applied sciences

MDPI

*Article*

# Contrastive Learning for Morphological Disambiguation Using Large Language Models in Low-Resource Settings

**Gulmira Tolegen [1,2,*], Alymzhan Toleu [1,2] and Rustam Mussabayev [1,2]**

[1] AI Research Laboratory, Satbayev University, Almaty 050040, Kazakhstan; ru.mussabayev@satbayev.university (R.M.)
[2] Laboratory of Analysis and Modelling of Informational Processes, Institute of Information and Computational Technologies, Almaty 050010, Kazakhstan
* Correspondence: gulmira.tolegen.cs@gmail.com or g.tolegen@satbayev.university

**Abstract:** In this paper, a contrastive learning approach for morphological disambiguation (MD) using large language models (LLMs) is presented. A contrastive loss function is introduced for training the approach, which reduces the distance between the correct analysis and contextual embeddings while maintaining a margin between correct and incorrect embeddings. One of the aims of the paper is to analyze the effects of fine-tuning an LLM on MD in morphologically complex languages (MCLs) with special reference to low-resource languages such as Kazakh, as well as Turkish. Another goal of the paper is to consider various distance measures for this contrastive loss function, aiming to achieve better results when performing disambiguation by computing the distance between the context and the analysis embeddings. The existing approaches for morphological disambiguation, such as HMM-based and feature-engineering approaches, have limitations in modeling long-term dependencies and in the case of large, sparse tagsets. These challenges are mitigated in the proposed approach by leveraging LLMs, thus achieving better accuracy in handling the cases of ambiguity and OOV tokens without the need to rely on other features. Experiments were conducted on three datasets for two MCLs, Kazakh and Turkish—the former is a typical low-resource language. The results revealed that the proposed approach with contrastive loss improves MD performance when integrated with knowledge from large language models.

**Keywords:** morphological disambiguation; large language models; low-resource language; contrastive learning

check for updates

**Citation:** Tolegen, G.; Toleu, A.; Mussabayev, R. Contrastive Learning for Morphological Disambiguation Using Large Language Models in Low-Resource Settings. *Appl. Sci.* **2024**, *14*, 9992. https://doi.org/10.3390/app14219992

Academic Editor: Andrea Prati

Received: 6 October 2024
Revised: 24 October 2024
Accepted: 27 October 2024
Published: 1 November 2024

**Copyright:** © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

## 1. Introduction

Morphological disambiguation (MD) is a long-standing problem in processing text for morphologically complex languages. It is similar to part-of-speech (POS) tagging [1]; however, for MD, not only the POS tag but the lemma/root along with its corresponding morphological tags should be correctly predicted.

Considering the analysis involves various components like root forms, part-of-speech tags, and morpheme chains, treating MD as a straightforward tagging task introduces complexity, leading to an expansive tagset with sparse data points. To address this issue, several strategies have been proposed. A common approach is to decompose the tag sequence into smaller, more manageable segments. For example, in the HMM-based method proposed by Hakkani-Tur et al. [2], the analysis is broken down into smaller parts like inflectional groups. This method operates under the assumption that the tags in the current analysis depend only on the previous one, simplifying the task of disambiguation. However, this assumption has significant drawbacks. First, it limits the model's ability to capture long-term dependencies, which are essential for this type of task. Second, despite the decomposition, the overall tagset remains quite large, making the approach less efficient.

To alleviate this issue, a voted perception approach is proposed in [3]; instead of using a certain part of the analysis, the author proposed a set of features and sought to represent a sequence analysis with feature vectors. The approach utilized tri-gram decoding, which relaxes the previous assumption and, compared to the HMM-based approach, may better capture longer dependencies between tags. The underlying hypothesis of this approach is that the model will maximize the objective function, which ensures the feature vectors from the correct path of analyses obtain larger values than those in the non-correct path of analyses. This is a discrete feature-based approach, which requires manual feature engineering and it still cannot capture long-term dependencies.

To address this issue, a deep learning-based approach was proposed [4], which is currently considered the best for this task. The authors segment an analysis into (i) the root, (ii) its POS and (iii) the morpheme chain (MC), then use a nonlinear layer to calculate a dense representation for analysis. In order to capture the long-term dependencies, a bi-directional long short-term memory (LSTM) method was applied for context learning from sentences. Another character-level LSTM is utilized for capturing word internal features. Combing the fine-grained information from characters, the contextualized representation obtained for each word, and guided by the intuition that the correct analysis should be most similar to the context's representation, a dot product of two representations is computed to perform disambiguation. One disadvantage of this approach is that it uses a binary vector to calculate analysis embeddings, which are not in continuous space.

Another limitation is that it relies on a biLSTM to compute the contextual representation, which may not fully capture the complexities of language use in syntactic and semantic contexts as effectively as embeddings generated from large language models. With the advent of large language models (LLMs) [5–7], many results for downstream tasks of natural language processing (NLP) [8,9] have improved significantly. However, the impact of pre-trained large language models on the performance of MD remains unclear.

In this paper, a contrastive learning approach for MD using LLM is presented. A contrastive loss function is introduced for training the approach, which reduces the distance between the correct analysis and contextual embeddings while maintaining a margin between correct and incorrect embeddings. One of the aims of this work is to analyze the effectiveness of LLMs on MD and how the knowledge encoded in LLMs can be transferred to low-resource languages to improve model performance. Another aim of this work is to experimentally analyze the effectiveness of different distance measurements for performing morphological disambiguation by calculating the distance between context and morphological analysis' embeddings via the contrastive loss function. Experimental results show that the model incorporated with knowledge from an LLM gives better results while not using any designed internal or external features. The results for comparing different distance measurements for performing disambiguation show that the Euclidean distance and dot product yield a better outcome than others. The results showed that LLMs are useful in enhancing the performance of MD for various MCLs, especially for low-resource languages.

The structure of the paper is organized as follows: (i) In Section 1, we introduce the task and the purpose of this work; (ii) Section 2 describes the existing work related to MD; (iii) Section 3 details the proposed model; (iv) In Section 4, we report the experimental results; and in Section 5, the most common error cases are discussed with statistics, and the paper concludes in Section 6.

## 2. Related Work

A morphological analyzer generates a set of morphological analyses for an input word, and the MD task performs morphological disambiguation by choosing a possible analysis among the candidates depending on the context. Morphological disambiguation has been studied extensively over the past decades, especially for agglutinative languages like Turkish and Kazakh.

Approaches for morphological ambiguity resolution can be categorized into three problem groups as follows: (i) sequence labeling problem; (ii) morphological disambiguation problem; (iii) sequence-to-sequence (Seq2Seq) problem.

Sequence labeling is a type of problem in NLP where the goal is to assign a categorical label to each token in a sequence of tokens. This type of problem is essential in various NLP tasks, such as POS tagging [10] and named entity recognition (NER) [11]. In a sequence labeling problem, an input sequence $X = (x_1, x_2, \ldots, x_n)$ of length $n$ is given, and the task is to predict a corresponding sequence of labels $Y = (y_1, y_2, \ldots, y_n)$. Each label $y_i$ corresponds to an input element $x_i$, and the labels are drawn from a predefined set of categories. MD is treated as a sequence labeling problem, similar to POS tagging. However, MD is more complex than POS tagging because its labels include the root, POS tag, and a sequence of morpheme tags.

There are two ways of performing MD as sequence labeling: (i) treat each morphological analysis as a label; usually they are drawn from the training as a predefined set of categories. Since a morphological analysis contains the root, POS tag, and a sequence of morpheme tags, it results in a large number of unique labels. It not only increases label data sparsity but also leads to a potential issue of out-of-vocabulary for a label. (ii) To avoid this issue, most of the sequence labeling approaches for MD use a multi-class and multi-label model to predict different morphological categories. For each morphological category, there is a separate classifier.

