SwaT5 - A T5 based morpheme tokenizer for Swahili

Alessio Tosolini, Mike Violette, and Ben Kosa
University of Washington, Seattle
{tosolini, bkosa2, mviol42}@uw.edu

Machine Learning Project Summary

Project Scope

Swahili, a low resource language with high morphological complexity, is underrepresented in the NLP world in part due to data scarcity. We attempt to construct a synthetic data set for Swahili verb conjugations, and fine-tuning the T5 model for translating Swahili verbs into morphological tokens to test (1) how much data a transformer model needs to achieve SOTA accuracy and (2) whether a model can morphologically parse verbs whose roots it has never seen before.

Methodology

We fine-tuned the "T5-small" transformer provided by HuggingFace and ran it on a Google Colab file. Our pipeline looked as follows:
(1) We generate synthetic verb datasets of various size (1,000, 10,000, 100,000, 1,000,000) using a conjugator that takes in inflectional (e.g. person) and derivational parameters (e.g. causative) using a set of 500 verb roots.
(2) We train the T5-small transformer model to provide a morphologically parsed verb given the surface form of the verb (e.g. given the input *"ninakupikia"* 'I cook for you', the output should be *"ni na ku pik i a"*, where the string is a series of morphemes separated by spaces). For this, we modified the existing T5-small's code to account for our goals.
(3) We define accuracy as the number of correct morphological parses returned by the model out of 1,000 verbs on one of three datasets described below.

Training each model took at most 1 hour on Google Colab's GPUs for the 100,000 and 1,000,000, with the other two models taking less time. It took around 3 hours to train all of our models, including times we needed to restart because of an error we made.

Results

Our project was successful in returning state of the arts results in Swahili verbal morphological parsing, achieving a more than 10% increase on the previous best algorithm Swaregex [3] (our best model had just under a 90% accuracy on a testset of verbs it has never seen before). We found that transformers are good models for parsing verbs, and that synthetic data generation is a viable method for training models in situations where little natural data exists.

A limitation of a transformer model is that it performs significantly worse on parsing verbs whose roots it has never seen before, achieving around 40% accuracy. The broader implications of these two results is that transformer based parsing models can be very successful in parsing morphologically complex tokens, as long as it has trained on each morpheme.

What was Easy

The implementation of the model using a fine-tuned pre-trained model from HuggingFace was relatively easy, thanks to the availability of documentation provided by HuggingFace. Creating the datasets for the project, particularly for the Swahili verb conjugator, was also relatively easy and successful since Swahili verbal morphology is also well documented. Overall, the ease of implementation and dataset creation suggest that certain aspects of the project can be easily applied or replicated with reasonable effort.

What was Difficult

Evaluating the model's generalizability was challenging due to the absence of a suitable gold standard dataset for Swahili verbs, resulting in reliance on synthetic data for testing. Choosing an accurate metric for accuracy also posed difficulties, with uncertainties between exact match and partial match definitions. These challenges may render certain aspects of reproducing the project difficult.

1 Introduction

Historically, the vast majority of NLP focuses on English and on languages with relatively similar morphologies to English [2]. When the technology is designed to be English-specific, overrepresenting English is not a large problem. However, many NLP papers tend to claim language-independence with the technology they create, but only test on a subset of languages that do not truly represent the linguistic diversity of the world's languages. Moreover, languages with higher speaker populations tend to show lower morphological complexity, leaving morphologically complex languaegs underrepresented in the NLP field [4].

Morphologically complex languages pose a unique problem to modern NLP models due to word-level tokenization creating data scarcity problems. Consider a morphologically complex language like Swahili (ISO 639-3: swa, Bantu language family) which is widely considered to be the lingua franca of Eastern Africa. Although it is spoken by around 100 million people as a first or second langauge, LLMs are not yet able to handle the high degree of morphological complexity that these languages exhibit [6]. For example, Google Translate incorrectly translates Swahili *hawakupenda* as 'they didn't like you', when a better translation would be 'they didn't like it'. Data scarcity disproportionately affects such languages, especially for models which tokenises on the word level, since many grammatically valid inflections of words may not appear in the training set simply due to the exceedingly large number of permutations of morphemes.

