UNIVERSITY OF CAPE TOWN

DEPARTMENT OF COMPUTER SCIENCE

CS Honours Project Final Paper 2024

Title: Deep Learning for Morphological Parsing of Nguni Languages

Author: Cael Marquard

Project Abbreviation: MorphParse

Supervisor(s): Francois Meyer

Category	*Min*	*Max*	Chosen
Requirement Analysis and Design	*0*	*20*	0
Theoretical Analysis	*0*	*25*	0
Experiment Design and Execution	*0*	*20*	20
System Development and Implementation	*0*	*20*	5
Results, Findings and Conclusions	*10*	*20*	20
Aim Formulation and Background Work	*10*	*15*	15
Quality of Paper Writing and Presentation	*10*		10
Quality of Deliverables	*10*		10
Overall General Project Evaluation (*this section allowed only with motivation letter from supervisor*)	*0*	*10*	0
Total marks		**80**	

Deep Learning for Morphological Parsing of Nguni Languages

Supervisor: Francois Meyer

Cael Marquard
mrqcae001@myuct.ac.za
University of Cape Town
Cape Town, Western Cape, South Africa

ABSTRACT

Morphemes are the smallest units of meaning in a language. They play an important role in creating meaning and grammatical syntax. This is especially true in agglutinative languages, such as the Nguni languages, which construct words by concatenating many morphemes. Many Natural Language Processing (NLP) tasks, such as machine translation, can be improved by incorporating morphological information. However, methods to extract this information for the Nguni languages still need to be refined. This paper evaluates the use of neural methods in morphological parsing, or the labelling of morphemes with their corresponding grammatical role. Two main approaches are compared: training neural models from scratch, and fine-tuning pre-trained language models. The performance of these models are compared with eachother as well as to traditional methods of solving the task, such as Finite-State Transducers (FST) models. It was found that models trained from scratch outperformed both fine-tuned pre-trained language models and traditional FST models. Models using morpheme-level embeddings and sentence-level context tended to perform the best.

1 INTRODUCTION

Morphological parsing is the task of identifying the grammatical role of each morpheme within a word [38]. For example, "aliqela" (meaning "they are several" in isiXhosa) is split into the morphemes "a-li-qela", which could be parsed as "a[RelConc6] - li[BPre5] - qela[NStem]". Each bracketed tag labels the preceding morpheme with its grammatical function, as well as noun class if applicable. The morpheme "qela" is the word's noun stem in this example. The goal of the task is to predict these morpheme tags at the sentence level.

Many common Natural Language Processing (NLP) tasks, such as machine translation and information retrieval rely on *lexical semantics*, or the meaning of individual words and their connections to other words [8]. Since the individual morphemes of which a word is composed create the whole word's meaning, morphological features are therefore important to NLP [38]. By developing improved tools for morphological parsing, these NLP tasks could be better solved through the incorporation of these features.

Few parsers exist for the Nguni languages, none of which use deep learning methods [13, 43]. Morphological information is especially important for NLP in Nguni languages due to their linguistic features:

(1) Nguni languages are agglutinative, meaning many words are created by aggregating multiple morphemes [8, 29, 42].
(2) Nguni languages are written conjunctively, meaning that morphemes are concatenated into a single word [42]. For example, in isiXhosa, "andikambuzi" means "I haven't yet asked him", and is composed of the morphemes "a", "ndi", "ka", "m", "buza", and "i".

This paper examines the use of neural methods for morphological parsing in the Nguni languages. Whilst my partner, Simbarashe Mawere, examined the fine-tuning of pre-trained language for this task, this paper evaluates models trained from scratch. The selected models are bi-directional Long Short-Term Memory (bi-LSTM) models and Conditional Random Fields (CRFs) with bi-LSTM features (bi-LSTM CRFs) [22]. These models were selected due to their previous application to related sequence labelling tasks in Nguni languages, such as part-of-speech tagging and morphological segmentation [29, 33].

Neural models will be examined in relation to traditional approaches to morphological parsing to determine their suitability for this task. The performance of pre-trained language models will be compared to models trained-from scratch to ascertain whether this is an area in which transfer learning techniques may be applied. Finally, the neural models will be evaluated both on surface and canonical segmentations to determine which segmentation approach is most conducive to the task of morphological parsing.

2 RELATED WORK

2.1 Neural morphological parsing

Traditionally, morphological parsing is done by manually incorporating grammatical and morphological rules about the language into a model. For example, the ZulMorph analyser is a rule-based morphological parser for isiZulu [8, 37].

This approach requires linguists to define grammatical and morphological rules about each language, which are then used to create the model. Therefore, these models require a high degree of linguistic expertise, are time-consuming to produce, and do not generalise well to dissimilar languages [10, 11]. Solutions based on machine learning, such as sequence-to-sequence [41] and sequence labelling models [26], address these issues as they do not require any linguistic expertise to produce, need only existing annotated data to produce, and may be created in a language-independent manner.

Learning-based approaches have become more common in morphological parsing [24]. They avoid the drawbacks of more manual approaches by instead using large datasets of annotated text to automatically learn how to classify morphemes with their grammatical role. For instance, Puttkammer and Du Toit [38] use MarMoT [6, 31], a trainable Conditional Random Field (CRF) [22] pipeline, to syntactically tag morphemes in Nguni languages.

While non-neural, learning-based approaches improved upon rule-based methods, neural methods tend to outperform non-neural, learning-based models on many tasks relating to computational morphology [24]. For instance, MarMoT's [31] approach was improved by Labeau et al. [21] on German by incorporating neural features [24]. Akyürek et al. [3] have applied a neural sequence-to-sequence model to the task of morphological parsing and achieved excellent results on a variety of European languages.

2.2 Computational linguistic analysis of Nguni languages

2.2.1 Datasets. Several datasets of linguistic annotations suitable for training and evaluating computational linguistic tools are available for the Nguni languages. The Ukwabelana corpus [40] has been used to train morphological segmenters for isiZulu [28]. Eiselen and Puttkammer [16] created a corpus of ten South African languages (including the Nguni languages), which has also been used to train morphological segmenters for the Nguni languages (e.g., by Moeng et al. [29]).

Eiselen and Puttkammer [16] used an older morphological annotation protocol, which is no longer the standard. It was not updated to more modern protocols until Gaustad and McKellar [17] did so after this research project was under way. Gaustad and Puttkammer's dataset [18], published in 2022, was used for this project, since it covers all official Nguni languages and uses the latest morphological annotation protocols.

2.2.2 Morphological segmentation. Morphological segmentation refers to decomposing a word into its constituent morphemes. There are two subcategories of this task: canonical and surface segmentation. Canonical segmentation refers to decomposing a word into its constituent morphemes, which are the smallest units of meaning in a language [11, 29, 39]. Surface segmentation refers to decomposing a word into its constituent *morphs*, which are the surface-form of morphemes as they appear in the word [29, 39]. A word's morphs may differ from its morphemes due to spelling changes that occur during morpheme attachment [29, 39].

ZulMorph [7] applies traditional, finite-state transducers to canonical segmentation for isiZulu. The approach has also been adapted to other languages, such as isiNdebele [36]. Puttkammer and Du Toit [38] applied memory-based (non-neural) machine-learning algorithms to the task of canonical segmentation. Neural machine-learning methods have also been explored. Mkhwanazi and Marais [28] applied Transformers [43] to the task of surface morphological segmentation for isiZulu, whilst Moeng et al. [29] apply several models, both neural and non-neural, to canonical and surface segmentations. Moeng et al. [29] found that (non-neural) feature-based CRFs performed the best for surface segmentation whilst Transformers performed the best for canonical segmentation.

2.2.3 Morphological parsing. ZulMorph [7] applies finite-state transducers to morphological analysis for isiZulu. The system both segments words into their canonical morphemes and parses their grammatical roles. Puttkammer and Du Toit [38] applied MarMoT [6], a combined part-of-speech and morphological parser based on CRFs, to syntactic morphological parsing for the Nguni languages. This is a variation of morphological parsing in which noun-class information is not parsed. For instance, the word "aliqela" would be parsed fully as "a[RelConc6] - li[BPre5] - qela[NStem]", but syntactically as "a[RelConc]-li[BPre]-qela[NStem]" [18, 38].