In [12], the authors proposed different models with different architectures for morphological tagging. In their multiclass, multilabel (McMI) model, which predicts POS, different morphological categories are employed separately as the output of the model; however, they share an input layer.

Since the morphological tagging contains a tag sensitivity issue (one tag may depend on previous tags), to capture this information, the authors proposed a hierarchical multiclass, multilabel model (HMcMI); in this architecture, the authors only consider the POS tags' sensitivity. Another approach was the testing of a sequence model (Seq); this takes each word in a sentence, and feeds it to a long short-term memory LSTM [13]) network as a context vector; then, for the decoder, it generates category-value pairs.

The authors take a multiclass model (MC) as their baseline, treating an analysis as a label. The experiments were conducted on UDv2.1 corpora for 49 languages. The experiments showed that, on average, the Seq model performed the best compared to the others. For the OOV case, no significant difference was observed between the HMcMI and McMI models. For large datasets, it seems that the baseline MC outperformed these two models. For POS tagging, HMcMI outperformed McMI. This type of approach is efficient for languages with less complex morphology, such as English and others; it treats the problem as a morphological tagging problem, which essentially involves predicting the entire set of morphological tags given the context.

In contrast to predicting the entire analysis given a word context, treating it as a disambiguation problem uses all candidates and selects the most probable analysis. In this direction, approaches to the MD task for Turkish began with a rule-based method reported in [14].

It uses a constrained lexical rule to select the most probable analysis among the candidates. The lexical rule contains the word case, POS tags, and the positional features of the word. Disambiguation involves using different combinations of these lexical rules to detect the pattern.

In another work, the authors explored a rule-based approach to MD, integrating a set of predefined constraint rules with an algorithm that automatically learns additional rules. It is denoted as constraint-based MD [15]. Following this constraint-based MD, a pure rules-based approach [16] was proposed for Turkish MD. The approach is similar to that described in [15], since all the rules are designed and chosen manually; the disambiguator uses more capable and descriptive formatting for the disambiguation rules.

Statistically based approaches have been proposed for extracting the lexical pattern of words to disambiguate the morphology ambiguity. In this direction, hidden Markov model (HMM)-based approaches [2] were proposed for MD by modeling the transition and emission probabilities from features of the observations. To avoid the complex structure of the MD label, the authors break down each analysis into smaller units called inflectional groups (IGs). Then, HMM is used to calculate the transition and emission probabilities between these inflectional groups. This approach makes assumptions that the current word prediction only depends on the previous words (tri-grams); in MD, it is used between IGs. For instance, in a model, the presence of IGs in a word only depends on the final IGs of the previous words.

The IGs-based model for Turkish achieves 92.08% accuracy, while the root-based model achieves 80.36%. The combined model improves the accuracy further, achieving 93.95%.

In [17], the authors proposed an open source toolkit for Turkish text processing including a morphological analyzer and disambiguation, in which they tried different methods of disambiguation as follows: (i) rote-learning disambiguator: this counts analyses observed in the training data; for testing, it picks the analysis with the highest frequency as correct; (ii) model without root: instead of performing a complete analysis, it decomposes the analysis into two parts—root $r$ and a sequence of morphological tags $a$. Then, they formulate the joint probability to $P(r, a) = p(r|a)p(a)$; (iii) model with IGs: instead of decomposing the analysis into two parts, in this model, it is split into IGs, and the joint probabilities between these IGs are modeled. Experimental results showed that the last two models achieved comparable accuracy on a test set similar to the training set (the text domain is news). The final model demonstrated better accuracy on a smaller test set that differed more significantly from the training set. It also showed that the models performed worse when the boundaries of the IGs or tags were not clear. In [18], a set of classifiers was proposed for MD; using the J48 Tree algorithm, a highest accuracy of 95.61% was obtained. Using a trigram sequence, a perceptron approach was proposed with 23 features [3], and using these features, the results of accuracy were improved from 93.95% to 96.80% in MD.

In [19], a bidirectional long short-term memory network-based neural network architecture was introduced for disambiguating morphological parses using different amounts of contextual information. The results demonstrated that the type and amount of context required for effective disambiguation vary across languages, depending on their linguistic characteristics. In languages like Turkish, where morphological information is largely conveyed by the surrounding context, models utilizing surface context can effectively capture long-range dependencies to resolve ambiguities. In contrast, languages like Arabic, where surface representations are less informative, there is a significant benefit from incorporating representations of surrounding parse candidates alongside the surface forms of neighboring words.

MD is treated as a Seq2Seq problem [20–22]. It takes the characters of a word as an input sequence, and for each word, generates a morphological analysis. Each analysis can be decomposed into a sequence of tags. Using two sequences, a sequence-to-sequence approach from machine translation can be applied. In [20], the authors proposed a neural architecture, namely, Morpheus, which is based on sequential neural encoder-decoders. It jointly solves the lemmatization and morphological tagging task. It uses a two-level LSTM network that produces context-aware vector encodings, and takes this vector as input for the decoders. The outputs are both the morphological tags associated with each word and the minimal edit operations required to transform the surface words into their respective lemmas. Previous work [21] employed an encoder-decoder framework with a bidirectional LSTM for encoding, paired with an attention mechanism during decoding to better understand the semantic relationships between the suffixes of words and other elements in a sentence. These approaches have been validated across multiple languages, including Finnish, Turkish, and English, demonstrating that such models can achieve performance that rivals or exceeds existing state-of-the-art techniques.

Overall, the shortcomings of existing approaches can be summarized as follows:

(i) The variety and complexity of morphological analysis result in a large number of unique labels. This not only increases label data sparsity but also leads to the potential issue of out-of-vocabulary labels.

(ii) The issue of long-term dependency, such as in HMM-based approaches, fails to capture the long-term dependencies between tags. While CRF-based approaches may mitigate the issue found in HMMs, they require feature engineering since features are extracted from the analysis. Additionally, CRFs still operate under a second-order HMM assumption, making it impractical to fully consider all the possible paths of different analyses.

The approach proposed in this work attempts to address these problems using contrastive learning methods. It models the analyses separately and employs a Transformer architecture to process input at the sentence level, incorporating knowledge encoded in LLMs. To our knowledge, there are few papers available on the MD task with LLMs, and the work presented here not only represents one of the first efforts to apply LLMs to this task but also introduces a contrastive learning method for the approach.

## 3. Proposed Approach

To analyze the impact of pre-trained large language models on MD and to effectively utilize the knowledge encoded in a PLM, this approach integrates contextual embeddings from a PLM with dense morphological representations to perform ambiguity resolution. Figure 1 illustrates a contrastive learning approach for morphological disambiguation. Given an input token, the LLM generates a context representation based on the token's context in the sentence. Multiple possible morphological analyses are transformed into morphological embeddings. The learning process aims to minimize the distance between the context representation and the correct morphological analysis (a1+, in blue), while maximizing the distance between the context and incorrect analyses (a2−, a3−, in red). During the learning process, the model improves its ability to select the correct morphological form by contrasting correct and incorrect analyses.

