Predictive text for agglutinative and polysynthetic languages

Anonymous ACL submission

Abstract

This paper presents a set of experiments in the area of morphological modelling and predictioning. We examine the tasks of segmentation and predictive text entry for two under-resourced and indigenous languages, K'iche' and Chukchi. We use different segmentation methods to make datasets for language modelling and then train models of different types: single-way segmented, which are trained using data from one segmentor; two-way segmented, which are trained using concatenated data from two segmentors; and finetuned, which are trained on two datasets from different segmentors. We measure word and character level perplexities of the language models and find that single-way segmented models trained using morphologically segmented data and finetuned models work the best. Finally, we test the language models on the task of predictive text entry using gold standard data and measure the average number of clicks per character and keystroke savings rate. We find that the models trained using morphologically segmented data work better, although with substantial room for improvement. At last, we propose the usage of morphological segmentation in order to improve the end-user experience while using predictive text and we plan on testing this assumption by training other models and experimenting on more languages.

1 Introduction

Nowadays text prediction is widely used in different cases such as autocomplete, smart keyboards, etc. The underlying models are limited by resources, so they save only the top-N highest frequency words, which may work well with analytic languages, but when it comes to the synthetic languages the out-of-vocabulary (OOV) problem becomes more and more noticeable. In order to deal with it, words are usually segmented in constituent parts, so that more of them can be saved in the model vocabulary.

The segmentation task is not new, there are many algorithms with BPE (Gage, 1994) being the most known and used for segmentation. Such methods do not lean on linguistics but only on statistics. In this paper, we tested whether morphological segmentation can improve language modelling and whether it can compete against statistical segmentation methods in predictive text entry task.

We have a particular interest in developing text prediction that is both effective and *ergonomic*. By ergonomic we mean that made predictions should be linguistically sound and intelligible for the end user. For example, imagine an English word *antidisestablishmentarianism*. An ergonomic segmentation would split the word into its constituent morphs [anti, dis, establish, ment, arian, ism], or an alternative [anti, dis, establishment, arianism]. An unergonomic segmentation might be [antid, isestab, lishme, ntarianism] or [an, tidises, tablishm, entarianism]. One of the issues with many current methods is that while they can produce segments that are meaningful units, in many cases the segments are not linguistically meaningful. We argue that for the task of predictive text entry producing non-linguistic units creates more cognitive load and so would result in slower text entry than predicting the same amount (or a greater number of) linguistic units.

The remainder of the paper is laid out as follows: in Section 2 we overview the languages we experiment on, in Section 3 we discuss the works that were an inspiration for this paper, in Section 4 we describe the experiments we are doing, in Section 5 we review used segmentation methods, choose the best morphological segmentor and do the segmentation, in Section 6 we provide results of language modelling, in Section 7 we speak about language modelling evaluation task, in Section 8 we discuss our thoughts on the results, in Section 9 we announce the planned future experiments. Examples in this paper will be mostly given in K'iche' and English, but there will also be a couple of examples in Chukchi and Turkish. English examples, while English being neither an agglutinative or polysynthetic language, are given in order for the reader to better understand the examples.

2 Languages

We performed the experiments using two languages: K'iche' (ISO-639: `quc`), a Mayan language of Guatemala that is of the agglutinating type, and Chukchi (ISO-639: `ckt`), a Chukotko-Kamchatkan language of Siberia of the polysynthetic type. Both of these types are characterised by words consisting of a large number of individual morphs, surface representations of morphemes.

The following examples in K'iche' (1) and Chukchi (2) demonstrate this tendency.[1]

(1) X-in-e'-ki-k'am-a'
CP-B1SG-MOV-A3PL-receive-DEP
'They went to take me'

Both languages exhibit polypersonal agreement (both the subject and object arguments of transitive verbs are encoded on the verb), and Chukchi, in addition, exhibits noun incorporation. As it can be seen in example 2, the object *манэ* /mane/ 'money' is incorporated, rendering intransitive the transitive root *ванԓя* /wanɬa/ 'ask'.

(2)

	Нэмыӄэй	ны-манэ-ванԓя-сӄэв-ӄэна-т.
	nemǝqej	nǝ-mane-wanɬa-sqew-qena-t
	also	ST-money-ask-MCP-ST.3SG-PL

'They also came to ask for money'

Languages of these types are widespread across the Americas but infrequent in Europe and, as a result, were less researched in terms of predictive text input.

2.1 Data

As K'iche' and Chukchi are low-resource languages, the availability of large corpora is limited. We used data annotated for morphological segments and unannotated text as well. For Chukchi, the annotated data came from the ChukLang[2] corpus, we used a version that was extracted and converted to Cyrillic orthography to make it compatible with the unannotated corpus. The unannotated data came from a collection of folklore and texts from the internet.