2.2.4 Sequence tagging. Deep-learning approaches have not yet been applied to morphological tagging for Nguni languages. However, these approaches have been applied to a few closely related tasks, including part-of-speech tagging and Named Entity Recognition (NER). Other systems for Nguni languages have already been developed for these tasks.

Part-of-speech tagging refers to the task of labelling each word in a sentence with its lexical category, thus classifying the grammatical role that it plays in the sentence [33]. This is similar to morphological parsing, but operates at a word level rather than at a morpheme level. Du Toit and Puttkammer [15] applied MarMoT, a CRF-based part-of-speech and morphological tagger, to part-of-speech tagging for the Nguni languages. Dione et al. [14], in their MasakhaPOS project, applied CRFs and pre-trained language models to part-of-speech tagging for several African languages. Pannach et al. [33] applied both non-neural and neural models to part-of-speech tagging for the Nguni languages, and found that the neural models performed the best.

NER refers to the identification of sequences in text such as names (proper nouns) and numeric expressions [32]. It relates to morphological parsing in that they can both be treated as sequence-labelling tasks [23]. Lample et al. [23] applied bidirectional Long Short-Term Memory networks (bi-LSTMs) and CRFs with bi-LSTM features to NER. MasakhaNER [2] applied pre-trained language models to NER for several African languages. MasakhaNER 2.0 [1] extended this with the notable inclusion of Nguni languages.

3 METHODOLOGY

3.1 Architecture

In order to simplify the task of the models, the models will tag already-segmented morphemes with their grammatical roles. This is similar to the second-part of the two-step approach taken by Puttkammer and Du Toit [38] for syntactic tagging of morphemes in Nguni languages (see Figure 1). The models will be trained on the gold-standard segmentations (both surface and canonical). In order to evaluate the models end-to-end on raw text, pre-existing canonical and surface segmenters will be used. This approach has several advantages.

Firstly, the task of the model is simpler, as it does not need to jointly learn the segmentation and tagging steps. Secondly, it is more modular. If a higher-quality segmenter is developed, the tagging model can simply be made to operate on those segmentations instead. Lastly, this approach allows sequence-labelling architectures to be used for this task, since almost all morphemes can be assigned only one tag. These architectures have already been used effectively for related tasks in the Nguni languages.

Two main approaches will be investigated: models operating on canonical segmentations, and models operating on surface segmentations. Both the surface and canonical models will be trained on the gold-standard segmentations and be evaluated both on the gold-standard and predicted segmentations. Evaluating on the gold-standard segmentations shows how well the models perform at the tagging step, whilst evaluating on the predicted segmentations show how well the systems function end-to-end on raw text.

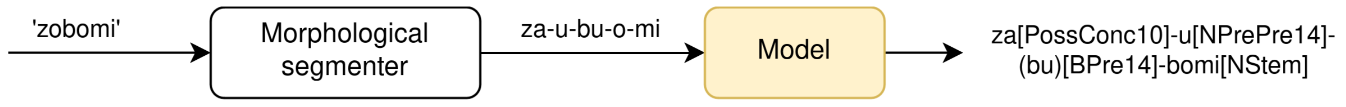

Figure 1: Architecture of the overall system operating on the word "zobomi". The model developed is highlighted in orange. Final analysis from Gaustad and Puttkammer [18].

3.2 Models

Two types of models were chosen: Long Short-Term Memory (LSTM) models and Conditional Random Fields (CRFs) with neural features.

3.2.1 Long Short-Term Memory models. LSTMs may be used to solve sequence processing tasks [20, 33]. LSTMs are a type of Recurrent Neural Networks (RNNs). These are a class of neural network able to represent recent past events for use in future computations [20]. This allows them to incorporate the context of the past sequence while processing its items one by one. Hochreiter and Schmidhuber [20] propose LSTMs as a way to avoid issues such as vanishing and exploding gradients which are present in older RNN architectures.

Bi-directional LSTMs (bi-LSTMs) are a combination of two separate LSTMs, where one reads the input from the start to the end, and the other reads the input from the end to the start in reverse [33]. The hidden states produced by each LSTM are concatenated and then often fed through another fully-connected layer to create the final output [33]. This allows the model to take into account both the future and past of the sequence, which is useful for many sequence-processing tasks [19]. Bi-LSTMs have been successfully applied to part-of-speech tagging for Nguni languages [33]

3.2.2 Conditional Random Fields. Conditional Random Fields (CRFs) are a probabilistic model often used for sequence labelling tasks [22]. A CRF models the probability of a given input sequence **X** being assigned any possible label sequence **Y**. A CRF estimates the probability of a given output sequence's correctness by modelling the interdependence of the individual labels on eachother as well as their dependence on the input sequence.

Linear-chain CRFs are CRFs in which the probability of each label depends only on the input datum, previous label and next label (see Figure 2). These were chosen for this paper due to their lower computational complexity compared to higher-order CRFs. This is especially important since the algorithm required to train CRFs has a runtime quadratic in the size of the possible tag-set [6, 15, 29], and the tag-set for morphological parsing is very large, comprising of over 200 tags per language [18]. Linear-chain CRFs with Bi-LSTM features have been applied to part-of-speech tagging [33] as well as morphological segmentation [29] for the Nguni languages.

Traditionally, CRFs use a set of hand-crafted features in order to assign probabilities [29]. However, instead of designing these features by hand, a neural network may instead be used to learn the relevant features from the data [23, 26, 29]. A bi-LSTM was chosen to generate these features, due to the successful application of bi-LSTM CRFs to NER [23] and part-of-speech tagging [33].

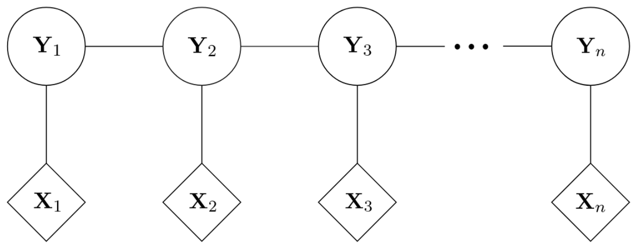

Figure 2: An example of a CRF [22] organised as a linear chain. Diamonds represent input variables $\mathrm{X_i}$ and circles represent output variables $\mathrm{Y_i}$, while edges represent statistical interdependence.

3.3 Evaluation

The quality of the models was evaluated by calculating the F_1 scores between the predicted tags and expected tags. The F_1 score is the harmonic mean of a model's precision and recall, and is a common measure of overall classification performance [12]. In the case of morphological parsing, precision is the proportion of tag identifications that were correct, whereas the recall is the proportion of all tag identifications that were correctly identified by the model [12]. Combining these two measures into the F_1 score balances the selectivity of predictions (precision) as well as their coverage (recall).

The F_1 score can be calculated in multiple ways. The two measures chosen for the experiment were the Micro F_1 score and Macro F_1 score. The Micro F_1 score is calculated on the entire predicted and expected tag sequences, and is equivalent to the accuracy of the model in this case. The Macro F_1 first calculates a per-tag F_1 score and averages them, weighting each tag equally. Because each class is not weighted according to its prevalence in the data, the Macro F_1 score penalises models which perform poorly on relatively rare tags. Because rare tags are harder to predict accurately, it is more difficult for the model to attain a higher Macro F_1 score. Therefore, the Macro F_1 score was chosen as the primary measure on which to evaluate the models.

The Micro and Macro F_1 scores for each model were averaged across five different random seeds (0, 1, 2, 3, and 4) used to initialise the models parameters. This makes the experiment more accurate, since the parameter initialisation is not an important factor to optimise for. The model with the best macro performance across all five training runs was also measured separately and saved in order to produce the best possible model for each language. The saved model is the one which is released, as it is the most useful to downstream users who simply need to predict the morphological analysis of text.