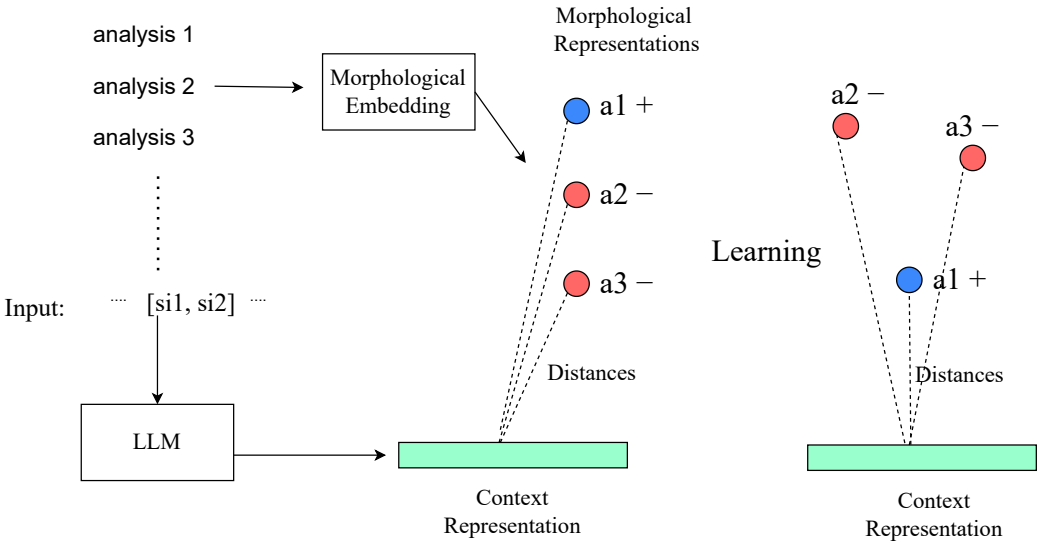



**Figure 1.** Contrastive learning for morphological disambiguation: aligning context and morphological representations.

### *3.1. Task Formulation*

Morphological disambiguation is the process of assigning the most appropriate morphological analysis to a given word form within its context. This task is important for languages with complex morphology, where a single word form can have multiple valid analyses depending on its use in a sentence. Formally, this task can be defined as follows:

Let $S = (w_1, w_2, \ldots, w_n)$ denote a sentence, where each $w_i$ is a word. For each word $w_i$, it has a set of morphological analyses, denoted by $A_i = \{a_{i1}, a_{i2}, \ldots, a_{ik}\}$. Each analysis $a_{ik}$ can be represented as a tuple $(r_{ikj}, p_{ikj}, m_{ikj})$, where:

- $r_{ikj}$ represents a root/lemma,
- $p_{ikj}$ denotes its part of speech,
- $m_{ikj}$ indicates a sequence of grammatical features.

The objective of morphological disambiguation is to identify the most probable analysis $a_i^*$ for each word $w_i$ in the sentence, such that $a_i^* \in A_i$. This prediction should maximize the overall probability of the sequence of analyses $A^* = (a_1^*, a_2^*, \ldots, a_n^*)$ given the sentence $S$.

The probability of the sequence of morphological analyses given the sentence $S$ is expressed as $P(A^* \mid S)$. This can be factorized using the chain rule:

$$P(A^* \mid S, A) = \prod_{i=1}^{n} P(a_i^* \mid S, a_1, a_2, \ldots, a_{i-1}) \tag{1}$$

where $P(a_i^* \mid S, a_1, a_2, \ldots, a_{i-1})$ represents the probability of the analyses $a_i^*$ for the word $w_i$, conditioned on the sentence $S$ and the analyses of the preceding words.

To approximate these probabilities, various models can be employed: (i) HMMs are used to capture the transition probabilities between analyses and the likelihood of words given the analyses. (ii) Models such as recurrent neural networks (RNNs) [23,24] and transformers [25] are capable of learning complex patterns and dependencies in the data, providing a better understanding of context.

### 3.2. Contextual Embedding

In MD, capturing the context in which a word appears is crucial for resolving ambiguity. Pre-trained language models (PLMs) are trained on large corpora and provide contextual embeddings that may contain syntactic and semantic information. To generate a sub-word context embedding using PLMs, a byte pair encoding (BPE) for tokenization is needed, which is a subword tokenization technique that splits words into smaller units.

Given a word $w_i$ in a sentence $S$, BPE tokenizes it into a sequence of subwords $s_{i1}, s_{i2}, \ldots, s_{im}$. To handle the alignment of these subwords with their analysis sequence, only the first subword of a word carries morphological labels, while subsequent subwords are assigned an "ignored" label. To ensure that each sentence has the same length, sequences are padded to a maximum length using special padding tokens and masks. To ensure that each word has the same number of analyses, analyses are padded to a maximum number using a special padding analysis and masks.

With a transformer-based PLM, we calculate contextual embeddings for input tokens by transferring knowledge from a PLM, capturing the fine-grained usage of words across various contexts as well as the long-term dependencies captured by multi-headed attentions. The subword tokenizer captures word-internal features, then a transformer layer calculates subword-aware representation for the input sequence, ranging from fine to coarse. More formally, let $T$ be the tokenization function using BPE. A word $w_i$ is tokenized into a sequence of subwords:

$$T(w_i) = (s_{i1}, s_{i2}, \ldots, s_{im}) \tag{2}$$

where $m$ is the number of subwords in $w_i$.

Let $E_{\text{PLM}}$ be the embedding function of PLM. For a given sentence $S$, the tokenized subwords are fed into the PLM to generate embeddings for each subword, considering the context of the entire sentence. A PLM generates embeddings for each subword in the tokenized sentence, taking the entire sentence $S$ into account for context:

$$T(S) = (T(w_1), T(w_2), \ldots, T(w_n)) \tag{3}$$

$$E_{\text{PLM}}(T(S)) = (E_{\text{PLM}}(s_1), E_{\text{PLM}}(s_2), \ldots, E_{\text{PLM}}(s_{L_{\max}})) \tag{4}$$

where $E_{\text{PLM}}(T(S))$ generates the embeddings for the sequence of subwords in the entire tokenized sentence.

### 3.3. Morphological Embedding

For a word $w_i$, its analyses can be denoted by $A_i = \{a_{i1}, a_{i2}, \ldots, a_{ik}\}$. For simplicity, we do not distinguish between the root, POS, and morpheme chain, and instead consider them as a sequence of morphological tags.

Let $a_{ik} = (t_{ik1}, t_{ik2}, \ldots, t_{ikl})$ represent a sequence of tags for the $k$-th analysis of the $i$-th word, where each tag $t_{ikl}$ can be a root, POS, or any morpheme tag. Let $\mathbf{t_{ikl}}$ (the bold indicates a vector, while non-bold represents tag.) be an embedding for the $l$-th tag of the $k$-th analysis of the $i$-th word and each analysis can have up to $L$ tags (the maximum tag length). The average embeddings of all tags in an analysis's tag sequence form a single embedding for an analysis:

$$\mathbf{t}_{ik} = \frac{1}{L} \sum_{l=1}^{L} \mathbf{t_{ikl}} \tag{5}$$

Use the averaged embedding vector with a nonlinear layer to compute the final analysis embedding $E_a(a_{ik})$:

$$E_a(a_{ik}) = \sigma(\mathbf{W}\mathbf{t}_{ik} + \mathbf{b}) \tag{6}$$

where $\mathbf{W}$ and $\mathbf{b}$ are the weight matrix and bias vector of the nonlinear layer, and $\sigma$ is a nonlinear activation function.

Collect all the analysis embeddings for the word $w_i$ into a matrix $\mathbf{A}_i$. If the number of analyses is less than $K$, pad the matrix with a special padding vector $\mathbf{p}$:

$$\mathbf{A}_i = \begin{pmatrix} E_a(a_{i1}) \\ E_a(a_{i2}) \\ \vdots \\ E_a(a_{ik}) \\ \mathbf{p} \\ \vdots \\ \mathbf{p} \end{pmatrix} \tag{7}$$

where $\mathbf{A}_i \in \mathbb{R}^{K \times d}$ is a matrix, and $k$ is the number of actual analyses (where $k \leq K$ ) and $d$ is the dimension of each analysis' embedding.

