Canonical and Surface Morphological Segmentation for Nguni Languages

Tumi Moeng Sheldon Reay Aaron Daniels Jan Buys
Department of Computer Science
University of Cape Town, South Africa
{MNGTUM007,RYXSHE002,DNLAAR001}@myuct.ac.za, jbuys@cs.uct.ac.za

Abstract

Morphological Segmentation involves decomposing words into morphemes, the smallest meaning-bearing units of language. This is an important NLP task for morphologically-rich agglutinative languages such as the Southern African Nguni language group. In this paper, we investigate supervised and unsupervised models for two variants of morphological segmentation: canonical and surface segmentation. We train sequence-to-sequence models for canonical segmentation, where the underlying morphemes may not be equal to the surface form of the word, and Conditional Random Fields (CRF) for surface segmentation. Transformers outperform LSTMs with attention on canonical segmentation, obtaining an average F1 score of 72.5% across 4 languages. Feature-based CRFs outperform bidirectional LSTM-CRFs to obtain an average of 97.1% F1 on surface segmentation. In the unsupervised setting, an entropy-based approach using a character-level LSTM language model fails to outperforms a Morfessor baseline, while on some of the languages neither approach perform much better than a random baseline. We hope that the high performance of the supervised segmentation models will help to facilitate the development of better NLP tools for Nguni languages.

1 Introduction

Morphological Segmentation is the task of separating of words into their composite *morphemes*, which are the smallest meaning-bearing units of a language (Creutz et al., 2007; Ruokolainen et al., 2013). This task is particularly important when applied to *agglutinative* languages, which have words that are composed of aggregating morphemes, generally without making significant alterations to the spelling of the morphemes. Obtaining these morphemes enables analysis that can be applied to further Natural Language Processing (NLP) tasks (Creutz et al., 2007). For example, breaking a word down to its composite morphemes before translation, or generating those morphemes one at a time, could lead to more accurate translation, especially in a low-resource scenario where limited training data is available. Morphological analysis could also be used in the development of tools that could benefit language learners and assist linguists researching these languages.

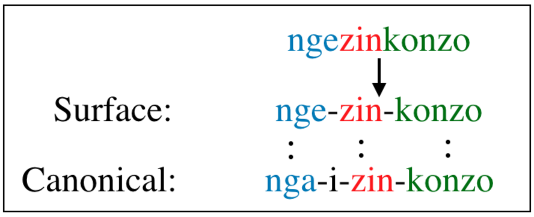

Figure 1: An isiZulu word with its canonical and surface segmentations.

In this paper, we develop models for morphological segmentation for the Nguni languages, a group of low-resource Southern African languages.[1] We train supervised and unsupervised models for isiNdebele, isiXhosa, isiZulu and siSwati, which are all official languages of South Africa. Morphological segmentation is particularly applicable to Nguni languages because they are agglutinative and written conjunctively (Taljard and Bosch, 2006; Bosch and Pretorius, 2017). The only previous work we are aware of based on the datasets we use is a rule-based approach (Eiselen and Puttkammer, 2014), which our supervised models outperform substantially. Cotterell et al. (2015) trained a semi-Markov CRF for isiZulu segmentation, but used a different corpus (Spiegler et al., 2010).

We investigate supervised sequence-to-sequence

[1] The code of our models is available at https://github.com/DarkPr0digy/MORPH_SEGMENT

models (Sutskever et al., 2014; Bahdanau et al., 2015; Vaswani et al., 2017) as well as Conditional Random Fields (CRFs; Lafferty et al., 2001; Lample et al., 2016; Ma and Hovy, 2016) for segmentation. For sequence-to-sequence models we interpret the process of transforming a word into its segmented form as a character-level sequence transduction problem, which has previously been shown to be effective when applied to other languages (Wang et al., 2016; Shao, 2017; Ruzsics and Samardžić, 2017). Sequence-to-sequence models are able to deal with input and output sequences of differing lengths, and subsequently to handle *canonical* segmentation, where a morpheme may not be equal to the segment of the word that it corresponds as written (Kann et al., 2016). The CRFs on the other hand are suitable for *surface* segmentation, where the morphemes are a pure segmentation of the orthography of the word. Figure 1 shows an example of a word with its canonical and surface segmentations.