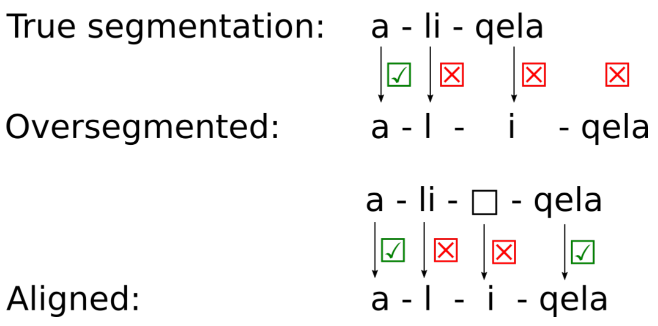

Figure 3: Over-segmented morphological segmentation with maximal alignment. Without alignment, the accuracy is 1/4. With alignment, the accuracy is 2/4.

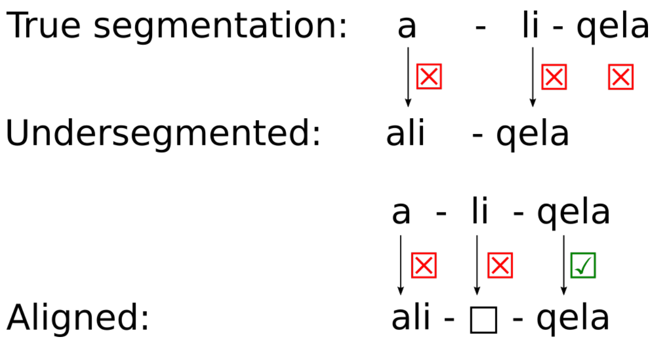

Figure 4: Under-segmented morphological segmentation with maximal alignment. Without alignment, the accuracy is 0/3. After alignment, the accuracy is 1/3.

3.3.1 Handling segmentation errors. For some of the models and baselines, such as the surface-segmented models and ZulMorph [37], the generated segmentations may not match the segmentations in the NCHLT annotated corpus. This results in under-segmentation (as seen in Figure 4) or over-segmentation (as seen in Figure 3) which is when there is a mismatch between the the number of morphemes in the segmentations. Because the aim of the project is to test morphological *parsing*, as opposed to *segmentation*, these segmentation errors should only be penalised insofar as they lead to incorrectly-classified morphemes. This was achieved by maximally aligning the predicted and target tag sequences. This prevents an early segmentation error causing every subsequent tag to be wrong.

The maximal alignment algorithm (Appendix E) aligns two sequences to maximise the number of matches between them by inserting padding items at certain indices. First, the shorter sequence of the two is identified. Then, the minimum number of padding elements are inserted into the shorter sequence such that it reaches the length of the longer sequence. The indices of the padding elements are identified through a brute force search, wherein every possible combination of insertion indices are tested. The best combination is the one yielding the most matches between the candidate padded sequence and longer sequence.

Because of the different natures of each type of model (canonical, surface, and ZulMorph baseline), some modifications had to be made to the evaluation methodology. However, in all cases, both the Macro F_1 and Micro F_1 scores were calculated (after aligning if necessary). The specific evaluation methodologies used are listed below.

3.3.2 Classifiers for canonical morphemes. The models which classified the gold-standard, canonical segmentations, for the most part, did not need much post-processing. This is because they were trained specifically to output the NCHLT tagset, and the vast majority of the gold-standard segmentations had a single tag. However, some demonstratives (e.g. ‘leso[Dem7][Pos1]” in siSwati [18]), were tagged with two tags, the first being a `DEM` tag and the second being a `POS` tag. Because the models were single-label classifiers, the `DEM` tag was chosen during training as the tag for the entire morpheme.

3.3.3 Classifiers for surface morphs. On the other hand, the models which classified surface segmentations required more modification to the evaluation methodology. This was due to the not-insignificant amount of segmentation errors. Therefore, the output of these models were aligned with the corresponding gold tag sequence by using the maximal alignment algorithm (Appendix E).

Additionally, whilst these models were trained on the surface segmentations generated from the canonical segmentations using a script, they were evaluated on the actual segmentation model output. This measures the performance of the models end-to-end on raw text. To aid in the evaluation process, the segmentation models were run over the entire test set and then these segmentations were fed as input to the surface tagger models.

3.3.4 Baseline - ZulMorph. The source code and binary of ZulMorph [37] are not currently available to the general public. However, there is a public demo[1] hosted by the South African Centre for Digital Language Resources[2] (SADiLaR). This demo was scraped and parsed using BeautifulSoup [5] to create a dataset of ZulMorph’s morphological analyses of the testset used in this project.

ZulMorph [37] makes use of a different tagset[3] than Gaustad and Puttkammer’s dataset [18]. Because of these differences, a script was created to map the ZulMorph tags to the NCHLT tags to the greatest extent possible. The full remapping process is described in Appendix B.

Because ZulMorph both segments and parses the morphemes of the provided words, there may be segmentation errors which impact the final tag sequence [37]. To avoid unfair penalisation, ZulMorph output was aligned to the expected tags using the maximal alignment algorithm (Appendix E).

4 EXPERIMENTAL SETUP

4.1 Data and materials

Gaustad and Puttkammer [18] produced an annotated corpus of the Nguni languages. This dataset contains 1 431 paragraphs in each language, comprising of roughly 50 000 tokens per language. Each word is annotated with its full morphological analysis, as seen in Table 1. This morphological analysis consists of the word broken

[1]https://portal.sadilar.org/FiniteState/demo/zulmorph/
[2]https://sadilar.org/en/
[3]Full tagset available at https://portal.sadilar.org/FiniteState/demo/zulmorph/doc.html

Table 1: Three examples from the isiXhosa part of the dataset used in our experiments [18].

Word	Morphological analysis
aliqela	a[RelConc6]-li[BPre5]-qela[NStem]
kwibhunga	ku[LocPre]-i[NPrePre5]-(li)[BPre5]-bhunga[NStem]
izincomo	i[NPrePre10]-zin[BPre10]-como[NStem]

into the canonical morphemes that it is composed of, as well as each morpheme's grammatical tag. The dataset is pre-split into a testing and training corpus, making up 10% and 90% of the dataset respectively.

To prepare the data for training, the morphological analysis of the words were split into its parallel morpheme sequence and tag sequence.

Training the surface tagging models required additional data preparation. A script developed by Dr Jan Buys was used to generate gold-standard surface segmentations from the canonical segmentations in the dataset. These surface segmentations were also aligned with the tags from the canonical analysis in the dataset. In cases where there were more canonical tags than surface morphs, some tags were dropped by the script. This is because the models are only able to output a single label for each input morph.

To create an end-to-end test dataset for the surface tagging models, Moeng et al.'s [29] surface segmentation models were used. The feature-based CRFs developed by Moeng et al. [29] were selected as they performed the best out of all models presented in the paper. These models were trained on the full train-set of the dataset developed by Gaustad and Puttkammer [18] in order to more closely match the data under study. A script was then used to run the trained models and combine the output segmentations with the gold-standard tag sequences. This allows the surface tagging models to be evaluated on their end-to-end performance, starting with raw words, being segmented by the segmentation model, and being tagged by the tagging model.

Similarly, to create an end-to-end testing dataset for the canonical tagging models, the Transformer models developed by Moeng et al. [29] were re-trained. Each language's model was trained for 150 epochs, and the epoch with the best validation loss was selected. All other hyperparameter choices were kept the same.

Further materials used, such as software and libraries, are listed in Appendix A.

4.2 Model configurations

Two types of models trained from scratch were selected: a bi-LSTM, and a bi-LSTM CRF. These models were then parameterised by other factors:

- **Segmentation type.** Models were trained on either the canonical or surface segmentations of the words. The canonical segmentations were simply taken from the dataset's annotations. The surface segmentations were generated from the canonical segmentations by using a script, which also aligned each surface morph to a single tag.
- **Feature level.** Models were trained on either morpheme-level features or character-level features. These features were given as input to the bi-LSTM and CRF models.
- **Feature embedding.** Once a feature level was selected, the features would be transformed into embeddings for processing. First, the features were embedded by using a learned embedding dictionary to produce an *intermediate* embedding representation. For the morpheme-level features, these intermediate representations were simply used as the final morpheme embedding. In cases where a morpheme occurred only once, it was replaced with an unknown morpheme symbol to help the model generalise to unseen data. For the character-level features, these intermediate embeddings were summed together to produce the final embedding.
- **Context level.** Models were trained on single words as well as on sentences.