In this paper, we take advantage of the relative grammatical regularity of Swahili to see how generating synthetic verb conjugation data for Swahili allowed for facilitated training of a tranformer model on a morphological segmentation task. The best performing morphological segmentation algorithm for Swahili is called SwaRegex and is rule-based. No morphological segmentation task has been performed with synthetic training data on a language as low-resource and morphologically complex as Swahili, and by testing whether such a model would be successful, we aim to see whether synthetic data can be used to train a modern model which required much data.

The verb conjugator, SwaRegex, and our data can be found at the following link: `https://github.com/mviol42/SwaBERT`.

2 Scope of the Project

To our knowledge, the current State of the Art (SOTA) method for parsing verbs in Swahili to morphemes is a heavily linguistically rule-based method introduced in 2022, called SwaRegex [3]. For many resource heavy and morphologically simpler languages (like English, Spanish, French, etc), SOTA methods have been making strong use of Transformer-based approaches [1], to great success. For low-resource languages like Swahili, these methods have not yet been utilized due to the large data requirements of Transformer-based architectures. This is why SwaRegex, along with the methods it builds on top of, have stayed rule-based.

In our project, we hope to remedy this by evaluating whether using synthetic data can allow modern Transformers-based morpheme parsers to achieve SOTA for parsing Swahili verbs into morphemes.

3 Methodology

3.1 Model

We fine-tuned the "T5-small" transformer provided by HuggingFace. We chose this model because we were looking for a relatively light-weight encoder-decoder architecture. We needed it to be light-weight because of the limited compute power and wanted encoder-decoder because this is a Seq2Seq task. We prompted the model with the phrase "Translate Swahili to morphological tokens: {`surface_form`}" as suggested by the tutorial. Furthermore, HuggingFace has very good documentation for how to implement fine-tuning on this model.

3.2 Hyperparameters

We trained the model using checkpoint and recommended hyperparameters from Hugging Face and as many epochs as our compute could handle in a reasonable amount of time (maxing out at 5 hours). We trained on datasets of sizes 1000, 10000, 100000, 1000000.

3.3 Data

Swahili is a low resource language, meaning there are few existing, annotated data sets for us to use. However, there exists an unannotated data set with >9 million Swahili words [5]. SwaRegex is a rule-based Swahili parser which

Table 1: Hyperparamters for Fine-tuning T5

Hyperparameters	Values
Number of Epochs	{50, 50, 30, 3}
Learning Rate	2e-5
Weight Decay	0.01
Per device train batch size	16

takes advantage of the grammatical regularity of Swahili verbs to parse Swahili verbs into morphemes with state of the art accuracy [3]. We have adapted SwaRegex from C# into Python so we can use SwaRegex to parse verbs from the unannotated data set. This lets us assemble a reasonably sized natural data set of over 2,000 morphologically complex Swahili verbs annotated with their morphemes. These verbs tended to be on the less complex side, so this dataset was used as a sanity check, since we can run the SwaRegex and T5 model on it to ensure that these models agree on an analysis for more morphologically transparent verbs. We can further take advantage of the grammatical regularity of Swahili verbs to generate synthetic data. Instead of scraping the web for data, we can directly generate the verbs given a set of grammatical rules. This means we can see exactly which morphemes are used to generate a verb.

The Swahili verb generator will take as parameters various aspects of the conjugation of the verb (e.g. 'root="penda", subject="1sg", tense="prs"') and output the morphological segmentation of the verb (e.g. ["ni", "na", "pend", "a"]) and the surface form of the verb (e.g. "ninapenda"). The generator is able to handle all major inflectional paradigms, but does not account for colloquialisms and dialect variation and always excludes optional morphemes. Given any regular root, the verb generator is able to return any paradigm. We will generate to generate an arbitrarily large synthetic dataset with the verb generator.

3.4 Experimental Setup

In brief, besides preliminary hyperparameter tuning, where we mostly used the default values for the model provided by HuggingFace, we tested the accuracy of the model on two parameters: (1) the amount of synthetic training data we fed into the model and (2) the testset that we used. We have three test datasets: one drawn from the same distribution as the training datasets, one drawn from the same distribution excluding the verbs that the model has already seen, and one drawn from a distribution with 50 verb roots that the model has never seen before. We tested each model (1k, 10k, 100k, 1m) on all three testsets to see how the accuracies can be a function of these parameters.

