that our models are performing well in classifying not-mention spans but is not able to classify mention spans at the same correctness rate.

Figure 5: The comparison of FL_Model and PFLN_Model on Common_Corpus.

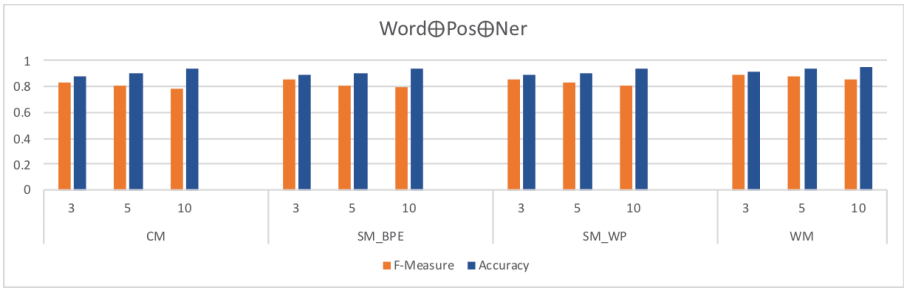

Figure 6: The performance of FL_Model on Any_Corpus according to different number of not-mentions.

5.2. Comparison with Sequence Labeling Model

The sequence labeling problem is formalized as given a sequence $X = (x_1, x_2, ..., x_n)$ of length n where x_i is the ith word, generate an output sequence of the same length $Y = (y_1, y_2, ..., y_n)$ such that y_i is the predicted label of x_i. Each output label y_i corresponds to a label from a predefined list (e.g., named-entity or part-of-speech tags). In the literature, mention detection task has also been addressed as a sequence labeling problem [38]. By following this approach, we developed a transformer-based sequence labeling model and compared its performance with our models. For this purpose, we first annotated all sentences in our dataset

using the BIO tag format with the labels 'B-MENT' (beginning of a mention span), 'I-MENT' (inside a mention span), and 'O' (outside of a mention span). For instance, the sentence with two mention spans ('bu kente' and 'onu') is annotated as "Bugün/O ,/O bu/B-MENT kente/I-MENT geleli/O beşinci/O gün/O ,/O hiçbir/O yerde/O bulabilmiş/O değilim/O onu/B-MENT ./O" *(Today is the fifth day since I came to this city , I can't find him anywhere .)*. The sequence labeling model received sentences and their labels as input-output pairs.

Our evaluation results with classification models demonstrated the superiority of word-based language model (WM) on both corpora. Thus, we fine-tuned Bidirectional Encoder Representations from Transformers (BERT) model with a sequence classification head using WM as our labeling model. In this experiment, the dataset was splitted into 80% training, 10% validation, and 10% test data. The model was trained using 10 and 30 epochs, respectively. The average scores of 10-fold cross validation were computed as the model performance. With 10 epochs, the model achieved an f-measure of 0.685 and an accuracy of 0.896. On the other hand, the model received 0.701 f-measure and 0.90 accuracy with 30 epochs. We compared these results with the scores measured for our classification models where only word embeddings are used on both corpora. Our analysis showed that the prediction accuracy and f-measure of our classification models on Any_Corpus are higher than the tagging accuracy and f-measure of the labeling model. On Common_Corpus, the classification models outperformed the labeling model according to f-measure. However, the tagging accuracy of the labeling model was found to be slightly higher than the classification accuracy of the FL_Model but lower than that of the PFLN_Model. Although our classification models outperformed the sequence labeling model in most cases, we argue that both models demonstrate acceptable performances on Turkish mention detection task.

6. Conclusion

This article describes our work on mention detection in Turkish where two neural models are introduced. Given a text, our models first extract all text spans up to a length as candidate mention spans. These candidates are then classified as mentions or not-mentions by using a feed-forward neural network with sigmoid function at top. The first model represents each candidate span by focusing on its first and last words whereas the second model also takes into account the previous and next words of the span as it appears in a sentence. The words in mention span representations are encoded using their contextual embeddings, part-of-speech embeddings, and named-entity embeddings. Different context-dependent Turkish language models are used to obtain contextual embeddings of words. Our evaluation studies reveal that the second model performs better than the first model in general and the use of word-based language model for obtaining word embeddings improves model performances more than what can be achieved with other language models. Moreover, our models surpass the performance of a transformer-based sequence labeling model in detecting mentions.

Although our work fills a gap in the existing literature by providing a neural mention detector for Turkish, we have several future research directions. We plan to integrate our best performing model into our end-to-end coreference resolution system [29] and measure its extrinsic performance in the underlying system. Another major area is to enhance our mention detection models so that nested coreferent mentions can be handled. An interesting research direction is to identify and filter singleton mentions from candidate mentions spans. Finally, we have plans to explore more features in mention span representations and assess the impact of the number of words in candidate spans on model performances.