4.3 Hyperparameter tuning

To prevent over-fitting of the hyperparameters to the test set, the models were trained on 90% of the training set and evaluated on 10% of the training set during the tuning process. The models were tuned in order to maximise their Macro F_1 scores on the validation dataset by training the models with different hyperparameters. The hyperparameters on which the models were trained are shown in Table 2. The models were tuned on isiZulu, and once the best parameters for isiZulu were found, these configurations were then applied to the other languages. The hyperparameter tuning process is described in greater depth in Appendix D.

4.4 Final training

Once the optimal hyperparameters had been selected, five models (each with a different seed) per configuration were trained for each language. These models were trained on the full training set. Sentence-level models were trained for 40 epochs, whilst word-level models were trained only for 30, since the word-level models took much longer to train. For each of these seeds, the epoch which had previously attained the best Macro F_1 on the validation set was chosen, along with the accompanying Micro F_1 for this epoch. These values were then averaged across all five seeds to determine the Macro and Micro F_1 scores. Additionally, the epoch with the best Macro F_1 score across all five seeds was saved for distribution to downstream use.

5 RESULTS AND DISCUSSION

All results reported herein include all tokens in the text, including punctuation, foreign words, and numbers. Additionally, all sequences are aligned using the maximal alignment algorithm (Appendix E).

The results for the models on the gold dataset segmentations are reported in Table 3. These models are, however, only evaluated on one tag per morpheme. In a few cases (e.g., "leso[Dem7][Pos1]" in siSwati [18]), the first tag only is taken. This is, overall, a very small portion of the dataset though.

The results for the models evaluated end-to-end by using an existing segmentation model are reported in Table 5. Because the segmentation models are not perfectly accurate, these results should be interpreted alongside the table of segmentation results (Appendix

Table 2: The set of hyperparameters which were tuned.

Hyperparameter	Search space	Explanation
Learning rate	$[10^{-6}, 10^{-1}]$	The rate at which the model learns based on the back-propagated error.
Weight decay	$\{0\} \cup [10^{-10}, 10^{-3}]$	Controls the rate at which which the weights tend toward zero over time.
Hidden state size	$\{2^x : 6 \leq x \leq 11\}$	The number of features/vector size of the hidden state of the model.
Dropout	$\{0, 0.1, 0.2, 0.3\}$	Proportion of parameters randomly zeroed during training for regularisation.
Gradient clip	$\{0.5, 1, 2, 4, \infty\}$	The maximum value of a gradient during back-propagation.

C.1). The models are tested on the full tagset, *including* both tags of double-tagged morphemes (as in “leso[Dem7][Pos1]” [18]).

5.1 Best models trained from scratch

Overall, sentence-level models tended to outperform word-level models—aside from isiZulu, sentence levels achieved the best Macro F_1 score (Table 3). Models using morpheme-level features tended to outperform those using character-summing features. The differences in performance between plain bi-LSTMs and bi-LSTM CRFs were quite small (often less than 0.01 difference).

Sentence-level models outperformed word-level models—aside from isiZulu, they achieved the best Macro F_1 scores (Table 3). These models are given the entire sentence as context, which may allow them to use grammatical dependencies to more accurately tag morphemes. For example, in the isiXhosa sentence “ipolisa liyahamba”, the word “ipolisa” is in noun class 5. The shorted prefix “i” (“**i**polisa”) is ambiguous and also appears in class 9 nouns, such as “iteksi”. However, combining it with the subject concord for class 5 “li” (“**li**yahamba”) would allow the model to place “ipolisa” in class 5 correctly.

Models using morpheme-level embeddings outperformed those using character-summing embeddings. This could be because morphemes are simply a more effective representation for syntactic tasks [44]. Additionally, morpheme-level embeddings allow the model to be more sensitive to small changes in the morpheme itself. Since each morpheme is mapped to its own learnt embedding, even a single differing character may yield an entirely different embedding. However, in the character summing approach, the contribution of each letter is much smaller. Because some of the tags in the dataset are extremely rare, the fact that each morpheme is given a unique embedding may assist the model to disambiguate these cases.

There did not tend to be a very large difference in performance between plain bi-LSTMs and bi-LSTM CRFs, with the difference in Macro F_1 scores often being less than 0.01. This could indicate that the advantage presented by explicitly modelling grammar is limited, and the bi-LSTMs are able to encode this simply through their memory cells.

5.1.1 Surface vs canonical taggers. The models which tagged the morphs from the surface segmentations of the word did not perform as well as those which operated on the canonical segmentations. When evaluated end-to-end (Table 5), the Macro F_1 attained was ≈0.04-0.07 lower than the Macro F_1 attained by the canonical segmenters. Since it was unlikely that optimising the models further would bridge this gap, only two models were trained for the surface taggers.

The feature-based CRF surface segmenter developed by Moeng et al. [29] did not perform as well as expected based on the reported accuracy in their original paper (see Appendix C.2). This could be one reason why the canonical taggers outperformed the surface taggers.

Firstly, poor segmentation quality leads to the morphs during training and evaluation diverging, since the morphs in training are gold-standard, whilst the morphs in evaluation are predicted. This impacts the morpheme-embedding models especially. These models map each unique morph in the training set to an embedding vector. This means that the morph “ntu” and “mntu” may have completely different embedded representations. In some cases, the model may even lack an embedding for an incorrectly surface-segmented morph, meaning that it would be mapped to the embedding for any unknown morpheme.

Secondly, if the number of predicted surface morphs differs from the correct number of surface morphs for a given word, the model’s maximum performance is strictly limited. This is because, being sequence labelling models, the models output one tag per morph. Thus, if the segmentation models predict that a word has four morphs, when it actually has eight, the model can, at best, get 50% of the tags correct—even with alignment (Appendix E).

This is reflected in the end-to-end parsing task, where models are given canonical segmentations as predicted by Moeng et al.’s [29] canonical transformer models. The sentence-level context, morpheme embedding bi-LSTM CRF achieved the highest Macro F_1 for all languages but isiZulu. Even so, it was only outperformed by the best model for this language by 0.0042.

5.2 Comparison to ZulMorph

ZulMorph [37] achieved a Macro F_1 of 0.3378 and Micro F_1 of 0.6471 on the test-set. This was significantly outperformed by the deep-learning models, which ranged from Macro F_1s of 0.59 to 0.6 and Micro F_1s of 0.80 to 0.82. This could be explained by a number of factors.

One mitigating factor is that the tag-set of ZulMorph is not the same as the tag-set used by Gaustad and Puttkammer [18] in the dataset. In order to perform the comparison, the ZulMorph tags were mapped to the tags which they matched most closely. However, this mapping is imperfect, and there may be errors impacting the final performance. Additionally, ZulMorph’s segmentation performance is poorer than the canonical segmentation transformers developed by Moeng et al. [29]. Since ZulMorph is an end-to-end system which accepts raw text, it could not be tested on this output, nor on the gold-standard morphemes themselves.

Since ZulMorph is rule-based and contains manually-incorporated stems and affixes, it likely struggles to generalise to unseen data. For

Table 3: Results for the canonical models on gold-standard canonical segmentation (tags reduced to one tag per morpheme). The best models for each approach (pre-trained or from scratch) is bolded, whilst the best overall is underlined.

Model and embedding	IsiZulu		IsiNdebele		IsiXhosa		SiSwati	
	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1
Word-level								
Bi-LSTM, morpheme	**0.677**	**0.926**	0.686	**0.922**	0.724	0.952	0.670	0.915
Bi-LSTM, character-sum	0.672	0.927	0.683	**0.922**	0.728	0.955	0.669	0.917
Sentence-level								
Bi-LSTM, morpheme	0.669	**0.926**	**0.689**	0.918	**0.764**	0.961	0.680	0.918
Bi-LSTM, character-sum	0.646	0.920	0.678	0.912	0.742	0.957	0.666	0.912
CRF, morpheme	0.661	**0.926**	0.687	0.919	0.762	0.961	**0.684**	**0.920**
CRF, character-sum	0.658	0.924	0.685	0.916	0.750	**0.962**	0.664	0.917
Pre-trained language models (word-level)								
XLM-RoBERTa [9]	0.6157	0.9132	**0.6425**	**0.9152**	0.6773	0.9467	0.6420	0.9095
AfroXLMR [4]	**0.6610**	**0.9282**	0.6273	0.9133	**0.7363**	**0.9583**	**0.6460**	**0.9100**
NguniXLMR [27]	0.6302	0.9187	0.6190	0.9104	0.6738	0.9488	0.6176	0.9042

Table 4: Results for the surface models on gold-standard surface segmentation. The best models are bolded and underlined.