Our approach involved training the model to accurately provide a morphologically parsed representation of a given surface form of a verb. This entailed transforming the input sentence into its constituent morphemes, including prefixes, roots, and suffixes. The input is given as a string with no spaces (i.e. "ninakupenda") and the output is the same string with spaces delimiting morpheme boundaries (i.e. "ni na ku pend a").

To begin, we implemented a pipeline that encompassed various stages of our research. The pipeline consisted of three main steps. First, we developed a synthetic verb dataset generation process. Using a conjugator, we created datasets of different sizes (ranging from 1,000 to 1,000,000 examples) by incorporating inflectional parameters such as person and derivational parameters such as causative forms. Our synthetic dataset generation relied on a set of 500 verb roots, ensuring diversity and coverage of various verb forms.

Next, we proceeded to train the T5-small transformer model on our generated datasets. Adapting the existing code provided by HuggingFace, we made modifications to align the model's architecture and objectives with our specific task of morphological parsing. By fine-tuning the T5-small transformer, we aimed to optimize its performance in accurately predicting the morphological structure of the input verbs.

Throughout our experiments, we defined accuracy as the number of correct morphological parses returned by the model out of 1,000 randomly selected verbs from our evaluation datasets. These evaluation datasets were designed to assess the model's performance under different conditions. We carefully constructed three distinct datasets: one with verb forms that the model had been trained on (seen forms), another with verb forms containing unseen roots, and a third dataset that included verb forms with no repeated root forms. By evaluating the model's accuracy across these datasets, we could gain insights into its generalization capabilities and ability to handle novel verb forms.

4 Results/Summary

We evaluated on three different datasets. The first contained words that the model had seen before. Unsurpisingly, this had great accuracy, maxing out at 90% accuracy on our largest training suite. Then, we gave it words it had never

seen before, but no new components. This had a reasonable result, again maxing out around 90% accuracy. This result indicates some form of generalizability existing in our model; it can learn general patterns about morpheme parsing. Finally, we gave the model words with components it had never seen before. We did this by withholding about 50 roots from our training data set, then creating a test set using these roots.

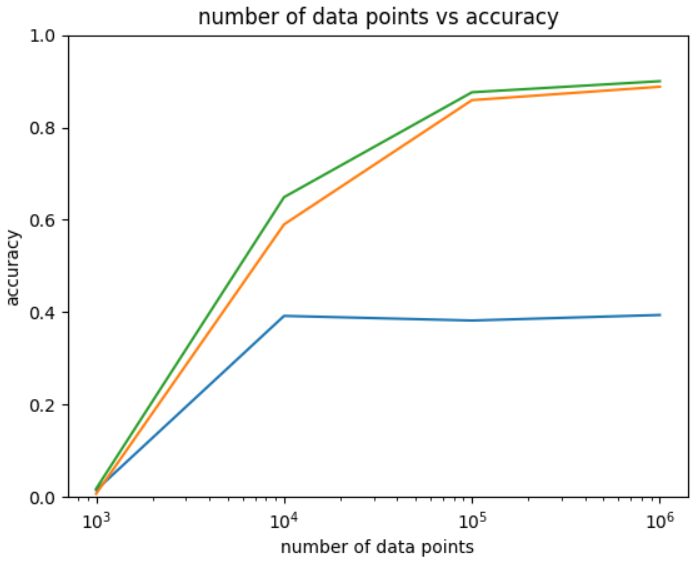
We got about 40% accuracy on these roots. While this isn't great, especially compared to the 90% accuracy from above, it is higher than the SOTA which gets 0% accuracy on roots it hasn't seen before based on how it processes. Further, we noticed the ones it tended to get wrong were correct, except the root which was over tokenized. For example, when examining the words our model got wrong here, we found the root was often split in half. Other than that, the surrounding morphemes tended to be correct, again indicating some generalizability of this approach.

In total, with a comprehensive list of roots, we could get highly accurate morpheme parser for Swahili verbs.