[1]Glossing symbols are from the original sources: CP 'completive', B1SG 'absolutive 1st person singular', MOV 'movement prefix', A3PL 'ergative 3rd person plural', DEP 'dependent status suffix, ST 'stative', MCP 'goal-oriented movement', ST.3SG '3rd person singular stative', PL 'plural'.

[2]https://chuklang.ru/

	Unannotated		Annotated	
	Sents	Words	Sents	Words
K'iche'	24,254	275,265	1,299	8,789
Chukchi	33,322	151,585	1,006	4,417

Table 1: Dataset sizes for the two languages measured in sentences and words. Unannotated and annotated datasets do not intersect. Annotation was done manually.

For K'iche' we also used annotated and unannotated texts. The annotated texts were a hand-segmented set of sentences used in constructing a morphologically and syntactically annotated corpus of K'iche', these sentences were from a range of sources including grammar-book and dictionary examples, stories and legal texts.

The second, unannotated, portion of the data was obtained from the *An Crúbadán* project (Scannell, 2007) that collects corpora from the web for indigenous and marginalised languages.

Table 1 shows the amount of data available for both languages.

2.2 Preprocessing

In order to segment the raw data using supervised learning methods, the annotated data was split into two disjoint subsets: train (50 percent) and test (50 percent). This ratio was chosen due to low annotated data volume – we suppose that a choice of a disbalanced ratio like 80 percent/20 percent can lead to unreliable results. The automatically segmented corpus was then used for language modelling, while the test split of annotated data was used for predictive text.

3 Related work

Being one of the latest works (Schwartz et al., 2020) on language modelling of indigenous languages, this paper proposed the usage of morphological segmentation in order to improve metrics of language modelling. They compared different segmentation methods, such as single words, dividing into characters, BPE, Morfessor, Finite-state transducers (FST). Even though FST is a good segmentation method used for lots of languages (Mittal, 2010; Hlaing and Mikami, 2014) and there are even ones for K'iche' (Richardson and Tyers, 2021) and Chukchi (Andriyanets and Tyers, 2018), we decided that we will not use them because the coverage for Chukchi is too low and it is hard to do disambiguation with FST because it requires a huge tagged corpus. Unfortunately, the authors could not do the end-task evaluation of the trained models but suggested doing predictive text. While also having no access to native speakers we decided to emulate the user input in order to evaluate the models. It still is not as good as end-user testing though it is better than nothing.

Another work (Boudreau et al., 2020) that gave us ideas on how to approach the language modelling task was devoted to Mi'kmaq language modelling evaluation. Mi'kmaq (ISO-639: `mic`), an Eastern Algonquian low-resourse polysynthetic language, is spoken primarily in Eastern Canada and has around 8700 speakers. Not only did the authors work with indigenous language, but they also did the keystroke savings evaluation, which is pretty similar to the idea of predictive text evaluation described in the previous work.

There are other works (Suhartono. et al., 2014; Yu et al., 2017) that describe keystroke savings evaluation. What is more important, the authors worked with agglutinative languages, Bahasa(ISO-639, `ind`), the official language of Indonesia, and Korean(ISO-639, `kor`), official and national language of both North Korea and South Korea (originally Korea). Though we do not want to use the same language modelling technics as were described in the papers, we still find it inspiring there are works dedicated to this task.

As we mentioned before, we assume that the usage of morphs while doing text prediction will make it both effective and ergonomic. However, there was a research (Lane and Bird, 2020) on Kunwinjku, a polysynthetic language of northern Australia, and Turkish, which states that morph-based autocomplete for polysynthetic languages can be troublesome due to long words and sparse vocabularies of such languages. Moreover, dialectal variations and dealing with input errors using edit distance makes the next-morpheme predictioning even harder, so, as it is shown in the paper, Turkish may be a more attractive language for morph-based predictioning than Kunwinjku

4 Tasks

As mentioned previously, our experiments are split into four distinct tasks, from the more fundamental to the more application-specific. In the following sections we describe the methodology for these tasks and the results obtained.

Morphological segmentation We train morphological segmentation models based on the corpora outlined in Table 1 and evaluate them. Here to be a correct segmentation models must match the reference sentence in the test set. The evaluation measure is F_1 score. F_1 score is defined using precision and recall:

$$\text{precision} = \frac{\text{correct boundaries found}}{\text{total boundaries found}} \quad (1)$$

$$\text{recall} = \frac{\text{number of found}}{\text{total correct boundaries}} \quad (2)$$

$$F_1 = 2 * \frac{\text{precision} * \text{recall}}{\text{precision} + \text{recall}} \quad (3)$$

Language modelling We take the best morphological segmentation model and the statistical segmentation ones in order to do language modelling. We do 10-fold cross-validation in order to train models for end-task evaluation. We also do experiments using one fold investigating how the data volume influences the model training results. The evaluation metric is word and character level perplexity. Although the model we chose allows both character and word level training, in this paper we do word level training with subwords serving as words.