Model and embedding	IsiZulu		IsiNdebele		IsiXhosa		SiSwati	
	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1
Sentence-level								
Bi-LSTM, morpheme	**0.659**	**0.912**	**0.677**	**0.911**	**0.767**	**0.952**	**0.657**	**0.905**
Bi-LSTM, character-sum	0.626	0.907	0.650	0.903	0.747	0.949	0.613	0.894

instance, ZulMorph failed to segment and parse “wezentuthuko”, and instead output “wezentuthuko +?”. Conversely, the neural models do not explicitly incorporate any information. The models are able to classify text even when there are unknown morphemes present in the text based on the surrounding context and the known grammatical morphemes.

These two factors help to explain why the surface segmentation models may perform worse than the canonical segmentation models on the end-to-end task. However, even when tested on gold-standard segmentations, the surface tagging models (Table 4) performed worse than the canonical tagging models (Table 3). This could be due to a few factors.

Firstly, the surface segmentation of a word may provide less information to the model than the canonical segmentation. For instance, the word “kwicandelo” is canonically segmented as “ku-i-(li)-candelo” and surface segmented as “kw-i-candelo” [18]. Critically, the “(li)” morpheme is lost, which is the noun prefix for class 5. Without this information, the model may not be as accurate in predicting the noun class of the “i” morpheme as class 5, as opposed to another class prefixed by “i” in its surface form (e.g. class 9).

Secondly, the script which generates the gold-standard surface segmentation selects only one tag for each morpheme. It is possible that the tags which are selected might be harder for the model to infer compared to other possible choices. Additionally, these choices may result in the same tags being assigned to many different morphemes, which could make it more difficult for the model to accurately classify them.

5.3 Comparison to pre-trained language models

Fine-tuning pre-trained language models for morphological parsing was the other approach taken in the project, and was investigated by my partner. In most cases, the models trained from scratch outperformed the pre-trained language models fine-tuned for the task of morphological parsing. This could be due to their smaller size. In general, the pre-trained language models are much large, featuring more parameters than the models trained from scratch. This has two main effects.

Firstly, these models are more computationally expensive to train than the comparatively smaller models trained from scratch. This means that, for the sake of time, the hyperparameter tuning process cannot be as in-depth, since the hyperparameter space cannot be explored as quickly.

Secondly, the size of these models may lead to over-fitting on the training set, which may limit their ability to generalise to the testing set. For instance, Nguni-XLMR-large [27], one of the pre-trained language models, consists of over 355M parameters. By comparison, the morpheme-level embedding, sentence-level context CRF trained herein has only $\approx$ 5 million parameters. In many cases, the largest embedding and hidden dimension sizes tested did not produce the best results. This could suggest that, for the task of morphological parsing for this dataset at least, fewer parameters are favoured to avoid over-fitting.

Lastly, these models make use of their own tokenisation scheme which may be negatively impacted by the splitting of morphemes. For instance, the model’s owned tokenisation scheme may rely on having the surface forms of the morphemes all concatenated. The canonical segmentation of a word may be very different to this.

Table 5: End-to-end parsing performance of the best models and baselines on the full, uncleaned tagset. Segmentation type is indicated through headers. The best models for each approach (canonical or surface) is bolded, whilst the best overall is underlined.

Model	IsiZulu		IsiNdebele		IsiXhosa		SiSwati	
	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1
ZulMorph [37]	0.3378	0.6471	-	-	-	-	-	-
Canonical segmentations using Moeng et al. [29]								
Word-level								
Bi-LSTM, morpheme	0.5936	0.8250	0.5769	0.8084	0.6807	0.9110	0.5717	0.8230
Bi-LSTM, char-sum	**0.6033**	0.8248	0.5837	0.8094	0.6816	0.9107	0.5760	0.8289
Sentence-level								
Bi-LSTM, morpheme	0.5989	0.8265	0.5834	0.8059	0.7171	0.9172	0.5792	0.8274
Bi-LSTM, char-sum	0.5903	0.8184	0.5814	0.8010	0.7020	0.9135	0.5749	0.8238
CRF, morpheme	0.5991	0.8280	**0.5961**	0.8049	**0.7224**	**0.9190**	**0.5860**	**0.8301**
CRF, char-sum	0.5973	0.8246	0.5893	0.8081	0.7084	0.9181	0.5782	0.8271
Pre-trained language models (word-level)								
XLM-RoBERTa [9]	0.5468	0.8254	0.5540	0.8159	0.6375	0.9149	0.5308	0.8293
AfroXLMR [4]	0.5532	0.8283	0.5488	0.8149	0.6523	0.9150	0.5405	0.8293
NguniXLMR [27]	0.5568	**0.8285**	0.5505	**0.8162**	0.6444	0.9161	0.5398	0.8290
Surface segmentations using Moeng et al. [29]								
Sentence-level								
Bi-LSTM, morpheme	**0.5529**	**0.8023**	**0.5409**	**0.7830**	**0.6724**	**0.8661**	**0.5274**	**0.8129**
Bi-LSTM, char-sum	0.5511	0.7976	0.5238	0.7742	0.6054	0.7994	0.5212	0.8053
Pre-trained language models (word-level)								
XLM-RoBERTa [9]	0.4349	0.6759	0.4868	0.7282	0.5208	0.7244	0.2231	0.5107
AfroXLMR [4]	0.4495	0.6794	0.4832	0.7275	0.5300	0.7273	0.2412	0.5216
NguniXLMR [27]	0.4512	0.6818	0.4746	0.7258	0.5270	0.7309	0.2505	0.5328

This may limit the degree of transfer learning that can occur. In the models trained from scratch, the level of feature tokenisation had a significant impact on model performance, and the same may be true of the pre-trained language models.

5.4 General trends

In general, the Macro F_1 scores of the models were significantly lower than the corresponding Micro F_1 scores. This is likely due to the fact that the tagset is very large, with many tags having very few examples in the dataset. This imbalance could be a reason why the models struggle to classify rarer classes compared to the more frequent ones. However, this is also the case for ZulMorph [7] (Table 5), which is not trained on the dataset. This could indicate that these tags are simply harder to classify correctly.

6 CONCLUSIONS

Morphological parsing for the Nguni languages is traditionally done using rule-based analysers. However, it is also possible to apply deep-learning approaches to this task. Deep-learning models trained from scratch outperformed pre-trained language models, and both deep-learning approaches outperformed the tested traditional, rule-based analyser. Models classifying canonical segmentations performed better than those classifying surface segmentations. This could be due to the fact that more information is provided in canonical segmentations, or to the fact that there are generally fewer surface morphs than canonical morphemes, which disadvantages sequence labelling approaches.

The deep-learning models developed herein are able to be used to analyse text end-to-end when coupled with canonical segmenters. To the best of our knowledge, cascading existing canonical segmenters into our models currently represents the state-of-the-art in morphological parsing on the full tagset. As and when improved canonical segmenters are developed, existing segmenters may be swapped out in favour of these new models to yield an overall improved system. This is an advantage over traditional, FST-based approaches in which segmentation and tagging are tightly coupled.

Our results show that deep-learning methods can effectively model computational linguistic tasks in Nguni languages. Future work could apply these methods to other morphological tasks, or use the models developed as additional features for models in downstream tasks. Additionally, with the concept proven, other avenues in neural morphological tagging could be explored, such as sequence-to-sequence models which jointly learn segmentation and tags, pre-trained language models for morphological tagging with better-aligned tokenisation schemes, and different architectures for models trained from scratch.

