

Combined Learning of Morphological Segmentation, POS Tagging and Contextual Lemmatization

Christian Wartena

Hochschule Hannover

Institute for Applied Data Science - Data|H

Expo Plaza 12

30539 Hannover, Germany

christian.wartena@hs-hannover.de

Abstract

Morphological analysis, part-of-speech tagging, and lemmatization are closely related tasks that are typically addressed with separate models or pipeline systems. We present Flexor, a neural architecture that jointly learns all three tasks from character-level input. Flexor performs morphological analysis via morpheme-level character labeling, while a shared contextual word representation is used for both POS classification and lemma generation. The model combines word- and sentence-level bidirectional LSTMs to capture local morphological structure and contextual information required for disambiguation. We evaluate on English, German, and Dutch and show that joint training yields strong performance across all tasks. Flexor consistently outperforms comparable taggers trained on the same data while maintaining a compact architecture with 3.5M parameters. These results show that a unified character-based model can integrate morphology, POS tagging, and lemmatization effectively.

1 Introduction

Part-of-speech (POS) tagging, lemmatization, and morphological analysis are core components of many natural language processing systems. While traditional approaches rely on rule-based or finite-state methods, and more recent systems frequently exploit pretrained contextual language models for feature extraction, neither line of work explicitly models fine-grained, structured morphological decomposition in context. In this paper, we propose a neural model that jointly predicts POS-tags, lemmas, and labeled morpheme structures for words in context.

Often POS-tagging and morphological analysis are treated as independent tasks, at least for languages that do not have a very rich morphology, like English or German. However, morphological structure gives much information about possible POS of a word, much more than just probabilities

computed for the final two, three or four characters of a word. On the other hand many words are morphologically ambiguous and the correct analysis and POS can only be decided in context. Thus we will integrate POS-tagging and morphological analysis to make optimal use of the internal structure of a word and its context at the same time. Furthermore, we treat lemmatization, the formation of the canonical representation of a word, as a task that is integrated in the word analysis as well.

The approach presented here considers morphological analysis and POS-tagging as a sequence labeling task and lemmatization as a sequence-to-sequence task. The characters of a word are labeled with IOB-style tags to indicate what part of the word is the stem, a prefix, a suffix etc. This is done with a stack of LSTMs for each word. The final hidden state of these LSTMs is fed into a decoder LSTM that generates the lemma for the word. The decoder uses cross-attention in addition because the word stem usually can just be copied from the analyzed word form to the lemma. The last hidden state of each word is used furthermore to predict the POS of the word. This is useful, since the lemmatizer needs information about the POS in order to generate the correct lemma. E.g. for a verb it needs to generate the infinitive form in many languages. The context information that is needed for the disambiguation of many words is provided by an LSTM that runs over all word representations.

The bottleneck of the presented approach is the need for annotated training data. For this study we take annotated data were created for HanTa (Wartena, 2019), a statistical tagger that does morphological analysis and POS-tagging by searching for the sequence of morphemes that optimizes the probabilities for each word and the whole sentence. These training data are generated from tagged and lemmatized corpora using heuristic rules that derive the morphological structure of each word given the lemma and the POS-tag of the word. HanTa was

developed for German, but training data and parameter files for English and Dutch are available as well (Wartena, 2023). We have extended the training data provided for English for this study. For German and Dutch we took the data from the HanTa project.

The remainder of the paper is structured as usual. We first review related work before we present the data and the exact problem definition. The architecture of the network and parameter settings follow. Finally, the results are presented and discussed.

2 Related Work

Our network addresses three closely related tasks: POS-tagging, morphological analysis, and lemmatization. Part-of-speech tagging is one of the most established tasks in natural language processing and is commonly formulated as sequence labeling over tokens in running text. Overviews of the task can be found in any introductory NLP textbook.

Morphological processing comprises a spectrum of tasks that differ in the amount of information they aim to recover from a word form. At one end lies *morphological segmentation*, whose goal is merely to identify morpheme boundaries. At the other end lies *full morphological analysis*, which predicts detailed morphosyntactic features such as tense, person, number, gender, or case, but typically does not explicitly decompose a word into its constituent morphemes. Between these extremes lie approaches that segment a word into morphemes and assign labels to the resulting segments. Our work belongs to this intermediate category.

For morphological analysis finite-state and rule-based systems remain highly competitive and are often regarded as the reference approach for many languages. In this context, the work of Koskenniemi (1983) is particularly influential.