### 3.4. Ambiguity Resolution

To perform disambiguation, the distance between the context embedding and each analysis embedding is calculated. Since context embedding is based on subwords, for a word $w_i$, we only compute the distance between the first subword $s_{i1}$ of that word with its corresponding analysis embeddings. For the subsequent subwords $s_{i2}, s_{i3}, \ldots, s_{im}$ the distance calculation uses the padding analysis embedding $E_a(\text{pad})$ as follows:

$$a_i^* = \arg\min_{a_{ik} \in \mathbf{A}_i} \mathrm{d}(E_{\text{PLM}}(s_{ij}), E_a(a_{ik})) \tag{8}$$

$$E_a(a_{ik}) = \begin{cases} E_a(a_{ik}) & \text{if } j = 1, \\ E_a(\text{pad}) & \text{if } j > 1, \end{cases} \tag{9}$$

The following distance metrics can be calculated for measuring the distance between the context and analysis embeddings:

(i) Euclidean distance: this measures the straight-line distance between two points in the embedding space.

$$\mathrm{d}(E_{\text{PLM}}(s_i), E_a(a_{ik})) = \|E_{\text{PLM}}(s_i) - E_a(a_{ik})\| \tag{10}$$

(ii) Cosine similarity: this measures the cosine of the angle between two vectors, reflecting how similar their directions are.

$$\mathrm{d}(E_{\mathrm{PLM}}(s_i), E_a(a_{ik})) = \frac{E_{\mathrm{PLM}}(s_i) \cdot E_a(a_{ik})}{\|E_{\mathrm{PLM}}(s_i)\| \|E_a(a_{ik})\|} \tag{11}$$

(iii) Dot product: this measures the magnitude of projection of one vector onto another, reflecting their similarity.

$$\mathrm{d}(E_{\mathrm{PLM}}(s_i), E_a(a_{ik})) = E_{\mathrm{PLM}}(s_i) \cdot E_a(a_{ik}) \tag{12}$$

(iv) Linear layer with a single output: this combines the embeddings through a weighted sum and a bias term to produce a scalar output.

$$\mathrm{d}(E_{\mathrm{PLM}}(s_i), E_a(a_{ik})) = \mathbf{W}_t \cdot \begin{bmatrix} E_{\mathrm{PLM}}(s_i) \\ E_a(a_{ik}) \end{bmatrix} + b \tag{13}$$

### *3.5. Contrastive Loss and Training*

The training process for the MD focuses on optimizing the model's parameters to compute embeddings for both subword-level contextual representations and morphological analyses. In order to effectively learn the parameters for both correct and incorrect analyses (for each tag in the analysis, we use the same dimensional dense vector for the root, POS, and morpheme tags, and we do not distinguish them separately here), we introduce a contrastive loss function, which reduces the distance between the correct analysis and contextual embeddings, while maintaining a margin between correct and incorrect embeddings. The goal is to reduce the distance between the contextual embedding and the correct analysis embedding, while increasing the separation between the contextual embedding and the incorrect analysis embeddings.

This process is set up as a metric-based approach, where the objective is to minimize the distance between the correct analysis and the context, while maximizing the margin between the correct and incorrect analyses from multiple candidates.

In order to illustrate how the contrastive loss works in the proposed approach, we take the dot product as the distance metric in the following notation, and for other distance metrics, the process should be similar and the gradient will be calculated with the corresponding formulae of the chosen distance metrics. For the contrastive loss, we calculate the distance (in the case of the dot product) between the context embedding $c_i = E_{PLM} s_{i1}$ and the morphological analysis embedding $E_a(a_{ik})$, treating it as the similarity score.

$$p_{ik} = softmax(\mathbf{c}_i \cdot E_a(a_{ik})) \tag{14}$$

where $p_{ik}$ is the similarity score between the context and the $k$-th analysis embedding. By taking the negative of the dot product, we ensure that higher similarity (larger value of dot product) results in a smaller distance, and lower similarity (smaller value of dot product) results in a greater distance.

The contrastive loss for MD includes two parts:

(i) Positive loss: minimizes the distance (maximize similarity) for the correct morphological analysis $a_i^*$ for word $w_i$

$$\mathcal{L}_{\mathrm{positive}} = -\log(p_{i,a_i^*}) \tag{15}$$

where $p_{i,a_i^*}$ is the probability for the correct morphological analysis $a_i^*$ for word $w_i$.

(ii) Negative loss: ensures that the sum of probabilities for incorrect analyses is below a margin $\delta$.

$$\mathcal{L}_{\mathrm{negative}} = \max\left(0, \delta - \sum_{k \neq a_i^*} p_{ik}\right) \tag{16}$$

where $\sum_{k \neq a_i^*} p_{ik}$ is the sum of probabilities for all incorrect analyses for the $i$-th word, and $\delta$ is the margin.

The total loss for the entire training set, averaged over $N$ words, can be defined as follows:

$$\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} \left( -\log(p_{i,a_i^*}) + \max\left(0, \delta - \sum_{k \neq a_i^*} p_{ik}\right)\right) \quad (17)$$

## 4. Experiments

Two sets of experiments were carried out in this work, and they are summarized as follows:

- To explore the effectiveness of various distance measurements for performing disambiguation, we chose two Kazakh datasets, and report the results for overall and out-of-vocabulary (OOV) and ambiguous tokens, providing detailed results for separate units of analysis like root, part-of-speech, and morpheme tags. It should be noted that morpheme tags form a sequence of multiple morphological tags with their specified order.
- After selecting the distance measurement that gives the highest result, the second experiment involves a model with a pre-trained language model, comparing it to a model without one. Two comparisons were reported for two languages and three datasets. A general comparison was conducted, and for each dataset, a detailed comparison was provided for predicting each root, POS, and morpheme tag.

### 4.1. Datasets

Table 1 presents the data statistics for the Kazakh and Turkish datasets in terms of dataset size, OOV rates, and AT. The size of the Turkish dataset is much greater than the size of the Kazakh datasets. The OOV rates for Kazakh are higher at 27.58% and 43.9% for kD1 and kD2, while the Turkish dataset has an OOV rate of 10.24%. This complexity is further manifest in the average number of analyses per token where the Kazakh datasets have an AT of 2.95 and 2.85 for kD1 and kD2, respectively, compared to 1.76 for the Turkish dataset. These statistics suggest that while both languages exhibit rich morphological structures, Kazakh presents more challenges due to its smaller datasets and higher OOV rates. These factors are important in model development because fine-tuning PLMs could enhance performance for low-resource languages such as Kazakh.

**Table 1.** Corpora statistics: *AT* denotes the average number of analyses per token.

| Language | Dataset | Train | Test | OOV | *AT* |
|---|---|---|---|---|---|
| Kazakh | kD1 [26] | 13,849 | 1617 | 27.58% | 2.95 |
| | kD2 [27] | 16.624 | 2324 | 43.9% | 2.85 |
| Turkish [28] | | 752,332 | 20,536 | 10.24% | 1.76 |

### 4.2. Model Setup

The model's hidden size is set at 768. The dimensions for both word embedding and tag embedding are set to 768, ensuring uniformity across the model's embedding spaces. The learning rate is set at $5 \times 10^{-4}$, providing a balanced approach between effective training and achieving convergence. A weight decay of 0.01 is employed to avoid overfitting by penalizing large weights. The model was trained for a total of 100 epochs. Additionally, a warmup proportion of 0.3 is utilized. The xlm-roberta-base model was used as the base model, as it is a multilingual pre-trained language model trained on a large dataset covering over 100 languages.

### 4.3. Results

In the experimental results, the accuracy for three types of tokens is reported as follows: (i) all tokens; (ii) ambiguous tokens (those with at least two analyses); (iii) OOV tokens.