Canonical segmentation results show that the bidirectional LSTM with attention outperforms the LSTM without attention, while the Transformer leads to the best performance on all languages, with an average F1 of 72.5%. For surface segmentation, the feature-based CRF obtains an average F1 score of 97.1% across the 4 languages, compared to 94.8% for the Bi-LSTM CRF.

We also implemented an unsupervised entropy-based approach to morphological surface segmentation, based on character-based LSTM language models in the forward and backward directions. We experimented with different entropy-based objective functions for segmentation, but none outperformed the Morfessor baseline. For some of the languages, neither approach perform much better than a random baseline.

2 Background: Morphological Segmentation

2.1 Nguni Languages

The Nguni language group consists of number of languages spoken in Southern Africa. isiZulu, isiXhosa, (Southern) isiNdebele and siSwati are all official languages of South Africa and constitute the majority of Nguni speakers (along with Zimbabwean Northern Ndebele). These languages are closely related to one another, with siSwati being a bit further apart from the rest, and Ndebele's vocabulary being influenced more by the neighbouring Sotho-Tswana languages. All of these languages can be described as low-resource (Spiegler et al., 2008; Bosch et al., 2008; Bosch and Pretorius, 2017). The Nguni languages are agglutinative and are written conjunctively, meaning that words can be made up of many morphemes written unseparated (Spiegler et al., 2008; Bosch et al., 2008).[2] The meaning of a word is a function of all its morphemes. Therefore extracting the morphemes is essential for syntactic analysis and various forms of further text processing of these languages.

Here is an example of agglutination in isiZulu (Bosch and Pretorius, 2017):

- -phind-a - "repeat"
- -phind-*is*-a - "cause to repeat"
- -phind-*el*-*el*-a - "repeat again and again"

2.2 Morphological Segmentation

We distinguish two ways in which a word *w* can be segmented, *surface segmentation* and *canonical segmentation* (Cotterell et al., 2016):

- **Surface segmentation**: *w* is segmented into a sequence of substrings, which when concatenated will result in *w*.
- **Canonical segmentation**: *w* is analyzed as a sequence of canonical morphemes representing the underlying forms of the morphemes, which may differ from their orthographic manifestation.

The canonical segments correspond to the underlying morphemes used by linguists, and may be more informative for downstream analysis than pure surface segmentation (Cotterell et al., 2016).

2.3 Data

The morphological annotations used in this paper come from the Annotated Text Corpora from the National Center for Human Technology (NCHLT; Eiselen and Puttkammer, 2014).[3] We use the isiNdebele, isiXhosa, isiZulu and siSwati corpora that are annotated with canonical morphological segmentations. The morphemes are also labelled with their grammatical functions, but in this paper we

[2] The Sotho-Tswana languages, the other major South African language group, are written *disjunctively*: morphemes are generally written as separate words, despite the languages being agglutinative.

[3] Datasets are available at `https://repo.sadilar.org/handle/20.500.12185/7`

Language	Train	Dev	Test
isiZulu	17 778	1 777	3 298
isiXhosa	16 879	1 688	3 004
isiNdebele	12 929	1 119	2 553
siSwati	13 278	1 080	1 347

Table 1: Sizes of the morphological segmentation datasets (number of words) after preprocessing.

will only consider the segmentation task and not the labelling task.

The original annotation for the isiZulu word *ngezinkonzo* is as follows, where the morpheme labels are given in square brackets:

```
[RelConc]-nga[NPre]-i[NPrePre]
  -zin[BPre]-konzo[NStem]
```

The data consists of annotated running text. We process the data to extract a set of annotated words. We exclude punctuation, numbers, and words that are unsegmented in the annotations (as many of them are actually unannotated or are loan words). The data is given with a training and test split. We ensure that there is no overlap between the training and test sets by removing all words appearing in both texts from the training data. This ensures that we are evaluating the ability of the models to generalize to unseen words. We split a development set from the training set in the same manner. The size of the processed datasets is given in Table 1.