Morphological segmentation was traditionally studied as an unsupervised learning problem. Influential early approaches were based on the Minimum Description Length (MDL) principle, which seeks a balance between maximizing the reuse of recurring morphemes and minimizing the size of the resulting morpheme lexicon. Examples include the work of Goldsmith (2001) and the Morfessor family of algorithms (Creutz and Lagus, 2007). A related line of research is represented by Adaptor Grammars (Johnson et al., 2007; Sirts and Goldwater, 2013; Botha and Blunsom, 2013), which model words by means of hierarchical Bayesian grammars and learn

recurring morphological units from unannotated corpora. An overview of unsupervised morphology learning is given by Goldsmith et al. (2017).

More recently, supervised and semi-supervised approaches have attracted increasing attention. Character-level neural models that formulate segmentation or morpheme boundary detection as a sequence-labeling problem, have been shown to learn accurate morpheme segmentations from relatively small amounts of annotated data (Ruokolainen et al., 2013, 2016; Abudouwaili et al., 2024). Würzner and Jurish (2015) extend this approach by distinguishing different types of segment boundaries, thereby moving closer to morphological analysis.

Full morphological analysis aims to predict the complete set of morphosyntactic features of a word, including categories such as tense, person, number, case, and gender. If the combination of all features is treated as a single label, the task becomes very similar to POS-tagging, albeit with a much larger number of categories. State-of-the-art systems typically employ character-level neural encoders that compute contextualized word representations, followed by classifiers that predict the relevant feature bundles. A representative example of this approach is given by Heigold et al. (2017).

Kann et al. (2016) propose a different approach. They train an encoder–decoder model in which the encoder reads a word form and the decoder generates a sequence representing the normalized morphemes. This approach is closely related to lemmatization, since lemma generation often requires not only the removal of inflectional affixes but also stem normalization.

Besides methods that learn morphology directly from annotated data, recent work has investigated the extent to which morphological information is encoded in large language models. The underlying idea is that contextual embeddings produced by pretrained language models contain substantial morphological information that can be extracted through probing or fine-tuning (Rogers et al., 2020; Acs et al., 2024).

Supervised lemmatization has been addressed in a wide range of approaches, including rule induction methods, classification-based systems, and more recently neural sequence-to-sequence models. Kanis and Müller (2005) and Jongejan and Dalianis (2009) learn transformation rules from lemmatized corpora that map inflected word forms to their corresponding lemmas. Gashkov and Eltsova

(2018) obtain strong results for German by combining a full-form dictionary with analogy-based methods. For unknown words, the dictionary entry sharing the longest suffix is identified and the corresponding transformation is applied. [Gesundo and Samardžić \(2012\)](#) propose annotating words with the type of transformation required to derive the lemma from the surface form, thereby reducing lemmatization to a tagging task. A similar idea is pursued by [Chrupala et al. \(2008\)](#), who define classes corresponding to mappings from word forms to lemmas and train a classifier to predict the appropriate class. This approach is extended by [Müller et al. \(2015\)](#), who employ richer feature representations and conditional random fields. More recently, contextualized embeddings produced by pretrained language models have become the dominant source of information for such classifiers. [Toporkov and Agerri \(2024\)](#) investigate which types of information contribute most strongly to lemmatization performance and conclude that most relevant information is already encoded in the embeddings.

Sequence-to-sequence neural models have also become popular for lemmatization. Instead of predicting transformation classes, these models directly generate the lemma from the surface form. Examples include the work of [Milintsevich and Sirts \(2021\)](#) and [Kann et al. \(2016\)](#). The SIGMORPHON shared tasks have played an important role in advancing this line of research by providing multilingual benchmarks for morphological inflection, lemmatization, and morphological analysis, thereby stimulating the development of increasingly powerful neural architectures.

Several recent systems jointly model lemmatization and morphological tagging, motivated by the observation that both tasks rely on similar morphological information. Shared encoder architectures have been shown to improve performance on both tasks. The system of [Kondratyuk \(2019\)](#) is particularly interesting in this regard, since it employs contextualized representations derived from a fine-tuned BERT model for both lemmatization and morphological feature prediction.

Our system draws on ideas from several of the approaches discussed above, while differing from them in a number of important respects. First, we combine three tasks: POS-tagging, morphological analysis, and contextual lemmatization. We employ relatively large tag sets, yet our approach remains distinct from full morphological analysis. Unlike most work on morphological tagging, our system ex-

plicitly decomposes words into morphemes, taking into account not only inflectional affixes but also many derivational affixes. Furthermore, all morphemes are assigned detailed morpheme classes, making the task substantially richer than simple morphological segmentation.

The tasks addressed by our system and the training data are the same as we used before in a statistical approach ([Wartena, 2019](#)). In that approach we collected morphemes from fully annotated training data and used an extension of the Viterbi algorithm to determine an optimal morpheme sequence during deployment. Our current approach differs fundamentally from this method. Like most modern systems, we compute contextualized word representations using a character-based neural encoder. However, unlike many recent architectures that incorporate convolutional or transformer-based components, our model relies exclusively on recurrent neural networks. Despite its architectural simplicity, the model achieves competitive results on all three tasks.