REFERENCES

[1] David Adelani, Graham Neubig, Sebastian Ruder, Shruti Rijhwani, Michael Beukman, Chester Palen-Michel, Constantine Lignos, Jesujoba Alabi, Shamsuddeen Muhammad, Peter Nabende, Cheikh M. Bamba Dione, Andiswa Bukula,

Rooweither Mabuya, Bonaventure F. P. Dossou, Blessing Sibanda, Happy Buzaaba, Jonathan Mukiibi, Godson Kalipe, Derguene Mbaye, Amelia Taylor, Fatoumata Kabore, Chris Chinenye Emezue, Anuoluwapo Aremu, Perez Ogayo, Catherine Gitau, Edwin Munkoh-Buabeng, Victoire Memdjokam Koagne, Allahsera Auguste Tapo, Tebogo Macucwa, Vukosi Marivate, Mboning Tchiaze Elvis, Tajuddeen Gwadabe, Tosin Adewumi, Orevaoghene Ahia, and Joyce Nakatumba-Nabende. 2022. MasakhaNER 2.0: Africa-centric Transfer Learning for Named Entity Recognition. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, Yoav Goldberg, Zornitsa Kozareva, and Yue Zhang (Eds.). Association for Computational Linguistics, Abu Dhabi, United Arab Emirates, 4488–4508. https://doi.org/10.18653/v1/2022.emnlp-main.298

[2] David Ifeoluwa Adelani, Jade Abbott, Graham Neubig, Daniel D'souza, Julia Kreutzer, Constantine Lignos, Chester Palen-Michel, Happy Buzaaba, Shruti Rijhwani, Sebastian Ruder, Stephen Mayhew, Israel Abebe Azime, Shamsuddeen Muhammad, Chris Chinenye Emezue, Joyce Nakatumba-Nabende, Perez Ogayo, Anuoluwapo Aremu, Catherine Gitau, Derguene Mbaye, Jesujoba Alabi, Seid Muhie Yimam, Tajuddeen Gwadabe, Ignatius Ezeani, Rubungo Andre Niyongabo, Jonathan Mukiibi, Verrah Otiende, Iroro Orife, Davis David, Samba Ngom, Tosin Adewumi, Paul Rayson, Mofetoluwa Adeyemi, Gerald Muriuki, Emmanuel Anebi, Chiamaka Chukwuneke, Nkiruka Odu, Eric Peter Wairagala, Samuel Oyerinde, Clemencia Siro, Tobius Saul Bateesa, Temilola Oloyede, Yvonne Wambui, Victor Akinode, Deborah Nabagereka, Maurice Katusiime, Ayodele Awokoya, Mouhamadane MBOUP, Dibora Gebreyohannes, Henok Tilaye, Kelechi Nwaike, Degaga Wolde, Abdoulaye Faye, Blessing Sibanda, Orevaoghene Ahia, Bonaventure F. P. Dossou, Kelechi Ogueji, Thierno Ibrahima DIOP, Abdoulaye Diallo, Adewale Akinfaderin, Tendai Marengereke, and Salomey Osei. 2021. MasakhaNER: Named Entity Recognition for African Languages. arXiv:2103.11811 [cs.CL]

[3] Ekin Akyürek, Erenay Dayanık, and Deniz Yuret. 2019. Morphological Analysis Using a Sequence Decoder. *Transactions of the Association for Computational Linguistics* 7 (2019), 567–579. https://doi.org/10.1162/tacl_a_00286

[4] Jesujoba O. Alabi, David Ifeoluwa Adelani, Marius Mosbach, and Dietrich Klakow. 2022. Adapting Pre-trained Language Models to African Languages via Multilingual Adaptive Fine-Tuning. In *Proceedings of the 29th International Conference on Computational Linguistics*, Nicoletta Calzolari, Chu-Ren Huang, Hansaem Kim, James Pustejovsky, Leo Wanner, Key-Sun Choi, Pum-Mo Ryu, Hsin-Hsi Chen, Lucia Donatelli, Heng Ji, Sadao Kurohashi, Patrizia Paggio, Nianwen Xue, Seokhwan Kim, Younggyun Hahm, Zhong He, Tony Kyungil Lee, Enrico Santus, Francis Bond, and Seung-Hoon Na (Eds.). International Committee on Computational Linguistics, Gyeongju, Republic of Korea, 4336–4349. https://aclanthology.org/2022.coling-1.382

[5] BeautifulSoup [n. d.]. Beautiful Soup: We called him Tortoise because he taught us. https://www.crummy.com/software/BeautifulSoup/

[6] Anders Björkelund, Özlem Çetinoğlu, Richárd Farkas, Thomas Mueller, and Wolfgang Seeker. 2013. (Re)ranking Meets Morphosyntax: State-of-the-art Results from the SPMRL 2013 Shared Task. In *Proceedings of the Fourth Workshop on Statistical Parsing of Morphologically-Rich Languages*, Yoav Goldberg, Yuval Marton, Ines Rehbein, and Yannick Versley (Eds.). Association for Computational Linguistics, Seattle, Washington, USA, 135–145. https://aclanthology.org/W13-4916

[7] Sonja Bosch, Laurette Pretorius, Kholisa Podile, and Axel Fleisch. 2008. Experimental Fast-Tracking of Morphological Analysers for Nguni Languages. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation (LREC'08)*, Nicoletta Calzolari, Khalid Choukri, Bente Maegaard, Joseph Mariani, Jan Odijk, Stelios Piperidis, and Daniel Tapias (Eds.). European Language Resources Association (ELRA), Marrakech, Morocco. http://www.lrec-conf.org/proceedings/lrec2008/pdf/643_paper.pdf

[8] Sonja E. Bosch and Laurette Pretorius. 2017. A Computational Approach to Zulu Verb Morphology within the Context of Lexical Semantics. *Lexikos* 27 (00 2017), 152 – 182. http://www.scielo.org.za/scielo.php?script=sci_arttext&pid=S2224-00392017000100007&nrm=iso

[9] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault (Eds.). Association for Computational Linguistics, Online, 8440–8451. https://doi.org/10.18653/v1/2020.acl-main.747

[10] Mathias Creutz and Krista Lagus. 2005. *Unsupervised morpheme segmentation and morphology induction from text corpora using Morfessor 1.0.* Helsinki University of Technology, Finland.

[11] Sajib Dasgupta and Vincent Ng. 2006. Unsupervised morphological parsing of Bengali. *Language Resources and Evaluation* 40, 3 (01 Dec 2006), 311–330. https://doi.org/10.1007/s10579-007-9031-y

[12] Google Developers. 2022. Precision and Recall. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall. Accessed: [19/08/2024].

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. arXiv:1810.04805 [cs.CL]

[14] Cheikh M. Bamba Dione, David Ifeoluwa Adelani, Peter Nabende, Jesujoba Alabi, Thapelo Sindane, Happy Buzaaba, Shamsuddeen Hassan Muhammad, Chris Chinenye Emezue, Perez Ogayo, Anuoluwapo Aremu, Catherine Gitau, Derguene Mbaye, Jonathan Mukiibi, Blessing Sibanda, Bonaventure F. P. Dossou, Andiswa Bukula, Rooweither Mabuya, Allahsera Auguste Tapo, Edwin Munkoh-Buabeng, Victoire Memdjokam Koagne, Fatoumata Ouoba Kabore, Amelia Taylor, Godson Kalipe, Tebogo Macucwa, Vukosi Marivate, Tajuddeen Gwadabe, Mboning Tchiaze Elvis, Ikechukwu Onyenwe, Gratien Atindogbe, Tolulope Adelani, Idris Akinade, Olanrewaju Samuel, Marien Nahimana, Théogène Musabeyezu, Emile Niyomutabazi, Ester Chimhenga, Kudzai Gotosa, Patrick Mizha, Apelete Agbolo, Seydou Traore, Chinedu Uchechukwu, Aliyu Yusuf, Muhammad Abdullahi, and Dietrich Klakow. 2023. MasakhaPOS: Part-of-Speech Tagging for Typologically Diverse African languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (Eds.). Association for Computational Linguistics, Toronto, Canada, 10883–10900. https://doi.org/10.18653/v1/2023.acl-long.609

[15] Jakobus S. du Toit and Martin J. Puttkammer. 2021. Developing Core Technologies for Resource-Scarce Nguni Languages. *Information* 12, 12 (2021). https://doi.org/10.3390/info12120520