2.4 Generating Surface Segmentations

We map the canonical segmentation annotations heuristically to corresponding surface segmentations. We first check if the de-segmented canonical form is the same as the orthographic word, in which case the canonical and surface segmentations are equivalent. Otherwise, we compute the Levenshtein distance minimal edit operations from the de-segmented canonical form to the orthographic word. The operations are constrained so that each character in the input word corresponds to a character in the de-segmented canonical form. For example, in Figure 1, the edit operations are to delete the single-character morpheme “i” and to replace the first “a” with “e”.

Finally, the sequence of edit operations is processed to align the canonical segments to the surface segments. This enables detecting the deletion of canonical segments, and mapping the morpheme boundaries in the canonical form to the orthography to create the surface segmentation.

We computed a number of statistics to determine the efficacy of the method for obtaining surface segmentations. On average over all four languages, 45% of the words’ canonical segmentations differ from their surface segmentations. Of all the edit operations, 38.83% of the operations were replacement operations, and the remaining 61.17% were deletion operations. Finally, of all the segments generated in the surface form, 60.26% of the segments are equal to the corresponding morpheme in the canonical form.

2.5 Evaluation

The segmentation models are evaluated using precision, recall and F1 score of the morphemes identified for each word, compared to those in the annotated segmentation. We follow Cotterell et al. (2016) in treating the segmentation as a set of morphemes for evaluation purposes and computing the micro-F1 over the test set. This contrast to the traditional approach to evaluating morphological segmentation with morpheme boundary identification accuracy. That method is not applicable to canonical segmentation, and we believe that basing the evaluation directly on morpheme identification is a better reflection of accuracy on this task.

3 Canonical Segmentation with Sequence-to-Sequence Models

We apply a Recurrent Neural Network (RNN) encoder-decoder model with attention (Bahdanau et al., 2015) as well as an encoder-decoder Transformer model (Vaswani et al., 2017) to the task of canonical morphological segmentation. The task is formulated as transducing the given word’s character sequence to an output character sequence consisting of the canonical form of the word together with the segment boundaries.

3.1 Bi-LSTM with Attention

In an RNN-based encoder-decoder (Sutskever et al., 2014) the encoder is an RNN which processes the input sequence x sequentially, updating the hidden state of the RNN after reading each input element. The decoder is another RNN which uses the final encoder hidden state as its initial hidden state to generate the output sequence. At each time step the previous output generated by the decoder is fed back into the RNN, which then produces the next output symbol, until a complete output sequence has been generated. We use RNNs with Long-Short-Term-Memory (LSTM) cells (Hochreiter and Schmidhuber, 1997) to avoid vanishing or exploding gradients.

The encoder can be extended to be bidirectional (Schuster and Paliwal, 1997), encoding the sequence using separate forward and backward LSTMs. For each input element x_j, the encoder produces hidden layers $\overrightarrow{h_j}$ and $\overleftarrow{h_j}$ using a Bi-Directional RNN. The concatenation of these, h_j, can be seen as a contextual representation of x_j.

To overcome limitations of using a single fixed context vector to encode the entire input sequence, an attention mechanism can be used, which enables the model to dynamically determine which parts of the input sequence to focus on as the sequence is traversed (Bahdanau et al., 2015). The attention mechanism computes an alignment score e_{ij} between the input at position j and the output at position i, using a feed-forward network a:

$$e_{ij} = a(s_{i-1}, h_j) \tag{1}$$

where s_{i-1} is the output from the previous decoder step and h_j is the encoder representation at time step j. The alignment scores are normalized with the softmax function, and the context vector c_i at decoder time step i is computed as a weighted average of the encoder representations h_j:

$$c_i = \sum_{j=1}^{T_x} \text{softmax}_j(e_{ij})h_j \tag{2}$$

The decoder is therefore using the attention mechanism to dynamically calculate a context vector representation of the input sequence rather than using the (fixed) final encoder hidden state.

3.2 Transformer

The Transformer is a sequence model which does not use recurrence and is instead based on the concept of self-attention (Vaswani et al., 2017). In the encoder-decoder framework, the transformer encoder uses multi-headed attention to calculate self-attention over the input sequence. The decoder uses two multi-headed attention blocks, over itself and over the encoder. Masking is used to prevent the first block from calculating self-attention over decoder positions which follow the current sequence position. A transformer block consists of alternating multi-headed self-attention and feed-forward layers. Self-attention can be calculated in parallel, increasing computational efficiency. Due to the lack of recurrence, positional embeddings are used to encode the order of sequence elements in the encoder and the decoder.