3 Problem Formulation and Data

In this section we first review the typical problems of morphological analysis and lemmatization and then present the creation and the formal of the data used to train and test our models.

3.1 Morphology

In the morphology of Indo-European (and many other) languages we have to distinguish between *inflectional* and *derivational* morphology. Inflectional morphology deals with the systematic formation of grammatical variants of word, like adding case, number, tense, etc. Derivational morphology studies the formation of new words. For many applications, like e.g. information retrieval, recognition of inflectional variants is important since inflection does not change the meaning of the word (eventually up to a very specific and well defined aspect). Derivational morphology is often much less regular and the meaning of derived words is in many cases not predictable. In our data for German, English and Dutch, inflectional morphology is covered completely, while for derivational morphology only the most frequent and regular cases are annotated in the data.

Usually the morphological structure is considered to be a hierarchical structure. In our data a word is modeled as a list of labeled morphemes

without structure. E.g. the word *unbelievable* is annotated as [(un, PREF_NEG), (believ, VV_VAR), (able, SUF_VVADJ)]. Furthermore, for each word the POS and the lemma are given and a mapping from morpheme variants to their canonical form. In the current example that would be the mapping from *believ* to *believe*. For our model we will not use these mappings as the lemma decoder will learn these relations itself.

Finally, we note that the lemmatization of a word is not extremely complicated but nevertheless not always obvious. The lemma is the canonical form of a word. Thus the lemma for *believes* is *believe*, but the lemma of *unbelievable* is *unbelievable*, since a new word was formed with its own canonical form. The situation now becomes tricky for participles, that are problematic for POS-tagging as well. The word *reading* can be a verb, an adjective or a noun (*John is reading / a reading boy / a reading room*). In the first case the lemma is *(to) read* but in both other cases the lemma is *reading* since obviously a new word with a different POS was formed. Though this is the standard analysis (see e.g. [Greenbaum and Quirk, 1990](#), pp133ff), some corpora might have been following other policies for these cases. In German the situation is even more complicated as the participle used as adjective can be turned subsequently to a noun with strong inflection, as e.g. in *Die Lesenden* (“*the reading ones*”) for which there are three different lemmas, depending on the (intended) gender. Here corpora differ substantially in the choices made to lemmatize these forms. Finally, the lemmatization of closed classes, like pronouns, is in fact completely useless, but for completeness is given in all annotated corpora. The British National Corpus e.g. uses *he* as lemma for the possessive pronoun *his*, while others use the word form itself for each pronoun.

3.2 Combining BNC and Brown

The original HanTa-data enriched the TIGER Corpus ([Brants et al., 2002](#)) with labeled morphemes for German and part of the Sonar 1 Corpus for Dutch ([Oostdijk, 2012](#)). For English the Brown Corpus was used with lemmas basically generated by WordNet ([Miller, 1995](#)) lookup. For the present study we decided to use the manually corrected part of the British National Corpus (BNC) ([Consortium et al., 2007](#)). Since the manually corrected version of the BNC does not give lemmas, we had to take them from the corresponding sentences in the machine annotated version. In case the POS-

tag did not match we used some heuristic rules to find the correct lemma. The manually corrected version uses the CLAWS7 tag set, that is extremely fine grained. We decided to map the tags to the CLAWS5 tag set that would also enable the use of much more training data from the automatic tagged corpus in the future. In order to include both British and American spelling we added the Brown Corpus to our data as well. To do so, we first trained a HanTa-tagger on the BNC data. Then we changed each sentence from Brown to British spelling and tagged it with the HanTa-BNC tagger. If the original Penn Treebank tag can be mapped unambiguously to a CLAWS5 tag, that tag is used, of course. Otherwise, if the CLAWS5-POS found by the tagger corresponds to the Penn Treebank tag provided by the Brown corpus we assume that the tag and the lemma are correct. If HanTa found a different tag, HanTa was forced to reanalyze the word with the given POS and the lemma that was found then was used. Finally, a few rule based changes were made to overcome differences in the tagging policy form both corpora.

The generation of the morpheme lists from the given POS and lemma is a two pass process. In the first pass the word stem is derived from the lemma and the word is split up in prefix, infix, stem and suffix (all affixes obviously can be empty). In the second pass frequent stems found in the first pass are used to find derived words.