Predictive text entry We take the trained models from the former task and compare their performance in the predictive text task. The task is to predict the next linguistic unit of output for a given input looking at the top-3 predictions. The evaluation measure is average number of clicks per character and keystroke savings rate. The fewer clicks per character the less the end-user has to type. It's important to mention that the first segment of each word is always typed character by character; this is caused by the model not having token `<bos>` (beginning of the sentence) in its design and the fact that we are doing word level training. As mentioned above, we use the cross-validation models for this task.

Significance testing As all of the tasks are done using cross-validation, we have sets of results for each model. These results can be tested in order to say if some models are significantly better than the others. To implement this, first, we do the one-way ANOVA[3] with the null hypothesis being "all the means are the same". In case the null hypothesis is rejected, we do pairwise Least Significant Difference test (LSD-test)[4] to group the models so that we can find the best performing ones which are not significantly different from each other. The LSD values for all the tasks are given in the appendix.

5 Segmentation

The idea to compare statistical and morphological segmentation was already tested (Pan et al., 2020); the results show that the usage of morphological segmentation significantly improves the BLEU and ChrF3 metrics in neural machine translation (NMT).

In this paper we tried six segmentation models. Four of these are unsupervised: Byte-pair encoding (BPE; Gage, 1994), which was popularised by (Sennrich et al., 2016), Unigram (Kudo, 2018), WordPiece (Schuster and Nakajima, 2012) and Morfessor (Virpioja et al., 2013). The other two are supervised: NeuralMorphemeSegmentation (NMS; Sorokin and Kravtsova, 2018) and NCRF++ (Yang and Zhang, 2018).

Morfessor, NMS and NCRF++ are morphological segmentation models and they have to be trained and evaluated against a test subset. The remaining three do not require evaluation although they can be evaluated on a hold-out dataset.

As an output format, we decided to use one of those in the mentioned work (Pan et al., 2020): we modified the stem with singular suffix strategy, so that all of the subwords are treated the same way: single-morpheme words remain unchanged, in composite words every morpheme except the last one ends with #, the last morpheme ends with $.

5.1 Systems

As we decided to compare statistical and morphological segmentation, we also wanted to compare the models within each type. For this reason we chose BPE, Unigram and Wordpiece as statistical segmentation models. While choosing morphological segmentation models, we were looking for the ones that were tested not only on English and also had F_1-score as a computed metric. Thus, alongside Morfessor, the best-known morphological segmentor, we also chose NMS and NCRF++.

	Chukchi	K'iche'
Morfessor	0.610	0.618
NMS	**0.840**	**0.907**
NCRF++	0.821	0.874

Table 2: Best F_1-score for morphological segmentation. The neural morphological segmentation (NMS) model outperforms both Morfessor and NCRF++ for both languages.

5.2 Results

We decided to choose the best segmentation model out of Morfessor, NMS and NCRF++, because the paper is not dedicated entirely to defining the best morphological segmentation model being rather examining if it is better than statistical segmentation in specific tasks. The results of morphological segmentation are presented in Table 2 and the models hyperparameters are included in the appendix.

While being designed for Russian, an inflective language, NMS outperformed Morfessor and NCRF++. Though, since the small size of the available data the metrics are not stable, fluctuating between $n - 0.3$ and $n + 0.3$, where n is the F_1-score on evaluation data. Thus the second-best model, NCRF++, was chosen to segment the whole corpus and to be used while evaluating the language model.

Hence, 3 segmentation methods were used to make datasets for the following experiments – NCRF++, Unigram and Wordpiece. In order to do Unigram segmentation we used a package made by Google (Kudo and Richardson, 2018), for Wordpiece we used BertWordPieceTokenizer model from tokenizers package (Moi, 2021). Table 3 shows the same sentence being segmented with different models.

6 Language modelling

In order to do the text prediction we decided to choose the model that achieved state-of-the-art word level perplexities on Penn treebank and WikiText-2 (Merity et al., 2017). This model was applied (Schwartz et al., 2020) to several indigenous languages, including Chukchi, and showed good performance. This model trains fast, allows to be trained both on character level and word level, and

[3](2008) One-Way Analysis of Variance. In: The Concise Encyclopedia of Statistics. Springer, New York, NY. https://doi.org/10.1007/978-0-387-32833-1_297

[4](2008) Least Significant Difference Test. In: The Concise Encyclopedia of Statistics. Springer, New York, NY. https://doi.org/10.1007/978-0-387-32833-1_226

Variant	Example
Input text	*Xke'x ri nukinaq'*
Canonical	x# ke'x$ ri nu# kinaq'$
Morph. segmen.	x# ke'x$ ri nu# kinaq'$
BPE	xke# '# x$ ri nukina# q'$
Unigram	xke# '# x$ ri nuki# na# q'$
WordPiece	xk# e'x$ ri nuk# inaq'$

Table 3: Segmentation variants for the K'iche' sentence *Xke'x ri nukinaq'* "My beans were ground". The canonical segmentation corresponds to /CP-grind.PASS[5] the POSS.1SG-bean/. The hash symbol, #, indicates that there is a segment after the current one and the dollar symbol, $, indicates the last segment in a multi-segment word.

also is good dealing with overfitting, which is essential while working with low-resource languages.