3.3 Experimental Setup

The BiLSTM with attention was implemented by adapting an existing PyTorch code-base to support character-level transduction and a bidirectional encoder. The Transformer was implemented in the same code base. As the model generates an output segmentation, it generates "-" to indicate a morpheme boundary. As a baseline we use an encoder-decoder without attention. Each model was trained on the dataset for each language independently.

Based on hyperparameter tuning, the batch size was set at 32 or 64 for both the Transformer and the BiLSTM+Attention models. The best learning rates were 0.0001 for the biLSTM with attention, and 0.0005 for the Transformer. The hidden dimension for both models was set at 256, and dropout was applied with a rate of 0.3. The biLSTM with attention used 2 layers. Adam (Kingma and Ba, 2017) was found to be a better optimizer than Stochastic Gradient Descent for both models.

In order to compare with previous work, we ran the NCHLT rule-based segmenters (Eiselen and Puttkammer, 2014), which used the same (original) training corpus for model development, on our test set.[4] The systems produce a set of multiple possible segmentations for some words; we compute an upper bound on performance by choosing the highest-scoring segmentation where multiple options are given.

3.4 Results

Table 2 shows the canonical segmentation performance of each model on each language. Eiselen and Puttkammer (2014) reported F1 scores of 82% to 85% for the NCHLT rule-based model on the Nguni languages. However, they do not fully explain their experimental setup and if or how they disambiguate between multiple system outputs. The rule-based systems perform substantially worse than any of our models, despite the fact that we are reporting an upper bound on its performance. The gap is narrower on precision than on recall. The rule-based system has higher precision

[4] Available at https://repo.sadilar.org/handle/20.500.12185/7/discover?filtertype=type&filter_relational_operator=equals&filter=Modules

Model	P	R	F1
Rule-based	60.26	43.97	50.72
isiZulu	63.21	47.95	54.42
isiXhosa	59.85	48.88	53.81
isiNdebele	65.81	43.29	52.22
siSwati	52.16	35.76	42.43
LSTM	64.78	56.92	60.59
isiZulu	68.72	60.25	64.2
isiXhosa	65.94	57.44	61.40
isiNdebele	61.90	53.89	57.61
siSwati	62.54	56.10	59.15
Bi-LSTM+Att	68.25	62.81	65.41
isiZulu	68.58	62.45	65.37
isiXhosa	70.06	62.86	66.26
isiNdebele	64.90	59.67	62.18
siSwati	69.45	66.25	67.82
Transformer	75.58	69.76	72.54
isiZulu	77.34	71.04	74.06
isiXhosa	75.76	68.36	71.87
isiNdebele	73.14	66.67	69.76
siSwati	76.07	72.96	74.48

Table 2: Canonical segmentation results. The Precision (P), Recall (R) and F1 Scores (F1) are given as percentages. The top line for each model reports the average over the 4 languages. For the rule-based systems we report the upper bound on accuracy among the possible segmentations produced.

Model	Output
Baseline	na-u-kuenza
Bi-LSTM+Attention	na-u-ku-enza
Transformer	na-u-ku-enz-a
Target	**na-u-ku-enz-a**

Table 3: Sample model outputs for canonical segmentation, compared to the target (reference) segmentation.

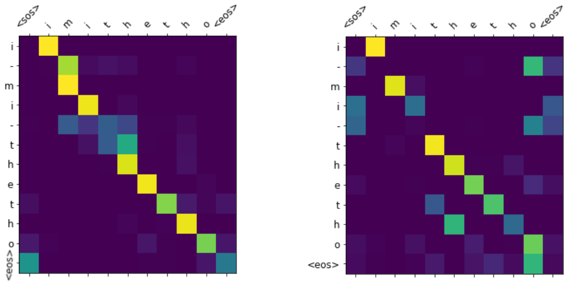

Figure 2: Attention distributions of two of the Transformer's attention heads.

than the LSTM-based model on isiNdebele, but everywhere else our models perform better. The rule-based systems produce canonical segmentations, but they produce segmentations matching the surface form for 92.7% of words in the test set (averaged over the 4 languages), while in the gold annotations only 73.6% have segmentations whose morphemes are equal to their surface forms.