References

[1] Liu R, Mao R, Luu AT, Cambria E. A brief survey on recent advances in coreference resolution. Artificial Intelligence Review 2023; 56: 14439–14481. https://doi.org/10.1007/s10462-023-10506-3

[2] Cruz AF, Rocha G, Cardoso HL. Coreference Resolution: Toward End-to-End and Cross-Lingual Systems. Information 2020; 11 (2): 74:1-74:23. https://doi.org/10.3390/info11020074

[3] Werlen LM, Popescu-Belis A. Using Coreference Links to Improve Spanish-to-English Machine Translation. In: The 2nd Workshop On Coreference Resolution Beyond OntoNotes (CORBON); Valencia, Spain; 2017. pp. 30-40.

[4] Bhattacharjee S, Haque R, de Buy Wenniger GM, Way A. Investigating Query Expansion and Coreference Resolution in Question Answering on BERT. In: Métais E, Meziane F, Horacek H, Cimiano P (editor). Natural Language Processing And Information Systems. Saarbrücken, Germany: Springer, 2020, pp. 47-59.

[5] Kriman S, Ji H. Joint Detection and Coreference Resolution of Entities and Events with Document-level Context Aggregation. In: The 59th Annual Meeting Of The Association For Computational Linguistics And The 11th International Joint Conference On Natural Language Processing Student Research Workshop; 2021. pp. 174-179.

[6] Lata K, Singh P, Dutta K. Mention Detection in Coreference Resolution: Survey. Applied Intelligence 2022; 52: 9816-9860. https://doi.org/10.1007/s10489-021-02878-2

[7] Toldova S, Ionov M. Mention Detection for Improving Coreference Resolution in Russian Texts: A Machine Learning Approach. Computacion Y Sistemas 2016; 20 (4): 681-696. https://doi.org/10.13053/cys-20-4-2480

[8] Li K, Huang H, Guo Y, Jian P. Singleton Detection for Coreference Resolution via Multi-window and Multi-filter CNN. In: China Workshop on Machine Translation; Dalian, China; 2017. pp. 9-19.

[9] Sapena E, Padró L, Turmo J. RelaxCor: A Global Relaxation Labeling Approach to Coreference Resolution. In: The 5th International Workshop On Semantic Evaluation; Beijing, China; 2010. pp. 88-91.

[10] Rahman A, Ng V. Coreference Resolution with World Knowledge. In: The 49th Annual Meeting Of The Association For Computational Linguistics: Human Language Technologies; Portland, Oregon, USA; 2011. pp. 814-824.

[11] Zhou H, Li Y, Huang D, Zhang Y, Wu C et al. Combining Syntactic and Semantic Features by SVM for Unrestricted Coreference Resolution. In: The Fifteenth Conference On Computational Natural Language Learning: Shared Task; Portland, Oregon, USA; 2011. pp. 66-70.

[12] Haponchyk I, Moschitti A. Making Latent SVMstruct Practical for Coreference Resolution. In: The First Italian Conference On Computational Linguistics (CLIC-it); Pisa, Italy; 2014. pp. 9-11.

[13] Veena G, Gupta D, Daniel AN, Roshny S. A learning method for coreference resolution using semantic role labeling features. In: International Conference On Advances In Computing, Communications And Informatics (ICACCI); Udupi, India; 2017. pp. 67-72.

[14] Fei H, Li X, Li D, Li P. End-to-end Deep Reinforcement Learning Based Coreference Resolution. In: The 57th Annual Meeting Of The Association For Computational Linguistics; Florence, Italy; 2019. pp. 660-665.

[15] Heusden R, Kamps J, Marx M. Neural Coreference Resolution for Dutch Parliamentary Documents with the DutchParliament Dataset. Data 2023; 8 (2): 34:1:34:16. https://doi.org/10.3390/data8020034

[16] Kim D, Park S, Han M, Kim H. Pipeline Coreference Resolution Model for Anaphoric Identity in Dialogues. In: The CODI-CRAC 2022 Shared Task On Anaphora, Bridging, And Discourse Deixis In Dialogue; Gyeongju, Republic of Korea; 2022. pp. 28-31.

[17] Clark K, Manning C. Deep Reinforcement Learning for Mention-Ranking Coreference Models. In: The Conference On Empirical Methods In Natural Language Processing; Austin, Texas, USA; 2016. pp. 2256-2262.

