Project 1 - Segmentation Tokenization

Jingyuan He
Carnegie Mellon University
jingyuah@cs.cmu.edu

Abstract

This report describes the different methods I used towards segmentation-based tokenization of the two languages, Shipibo-Konibo and Rarámuri. The frontier is an indigenous language of Northern Mexico whose ground truth tokenization consists of only segmentation, while the latter, a Panoan language of the Peruvian Amazon, has a tokenization that involves both segmentation and transformation. The performance of these methods are evaluated against the two unsupervised baselines: the EM-based Unigram model and Morfessor, which segments of words from a more linguistic perspective.

1 Rule-Based Segmentation

The rule-based segmentation of the words in the two languages are constructed based on my observation over the training set. It is easy to observe that both of the languages segment some parts at the end of the word. For instance, "ai" in shp and "ma" in tar as illustrated in Table 1. The prevalent segmentations are easy to capture, like the 66 such "ai" segmentation in shp and 73 such "ma" segmentation in tar. Therefore, I collected these easily-observed patterns and check to segment these affixes for each of the words in the validation/test set. Additionally, I realize that such segmentation patterns not only appear at the end of words. For instance, the segmented affix "ti" could appear at the end of words like in "murubé n ti", which is the tokenization of "murubénti". It could also appear in the middle of a word, like "opéchrtima" and its tokenization "opéch r ti ma". Therefore, I iteratively check from the longest pattern towards the shortest pattern at the back of the unsegmented portion of the word and segment if the pattern fits, until reaching the beginning of the word. This algorithm is described in Algorithm 1.

Language	Word	Segmentation
Shipibo-Konibo [shp]	bokasai	boti kas ai
Rarámuri [tar]	rarárima	rará ri ma

Table 1: Segmentation Examples

Algorithm 1 Iterative Rule-Based Segmentation

```
Input: word w, i-character patterns p_i
idx = len(w)
while idx < len(w) do
    for i in 5, 4, ..., 1 do
        if w[idx − i : idx] fit p_i then
            segment p_i
            idx -= i
            break
        end if
    end for
    if no pattern matches: then
        idx -= 4
    end if
```

2 Character-Level Supervised Learning

Even though rule-based approaches are accurate, it takes quite some time to explore the golden rule. In addition, it is really hard to develop rules to handle languages like shp, which involve not only segmentation, but also transformations in affixes in its tokenization.

Therefore, I tried character-level supervised learning by fine-tuning the $byt5 - small$ model on the training sets of the two languages. The $byt5 - small$ is a token-free model that map raw text to predictions directly, generalizing across terms by migrating sparse word representations into the dense model layers while saving the number of parameters for possible large vocabularies (Xue et al., 2022). The fine-tuning specifics are recorded in Table 2.

Hyper-Parameter	Value
Learning Rate	1e-4
Weight Decay	1e-5
Optimizer	Adam
Learning Rate Scheduler	cosine
Warm Up Portion	0.1
Batch Size	4

Table 2: Fine-Tuning Hyper-Parameters

3 Combination of Rule-Based and Supervised Approach

The third experiment was done by transforming the data with the algorithms described in 1 and then feeding these data into the $byt5 - small$ model for fine-tuning. This allow the model to build on the existing rules, which ensures the precision aspect of our model by the golden rules observed while gaining improvement on the recall by learning more subword transformation through supervised learning. The selection of rule-based algorithms is based on their performance: shp uses the last pattern detection algorithm, whereas tar uses the iterative algorithms as in Algorithm 1 as a pre-processing step prior to fine-tuning.

4 Performance and Analysis

As we could observe in Table 5, the rule-based methods have high precision because most of the rules are golden (according to my observation) to be applied on the tokenization of most words. However, rule-based methods tend to have lower recall, since my algorithm makes no segmentation to the original word if the given window capture no patterns. Therefore, many of the comparably rare segmentation rules or transformation, in shp, would be omitted, which results in the low recall rate.

The fine-tuning approaches claims much higher recall, meaning that the model prone to segment or transform a word compared to the rule-based models, and thus, make more matches towards the ground truth tokenization.

For tar, the fine-tuned model gave the best performance, meaning that its 83% of its segmentations are correct, while 73% of the ground truth segmentations are captured in the model prediction. In contrast to my expectation, the combination of the rule-base approach and the fine-tuning approach didn't work as good as the fine-tuning approach. One reason might be that I developed a considerable set of rules for tar, in which some of these rules might not always hold in some specific words. Therefore, forcing the model to further segment on top of the falsely segmented words lead to lower performance.

Experiment	F1	Precision	Recall
Rule-Based (Last Pattern)	0.4356	0.5287	0.3705
Rule-Based (Iterative)	0.3554	0.3197	0.4
ByT5 Fine-Tuned	0.4490	0.5238	0.3929
Combined	**0.5217**	**0.6108**	**0.4554**
Unigram (Baseline)	0.3069	0.2442	0.4130
Morfessor (Baseline)	0.3259	0.2996	0.3571

Table 3: Shipibo-Konibo [shp]

Experiment	F1	Precision	Recall
Rule-Based (Last Pattern)	0.4557	0.5625	0.3829
Rule-Based (Iterative)	0.5449	0.5987	0.5
ByT5 Fine-Tuned	**0.7797**	**0.8313**	**0.7340**
Combined	0.6089	0.6411	0.5798
Unigram (Baseline)	0.3182	0.2601	0.4096
Morfessor (Baseline)	0.3981	0.3524	0.4574

Table 4: Rarámuri [tar]

Table 5: Experiment Evaluations on Validation Set

The performance of the different approaches on shp are not consistent with their perfomance on tar. The iterative algorithm received a downgraded performance compared to merely segment for the one pattern from the back of the word, if exists. A possible reason for this is that the segmentation for shp is more complex than tar, claiming for more delicate rules in the middle of the words, rather than simply matching patterns. Another factor is that I observe less rules for shp compared to tar, which might hinder the performance of the rule-based method, especially the complex iterative approach. Furthermore, while the combined approach on tar is non-optimal, the combined approach on shp gave the best performance in all my experiments. Both recall and precsion are improved from the fine-tuned method and the rule-based method, meaning that the combined approach did balance between the golden precision-oriented segmentation rules and the recall-orientated segmentation or transformation based on the data distribution learnt from the training set.

Finally, the best approaches for each of the languages are selected as the final model to tokenize the test sets.

References

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Transactions of the Association for Computational Linguistics*, 10:291–306.