The Bi-LSTM+Attention model performs better than the encoder-decoder LSTM for all four languages, with an average increase in F1 Score by 4.82%. siSwati, which had an average word length of 7 in the dataset (compared to 9 for the other three languages), achieved the highest increase out of all four languages with an increase of 8.67%.

The Transformer model showed the best results across all languages and metrics. It outperformed the baseline LSTM model by 11.95%, and the Bi-LSTM+Attention model by 7.13%. The F1 score for isiZulu improved by 9.7 percentage points from 65.37% to 74.06%. This shows the model's ability to learn features within the low resource language. The Transformer performed well within the low resource environment, and more data would likely improve its performance even further. The Transformer's performance was also the most consistent across languages: Its F1 scores have the lowest Standard Deviation among the three models. The Transformer results shows that relying solely on attention is a feasible approach for morphological segmentation. Table 3 shows sample outputs from all three models compared to the target output.

Figure 2 shows two of the 8 encoder-to-decoder attention heads of the Transformer model. The attention matrices show that the attention mechanism is able to align input segments to the correct corresponding output morphemes; this can be seen consistently across attention heads.

The results show that in a low-resource setting the models are able to learn and extract features from the languages at a satisfactory level. Nevertheless, the size of the datasets offers the best explanation of why the models didn't achieve higher accuracies. isiNdebele has the smallest training set, as well as the lowest performance across all the models. siSwati also has a smaller dataset, but the words are shorter and consequently there are less segments, which makes the problem simpler.

Consequently, the attention-based models' highest performance is on the siSwati dataset. In contrast, the rule-based model and the baseline LSTM perform relatively worse on siSwati.

4 Surface Segmentation with Conditional Random Fields

4.1 CRFs

Conditional Random Fields (CRFs) are a class of discriminative, globally-normalized, probabilistic sequence labelling models (Lafferty et al., 2001). We apply traditional feature-based (log-linear) CRFs as well as neural CRFs based on bidirectional LSTMs to the task of surface morphological segmentation. A CRF takes a sequence X as input and estimates the probability distribution $P(Y|X)$, where Y is a sequence of the same length as X. Every label y_i in Y corresponds to an input token x_i.

For morphological segmentation, the input is the word, represented as a character sequence, and the output is a label sequence encoding the segmentation. The possible labels are *B*, *M*, *E*, and *S*, representing, respectively, whether the character is the start of a new morpheme, a part of the current morpheme, the end of the current morpheme, or a single length morpheme.

A CRF computes a feature score $s(X, y_i, y_{i+1})$ for each position i. That is, it scores every possible assignment of a pair of labels in adjacent positions, so that the labels probabilities are correlated with each other and not independent. The total unnormalized score that the CRF assigns to a given label sequence Y is:

$$S\left(X, Y\right) = \sum_{i=0}^{n-1} s(X, y_i, y_{i+1}) \tag{3}$$

The probability can then be computed by normalizing the score over $Y^{|X|}$, the set of all possible label sequences of the same length as X:

$$p(Y|X) = \frac{e^{S(X,Y)}}{\sum_{\widetilde{Y} \in Y^{|X|}} e^{S(X,\widetilde{Y})}} \tag{4}$$

The summation needs to be computed efficiently during training: This can be done using dynamic programming with the forward algorithm. During testing, the model makes use of a related dynamic programming algorithm known as the Viterbi Algorithm to calculate the highest scoring label sequence for a given input. The run time of the dynamic program is quadratic in the number of possible labels, but here that is a small set.

4.2 Bi-LSTM CRFs

A traditional CRF is a log-linear model with a large number of manually-designed sparse, usually binary-valued, features. The model learns a weight for each feature. Alternatively, the scoring function can also be parameterized using neural networks, which can learn the features instead. We use a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to encode the input sequence. Our Bi-LSTM CRF follows that of previous work on neural sequence labelling (Lample et al., 2016; Ma and Hovy, 2016).