[16] Roald Eiselen and Martin Puttkammer. 2014. Developing Text Resources for Ten South African Languages. In *Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, Nicoletta Calzolari, Khalid Choukri, Thierry Declerck, Hrafn Loftsson, Bente Maegaard, Joseph Mariani, Asuncion Moreno, Jan Odijk, and Stelios Piperidis (Eds.). European Language Resources Association (ELRA), Reykjavik, Iceland, 3698–3703. http://www.lrec-conf.org/proceedings/lrec2014/pdf/1151_Paper.pdf

[17] Tanja Gaustad and Cindy A. McKellar. 2024. Updated Morphologically Annotated Corpora for 9 South African Languages. *Journal of Open Humanities Data* 10, 1 (Jun 2024), 38. https://doi.org/10.5334/johd.211

[18] Tanja Gaustad and Martin J. Puttkammer. 2022. Linguistically annotated dataset for four official South African languages with a conjunctive orthography: IsiNdebele, isiXhosa, isiZulu, and Siswati. *Data in Brief* 41 (2022), 107994. https://doi.org/10.1016/j.dib.2022.107994

[19] Alex Graves and Jürgen Schmidhuber. 2005. Framewise phoneme classification with bidirectional LSTM and other neural network architectures. *Neural Networks* 18, 5 (2005), 602–610. https://doi.org/10.1016/j.neunet.2005.06.042 IJCNN 2005.

[20] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation* 9, 8 (11 1997), 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735 arXiv:https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf

[21] Matthieu Labeau, Kevin Löser, and Allauzen Alexandre. 2015. Non-lexical neural architecture for fine-grained POS tagging. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing* (2015). Association for Computational Linguistics, Lisbon, Portugal, 232–237.

[22] John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning (ICML '01)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 282–289. https://api.semanticscholar.org/CorpusID:219683473

[23] Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural Architectures for Named Entity Recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Kevin Knight, Ani Nenkova, and Owen Rambow (Eds.). Association for Computational Linguistics, San Diego, California, 260–270. https://doi.org/10.18653/v1/N16-1030

[24] Ling Liu. 2021. Computational Morphology with Neural Network Approaches. arXiv:2105.09404 [cs.CL] https://arxiv.org/abs/2105.09404

[25] Ilya Loshchilov and Frank Hutter. 2017. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*. https://api.semanticscholar.org/CorpusID:53592270

[26] Xuezhe Ma and Eduard Hovy. 2016. End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Katrin Erk and Noah A. Smith (Eds.). Association for Computational Linguistics, Berlin, Germany, 1064–1074. https://doi.org/10.18653/v1/P16-1101

[27] Francois Meyer, Haiyue Song, Abhisek Chakrabarty, Jan Buys, Raj Dabre, and Hideki Tanaka. 2024. NGLUEni: Benchmarking and Adapting Pretrained Language Models for Nguni Languages. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, Nicoletta Calzolari, Min-Yen Kan, Veronique Hoste, Alessandro Lenci, Sakriani Sakti, and Nianwen Xue (Eds.). ELRA and ICCL, Torino, Italia, 12247–12258. https://aclanthology.org/2024.lrec-main.1071

[28] Sthembiso Mkhwanazi and Laurette Marais. 2024. Generation of segmented isiZulu text. *Journal of the Digital Humanities Association of Southern Africa* 5, 1 (Feb. 2024). https://doi.org/10.55492/dhasa.v5i1.5034

[29] Tumi Moeng, Sheldon Reay, Aaron Daniels, and Jan Buys. 2021. Canonical and Surface Morphological Segmentation for Nguni Languages. arXiv:2104.00767 [cs.CL]

[30] Philipp Moritz, Robert Nishihara, Stephanie Wang, Alexey Tumanov, Richard Liaw, Eric Liang, Melih Elibol, Zongheng Yang, William Paul, Michael I. Jordan, and Ion Stoica. 2018. Ray: a distributed framework for emerging AI applications. In *Proceedings of the 13th USENIX Conference on Operating Systems Design and Implementation* (Carlsbad, CA, USA) *(OSDI'18)*. USENIX Association, USA, 561–577.

[31] Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient Higher-Order CRFs for Morphological Tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, David Yarowsky, Timothy Baldwin, Anna Korhonen, Karen Livescu, and Steven Bethard (Eds.). Association for Computational Linguistics, Seattle, Washington, USA, 322–332. https://aclanthology.org/D13-1032

[32] David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Lingvisticæ Investigationes* 30, 1 (2007), 3–26. https://doi.org/10.1075/li.30.1.03nad

[33] Franziska Pannach, Francois Meyer, Edgar Jembere, and Sibonelo Zamokuhle Dlamini. 2022. NLAPOST2021 1st Shared Task on Part-of-Speech Tagging for Nguni Languages. *Journal of the Digital Humanities Association of Southern Africa* 3, 01 (Feb. 2022). https://doi.org/10.55492/dhasa.v3i01.3865

[34] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. *PyTorch: an imperative style, high-performance deep learning library.* Curran Associates Inc., Red Hook, NY, USA.

[35] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.

[36] Laurette Pretorius and Sonja Bosch. 2012. Semi-automated extraction of morphological grammars for Nguni with special reference to Southern Ndebele. *Language Technology for Normalisation of Less-Resourced Languages* (2012), 73–78.

[37] L. Pretorius and S. Bosch. 2018. ZulMorph: Finite state morphological analyser for Zulu (Version 20190103. https://portal.sadilar.org/FiniteState/demo/zulmorph/ Software]. Web demo at.

[38] Martin Puttkammer and Jakobus Du Toit. 2021. Canonical Segmentation and Syntactic Morpheme Tagging of Four Resource- scarce Nguni Languages. *Journal of the Digital Humanities Association of Southern Africa (DHASA)* 3 (01 2021). https://doi.org/10.55492/dhasa.v3i03.3818

[39] 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*, Julia Hockenmaier and Sebastian Riedel (Eds.). Association for Computational Linguistics, Sofia, Bulgaria, 29–37. https://aclanthology.org/W13-3504

[40] Sebastian Spiegler, Andrew van der Spuy, and Peter A. Flach. 2010. Ukwabelana - An open-source morphological Zulu corpus. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Chu-Ren Huang and Dan Jurafsky (Eds.). Coling 2010 Organizing Committee, Beijing, China, 1020–1028. https://aclanthology.org/C10-1115

[41] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2* (Montreal, Canada) *(NIPS'14)*. MIT Press, Cambridge, MA, USA, 3104–3112.

[42] Elsabe Taljard and Sonja Bosch. 2006. A Comparison of Approaches to Word Class Tagging: Disjunctively vs. Conjunctively Written Bantu Languages. *Nordic Journal of African Studies* 15 (01 2006). https://doi.org/10.53228/njas.v15i4.37

[43] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2023. Attention Is All You Need. arXiv:1706.03762 [cs.CL]

[44] Ahmet Üstün, Murathan Kurfalı, and Burcu Can. 2018. Characters or morphemes: how to represent words?. In *Proceedings of the 3rd Workshop on Representation Learning for NLP*. Association for Computational Linguistics, 144––153. https://doi.org/10.18653/v1/w18-3019 In this paper, we investigate the effects of using subword information in representation learning. We argue that using syntactic subword units effects the quality of the word representations positively. We introduce a morpheme-based model and compare it against to word-based, character-based, and character n-gram level models. Our model takes a list of candidate segmentations of a word and learns the representation of the word based on different segmentations that are weighted by an attention mechanism. We performed experiments on Turkish as a morphologically rich language and English with a comparably poorer morphology. The results show that morpheme-based models are better at learning word representations of morphologically complex languages compared to character-based and character n-gram level models since the morphemes help to incorporate more syntactic knowledge in learning, that makes morpheme-based models better at syntactic tasks..

A MATERIALS

A.1 Software

The models were programmed in the Python programming language[4] (version 3.9.7) using the Torch machine learning framework (version 1.13.1+cu117) [34]. For all models, the `AdamW` optimiser was used [25]. To aid in the hyperparameter tuning process, the Ray library was used (version 2.24.0) [30]. The evaluation scripts were written using scikit-learn [35]. Supplementary scripts were also written in Python.