[18] Stylianou N, Vlahavas I. A neural Entity Coreference Resolution review. Expert Systems With Applications 2021; 168: 114466:1-114466:20. https://doi.org/10.1016/j.eswa.2020.114466

[19] Lee K, He L, Lewis M, Zettlemoyer L. End-to-end Neural Coreference Resolution. In: The Conference On Empirical Methods In Natural Language Processing; Copenhagen, Denmark; 2017. pp. 188-197.

[20] Aloraini A, Yu J, Poesio M. Neural Coreference Resolution for Arabic. In: The Third Workshop On Computational Models Of Reference, Anaphora And Coreference; Barcelona, Spain; 2020. pp. 99-110.

[21] Matej K, Slavko Ž. Neural coreference resolution for Slovene language. Computer Science And Information Systems 2022; 19 (2): 495-521. https://doi.org/10.2298/CSIS201120060K

[22] Nitoń B, Morawiecki P, Ogrodniczuk M. Deep Neural Networks for Coreference Resolution for Polish. In: The Eleventh International Conference On Language Resources And Evaluation (LREC); Miyazaki, Japan; 2018. pp. 395-400.

[23] Schröder F, Hatzel H, Biemann C. Neural End-to-end Coreference Resolution for German in Different Domains. In: The 17th Conference On Natural Language Processing (KONVENS); Düsseldorf, Germany; 2021. pp. 170-181.

[24] Grobol L. Neural Coreference Resolution with Limited Lexical Context and Explicit Mention Detection for Oral French. In: The Second Workshop On Computational Models Of Reference, Anaphora And Coreference; Minneapolis, USA; 2019. pp. 8-14.

[25] Lata K, Singh P, Dutta K. A comprehensive review on feature set used for anaphora resolution. Artificial Intelligence Review 2021; 54: 2917-3006. https://doi.org/10.1007/s10462-020-09917-3

[26] Yıldırım S, Kılıçaslan Y, Yıldız T. Pronoun Resolution in Turkish Using Decision Tree and Rule-Based Learning Algorithms. In: Vetulani Z, Uszkoreit H (editor). Human Language Technology, Challenges of the Information Society (LTC). Berlin, Germany: Springer, 2009, pp. 270-278.

[27] Kılıçaslan Y, Güner E, Yıldırım S. Learning-based pronoun resolution for Turkish with a comparative evaluation. Computer Speech & Language 2009; 23 (3): 311-331. https://doi.org/10.1016/j.csl.2008.09.001

[28] Pamay T, Eryiğit G. Turkish Coreference Resolution. In: Innovations In Intelligent Systems And Applications (INISTA); Thessaloniki, Greece; 2018. pp. 1-7.

[29] Demir Ş. Neural Coreference Resolution for Turkish. Journal Of Intelligent Systems: Theory And Applications 2023; 6 (1): 85-95. https://doi.org/10.38016/jista.1225097

[30] Pamay Arslan T, Eryiğit G. Incorporating Dropped Pronouns into Coreference Resolution: The case for Turkish. In: The 17th Conference Of The European Chapter Of The Association For Computational Linguistics: Student Research Workshop; Dubrovnik, Croatia; 2023. pp. 14-25.

[31] Schüller P, Cıngıllı K, Tunçer F, Sürmeli B, Pekel A et al. Marmara Turkish Coreference Corpus and Coreference Resolution Baseline. ArXiv 2018.

[32] Lee H, Chang A, Peirsman Y, Chambers N, Surdeanu M et al. Deterministic Coreference Resolution Based on Entity-Centric, Precision-Ranked Rules. Computational Linguistics 2013; 39 (4): 885-916. https://doi.org/10.1162/COLI_a_00152

[33] Ogrodniczuk M, Wójcicka A, Głowińska K, Kopeć M. Detection of Nested Mentions for Coreference Resolution in Polish. In: Przepiórkowski A, Ogrodniczuk M (editor). Advances In Natural Language Processing. Germany: Springer, 2013, pp. 270-277.

[34] Ittycheriah A, Lita L, Kambhatla N, Nicolov N, Roukos S et al. Identifying and Tracking Entity Mentions in a Maximum Entropy Framework. In: North American Chapter of the Association for Computational Linguistics; Edmonton, Canada; 2003. pp. 40-42.

[35] Uryupina O, Moschitti A. Multilingual Mention Detection for Coreference Resolution. In: The Sixth International Joint Conference On Natural Language Processing; Nagoya, Japan; 2013. pp. 100-108.

[36] Daumé III H, Marcu D. A Large-Scale Exploration of Effective Global Features for a Joint Entity Detection and Tracking Model. In: Human Language Technology Conference And Conference On Empirical Methods In Natural Language Processing; Vancouver, British Columbia, Canada; 2005. pp. 97-104.