Although BERT (Devlin et al., 2019) has been successfully used for low-resource languages (Ngoc Le and Sadat, 2020; Wang et al., 2020), models based on BERT models usually have hundreds of millions of parameters and as such are not efficient enough in terms of space for existing mobile phones. This is not suitable for us as our main goal is to use the model for a phone keyboard in order to do predictive text.

The data for language modelling was at first split into modelling (80 percent) and test (20 percent) subsets. Then for the 10-fold cross-validation the modelling subset was split into train (75 percent) and validation (25 percent) subsets. The folds were made using ShuffleSplit[6] with the same seed as the one used while language modelling. The dictionaries for the embeddings consist of all the subwords of train dataset plus the `<unk>` token; the validation subset is used to calculate perplexity in the end of each epoch. The models were trained until 5 epochs without perplexity improvement on a validation subset.

In order to investigate how the volume of data influences the model perplexity we decided to do language modelling increasing the data volume from 10 percent to 100 percent with a 10 percent step. The data is shuffled using the same seed as in other tasks and then we take the first *n* percent of lines. Thus we can be sure that two-way segmented models won't get the data only from one segmentor. We did these experiments using only one fold as we did not intend to show that a certain amount of data lets us train a better model.

The training hyperparameters are included in the appendix.

6.1 Modelling type

All the models we trained can be divided into three types: single-way segmented, two-way segmented and finetuned models.

In order to distinguish a language model from a segmentation method the model names will be given in **bold** e.g. Unigram is a segmentation model while **Unigram** is a model trained on data processed by the corresponding segmentation model.

6.1.1 Single-way segmented

Models of this type – **Morph. segm.**, **BPE**, **Unigram**, **Wordpiece** – were trained using single datasets we got in Section 5.

6.1.2 Two-way segmented

Models of this type – **MS+BPE**, **MS+Unigram**, **MS+Wordpiece** – were trained using two datasets we got in Section 5 combined together. The idea behind this modelling type is that we want to see if having data processed by different segmentation methods can help us solve both evaluation tasks on a high level.

6.1.3 Finetuning

As it was proposed in one of the related works (Boudreau et al., 2020), pretrained embeddings can be used in order to improve the performance of the language models. We decided to try finetuning though we chose to pretrain not only embeddings but also RNN layers.

Models of this type – **BPE2MS**, **Unigram2MS**, **Wordpiece2MS** – were at first trained using the Unigram/Wordpiece data and then morphologically segmented data was used to finetune the trained model. It's important to mention that while training models of this type either datasets volumes were step-by-step increased by 10 percent e.g. the first **Unigram2MS** model was trained on 10 percent of Unigram data and then on 10 percent of morphologically segmented data. Looking ahead we should also mention that it turned out there is no need to lower the learning rate of the model while finetuning it – the perplexity of the model in the end of training is the same while epoch count (and, accordingly, the training time) becomes approximately 10 times higher.

[6]https://scikit-learn.org/0.24/modules/generated/sklearn.model_selection.ShuffleSplit.html

6.2 Results

As we can see in Table 4, the best models for K'iche' and Chukchi according to perplexity are **Morph. segm.** and finetuning models.

	K'iche'		Chukchi	
	Wd	Ch	Wd	Ch
MS	**32.59**	**7.57**	**176.56**	**27.04**
BPE	38.53	8.95	2553.62	391.11
Uni	35.29	8.20	464.43	71.13
WP	148.24	34.45	2745.33	420.48
BPE2MS	**34.03**	**7.91**	**166.58**	**25.51**
Uni2MS	34.32	7.97	**163.58**	**25.05**
WP2MS	**32.06**	**7.45**	**165.90**	**25.41**
MS+BPE	35.46	8.24	500.85	76.75
MS+Uni	34.10	7.92	265.67	40.71
MS+WP	54.27	12.61	524.28	80.34

Table 4: Word (Wd) level and character (Ch) level perplexities for the models (mean scores of 10-fold cross-validation). **MS** stands for **Morph. segm.**, **Uni** stands for **Unigram**, **WP** stands for **Wordpiece**. We do not give subword level perplexities as they are not comparable. The best scores are in bold being significantly better according to ANOVA than the others but not outperform each other.