The script written to scrape the ZulMorph [37] demo was written in Python and made use of the BeautifulSoup library (version 4.12.3) [5].

The code developed by Moeng et al. [29] for their feature-based CRF segmenters and Transformer-based canonical segmenters was also used. The code is freely available on GitHub[5]. A script developed by Dr Jan Buys was also used to generate gold-standard surface segmentations. This script uses the minimum edit distance between the canonical segmentations and the surface form of the world to generate these gold-standard surface morphs.

A.2 Hardware

The models were hyperparameter tuned and trained on the NICIS CHPC[6] cluster's GPU nodes[7].

B ZULMORPH-TO-NCHLT MAPPING ALGORITHM

The remapping algorithm begins by concatenating all noun-class tags with their preceding syntactic tags (e.g., "[BPre][10]" becomes "[BPre10]"). After this step, the tags are normalized to the NCHLT tag scheme. The vast majority of ZulMorph [37] tags could simply be re-named. The mappings chosen are shown in Table 6. However, there are some special cases.

Firstly, whilst ZulMorph emits a noun-class tag for noun stems, the NCHLT tagset does not include this information, so it is dropped. Secondly, the NCHLT tagset does not include *hlonipha* (respectful language) information (ZulMorph's Hlon tag [37]), so this information is dropped as well.

Finally, the "VerbTermPerf" tag is mapped depending ont he morpheme that it is applied to. When applied to the morpheme "ile" (long-form perfect-tense verb ending), the tags "Perf" (perfect tense) and "VerbTerm" (verb-terminative) are emitted. When applied to the morpheme "e" (short-form perfect-tense verb ending), only the tag "VerbTerm" is emitted. This is because, whilst Gaustad and Puttkammer's [18] dataset parses "ile" into "il[Perf]-e[VerbTerm]" and "e" into "e[VerbTerm]", ZulMorph parses them into "ile[Perf][VerbTerm]" and "e[Perf][VerbTerm]" respectively. Regardless of this segmentation difference, the meaning and intent behind the "VerbTermPerf" tag is clear enough to make this mapping.

[4]https://www.python.org/
[5]https://github.com/DarkPr0digy/MORPH_SEGMENT
[6]https://www.chpc.ac.za/
[7]https://wiki.chpc.ac.za/chpc:gpu

Table 6: ZulMorph tags and their corresponding NCHLT tags

ZulMorph tags	NCHLT tag
PC	PossConc
VT, VTSubj, VTNeg	VerbTerm
QC	QuantConc
RC, RCPT	RelConc
AdjPre, AC	AdjPref
SCPT, SCHort, SCSit	SC
PotNeg	NegPre
EC	EnumConc
Item	Foreign
LongPres	Pres
Punct	Punc
ComplExt	IntensExt

C CANONICAL AND SURFACE SEGMENTERS

C.1 Comparison of approaches

The Macro and Micro F_1 accuracies for each segmentation approach are listed in Table 7. This helps to contextualise the end-to-end tagging results.

C.2 Re-evaluation of surface segmenters

Despite the Micro F_1 of the feature-based CRF surface segmenter developed by Moeng et al. [29] exceeding 0.98 when calculating using their methodology, the measured morpheme-level Micro F_1s were all between 0.57 and 0.66 (Table 8). This difference is most likely due to the measuring methodology. In Moeng et al.'s paper [29], the F_1 scores were measured for the task of morph-boundary classification. This means that, for a word such as "aliqela" to be correctly segmented as "a-li-qela" [18], the model would need to output something like "SBEBMME", where "B" identifies the start of a morph, "M" identifies a part of the current morpheme, "E" identifies the end of the current morpheme, and "S" identifies a single-letter morpheme. If the model were to output "BMEBMME" ("ali-qela"), it would obtain a Micro F_1 of 0.71 using this evaluation methodology. However, if the Micro F_1 is instead calculated on the aligned morpheme output sequences ("a-li-qela" and "ali-[PAD]-=qela"), then it would obtain a Micro F_1 of only 0.33. This could explain the sharp decrease in accuracy.

However, there is also a discrepancy between the boundary identification Micro-F_1 as measured using Moeng et al.'s [29] methodology and between our own methodology for measuring the performance of the same task (boundary identification). Moeng et al. [29] apply scikit-learn's [35] `MultiLabelBinarizer.fit_transform` function to both the list of all predicted and target sequences in the dataset. Our methodology simply flattens these sequences, appending each word's predicted and target sequences into the corresponding flattened lists. The same function, namely scikit-learn's [35] `f1_score`, is used to calculate the F_1 score itself in both cases.

A small experiment can be devised to test why there is such a discrepancy—calculating the boundary Micro F_1 score using Moeng et al.'s [29] methodology for the case of a word with expected boundaries "BES" and predicted boundaries "SBE". This results in a score of 1.0. This suggests that the `MultiLabelBinarizer.fit_transform` leads to an *order-independent* Micro F_1 score, rather than an order-dependent one. This is not as meangingful on the task of boundary identification, though.

D HYPERPARAMETER TUNING

Hyperparameters for the model were chosen through an iterative process. Models were tuned in batches of 12 hours. Before each hyperparameter tuning batch, two to four hyperparameters were chosen to be grid-searched. This meant that all combinations (time-permitting) of these parameters would be tested. The reason that only two to four parameters were grid-searched at a time was to prevent combinatorial explosion.

After each batch, the hyperparameters that had been grid-searched were roughly fixed to the values which maximised the Macro F_1 score. This meant that they would be randomly chosen as either the value, one step below it, or one step above it. The process was repeated (as many times as time permitted) until optimum parameters for each model configuration were obtained.

E MAXIMAL ALIGNMENT ALGORITHM

Algorithm 1 The maximal alignment algorithm

```
function ALIGN_SEQS($s$, $l$)
    $d \leftarrow$ LENGTH($l$) − LENGTH($s$)
    $m \leftarrow$ LENGTH($l$) − 1
    possible_indices $\leftarrow$ GENERATE_INDICES($d, m$)
    best_correct $\leftarrow 0$
    best_indices $\leftarrow$ []
    for all $i \in$ possible_indices do
        $p \leftarrow$ PAD_AT_INDICES($s, i$)
        $c \leftarrow$ COUNT_MATCHES($p, l$)
        if $c \geq$ best_correct then
            best_correct $\leftarrow c$
            best_indices $\leftarrow i$
        end if
    end for
    return PAD_AT_INDICES($s$, best_indices)
end function

function GENERATE_INDICES(length_diff, max_index)
    if length_diff = 0 then
        return []
    else
        indices $\leftarrow$ []
        for $f \leftarrow$ max_index to 0 do
            $i \leftarrow [f] +\!\!+$ GENERATE_INDICES(length_diff − 1, $f$)
            indices $\leftarrow$ indices $+\!\!+ [i]$
        end for
        return indices
    end if
end function
```

Table 7: Segmentation results for different approaches. All scores are maximally-aligned morpheme F_1 scores.

Model	IsiZulu		IsiNdebele		IsiXhosa		SiSwati	
	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1
Canonical segmentation models								
ZulMorph [37]	0.3604	0.6944	-	-	-	-	-	-
Moeng et al.'s [29] canonical transformers	0.5669	0.8612	0.5583	0.8617	0.7910	0.9563	0.5898	0.8811
Surface segmentation models								
Moeng et al.'s [29] feature-based CRFs	0.4842	0.8075	0.4298	0.7892	0.6236	0.8672	0.5115	0.8405

Table 8: Results for the re-evaluation of Moeng et al.'s [29] feature-based CRF surface segmenters.

Methodology	IsiZulu		IsiNdebele		IsiXhosa		SiSwati	
	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1	Macro F_1	Micro F_1
Our methodology, morpheme-level	0.4842	0.8075	0.4298	0.7892	0.6236	0.8672	0.5115	0.8405
Our methodology, boundary-level	0.9151	0.9223	0.8956	0.9102	0.9495	0.9533	0.9189	0.9291
Moeng et al.'s [29] methodology, boundary	0.9791	0.9835	0.9723	0.9801	0.9874	0.9896	0.9742	0.9813