[37] Rahman A, Ng V. Supervised Models for Coreference Resolution. In: The Conference On Empirical Methods In Natural Language Processing; Singapore; 2009. pp. 968-977.

[38] Nguyen T, Sil A, Dinu G, Florian R. Toward Mention Detection Robustness with Recurrent Neural Networks. ArXiv 2016.

[39] Miculicich L, Henderson J. Partially-supervised Mention Detection. In: The Third Workshop On Computational Models Of Reference, Anaphora And Coreference; Barcelona, Spain; 2020. pp. 91-98.

[40] Park C, Lee C, Lim S. Mention Detection Using Pointer Networks for Coreference Resolution. ETRI Journal 2017; 39 (5): 652-661. https://doi.org/10.4218/etrij.17.0117.0140

[41] Wang B, Lu W, Wang Y, Jin H. A Neural Transition-based Model for Nested Mention Recognition. In: The Conference On Empirical Methods In Natural Language Processing; Brussels, Belgium; 2018. pp. 1011-1017.

[42] Grenander M, Cohen S, Steedman M. Sentence-Incremental Neural Coreference Resolution. In: The 2022 Conference On Empirical Methods In Natural Language Processing; Abu Dhabi, United Arab Emirates; 2022. pp. 427-443.

[43] Zhang R, Santos C, Yasunaga M, Xiang B, Radev D. Neural Coreference Resolution with Deep Biaffine Attention by Joint Mention Detection and Mention Clustering. In: The 56th Annual Meeting Of The Association For Computational Linguistics; Melbourne, Australia; 2018. pp. 102-107.

[44] Yu J, Bohnet B, Poesio M. Neural Mention Detection. In: The Twelfth Language Resources And Evaluation Conference; Marseille, France; 2020. pp. 1-10.

[45] Wang Y, Jin H. Hybrid Rule-Neural Coreference Resolution System based on Actor-Critic Learning. ArXiv 2022.

[46] Barua A, Sharma P, Clark K. Coreferent Mention Detection using Deep Learning. Semantic Scholar 2017.

[47] Haagsma H. Singleton Detection using Word Embeddings and Neural Networks. In: The ACL 2016 Student Research Workshop; Berlin, Germany; 2016. pp. 65-71.

[48] Lata K, Singh P, Dutta K. SMDDH: Singleton Mention detection using Deep Learning in Hindi Text. ArXiv 2023.

[49] Xu M, Jiang H, Watcharawittayakul S. A Local Detection Approach for Named Entity Recognition and Mention Detection. In: The 55th Annual Meeting Of The Association For Computational Linguistics; Vancouver, Canada; 2017. pp. 1237-1247.

[50] Say B, Zeyrek Bozşahin D, Oflazer K, Özge U. Development of a corpus and a treebank for present-day written Turkish. In: The Eleventh International Conference Of Turkish Linguistics; Famagusta, North Cyprus; 2002. pp. 183-192.

[51] Biś D, Podkorytov M, Liu X. Too Much in Common: Shifting of Embeddings in Transformer Language Models and its Implications. In: The Conference Of The North American Chapter Of The Association For Computational Linguistics: Human Language Technologies; 2021. pp. 5117-5130.

[52] Demir S. What Word Embeddings Tell Us About Turkish Coreference Resolution. In: The International Conference on Informatics and Computer Science; 2022. pp. 39-45.

[53] Toraman C, Yilmaz E, Sahinuc F, Ozcelik O. Impact of Tokenization on Language Models: An Analysis for Turkish. ACM Transactions on Asian Low-Resource Language Information Processing 2023; 22 (4): 1-21. https://doi.org/10.1145/3578707

[54] Sennrich R, Haddow B, Birch A. Neural Machine Translation of Rare Words with Subword Units. In: The 54th Annual Meeting Of The Association For Computational Linguistics; Berlin, Germany; 2016. pp. 1715-1725.

[55] Schuster M, Nakajima K. Japanese and Korean voice search. In: IEEE International Conference On Acoustics, Speech And Signal Processing (ICASSP); Kyoto, Japan; 2013. pp. 5149-5152.

[56] Sak H, Güngör T, Saraçlar M. Morphological Disambiguation of Turkish Text with Perceptron Algorithm. In: Gelbukh A (editor). Computational Linguistics And Intelligent Text Processing. Berlin, Germany: Springer, 2007, pp. 107-118.

[57] Güngör O, Uskudarli S, Güngör T. Improving Named Entity Recognition by Jointly Learning to Disambiguate Morphological Tags. In: The 27th International Conference On Computational Linguistics; New Mexico, USA; 2018. pp. 2082-2092.