To implement the CRF scoring function, the Bi-LSTM output is represented as an n by k matrix P, where n is the number of words in the sequence and k is the number of labels in the label alphabet. Each element P_{ij} represents the (unnormalized) score of assigning label $y_i = j$ (as an index into the label vocabulary) to x_i. The CRF score is defined as

$$s\left(X, y_i, y_{i+1}\right) = P_{i,y_i} + A_{y_i,y_{i+1}}, \tag{5}$$

where A is a learned square matrix of dimension $L + 2$, where L is the number of labels in the label alphabet, and the 2 added labels represent the start and the end of the sequence (Lample et al., 2016).

4.3 Implementation

Our implementation of the feature-based CRF uses the `sklearn-crfsuite` library.[5] The features used are character n-grams with n in the range 0 to 6, whether the character is a vowel or a consonant, and whether the character is uppercase or lowercase. We adapt an existing Bi-LSTM-CRF implementation suited to the segmentation task.

For the feature-based CRF we tuned the choice of features, epsilon (which determines the convergence condition), and the number of training iterations. The best hyperparameters were an epsilon of 1e-7 and a maximum of 160 training iterations.

For the Bi-LSTM CRF we tuned the number of training epochs and the learning rate. The best models were trained for 20 epochs, with a learning rate of 9e-4 for isiXhosa and isiZulu, and 4e-4 for isiNedebele and siSwati.

	Feature-based CRF			Bi-LSTM CRF		
Language	**P**	**R**	**F1**	**P**	**R**	**F1**
isiZulu	97.88	96.82	97.35	96.64	96.64	96.64
isiXhosa	97.16	97.13	97.14	94.88	95.61	95.24
isiNdebele	97.94	96.62	97.27	96.59	96.21	96.40
siSwati	97.17	96.40	96.78	90.59	91.48	91.03
Average	97.54	96.74	97.14	94.68	94.99	94.83

Table 4: Results for the Surface Segmentation Task.

4.4 Results

Table 4 shows the results of the two CRF models on the task of surface segmentation. The feature-based CRF yielded very high performance in the surface segmentation task with an average F1 score of 97.13%. Surprisingly, the performance of the Bi-LSTM CRF is more than 3% lower, with an average F1 score of 93.81% across the four languages. The gap is substantially larger on SiSwati than on the other languages, with an F1 score of about 5% lower. One potential reason for the lower performance of the BiLSTM CRF is the small size of the dataset. In contrast to the sequence-to-sequence models, the performance drops on SiSwati rather than on isiNdebele, suggesting that it may be harder to tune the CRF on short sequences than the sequence-to-sequence models.

While not directly comparable, Cotterell et al. (2015) reported 90.16% F1 for surface segmentation of isiZulu using a semi-Markov CRF on the Ukwabelana corpus. However, in addition to using a different corpus, they performed semi-supervised training using only 1 000 annotated training examples (together with a larger unannotated corpus). For future work, we'd like to investigate further to what extend the performance gap between canonical and surface segmentation is due to the models compared to the greater inherent difficulty of the canonical segmentation task.

5 Unsupervised Segmentation

Unsupervised segmentation is important for low-resource languages as morphological annotations are unavailable for most of them. We use Morfessor (Creutz and Lagus, 2007), a widely-used model for unsupervised segmentation, to benchmark unsupervised segmentation on these datasets, following the same preprocessing and evaluation setup as for supervised surface segmentation. We use the Morfessor-Baseline model, which uses a segmentation optimization criteria based on Minimal Description Length. We compare this to a random segmentation baseline which inserts segment boundaries at random positions in a word, as a way to check whether the unsupervised models are learning anything useful. We also implemented an entropy-based model, following previous work on unsupervised segmentation of isiXhosa (Mzamo et al., 2019a,b).

5.1 Entropy-based Model

Shannon (1948) introduced the concept of entropy, which measures the amount of information produced by an event or process. The conditional entropy of x_i in a sequence $x_{1:i}$ for a given probability model $P(x|x_{1:i-1})$ can be defined as

$$-\sum_{\tilde{x}\in x^*} P(\tilde{x}|x_{1:i-1}) \log P(\tilde{x}|x_{1:i-1}) \qquad (6)$$

where x^* is the set of possible values of sequence element x.