3.3 Data Sets

Table 1 gives an overview of the sizes of the datasets for German, English and Dutch in terms of words and sentences. The number of POS-tags used for English and German is roughly the same, while the Dutch tags are much more fine grained and in fact already include detailed morphological features. Figure 1 shows that in all three languages the number of different lemmas is clearly smaller than the number of different words (types). The

Table 1: Sizes of the data sets and number of tags assigned to words and morphemes.

	English	German	Dutch
Sentences	106,709	50,474	52,922
Words	2,171,346	952,110	911,857
POS-tags	63	55	216
Morph-tags	87	95	223

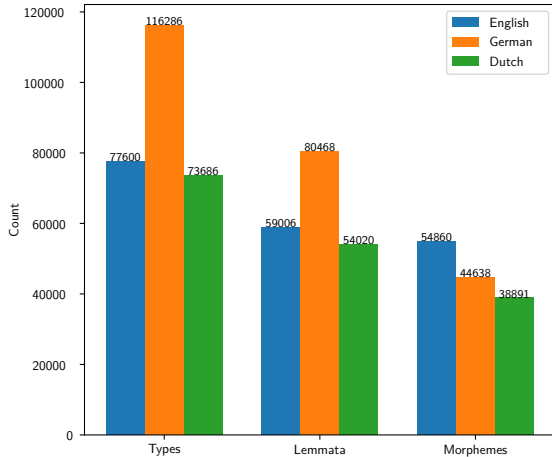


Figure 1: Absolute number of types, different lemmas and morphemes in the three datasets.

reduction ranges from 24% in English to 31% in German. The number of different morphemes in each language is again smaller, especially in the German dataset in which there are almost twice as many lemmas than morphemes. In Figure 2 we see that in each language the majority of the words consists of one morpheme. In the English data set almost all words consist of one or two morphemes while in the German and Dutch datasets a few percent of the words are composed from three or four morphemes. English still has a substantial number of four-morpheme words (e.g. *un-expect-ed-ly*), while both Dutch and German have some examples of words consisting of nine morphemes (e.g. German *Bund-es-tag-s-ab-ge-ordn-et-er* ‘member (delegate) of the Bundestag’ or Dutch *ont-wikkel-ing-s-samen-werk-ing-s-beleid* ‘policy on development cooperation’).

The original data do not give a split into training and test data. We divided the corpus into 8 equal contiguous bands and drew test and validation sentences from the tail of each band, ensuring representation across the full positional range of the corpus.

3.4 Data representation for the LSTM model

In order to handle the morphological analysis and the POS-tagging as one sequence labeling task we use IOB-type of tags. The first character of a morpheme X gets the label X and all subsequent characters the label X/c. This doubles the number of morpheme tags that has to be assigned. At the end of every word we add a <EOS> character that will be annotated with the POS of the word.

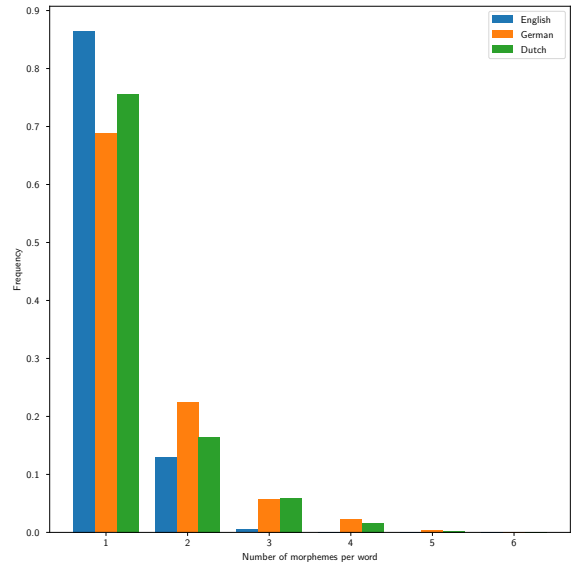


Figure 2: Distribution of the number of annotated morphemes per word for English, German, and Dutch data.

The total number of tags that the classifier has to learn thus ranges from 237 for English to 662 for Dutch. E.g. for the word *findings* in the original annotated data the lemma *finding* was given, the pos NN2 and the morphological analysis [('find', 'VV'), ('ing', 'SUF_ING'), ('s', 'SUF_NN_S')]. In order to train the network we have the lemma as target string and the following labels for the extended input string: ['VV', 'VV/c', 'VV/c', 'VV/c', 'SUF_ING', 'SUF_ING/c', 'SUF_ING/c', 'SUF_NN_S', 'END_NN2'].

4 Method

As stated above, our goal is to train a single neural network that learns morphological analysis, POS-tagging, and lemmatization jointly. The central design choice of Flexor is that a single contextualized word representation is shared between the morphological classifier and the lemma decoder, allowing both tasks to benefit from the same structural and contextual information.