The two-way segmented models show lower scores than **Morph. segm.** ones, though they are better than the models trained on data of the second origin (BPE, Unigram, Wordpiece segmentors). It does seem like the usage of morphologically segmentated data allows us to improve the performance of the models.

It is worth saying that perplexity scores for different segmentations can't be compared to each other as is due to the dictionary sizes of all the models being different. This is why we computed the word and character perplexities using the subword ones (Mielke, 2019). Basically, it is just a normalisation of metrics in order to be able to compare them correctly. To do that, we computed the negative log-likelihood of the strings:

$$nll = \log ppl^{sw} * (C_{sw} + k) \tag{4}$$

where *nll* is negative log-likehood, ppl^{sw} is the computed subword level perpelxity, C_{sw} is the total count of subwords in the set and *k* is the total count of lines in the set that stands for the count of `<eos>` tokens which the model also predicted.

Then we computed word level and character level perplexities using the negative log-likehood we got on a previous step:

$$ppl^{w} = \exp \frac{nll}{C_w + k} \tag{5}$$

$$ppl^{c} = \exp \frac{nll}{C_c + k} \tag{6}$$

where ppl^w is word level perplexity, ppl^c is character level perplexity, *nll* is negative log-likehood, C_w is the total count of words in the set, C_c is the total count of characters in the set and *k* is the total count of lines in the set.

As we also did the modelling of one fold using different data volumes, we decided to look at the dependency of the finetuning metrics on the data volume and to compare them to the results of **Morph. segm.**. As we can see at Figure 1, 40 percent of data is a threshold at which the most models epoch count starts getting lower and the **Morph. segm.** model perplexity starts becoming smooth; the perplexity is rising due to the growing count of tokens in data. Moreover, the epoch count while doing the finetuning is lower than when training the **Morph. segm.**, while the perplexity scores are being close to each other after the 40 percent threshold, so we can confirm our conclusion that finetuning works well.

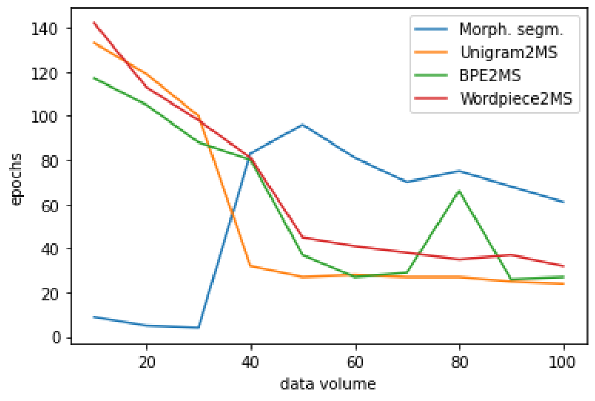
7 Predictive text input

Another task to evaluate language models is predictive text input. The idea is that we emulate a person using a smart keyboard while it is offering some predictions, which have to be meaningful. The meaningfulness is important because we assume that the typing person would like to choose from real words/morphs and not some artificial subwords that make at best no sense and in a worst case scenario they may mean something totally wrong (3). The example is given in Turkish because it illustrates the problem really good.

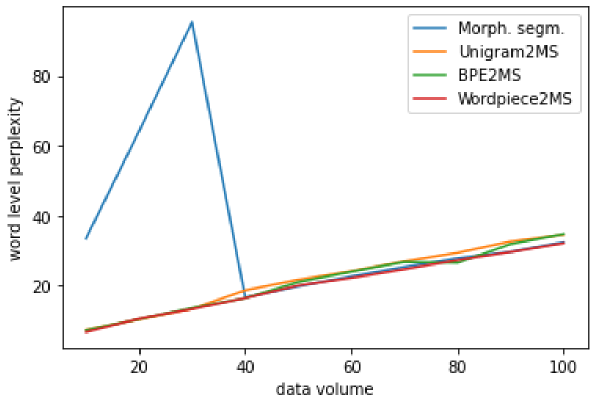
(3) a. araba-m-a
'into my car'

b. arab-am-a
'arab into *vulgar word*'

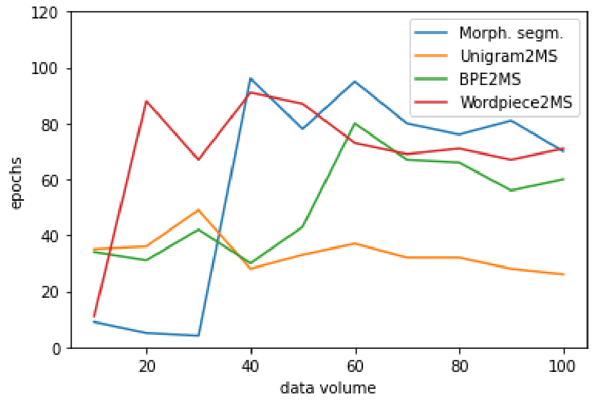
While evaluating, we look through top 3 model predictions and compare them to the subword we are currently predicting. If they are similar, that prediction is chosen, otherwise we look at the next one. If none of the predictions were correct, we consider