The intuition behind entropy-based morphological segmentation is that inside a morpheme, each next character will be less surprising (so have a lower entropy) than the previous one, while at the start of a new morpheme the entropy will increase as the character is less predictable.

Previous work used smoothed character n-gram language models to estimate the entropy (Mzamo et al., 2019a,b). Here we use a character-level LSTM language model instead that encodes the entire word, learning to estimate the probabilities of successive characters. We trained language models in both forward (left-to-right) and backward directions. These models are used to obtain the left and right entropies of words, respectively. We trained 2-layer LSTMs with hidden state size of

[5] https://github.com/TeamHG-Memex/sklearn-crfsuite

Model	P	R	F1
Random	24.83	14.76	18.51
isiZulu	24.15	14.97	18.48
isiXhosa	23.23	14.32	17.72
isiNdebele	25.91	15.27	19.22
siSwati	26.03	14.49	18.62
Morfessor	28.06	27.57	27.69
isiZulu	20.37	23.19	21.69
isiXhosa	27.21	29.04	28.10
isiNdebele	20.60	21.36	20.97
siSwati	44.05	36.67	40.02
Constant Entropy	30.49	25.82	25.73
isiZulu	28.74	14.20	19.01
isiXhosa	22.79	15.41	18.38
isiNdebele	34.19	14.81	20.67
siSwati	36.22	58.85	44.85

Table 5: The Precision (P), Recall (R) and F1 scores for unsupervised morphological segmentation.

200, dropout rate 0.2, and SGD with an initial learning rate of 20 that is decreased during training.

We experimented with a number of different objective functions that use the entropies to decide where to segment a word. We found that the best strategy was to consider the sum of left and right entropies at each character position and to insert a segment boundary if the sum exceeds experimentally determined constants. We refer to this model as *Constant Entropy*. Constants 4, 3, 12 and 2.5 were used for isiNdebele, siSwati, isiXhosa and isiZulu, respectively.

We also experimented with inserting a segment boundary based on whether the entropy increases between adjacent positions, as well as an objective that compares the sum of the left and right entropies to the mean over all the entropies in the word to perform relatively thresholding. These are similar to objective functions proposed by Mzamo et al. (2019b). However, in our experiments the constant entropy objective performed substantially better than either of those approaches.

5.2 Results

The results for the unsupervised models are given in Table 5. Our entropy-based models do not outperform the Morfessor baseline. The entropy-based approach outperforms the random baseline by 7.22% on average, but was 1.96% lower than Morfessor.

The results suggest that there are substantial structural differences between the languages. In particular, both models perform substantially better on siSwati than on the other languages. This could partly be explained by siSwati words having less segments on average than the other languages. The entropy-based model outperforms Morfessor on siSwati, with a particularly high recall of 58.8%. Both models perform only slightly above the random baseline for isiNdebele and isiZulu. For isiXhosa, the entropy model has similarly low performance, while Morfessor does 10% better, indicating that it was able to learn more structure of the language than our model.

Mzamo et al. (2019b,a) also found that entropy-based models do not outperform Morfessor, although theirs were based on n-gram language models instead of LSTMs. They reported scores of up to 77% boundary identification accuracy. We obtained similar results when evaluating Morfessor using that metric, but on the morpheme-based F1 metric the results are much lower.

6 Conclusions

We developed supervised models for surface and canonical segmentation, as well as an unsupervised segmentation model, for 4 Nguni languages. Sequence-to-sequence models outperformed a rule-based baseline for canonical segmentation by a large margin, with Transformers obtaining the highest performance. The feature-based CRF obtained very high accuracies on the surface segmentation task, with an average F1 score of 97.1%, outperforming the Bi-LSTM CRF.

The strong supervised results opens new avenues to apply models for morphological segmentation to downstream language processing tasks for Nguni languages. For future work, performance could possibly be improved further through semi-supervised training, as sequence-to-sequence models usually benefit when trained on larger datasets. Multilingual training could also be investigated, due to the similarities between the languages.

The performance of all the unsupervised models are substantially lower than the supervised models. We hypothesize that substantially different models will be required to make progress in unsupervised morphological segmentation.