Table 2 shows the overall, OOV, and ambiguous case accuracy for four distance measures: cosine similarity, Euclidean distance, dot product, and linear layer. For the kD1 dataset, the dot product distance metric achieved the highest overall accuracy, with 92.39% accuracy, 86.46% for OOV cases, and 88.36% for ambiguous cases. The Euclidean distance metric also performed strongly, with 92.14% overall accuracy. The cosine similarity achieved 89.67% overall accuracy, while the linear layer produced the lowest results, with an overall accuracy of 73.47%. In the kD2 dataset, the Euclidean distance yielded the highest overall accuracy (88.46%), followed by dot product at 87.83%. The cosine similarity showed an overall accuracy of 87.06%, and the linear layer again demonstrated the lowest performance, with 80.97% accuracy.

Table 3 reports the accuracy results for MD on the Kazakh datasets, measured across three aspects: root, POS, and morpheme. For the kD1 dataset, the Euclidean distance yielded the highest accuracy for root disambiguation, with 98.33% overall accuracy, followed by the dot product at 98.27%. In the OOV and ambiguous cases, the Euclidean distance achieved 96.44% and 97.45%, respectively. The cosine similarity yielded 97.71% overall accuracy. The linear layer achieved 93.75% overall accuracy, with a drop to 90.44% for ambiguous cases. In POS disambiguation, the Euclidean distance reached 96.60% overall accuracy, while the dot product scored 96.35%. The linear layer performed at 81.08%, with a significant decrease to 71.05% for ambiguous cases. For morpheme disambiguation, the dot product reached 93.94% overall accuracy and 88.36% OOV accuracy, with the Euclidean distance at 93.69% overall. The cosine similarity reached 91.22% overall accuracy, and the linear layer reached 80.27%, with the lowest ambiguous case accuracy of 69.82%.

In the kD2 dataset, the Euclidean distance achieved 98.65% overall accuracy for root disambiguation, followed by the dot product at 98.17% and the cosine similarity at 98.16%. The linear layer achieved 96.42%. For POS disambiguation, the Euclidean distance reached 94.40%, while the dot product followed with 93.81%. The linear layer achieved 89.52%. For morpheme disambiguation, the Euclidean distance reached 88.85% overall accuracy, with the dot product at 88.27%. The cosine similarity achieved 87.64%, and the linear layer reached 81.46%. Across both datasets, the Euclidean distance and the dot product consistently produced higher accuracy in all categories. The linear layer consistently showed lower results, particularly in ambiguous cases.

The possible reason behind why the linear layer showed lower results compared to the other metrics is the large tagset in morphological disambiguation (MD). The linear layer uses a parameter $\mathbf{W}_t \in \mathbb{R}^{d \times 1}$, which computes the concatenation of the context and the representations of different analyses. Here, $d$ represents the dimension of the concatenation of the context and one analysis embedding. Since the number of unique analyses is large and $W_t$ is a vector parameter, it becomes difficult to effectively capture the relationship between the context and the numerous different analyses. In comparison, in named entity recognition (NER), there is a fixed tagset, and in that case, the parameter's dimension is fixed to a specific label, making it easier to model the relationships.

Table 4 shows that fine-tuning with PLMs ($\text{cMD}_{plm}$) results in improved accuracy across both languages compared to the baseline ($\text{cMD}_{mlp}$) that uses a multilayer perceptron. Both of the models $\text{cMD}_{plm}$ and $\text{cMD}_{mlp}$ were trained and tested without using any external designed features. For the Turkish dataset, the overall accuracy increases slightly from 91.47% to 92.44%, with a gain in the OOV accuracy from 87.97 to 88.45%. The ambiguous case accuracy improves by nearly 2%. In the kD1 dataset, the overall accuracy improves by 6%, with the OOV accuracy reaching 85.27% and the ambiguous case accuracy increasing by over 9%. The kD2 dataset shows an over 6% improvement in overall accuracy, with similar increases in the OOV and ambiguous case accuracy.

**Table 2.** Accuracy results of MD for Kazakh datasets for overall, OOV, and ambiguous cases.

| Dataset | Distance | Overall Acc. | OOV Acc. | Ambig. Acc. |
|---|---|---|---|---|
| kD1 | cosine similarity | 89.67 | 82.18 | 84.20 |
| | Euclidean | 92.14 | 85.98 | 87.98 |
| | dot product | 92.39 | 86.46 | 88.36 |
| | linear layer | 73.47 | 72.68 | 59.41 |
| kD2 | cosine similarity | 87.06 | 84.34 | 79.99 |
| | Euclidean | 88.46 | 86.24 | 82.15 |
| | dot product | 87.83 | 85.71 | 81.18 |
| | linear layer | 80.97 | 80.88 | 70.58 |

**Table 3.** Accuracy results of MD for Kazakh datasets for root, POS, and morpheme.

| Dataset | Distance Metrics | Overall | OOV | Ambig. |
|---|---|---|---|---|
| kD1 | | root | | |
| | cosine similarity | 97.71 | 94.77 | 96.50 |
| | Euclidean | 98.33 | 96.44 | 97.45 |
| | dot product | 98.27 | 95.96 | 97.35 |
| | linear layer | 93.75 | 93.59 | 90.44 |
| | | POS | | |
| | cosine similarity | 96.04 | 94.06 | 93.95 |
| | Euclidean | 96.60 | 95.01 | 94.79 |
| | dot product | 96.35 | 94.29 | 94.42 |
| | linear layer | 81.08 | 85.51 | 71.05 |
| | | morpheme | | |
| | cosine similarity | 91.22 | 84.09 | 86.57 |
| | Euclidean | 93.69 | 87.89 | 90.35 |
| | dot product | 93.94 | 88.36 | 90.73 |
| | linear layer | 80.27 | 80.04 | 69.82 |
| kD2 | | root | | |
| | cosine similarity | 98.16 | 97.89 | 97.16 |
| | Euclidean | 98.65 | 98.21 | 97.91 |
| | dot product | 98.17 | 97.58 | 97.09 |
| | linear layer | 96.42 | 96.32 | 94.47 |
| | | POS | | |
| | cosine similarity | 92.85 | 92.96 | 88.95 |
| | Euclidean | 94.40 | 95.06 | 91.34 |
| | dot product | 93.81 | 94.32 | 90.44 |
| | linear layer | 89.52 | 91.81 | 83.79 |
| | | morpheme | | |
| | cosine similarity | 87.64 | 84.77 | 80.88 |
| | Euclidean | 88.85 | 86.45 | 82.75 |
| | dot product | 88.27 | 90.35 | 81.85 |
| | linear layer | 81.46 | 81.59 | 71.32 |

**Table 4.** Accuracy results of MD for Kazakh and Turkish datasets before and after fine-tuning on PLMs.

| Dataset | Model | Overall Acc. | OOV Acc. | Ambig. Acc. |
|---|---|---|---|---|
| Turkish | $cMD_{mlp}$ | 91.47 | 87.97 | 82.61 |
| | $cMD_{plm}$ | 92.44 | 88.45 | 84.58 |
| kD1 | $cMD_{mlp}$ | 86.58 | 69.35 | 79.47 |
| | $cMD_{plm}$ | 92.64 | 85.27 | 88.74 |
| kD2 | $cMD_{mlp}$ | 83.05 | 80.04 | 74.53 |
| | $cMD_{plm}$ | 89.23 | 87.61 | 83.35 |

Table 5 provides a detailed breakdown of the accuracy improvements across the root, POS, and morpheme categories. After using LLM, the accuracy for the ambiguous case improves across all categories. For the Turkish dataset, the accuracy for the root increases slightly, with the POS and morpheme categories seeing gains of around 0.9% and 2%, respectively. In the Kazakh datasets, the model $cMD_{plm}$ shows significant improvements. For kD1, there is a nearly 1.5% increase in the root ambiguous accuracy, an over 5% gain in POS, and a nearly 8% improvement in morpheme ambiguous accuracy compared with $cMD_{mlp}$.

In the kD2 dataset, the model shows similar results, with improvements of 1.5% in root, 6% in POS, and 10% in morpheme ambiguous accuracy.