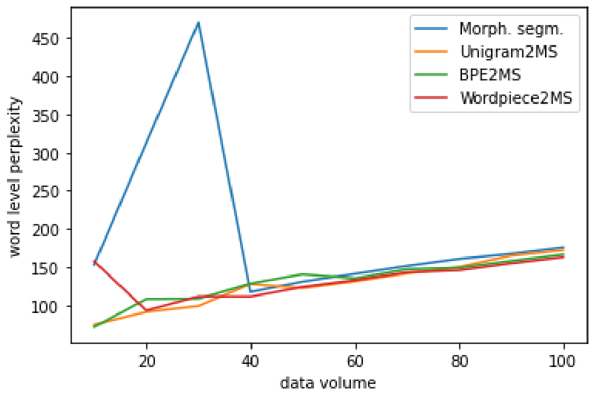
(a) K'iche' models best epochs.



(b) K'iche' models word level perplexity.



(c) Chukchi models best epochs.



(d) Chukchi models word level perplexity.

Figure 1: Best epochs and word level perplexity scores for finetuned models. Best epochs are the numbers of latest best epochs.

that the user will have to input the word to end character by character. The input for the model is built up using the remaining morphemes.

7.1 Results

As we can see in Table 5, for K'iche' the best model is **MS+Wordpiece** and for Chuckhi the best ones are **Morph. segm**, **BPE2MS**, **Wordpiece2MS** and **MS+Unigram**. We also decided to include the keystroke savings rate used in the Mi'kmaq paper (Boudreau et al., 2020) in order to be able to compare the results. The group of models which are the best for Chukchi is the second best for K'iche'.

Predictive text metrics do correlate with language-modelling metrics; even though **MS+Wordpiece** performs the best for K'iche', the group of **Morph. segm**, **BPE2MS** and **Wordpiece2MS** has both best perplexity and clicks per character scores. We suppose that the models connected with morphological segmentation perform better in this task because the language model, while not being trained on the evaluation data, got resembling training data.

The results for Chukchi are worse than the results for Ki'che'. The reason may be that gold standard for Chukchi is in telqep Chukchi, while the corpus used for training is in standard Chukchi. Another reason may be that words in Ki'che' evaluation data are shorter both segmentwise and characterwise than the Chukchi words. In case a model can not predict a correct morph, we penalize it by making the whole word be typed character-by-character, so the longer the word is, the more significant mistakes become.

Another issue that may influence the results is that the first subword of each word is typed character by character. We plan to get rid of it in the future in order to be able to evaluate the results better.

8 Discussion

As we can see, the evaluation shows that different models are good at different tasks which paves the way for a discussion if we can say that one model is better than another or not. While not being able to tell the correct answer for this question, we would recommend to try morphological segmentation as it can be used with a statistical one (alongside or by finetuning).

Morphological segmentation can also improve the model performance in predictive text task and other tasks, which were not discussed in this pa-

	K'iche'		Chukchi	
	CpC	KSR	CpC	KSR
No prediction	1.00	0.00	1.00	0.00
Morph. segm.	0.96	3.03	**0.99**	**0.78**
BPE	0.98	1.69	0.99	0.27
Unigram	0.98	1.46	0.99	0.26
Wordpiece	0.97	2.35	0.99	0.20
BPE2MS	0.96	3.45	**0.99**	**0.77**
Unigram2MS	0.96	3.49	0.99	0.69
Wordpiece2MS	0.96	3.53	**0.99**	**0.79**
MS+BPE	0.96	3.35	0.99	0.62
MS+Unigram	0.96	3.53	**0.99**	**0.73**
MS+Wordpiece	**0.95**	**4.26**	0.99	0.68

Table 5: Predictive keyboard metrics, the number of clicks per character (CpC) and keystroke savings rate (KSR) for each of the methods. 'No prediction' means that the user has to input all the words character by character including spaces, serving as baseline. The best scores are in bold being significantly better according to ANOVA than the others but not than each other.

per. What is important to mention is that there is no need in training models using morphologically segmented data from scratch, the existing models can be finetuned and the results will not differ significantly from the ones of **Morph. segm.** while the training time will be much lower as shown on Figures 1c and 1a.

As we can see, in all the tasks K'iche' models have better performance than Chukchi models. While we do not know the particular reason for this, we assume that the polysynthetic language complexity may be hindering the model from training. In the mentioned above paper (Lane and Bird, 2020) the authors also reported that polysynthetic languages have their special challenges such as high word length, complexity, etc.