Table 2: Test Results

Test Set Size	**1k**	**10k**	**100k**	**1m**
Accuracy on Train Distribution	0.018	0.649	0.876	0.9
Accuracy on Unique Train Distribution	0.007	0.590	0.859	0.888
Accuracy on Unique Test Distribution	0.015	0.392	0.382	0.394

Chart 1: Test Results



Green line is Test on Train Distribution,
Orange line is Test on Unique Train Distribution,
Blue line is Test on Test Distribution

5 Discussion

The synthetic creation process was very successful in generating Swahili verb data for training the model. This allowed us to have a reasonable amount of labeled data to train the transformer model effectively. Additionally, the overall pipeline for the project worked well, including data preprocessing, training the transformer model, and evaluating the results. The pipeline we ended up using provided a clear and organized structure to follow, ensuring smooth progress throughout the project. Collaboration among team members was another positive aspect of our research. The diverse backgrounds and skill sets brought by each team member made idea sharing and problem-solving easier, and ultimately helped greatly the overall success of the project.

Although we encountered several challenges of different gravity, they ended up being learning experiences. One issue we faced was with Git when handling larger datasets. This resulted in difficulties with version control and efficient collaboration, and ended up being a large time sink the issues with rebasing and reverting to prior versions took many hours. Another problem arose from machine learning issues when working with larger datasets. As the dataset size increased, we experienced longer training times and resource constraints, which adversely affected the performance

of the model. Because of this, we had to reduce the number of epochs for the larger datasets, potentially skewing out results. Furthermore, our computational resources ran out faster than expected due to the resource-intensive nature of the training process and the Google Colab was having some trouble accessing all the data. This limited our ability to experiment with larger models or conduct extensive hyperparameter tuning. We underestimated how long resolving logistical errors would take, and we now know that we should make time for that moving forward.

Our analysis of the results provides valuable insights into the performance and capabilities of the transformer model for morphological parsing. Firstly, we observed that the transformer model generally accurately segmented verbs with roots that it had encountered during training. This demonstrates the model's ability to learn and generalize from known root forms. Unsurprisingly, we found that randomly sampling from the distribution during training yielded slightly higher accuracy compared to ensuring no overlap between the training and test sets. This suggests that a certain degree of overlap can help the model better generalize to unseen verb roots. We also noticed that the model's accuracy seemed to plateau between 100,000 and 1,000,000 datapoints, indicating diminishing returns beyond a certain dataset size. This may be a result of the 10-fold decrease in the number of epochs we trained for, and more research is needed to see if increasing the number of times each datapoint is seen will increase acuracy. Additionally, even a relatively small increase of 9,000 datapoints between the 1,000 and 10,000 datasets resulted in a significant improvement in accuracy for the test set with no repeated root forms.

Our research establishes two main theoretical results. Firstly, we have demonstrated that a Transformer model outperforms the current state-of-the-art approach, which relies on regular expressions to morphologically parse complex verbs. This finding highlights the potential of Transformer models in improving parsing accuracy for low-resource morphologically complex languages. Secondly, our results show that the model exhibits reasonable generalization abilities when encountering verb roots it has not seen before. By identifying patterns in the morphemes preceding and following the root, the model can effectively infer the root form. These theoretical results lay the groundwork for further research and improvements in morphological parsing.

While we made progress towards our research goals, there are a few gaps and limitations that we identified which may limit the credibility of our results. One limitation was the lack of review by a native Swahili speaker for the verb generator that created the synthetic Swahili verb data. This could potentially affect the quality and accuracy of the dataset. Additionally, due to resource limitations, we were unable to allow the training process to fully converge. Allocating more computational resources would have been beneficial to achieve even better results. Furthermore, we did not spend much time on hyperparameter tuning, instead mostly opting for the T5's default hyperparameters. Hyperparameter tuning for long may have potentially improved the model's performance.

Our findings indicate promising directions for future computational morphology and low-resource NLP research. We observed that the model achieved a notable parsing accuracy of 40% even without prior exposure to the verb root form. This suggests that further improvements can be made to enhance the model's ability to parse morphologically complex verbs. The exact way that these improvements can be executed is still an open question for our project. Additionally, our research establishes the proposed Transformer model as the current state-of-the-art approach for morphological parsing, surpassing existing methods. This opens up possibilities for exploring and expanding the application of Transformer models in other linguistic tasks.

Our empirical findings align with established machine learning knowledge. The model's performance improved with larger amounts of training data, consistent with the well-known principle of "more data, better performance". Additionally, testing the model on verbs that were potentially included in the training set naturally led to higher accuracy, demonstrating the model's ability to learn and recognize previously encountered patterns.