Acknowledgments

This work is based on research supported in part by the National Research Foundation of South Africa (Grant Number: 129850) and the South African Centre for High Performance Computing. We thank Zola Mahlaza for valuable feedback, and Francois Meyer for running an additional baseline.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *International Conference on Learning Representations*.

Sonja Bosch, Laurette Pretorius, Kholisa Podile, and Axel Fleisch. 2008. Experimental fast-tracking of morphological analysers for nguni languages. In *LREC 2008*.

Sonja E. Bosch and Laurette Pretorius. 2017. A Computational Approach to Zulu Verb Morphology within the Context of Lexical Semantics. *Lexikos*, 27:152 – 182.

Ryan Cotterell, Thomas Müller, Alexander Fraser, and Hinrich Schütze. 2015. Labeled morphological segmentation with semi-Markov models. In *Proceedings of the Nineteenth Conference on Computational Natural Language Learning*, pages 164–174, Beijing, China. Association for Computational Linguistics.

Ryan Cotterell, Tim Vieira, and Hinrich Schütze. 2016. A joint model of orthography and morphological segmentation. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 664–669, San Diego, California. Association for Computational Linguistics.

Mathias Creutz, Teemu Hirsimäki, Mikko Kurimo, Antti Puurula, Janne Pylkkönen, Vesa Siivola, Matti Varjokallio, Ebru Arisoy, Murat Saraçlar, and Andreas Stolcke. 2007. Morph-based speech recognition and modeling of out-of-vocabulary words across languages. *ACM Transactions on Speech and Language Processing (TSLP)*, 5(1):1–29.

Mathias Creutz and Krista Lagus. 2007. Unsupervised models for morpheme segmentation and morphology learning. *ACM Trans. Speech Lang. Process.*, 4(1).

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-2014)*, pages 3698–3703, Reykjavik, Iceland. European Languages Resources Association (ELRA).

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

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.

Diederik P. Kingma and Jimmy Ba. 2017. Adam: A method for stochastic optimization. *arXiv preprint 1412.6980*.

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, page 282–289, San Francisco, CA, USA. Morgan Kaufmann Publishers Inc.

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*, pages 260–270, San Diego, California. Association for Computational Linguistics.

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)*, pages 1064–1074, Berlin, Germany. Association for Computational Linguistics.

Lulamile Mzamo, Albert Helberg, and Sonja Bosch. 2019a. Evaluation of combined bi-directional branching entropy language models for morphological segmentation of isixhosa. In *South African Forum of Artificial Intelligence Research*, pages 77–89.

Lulamile Mzamo, Albert Helberg, and Sonja Bosch. 2019b. Towards an unsupervised morphological segmenter for isixhosa. In *SAUPEC/RobMech/PRASA*, pages 166–170.

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.

Tatyana Ruzsics and Tanja Samardžić. 2017. Neural sequence-to-sequence learning of internal word structure. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 184–194, Vancouver, Canada. Association for Computational Linguistics.

Mike Schuster and Kuldip K Paliwal. 1997. Bidirectional recurrent neural networks. *IEEE transactions on Signal Processing*, 45(11):2673–2681.

C. E. Shannon. 1948. A Mathematical Theory of Communication. *Bell System Technical Journal*, 27(3):379–423.

Yan Shao. 2017. Cross-lingual word segmentation and morpheme segmentation as sequence labelling. In *The First Workshop on Multi-Language Processing in a Globalising World*, pages 75–80. Dublin City University.

Sebastian Spiegler, Bruno Golénia, Ksenia Shalonova, Peter Flach, and Roger Tucker. 2008. Learning the morphology of zulu with different degrees of supervision. In *2008 IEEE Spoken Language Technology Workshop*, pages 9–12. IEEE.

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)*, pages 1020–1028, Beijing, China. Coling 2010 Organizing Committee.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in Neural Information Processing Systems*, pages 3104–3112.

Elsabé Taljard and Sonja E Bosch. 2006. A comparison of approaches to word class tagging: Disjunctively vs. conjunctively written bantu languages. *Nordic journal of African studies*, 15(4).

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, pages 5998–6008.

Linlin Wang, Zhu Cao, Yu Xia, and Gerard de Melo. 2016. Morphological segmentation with window lstm neural networks. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, AAAI'16, page 2842–2848. AAAI Press.