4.1 Network architecture

As explained in Section 3.4, we have reduced POS-tagging and morphological analysis to a character-labeling problem. The basic idea is to use an LSTM running over the character embeddings of each word for this task. The same LSTM computes a word representation that is used by a decoder LSTM to generate the lemma. Since the POS and morpholog-

ical features of a word can be ambiguous without context, we use a global bidirectional LSTM that takes word representations as both input and output. As input, we use the last hidden state of each word. The output of this global LSTM provides an embedding for each word, which plays a central role in our approach. This embedding is used to initialize a final LSTM running over the characters of each word and producing the character labels. The same embedding is also used to initialize the decoder that generates the lemma. In addition, the decoder uses cross-attention over the original character embeddings. As we observed in earlier versions, the cross-attention mechanism struggles to focus on the correct position in the input string, especially for long words with repeated character sequences. We address this problem by adding a simple sinusoidal positional encoding to the character embeddings. The encoder and decoder share the same character embeddings. The overall architecture is shown in Figure 3. Note that the Word Encoder is first run for each word in the sentence, then the Sentence Encoder is run once for the whole sentence, and finally the Morph Classifier and Lemma Decoder are run again for each word in the sentence.

We start with character embeddings of 128 dimensions. After the first bidirectional LSTM, we obtain representations of 256 dimensions. The Sentence Encoder uses a hidden size of 128 dimensions and thus yields 256-dimensional representations. All LSTMs consist of a single layer. The cross-attention mechanism uses an attention mask centered around the character following the character that received the highest attention in the previous step. Characters located more than 5 positions away from the assumed new center are masked out. A much smaller window would in fact be sufficient (the attention stays at the same character or skips the length of an infix, i.e. at most 2 characters in our data), but we leave it to the attention mechanism itself to learn a useful alignment.

The number of parameters differs slightly between the three languages, as this depends on the number of characters found in the training data and on the number of labels. For English and German, we have 3.5 million trainable parameters, and slightly more for Dutch due to the large number of POS-classes.

4.2 Training objectives

We use negative log-likelihood loss for both the classifier and decoder, summing over all words in

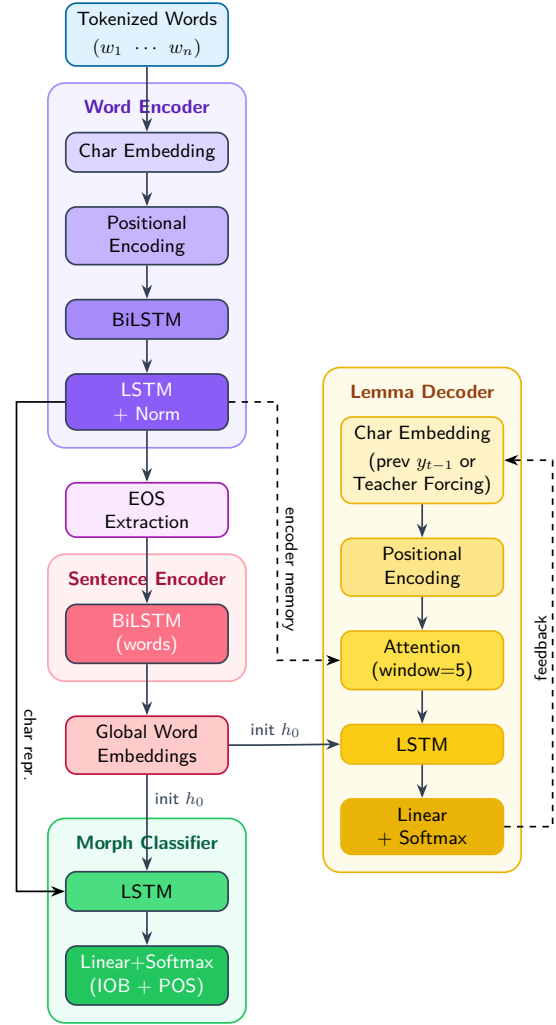


Figure 3: Network architecture of Flexor.

the sentence. We further add a transition penalty to the classifier loss that discourages invalid label sequences. For each pair of adjacent character positions, we compute the expected probability mass assigned to invalid label transitions and add it to the loss, scaled by 0.05.

4.3 Training

For English, German, and Dutch, most lemmas are identical or highly similar to their surface forms. We therefore initialize training with a copying task, encouraging the decoder to adopt copying as a default strategy before learning deviations required for lemmatization.

We select training instances that are unambiguous with respect to POS and morphology and whose lemma is identical to the surface form. We construct single-word sentences from these instances and train for one epoch before full training.