As we referenced the Mi'kmaq (Boudreau et al., 2020), it seems reasonable to compare the results of their experiments with the results of ours. The results of the evaluation cannot be compared easily because our task was to predict *linguistic* units, not any kind of units, while in the Mi'kmaq paper words and BPE segments were being predicted; though if we do compare the results, we can see that the best KSR score for Mi'kmaq is **3.81**, while the best score for Ki'che' is **4.26**. At the same time, the best Chukchi KSR is much worse that the Mi'kmaq score being only **0.79**.

Alongside the metrics computed while experimenting there is also a metric which cannot be measured without end-user testing – the sanity check. As mentioned before, the issue with statistical segmentation is that subwords predicted and offered to the user may have no sense for the user or, what is much worse, may carry the wrong meaning. We do suppose that this alone can be a reason to choose morphological segmentation over the regular one because segmentation task is not done in isolation – it serves a purpose in a larger scheme of things. We think that in case the language model will be used in predictive text settings, where the user experience and user reaction is highly relevant, morphological segmentation should be chosen as a subword tokenization method, while statistical segmentation may be chosen while doing machine translation, for example.

9 Future work

We are planning to test several other language models and language modelling metrics in order to find out what correlates best with text prediction scores.

We find it reasonable to experiment on other languages, for example, Turkish, Nahuatl and Yupik, in order to get a better understanding when the use of morphological segmentation is reasonable.

Another task to do is to run an end-user evaluation of multiple segmentations and determine which units are preferred. In order to do this, we would also need to solve the problem of predictive text evaluation that the user has to input the first word character by character. In order to do this, we will possibly have to combine word level and character level based models.

Acknowledgements

Removed for review

References

Vasilisa Andriyanets and Francis Tyers. 2018. A prototype finite-state morphological analyser for Chukchi. In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, pages 31–40, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Jeremie Boudreau, Akankshya Patra, Ashima Suvarna, and Paul Cook. 2020. Evaluating the impact of sub-word information and cross-lingual word embeddings on mi'kmaq language modelling. In *Proceedings of the 12th Language Resources and Evaluation Conference*, pages 2736–2745, Marseille, France. European Language Resources Association.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding.

Philip Gage. 1994. A new algorithm for data compression. *C Users J.*, 12(2):23–38.

Tin Hlaing and Yoshiki Mikami. 2014. Automatic syllable segmentation of myanmar texts using finite state transducer. *International Journal on Advances in ICT for Emerging Regions (ICTer)*, 6.

Taku Kudo. 2018. Subword regularization: Improving neural network translation models with multiple subword candidates.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing.

William Lane and Steven Bird. 2020. Interactive word completion for morphologically complex languages. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4600–4611, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Stephen Merity, Nitish Shirish Keskar, and Richard Socher. 2017. Regularizing and Optimizing LSTM Language Models. *arXiv preprint arXiv:1708.02182*.

Sabrina J. Mielke. 2019. Can you compare perplexity across different segmentations?

Vipul Mittal. 2010. Automatic sanskrit segmentizer using finite state transducers. pages 85–90.

Anothony Moi. 2021. Tokenizers.

Tan Ngoc Le and Fatiha Sadat. 2020. Revitalization of indigenous languages through pre-processing and neural machine translation: The case of Inuktitut. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 4661–4666, Barcelona, Spain (Online). International Committee on Computational Linguistics.

Yirong Pan, Xiao Li, Yating Yang, and Rui Dong. 2020. Morphological word segmentation on agglutinative languages for neural machine translation.

Ivy Richardson and Francis M. Tyers. 2021. A morphological analyser for k'iche'.

Kevin Scannell. 2007. The Crúbadán Project: Corpus building for under-resourced languages. In *Proceedings of the 3rd Web as Corpus Workshop*, pages 5–15.

Mike Schuster and Kaisuke Nakajima. 2012. Japanese and korean voice search. In *International Conference on Acoustics, Speech and Signal Processing*, pages 5149–5152.

Lane Schwartz, Francis Tyers, Lori Levin, Christo Kirov, Patrick Littell, Chi kiu Lo, Emily Prud'hommeaux, Hyunji Hayley Park, Kenneth Steimel, Rebecca Knowles, Jeffrey Micher, Lonny Strunk, Han Liu, Coleman Haley, Katherine J. Zhang, Robbie Jimmerson, Vasilisa Andriyanets, Aldrian Obaja Muis, Naoki Otani, Jong Hyuk Park, and Zhisong Zhang. 2020. Neural polysynthetic language modelling.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units.

Alexey Sorokin and Anastasia Kravtsova. 2018. Deep convolutional networks for supervised morpheme segmentation of russian language. In *Artificial Intelligence and Natural Language*, pages 3–10, Cham. Springer International Publishing.

Derwin Suhartono., Garry Wong., Polim Kusuma., and Silviana Saputra. 2014. Predictive text system for bahasa with frequency, n-gram, probability table and syntactic using grammar. In *Proceedings of the 6th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART,*, pages 305–311. INSTICC, SciTePress.