**Table 5.** Accuracy results of MD for Kazakh and Turkish datasets for root, POS and morpheme before and after fine-tuning on PLMs.

| Dataset | Model | Category | Overall | OOV | Ambig. |
|---|---|---|---|---|---|
| Turkish | $cMD_{mlp}$ | root | 98.12 | 95.81 | 96.16 |
| | | POS | 96.26 | 97.82 | 92.38 |
| | | morpheme | 91.90 | 88.18 | 83.46 |
| | $cMD_{plm}$ | root | 98.26 | 96.13 | 96.45 |
| | | POS | 96.56 | 97.82 | 92.98 |
| | | morpheme | 92.87 | 88.77 | 85.44 |
| kD1 | $cMD_{mlp}$ | root | 97.34 | 91.92 | 95.93 |
| | | POS | 93.44 | 84.80 | 89.97 |
| | | morpheme | 88.80 | 73.87 | 82.88 |
| | $cMD_{plm}$ | root | 98.33 | 96.68 | 97.44 |
| | | POS | 97.09 | 96.19 | 95.55 |
| | | morpheme | 94.06 | 86.93 | 90.91 |
| kD2 | $cMD_{mlp}$ | root | 96.61 | 94.95 | 94.77 |
| | | POS | 91.54 | 90.12 | 86.93 |
| | | morpheme | 84.11 | 80.46 | 75.42 |
| | $cMD_{plm}$ | root | 98.26 | 96.13 | 96.45 |
| | | POS | 96.55 | 97.82 | 92.97 |
| | | morpheme | 92.87 | 88.77 | 85.44 |

The results show that the proposed model with PLMs trained with contrastive loss improves the model's performance in terms of ambiguous cases and OOV tokens, especially in the context of low-resource languages, including Kazakh, with a small-sized dataset, without the help of any designed features.

Table 6 compares the proposed approach with the previous work. Since the datasets are the same as for the previous work, the results are comparable. The results for the HMM and voted perceptron approaches were obtained from [26]. The voted perceptron approach for Turkish was trained with the tool developed by [3]. It can be observed that the $cMD_{plm}$ outperforms the other approaches for the Kazakh language and Turkish without using any feature and decoding techniques.

**Table 6.** Comparison with previous work; fea. stands for a set of features, and decoding indicates that the method uses Viterbi decoding to find the best path among the analyses.

| | **Model** | **Overall** | **OOV** |
|---|---|---|---|
| kD1 | HMM (decoding) [26] | 84.91 | 71.73 |
| | Voted-Perceptron (fea.+decoding) [26] | 90.53 | 82.42 |
| | $cMD_{plm}$ | 92.64 | 85.27 |
| Turkish | Voted-Perceptron (fea.+decoding) | 91.89 | 87.98 |
| | $cMD_{plm}$ | 92.44 | 88.45 |

## 5. Discussion

The error distribution for the kD1 dataset shows several key misclassification patterns in Figure 2. The most frequent error (over 5.5%) is the misclassification of singular accusative forms (n px3sg acc) as plural accusative forms (n px3pl acc). Another common error is the prediction of adjectival forms instead of plural noun forms (adj px3pl acc → n px3pl acc), contributing to nearly 4% of the errors. Additionally, there are misclassifications in the locative and ablative cases, such as (n px3pl loc → n px3sg loc), reflecting the model's difficulty in distinguishing singular and plural forms across these cases.

Verb-related errors, especially between present and past forms (v-iv presct → v-iv pastvadv), also appear in the error distribution. The model struggles to differentiate between singular and plural noun forms, shows challenges with noun-adjective agreement in number and case, and has difficulty distinguishing between tense and auxiliary verb forms.

Figure 3 shows the error distribution for the kD2 dataset. The most frequent error, making up over 5% of the misclassifications, involves confusing masculine anthroponym noun phrases (np ant m nom) with simple nouns (n nom). Another error above 4% occurs between perfect participles (v tv prc_perf) and verbal adverbs (v tv gna_perf). The third most common error, around 3%, involves the misclassification of topological nouns (np top nom) as attributive nouns (np top attr).

Figure 4 presents the error distribution for the Turkish dataset and shows several cases where the model has difficulty. The most common error, accounting for almost 10% of cases, is the misclassification of adjectives as singular nouns in the nominative case (adj → noun a3sg pnon nom), indicating difficulty in separating these two grammatical functions. About 8% of errors involve confusing proper nouns with common singular nouns (noun prop a3sg pnon nom → noun a3sg pnon nom), suggesting challenges in distinguishing proper from regular nouns. Approximately 6% of the errors occur when singular accusative nouns are misclassified as singular nominative nouns with possession (noun a3sg pnon acc → noun a3sg p3sg nom). In addition, the model frequently confuses verbs in the past tense with narrative mood and singular past-tense verbs (verb pos narr adj zero → verb pos narr a3sg), reflecting difficulty with tense, mood, and number. Lastly, about 2.5% of errors are due to misclassifying adverbs as adjectives (adverb → adj).