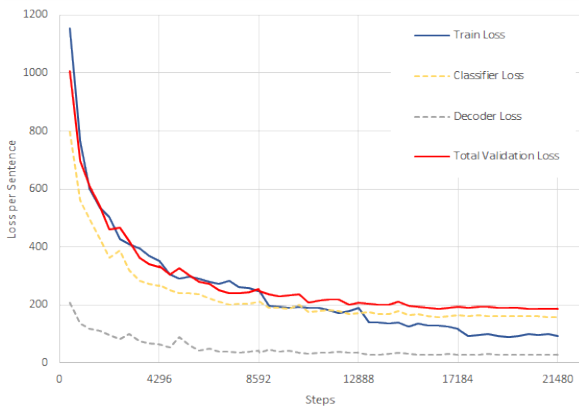


Figure 4: Train and validation loss for the German training data. The dotted lines for classifier and decoder loss sum up to the total validation loss.

We train for 4 epochs on English and German and 5 epochs on Dutch, as validation loss stabilizes at these points. We do not use early stopping and select the final checkpoint after all epochs. We use a batch size of 32 and the Adam optimizer with a learning rate annealed from $1.0 \cdot 10^{-3}$ to $1.0 \cdot 10^{-6}$. For the decoder, the teacher forcing rate is linearly reduced from 0.9 to 0.7.

Figure 4 shows training and validation loss on the German dataset (averaged over 200 batches). Validation loss is further decomposed into classifier and decoder components. The classifier dominates the total loss. The decoder loss stabilizes after approximately 1.5 epochs, while training loss continues to decrease in the final epoch without further gains in validation performance.

4.4 Evaluation

As evaluation measure we use accuracy for all three tasks. For the morphological analysis this is quite a strong assessment method: we consider an analysis to be correct or wrong. If a word is composed of three morphemes and Flexor finds the correct parts of the words and correct labels for two of the morphemes but gives the wrong label for one morpheme, we consider the analysis as incorrect.

As baseline we use the HanTa tagger version 1.2.0 from which we also took our training data. The training data were designed to train HanTa. Thus HanTa will give a strong baseline. It is tempting to compare the results with other POS-taggers and lemmatizers. However, this is an extremely difficult task, since taggers are trained on different data using different tags, different tagging policies and different underlying linguistic theories. Since

Table 2: Accuracy for all three tasks and three tests sets.

	Flexor	Hanta
English		
Lemma	0.9930	0.9867
POS	0.9786	0.9660
Morphology	0.9797	0.9287
German		
Lemma	0.9892	0.9591
POS	0.9780	0.9631
Morphology	0.9618	0.9542
Dutch		
Lemma	0.9901	0.9273
POS	0.9717	0.9469
Morphology	0.9625	0.9420

accuracy of most POS-taggers is very high, slightest mismatches in a mapping from one tag set to another will completely obfuscate the comparison of different taggers. Thus, we will only evaluate how well our method is able to reproduce the kind of tags and analyses provided in the HanTa training data. For the comparison we train HanTa with exactly the same split of the data as we used to train Flexor.

5 Results

Table 2 gives the results for the three tasks and the three languages. We observe that for every single task and language pair Flexor clearly outperforms HanTa. Furthermore, we see that the accuracy of lemmatization is 99% for each language. For POS-tagging 98% is reached for English and German and 97% for the Dutch data, that has roughly four times as much different POS-labels than the German and English datasets. For the morphological analysis we obtain 96% accuracy for German and Dutch and 98% for English, where lemmatization is easier.

If we inspect the errors we can make some interesting observations: In the first place we see that most lemmatization errors made by HanTa are related to capitalization. Many lemmas have capitals inside the word, especially proper names with a hyphen (e.g. *Saudi-Arabien*). HanTa internally works only with lower case and learns what lemmas have to be capitalized, but only considers the first character.

If we look at the errors from Flexor, we see that

most errors are related to ambiguity of POS-tags and corresponding morphological analysis. Many cases of ambiguity are solved correctly but some cases require a deeper syntactic and semantic analysis than Flexor can make. The second type of errors is related to compounds, where either Flexor analyzes a word as one morpheme, but it was annotated as a compound in the test data, or the other way around! E.g. the word *fingerprint* consists of two morphemes in the test data but is analyzed as one morpheme by Flexor. In contrast we find the word *Syrian-controlled* as one large morpheme in the test data but is analyzed (almost) correctly by Flexor as composed of a proper noun and an adjective. We find similar examples in the German and Dutch data.