Sami Virpioja, Peter Smit, Stig-Arne Grönroos, and Mikko Kurimo. 2013. Morfessor 2.0: Python implementation and extensions for morfessor baseline. D4 julkaistu kehittämis- tai tutkimusraportti tai -selvitys.

Zihan Wang, Karthikeyan K, Stephen Mayhew, and Dan Roth. 2020. Extending multilingual bert to low-resource languages.

Jie Yang and Yue Zhang. 2018. Ncrf++: An open-source neural sequence labeling toolkit. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics*.

Seunghak Yu, Nilesh Kulkarni, Haejun Lee, and Jihie Kim. 2017. Syllable-level neural language model for agglutinative language.

A Hyperparameters

Here we provide hyperparameter values for the various models to aid in reproduction of the results.

A.1 Morphological segmentation

In this section we describe the best hyperparameter settings that we found for the various tasks.

A.1.1 Morfessor

The best results for both K'iche' and Chukchi were achieved with this hyperparameters:

A.1.2 NeuralMorphemeSegmentation

The best results for K'iche' were achieved with this hyperparameters:

Parameter	Value
learning algorithm	recursive
training	type based

Table 6: Morfessor hyperparameters.

Parameter	Value
convolutional layers	3
window size	3 – 4
filters	96
dense output users	64
context dropout	0.3
memorize morphemes	no
memorize ngram counts.	no

Table 7: NMS hyperparameters (K'iche').

The best results for Chukchi were achieved with this hyperparameters:

Parameter	Value
convolutional layers	3
window size	4–6
filters	96
dense output users	20
context dropout	0.3
memorize morphemes	no
memorize ngram counts.	no

Table 8: NMS hyperparameters (Chuckhi).

A.1.3 NCRF++

The best results for K'iche' were achieved with this hyperparameters:

Parameter	Value
char. embedding dim	200
char. hidden vector dum	200
optimizer	Adagrad
convolutional layers	4
use CRF layer	yes
use char. sequence layer	yes
use CNN to train for chars	yes
use CNN to train for words	yes

Table 9: NCRF++ hyperparameters (K'iche').

The best results for Chukchi were achieved with this hyperparameters:

Parameter	Value
char. embedding dim	400
char. hidden vector dum	400
optimizer	Adagrad
convolutional layers	16
use CRF layer	yes
use char. sequence layer	yes
use CNN to train for chars	yes
use CNN to train for words	yes

Table 10: NCRF++ hyperparameters (Chukchi).

A.2 Least Significant Deviation values

The LSD-test results for language modelling and predictive text tasks:

Task	K'iche'	Chukchi
language modelling	1.494	17.806
predictive text	14.22e-4	6.779e-4

Table 11: LSD values

A.3 Language modelling

All the models were built results for Chukchi were achieved with this hyperparameters:

Parameter	Value
LSTM layers	3
embedding dim	256
hidden units per layer	3000
use regularization	no
layers dropout	0.4
RNN layers dropout	0.1
embeddings dropout	0.1
remove words from embeddings dropout	0.0
sequence length	100
optimizer	Adam
learning rate	1e-3
weight decay	1.2e-6
seed	1111

Table 12: Awd-lstm hyperparameters.

B Evaluation

Case	Sentence
Raw	naqaj at k’o wi chi wech
Morph. Segm.	n a q a j _ a t _ k ’ o _ wi _ c h i _ w ech _
Unigram	n a q a j _ a t _ k ’ o _ w i _ c h i _ w e c h _
Wordpiece	n a q a j _ a t _ k ’ o _ wi _ c h i _ w e c h _
Raw	qonojel wa’ pa q’ab’ la oj k’o wi nudyos
Morph. Segm.	r i _ n u m i ’ a l _ x u t a q _ l o q _ jun _ t a q o ’ m _ w u j _ chi _ w e _
Unigram	r i _ n u m i ’ a l _ x u t a q _ l o q _ jun _ t a q o ’ m _ w u j _ c h i _ w e _
Wordpiece	r i _ n u m i ’ a l _ x u t a q _ l o q _ jun _ t a q o ’ m _ w u j _ c h i _ w e _
Raw	xa rumal keta’m konojel ri winaq kkiqumuj le ja’
Morph. Segm.	x a _ r umal _ k e t a ’ m _ k o n o j e l _ ri _ winaq _ k ki q u m u j _ l e _ j a ’ _
Unigram	x a _ r u m a l _ k e t a ’ m _ k o n o j e l _ ri _ winaq _ k k i q u m u j _ l e _ j a ’ _
Wordpiece	x a _ r u m a l _ k e t a ’ m _ k o n o j e l _ r i _ w i n a q _ k k i q u m u j _ l e _ j a ’ _

Table 13: Text prediction by single-way segmented models(K’iche’).