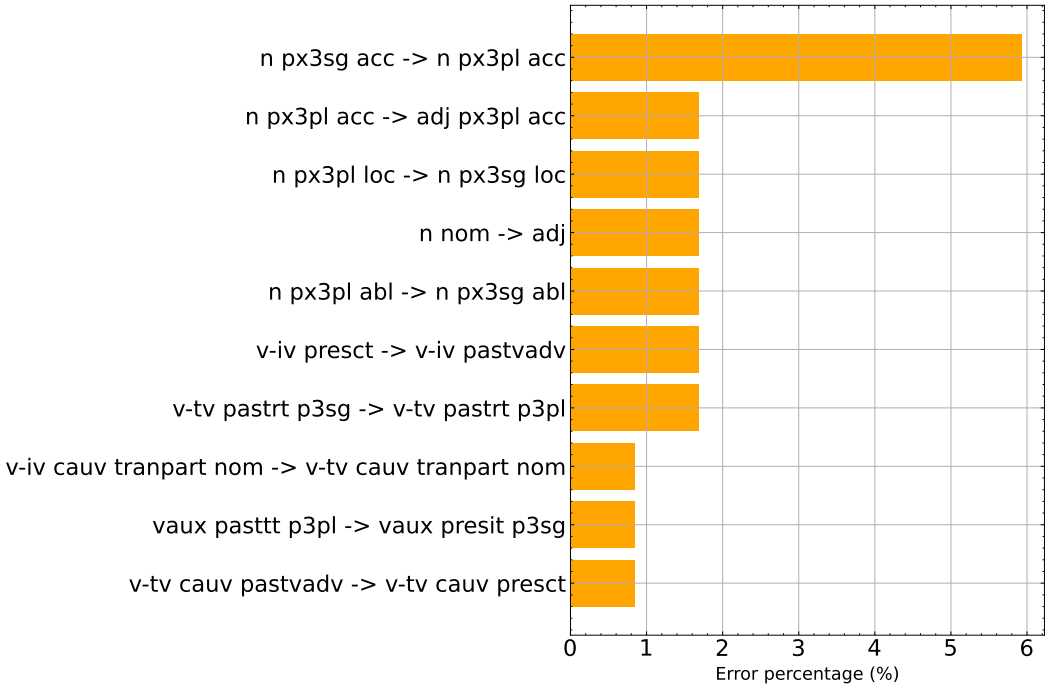



**Figure 2.** Error distributions of analysis (roots are excluded) for kD1 dataset.

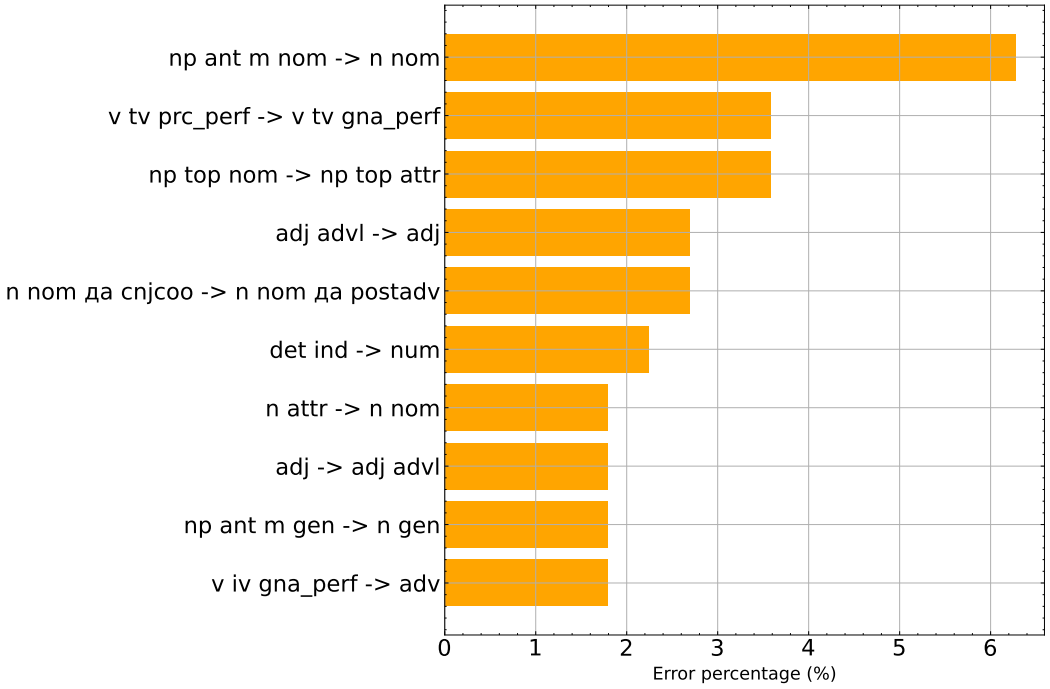



**Figure 3.** Error distributions of analysis (roots are excluded) for kD2 dataset.

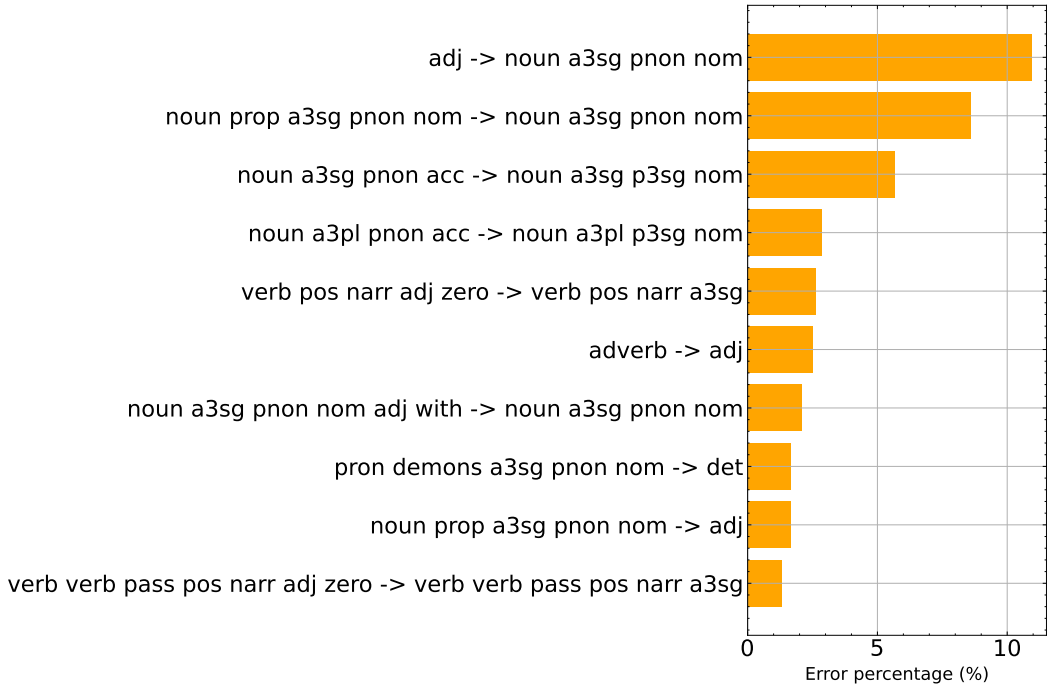



**Figure 4.** Error distributions of analysis (roots are excluded) for Turkish dataset.

## 6. Conclusions

Morphological disambiguation is a long-standing challenge in processing MCLs. It is akin to POS tagging, but with a key distinction: in MD, it is not just the POS tag that needs to be correctly predicted, but also the lemma or root form of the word along with its corresponding morphological tags. This adds layers of complexity to the task compared to traditional POS tagging.

In this paper, a contrastive learning approach for morphological disambiguation (MD) using pre-trained language models (PLMs) was presented. The intention behind the contrastive loss of the proposed approach is to reduce the distance between the correct analysis and contextual embeddings while maintaining a margin between incorrect with contextual embeddings.

One of the aims of the paper is to explore the impact of fine-tuning a PLM for MD on Kazakh, a low-resource language as well as Turkish. Traditional approaches to MD, like HMM-based and feature-engineered models, will not be able to capture long-term dependencies. In this regard, the study leveraged the contextualized embeddings from the PLMs, which allowed the model to enhance its handling of ambiguous cases and OOV tokens without depending on other features. The experimental results show that the fine-tuned PLM gave better MD results without using any designed features, especially for Kazakh. Further, the disambiguation analysis performed with different distance measurements shows that the Euclidean distance and dot product yield the best distance between context and morphological analysis embeddings. These results show that PLMs are useful in enhancing the performance of MD tasks for various morphologically complex languages, including low-resource ones like Kazakh. The error analysis revealed that the model faces challenges in differentiating singular from plural noun forms (it has strong long-term dependency), struggles with maintaining noun-adjective agreement in both number and case, and encounters difficulties in distinguishing between tense and auxiliary verb forms in the case of Kazakh. The results for the Turkish dataset revealed that the model struggled to distinguish between adjectives and nouns, properly identifying proper nouns versus common nouns, handling case and number distinctions (especially with possession and plurality), and managing complex verb forms involving tense and mood.

Overall, $cMD_{plm}$, which uses a contrastive learning method and is incorporated with LLM, achieves an overall accuracy of 92.64% for Kazakh (kD1). It outperforms the HMM-based approach by approximately 7% and the voted-perceptron approach (a set of features combined with decoding techniques) by 2%. For Turkish, it slightly outperforms the voted-perceptron approach which uses a group of features with decoding.

Future work can be summarized as follows: (i) to improve the model seeking to solve the existing errors; (ii) to explore how morphological analysis and disambiguation can be integrated into a unified learning framework through large language models, thus significantly improving the performance of the MD models in various linguistic contexts; (iii) to explore the question of how to effectively use the morphological features, since long-term dependency exits between these tags.

**Author Contributions:** Conceptualization, A.T. and G.T.; methodology, G.T. and A.T.; software, G.T. and A.T.; validation, A.T. and G.T.; formal analysis, A.T. and G.T.; investigation, G.T. and A.T.; resources, A.T. and G.T.; data curation, G.T. and A.T.; writing—original draft preparation, A.T. and G.T.; writing—review and editing, A.T. and G.T.; visualization, A.T. and G.T.; supervision, A.T. and G.T.; project administration, R.M.; funding acquisition, R.M. All authors have read and agreed to the published version of the manuscript.

**Funding:** The research was funded by the Committee of Science of the Ministry of Science and Higher Education of the Republic of Kazakhstan under grant number BR21882268.