A third type of error is frequently found in the English data. In English the POS-ambiguity of words is extremely high. In the data we find two ways to deal with this: in the first case it is assumed that there are different morphemes with a different POS-label. In the second case it is assumed that the morphemes are the same in each reading, but that a different POS-tag is assigned to the same morpheme sequence. An example of the first case is the word *little*. This word can be adjective, adverb, pronoun or noun (see e.g. Mariam Websters dictionary). in case of an adjective Flexor should simply tag the word as ['AJ', 'AJ/c', 'AJ/c', 'AJ/c', 'AJ/c', 'AJ/c', 'END_AJO'] but instead it produces ['AV', 'AV/c', 'AV/c', 'AV/c', 'AV/c', 'AV/c', 'END_AJO']. This kind of analysis, however is intended for present participles that are analyzed as verb if used with an auxiliary in the continuous tense but as an adjective if used in combination with a noun. In both cases the word, however, is analyzed as a verb stem with the suffix *-ing*. In fact in these cases the morph classifier produces an invalid label sequence. Eventually, the results here can be improved if we first collect all valid label transitions and use these in the loss function. Currently only a small penalty is given for POS-tags that do not appear at the end of a sequence and X/c-tags that do not follow an X-tag.

5.1 Ablation Studies

The central idea of our architecture is that we jointly train three different but related tasks and that the POS-tagging and lemmatization are based on the same representation of the word. We finally would like to investigate whether these three tasks support each other or contrary hinder each other. To answer

Table 3: Ablation: results on the German test data in case the model is trained after removing one training goal.

Removed	Lemma	POS	Morphology
–	0.9892	0.9780	0.9618
Lemma	–	0.9773	0.9629
POS	0.9885	–	0.9617
Morphology	0.9890	0.9755	–

this question we train the German model three times removing one of the training goals. We use the German model, because the quality of the data is quite high for German and German has the most complex morphology of the three studied languages. The results are shown in Table 3. The ablation study shows no significant task interference or beneficial interaction between tasks – each task appears to be largely independent. This means that the joint training and the joint representation are possible and useful from an efficiency point of view. The multi-task setup is not hurting anything, but it is not helping either.

6 Discussion and Future Work

We have presented a neural architecture for joint morphological analysis, POS-tagging, and lemmatization. In contrast to conventional approaches that predict morphological features directly, our model performs morphological analysis through morpheme-level character labeling. A contextualized word representation is shared between the morphological classifier and lemma decoder, allowing all three tasks to be learned jointly. The results demonstrate that this approach achieves high accuracy across all tasks and substantially outperforms existing taggers trained on the same data.

The proposed architecture is also relevant for practical NLP pipelines, where tokenization, POS-tagging, and lemmatization remain common preprocessing steps. However, the current model assumes pre-tokenized input, and its performance may be affected by mismatches between the tokenization conventions used during training and deployment. An important direction for future work is therefore the integration of tokenization into the model, enabling full end-to-end analysis. A further obvious extension would be to include full morphological feature prediction, which now in fact already is included in the Dutch data but not yet in the German and English data.

A major limitation of the approach is the availability and quality of annotated training data. Our experiments suggest that annotation errors and inconsistencies already impose a practical upper bound on achievable performance. Future work could therefore investigate semi-supervised or bootstrapping approaches, in which automatically analyzed data are manually corrected and incorporated into the training set. Given the high accuracy of the current model, such approaches appear promising for reducing annotation effort and extending the system to additional languages and domains.