5.1 What was Easy

We spent a lot of time deliberating on how to implement the model. We went back and forth on training our own versus finding someone else's model online. In the end, we did a middle ground and fine-tuned a pre-trained model from HuggingFace. It was surprisingly easy to implement as HuggingFace provides documentation for fine-tuning a translation model. That being said, once we settled on T5, we didn't experiment with other models due to time constraints. It would be interesting to see a comparison of this model versus one based on other models, like DeepSPIN3.

Although time consuming, it was also realtively easy to create the datasets. One of our goals with this project was to show that a field linguist with a good understanding of the inflectional and derivational paradigms of a language would be able to train a model on a language with few resources by creating their own data, and the ease by which we were able to create a Swahili verb conjugator given grammatical parameters made the synthetic data generation portion of this project a success.

5.2 What was Difficult

It is difficult to evaluate how general our model is. While we did some generalization experiments, we don't have a good gold standard dataset of naturally generated Swahili verbs annotated with their morphological tokens. Instead, we had to do testing on our synthetic dataset. So while we can test our model on the verbs we created, we don't know how it would perform in the wild. From this, we learned the importance of having a good test set. When we began the project, we overlooked how much it would impact the significance of our results, but we now understand that although generating synthetic training data is interesting and valid, testing on the same synthetic distribution yields unconvincing results.

It is also difficult to measure accuracy. We had some debate surrounding whether an exact match approach or partial match approach would be best (i.e. seeing if the whole tokenized output is correct or if just most of it is correct). Ultimately, we decided on exact match as this is the most applicable use-case vs. partial match - someone using this model needs to know their parsing is correct. From this, we now understand the importance of choosing a good accuracy metric before beginning to work on the project. We had a good idea in mind, but we didn't have any of the details, our difficulties with measuring accuracy would've been rendered null if we had properly planned the pipeline before starting.

5.3 Recommendations for Future Work

We would tell other groups not to overthink the model. We spent a lot of our time early on trying to figure out which model to use and how to tokenize the words. We researched into various existing morpheme tokenizers, such as DeepSPIN, but ultimately ended up going with a prebuilt model which worked very well. The simplicity of implementing T5 saved the project as we didn't have to learn any new tools. From this, we learned the importance of working off of existing code bases. There are so many well documented models out there that it wasn't worth it to implement a new one for scratch when we could just tune one someone else made. We still were able to get a lot out of working through minor hyperparameter tuning, creating our own evaluation function, etc., but learning how to use an existing model was definitely a skill we will keep and expand on in the future.

Another major limitation in demonstrating that our results are valid is that we didn't have a gold standard annotated test dataset. If another group were to repeat this project, I would suggest they pick a language with at least a few hundred annotated examples. This would allow them to have more conclusive results, even if it means spending more time finding useable data.

References

[1] Khuyagbaatar Batsuren, Gábor Bella, Aryaman Arora, Viktor Martinovic, Kyle Gorman, Zdeněk Žabokrtský, Amarsanaa Ganbold, Šárka Dohnalová, Magda Ševčíková, Kateřina Pelegrinová, Fausto Giunchiglia, Ryan Cotterell, and Ekaterina Vylomova. The SIGMORPHON 2022 shared task on morpheme segmentation. In *Proceedings of the 19th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*, pages 103–116, Seattle, Washington, July 2022. Association for Computational Linguistics.

[2] Emily M. Bender. Linguistically naïve != language independent: Why NLP needs linguistic typology. In *Proceedings of the EACL 2009 Workshop on the Interaction between Linguistics and Computational Linguistics: Virtuous, Vicious or Vacuous?*, pages 26–32, Athens, Greece, March 2009. Association for Computational Linguistics.

[3] Mutwiri George Muthee, Mutua Makau, and Omamo Amos. Swaregex: a lexical transducer for the morphological segmentation of swahili verbs. *African Journal of Science, Technology and Social Sciences*, 1(2):77–84, Dec. 2022.

[4] Daniel Nettle. Social scale and structural complexity in human languages. 2012.

[5] Shivachi Casper Shikali and Mokhosi Refuoe. Language modeling data for swahili. Zenodo, November 2019.

[6] SIL International. Ethnologue: Languages of the World, 2021.