**Institutional Review Board Statement:** Not applicable.

**Informed Consent Statement:** Not applicable.

**Data Availability Statement:** The data presented in this study are available in kD2: [apertium-kaz] at [https://svn.code.sf.net/p/apertium/svn/branches/]. kD1 dataset from this paper https://aclanthology.org/2020.sltu-1.36.pdf. Turkish dataset from https://github.com/ai-ku/TrMor2018 (all accessed on 10 August 2023).

**Conflicts of Interest:** The authors declare no conflicts of interest.

## References

1. Eşref, Y.; Can, B. Using Morpheme-Level Attention Mechanism for Turkish Sequence Labelling. In Proceedings of the 2019 27th Signal Processing and Communications Applications Conference (SIU), Sivas, Turkey, 24–26 April 2019; pp. 1–4. [CrossRef]
2. Hakkani-Tür, D.Z.; Oflazer, K.; Tür, G. Statistical Morphological Disambiguation for Agglutinative Languages. In Proceedings of the COLING 2000 Volume 1: The 18th International Conference on Computational Linguistics. Available online: https://aclanthology.org/C00-1042/ (accessed on 26 October 2024).
3. Sak, H.; Güngör, T.; Saraçlar, M. Morphological Disambiguation of Turkish Text with Perceptron Algorithm. In Proceedings of the 8th International Conference Computational Linguistics and Intelligent Text Processing, Mexico City, Mexico, 18–24 February 2007 pp. 107–118.
4. Toleu, A.; Tolegen, G.; Makazhanov, A. Character-Aware Neural Morphological Disambiguation. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), Vancouver, BC, Canada, 30 July–4 August 2017; pp. 666–671. [CrossRef]
5. Wei, J.; Tay, Y.; Bommasani, R.; Raffel, C.; Zoph, B.; Borgeaud, S.; Yogatama, D.; Bosma, M.; Zhou, D.; Metzler, D.; et al. Emergent Abilities of Large Language Models. *arXiv* **2022**, arXiv:2206.07682.
6. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language Models are Unsupervised Multitask Learners. *Open AI Blog* **2019**, *1*, 9.
7. Devlin, J.; Chang, M.W.; Lee, K.; Toutanova, K. 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, Volume 1 (Long and Short Papers), Minneapolis, MN, USA, 2019; pp. 4171–4186. [CrossRef]
8. Brown, T.; Mann, B.; Ryder, N.; Subbiah, M.; Kaplan, J.; Dhariwal, P.; Neelakantan, A.; Shyam, P.; Sastry, G.; Askell, A.; et al. Language Models are Few-Shot Learners. In Proceedings of the 34th International Conference on Neural Information Processing Systems, Vancouver, BC, Canada, 6–12 December 2020; pp. 1877–1901. Available online: https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf (accessed on 26 October 2024).
9. Touvron, H.; Lavril, T.; Izacard, G.; Martinet, X.; Lachaux, M.A.; Lacroix, T.; Rozière, B.; Goyal, N.; Hambro, E.; Azhar, F.; et al. LLaMA: Open and Efficient Foundation Language Models. *arXiv* **2023**, arXiv:2302.13971.
10. Toleu, A.; Tolegen, G.; Mussabayev, R. Deep Learning for Multilingual POS Tagging. In *International Conference on Computational Collective Intelligence*; Springer: Da Nang, Vietnam, 2020; pp. 15–24.

11. Tolegen, G.; Toleu, A.; Mamyrbayev, O.; Mussabayev, R. Neural Named Entity Recognition for Kazakh. In *Computational Linguistics and Intelligent Text Processing*; CICLing 2019; Lecture Notes in Computer Science; Springer: Cham, Switzerland, 2019; Volume 13452. [CrossRef]
12. Tkachenko, A.; Sirts, K. Modeling Composite Labels for Neural Morphological Tagging. In Proceedings of the 22nd Conference on Computational Natural Language Learning, Brussels, Belgium, 31 October–1 November 2018; pp. 368–379. [CrossRef]
13. Hochreiter, S.; Schmidhuber, J. Long Short-Term Memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
14. Oflazer, K.; Kuruoz, I. Tagging and Morphological Disambiguation of Turkish Text. In Proceedings of the Fourth Conference on Applied Natural Language Processing, Stuttgart, Germany, 13–15 October 1994; pp. 144–149. [CrossRef]
15. Oflazer, K.; Tur, G. Combining Hand-crafted Rules and Unsupervised Learning in Constraint-based Morphological Disambiguation. In Proceedings of the Conference on Empirical Methods in Natural Language Processing, Philadelphia, PA, USA, 17–18 May 1996.
16. Daybelge, T.; Çiçekli, I. A Rule-Based Morphological Disambiguator for Turkish. In Proceedings of the Recent Advances in Natural Language Processing, Borovets, Bulgaria 27–29 September 2007.
17. Çöltekin, Ç. A set of open source tools for Turkish natural language processing. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), Reykjavik, Iceland, 26–31 May, 2014; pp. 1079–1086.
18. Görgün, O.; Yildiz, O.T. A Novel Approach to Morphological Disambiguation for Turkish. In *Computer and Information Sciences II*; Gelenbe, E., Lent, R., Sakellari, G., Eds.; Springer: London, UK, 2012; pp. 77–83.
19. Shen, Q.; Clothiaux, D.; Tagtow, E.; Littell, P.; Dyer, C. The Role of Context in Neural Morphological Disambiguation. In *COLING 2016, Proceedings of the 26th International Conference on Computational Linguistics: Technical Papers*; Matsumoto, Y., Prasad, R., Eds.; The COLING 2016 Organizing Committee: Osaka, Japan, 2016; pp. 181–191.
20. Yildiz, E.; Tantuğ, A.C. Morpheus: A Neural Network for Jointly Learning Contextual Lemmatization and Morphological Tagging. In Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology, Florence, Italy, 2 August 2019; pp. 25–34. [CrossRef]
21. Zhu, S. A Neural Attention Based Model for Morphological Segmentation. *Wirel. Pers. Commun.* **2018**, *102*, 2527–2534. [CrossRef]
22. Seker, A.; Tsarfaty, R. A Pointer Network Architecture for Joint Morphological Segmentation and Tagging. In *Findings of the Association for Computational Linguistics: EMNLP, Online, 1 June 2020*; Cohn, T., He, Y., Liu, Y., Eds.; Association for Computational Linguistics: Stroudsburg, PA, USA, 2020; pp. 4368–4378. [CrossRef]
23. Rumelhart, D.E.; McClelland, J.L. Learning Internal Representations by Error Propagation. In *Parallel Distributed Processing: Explorations in the Microstructure of Cognition: Foundations*; MIT Press: Cambridge, MA, USA, 1987; pp. 318–362.
24. Schmidt, R.M. Recurrent Neural Networks (RNNs): A gentle Introduction and Overview. *arXiv* **2019**, arXiv:1912.05911.
25. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.U.; Polosukhin, I. Attention is All you Need. In *Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Red Hook, NY, USA, 2017; Volume 30.
26. Tolegen, G.; Toleu, A.; Mussabayev, R. Voted-Perceptron Approach for Kazakh Morphological Disambiguation. In Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL), Marseille, France, 11–12 May 2020; pp. 258–264.
27. Assylbekov, Z.; Washington, J.N.; Tyers, F.M.; Nurkas, A.; Sundetova, A.; Karibayeva, A.; Abduali, B.; Amirova, D. A free/open-source hybrid morphological disambiguation tool for Kazakh. In Proceedings of the 1st International Workshop on Turkic Computational Linguistics, Konya, Turkey, 3–9 April 2016.
28. Yuret, D.; Türe, F. Learning Morphological Disambiguation Rules for Turkish. In Proceedings of the Human Language Technology Conference of the NAACL, Main Conference, New York, NY, USA, 4–9 June 2006; pp. 328–334.


**Disclaimer/Publisher's Note:** The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.