References

- Gulinigeer Abudouwaili, Sirajahmat Ruzmamat, Kahaerjiang Abiderexiti, Binghong Wu, and Aishan Wu-maier. 2024. [A benchmark for morphological segmentation in uyghur and kazakh](#). *Applied Sciences*, 14(13).
- Judit Acs, Endre Hamerlik, Roy Schwartz, Noah A. Smith, and Andras Kornai. 2024. [Morphosyntactic probing of multilingual bert models](#). *Natural Language Engineering*, 30(4):753–792.
- Jan A. Botha and Phil Blunsom. 2013. [Adaptor Grammars for learning non-concatenative morphology](#). In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 345–356, Seattle, Washington, USA. Association for Computational Linguistics.
- Sabine Brants, Stefanie Dipper, Silvia Hansen, Wolfgang Lezius, and George Smith. 2002. The tiger treebank. In *Proceedings of the workshop on treebanks and linguistic theories.*, volume 168.
- Grzegorz Chrupala, Georgiana Dinu, and Josef van Genabith. 2008. [Learning morphology with Morfette](#). In *LREC 2008*.
- BNC Consortium and 1 others. 2007. British national corpus 1994. *Oxford Text Archive Core Collection*.
- Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(1):3.
- Alexander Gashkov and Mariia Eltsova. 2018. Lemmatization with reversed dictionary and fuzzy sets. In *SHS Web of Conferences*, volume 55, page 04007. EDP Sciences.
- Andrea Gesmundo and Tanja Samardžić. 2012. [Lematisation as a tagging task](#). In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 368–372, Jeju Island, Korea. Association for Computational Linguistics.
- John Goldsmith. 2001. [Unsupervised learning of the morphology of a natural language](#). *Computational Linguistics*, 27(2):153–198.
- John A Goldsmith, Jackson L Lee, and Aris Xanthos. 2017. Computational learning of morphology. *Annual Review of Linguistics*, 3:85–106.
- Sidney Greenbaum and Randolph Quirk. 1990. *A Student’s Grammar of the English Language*. Longman, Harlow, Essex, England.
- Georg Heigold, Guenter Neumann, and Josef van Genabith. 2017. [An extensive empirical evaluation of character-based morphological tagging for 14 languages](#). In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 1, Long Papers*, pages 505–513, Valencia, Spain. Association for Computational Linguistics.
- Mark Johnson, {Thomas L.} Griffiths, and Sharon Goldwater. 2007. Adaptor grammars: A framework for specifying compositional nonparametric bayesian models. In *Advances in Neural Information Processing Systems 19*, pages 641–648, United States. MIT Press.
- Bart Jongejan and Hercules Dalianis. 2009. [Automatic training of lemmatization rules that handle morphological changes in pre-, in- and suffixes alike](#). In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 145–153, Suntec, Singapore. Association for Computational Linguistics.
- Jakub Kanis and Luděk Müller. 2005. Automatic lemmatizer construction with focus on oov words lemmatization. In *International Conference on Text, Speech and Dialogue*, pages 132–139. Springer.
- Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. [Neural morphological analysis: Encoding-decoding canonical segments](#). In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 961–967, Austin, Texas. Association for Computational Linguistics.
- Dan Kondratyuk. 2019. [Cross-lingual lemmatization and morphology tagging with two-stage multilingual BERT fine-tuning](#). In *Proceedings of the 16th Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 12–18, Florence, Italy. Association for Computational Linguistics.
- Kimmo Koskeniemi. 1983. *Two-level morphology: A general computational model for word-form recognition and production*, volume 11. University of Helsinki, Department of General Linguistics Helsinki.

- Kirill Milintsevich and Kairit Sirts. 2021. [Enhancing sequence-to-sequence neural lemmatization with external resources](#). In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3112–3122, Online. Association for Computational Linguistics.
- George A. Miller. 1995. [Wordnet: A lexical database for english](#). *Commun. ACM*, 38(11):39–41.
- Thomas Müller, Ryan Cotterell, Alexander Fraser, and Hinrich Schütze. 2015. [Joint lemmatization and morphological tagging with lemming](#). In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 2268–2274, Lisbon, Portugal. Association for Computational Linguistics.
- Nelleke Oostdijk. 2012. Sonar-1. corpus van 1 miljoen woorden verrijkt met diverse semantische annotaties.
- Anna Rogers, Olga Kovaleva, and Anna Rumshisky. 2020. [A primer in BERTology: What we know about how BERT works](#). *Transactions of the Association for Computational Linguistics*, 8:842–866.
- Teemu Ruokolainen, Oskar Kohonen, Kairit Sirts, Stig-Arne Grönroos, Mikko Kurimo, and Sami Virpioja. 2016. [A comparative study of minimally supervised morphological segmentation](#). *Computational Linguistics*, 42(1):91–120.
- Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. 2013. [Supervised morphological segmentation in a low-resource learning setting using conditional random fields](#). In *Proceedings of the Seventeenth Conference on Computational Natural Language Learning*, pages 29–37, Sofia, Bulgaria. Association for Computational Linguistics.
- Kairit Sirts and Sharon Goldwater. 2013. [Minimally-supervised morphological segmentation using Adaptor Grammars](#). *Transactions of the Association for Computational Linguistics*, 1:255–266.
- Olia Toporkov and Rodrigo Agerri. 2024. [On the role of morphological information for contextual lemmatization](#). *Computational Linguistics*, 50(1):157–191.
- Christian Wartena. 2019. A probabilistic morphology model for german lemmatization. In *Proceedings of the 15th Conference on Natural Language Processing (KONVENS 2019): Long Papers*, pages 40–49, Erlangen, Germany. German Society for Computational Linguistics & Language Technology.
- Christian Wartena. 2023. [The Hanover Tagger \(version 1.1.0\) - lemmatization, morphological analysis and pos tagging in python](#). Technical report, Hochschule Hannover.
- Kay-Michael Würzner and Bryan Jurish. 2015. Dsolve—morphological segmentation for german using conditional random fields. In *Systems and Frameworks for Computational Morphology*, pages 94–103, Cham. Springer International Publishing.