ISSN 2007-9737

Predictive Text for Agglutinative and Polysynthetic Languages

Sergey Kosyak[1,*], Francis M. Tyers[2]

[1] Higher School of Economics, School of Linguistics, Russian Federation

[2] Indiana University, Department of Linguistics, United States of America

ser97Nikost@gmail.com, ftyers@iu.edu

Abstract. This paper is devoted to the usage of morphological segmentation for language modelling and predictive text. Having interest in providing effective and ergonomic text prediction for low-resource languages, we examine the task of predictive text entry for five under-resourced and Indigenous languages: Bashkir, Chuvach, K'iche', Mari and Chukchi. In segmentation section, we overview used segmentation methods, both statistical and morphological. They are used to create datasets from unannotated corpus for language modelling. We train models and measure normalized word and character level perplexities. In order to evaluate the models, we use predictive text entry task and measure keystroke savings rate. We provide two variations of evaluation algorithm which differ in how they process the latest user input: one using it as a start of a new prediction, the other using only the unsegmented part of it as a start of a new prediction. The best score is achieved with BPE for K'iche' (16.08) with morphological segmentation being the second best (15.30). We find that neither of the segmentation models performs the best in both language modelling and predictive text tasks. In order to define the best performing one and to test our hypothesis about predictive text ergonomics, we plan to do end-user predictive text evaluation.

Keywords. Morphological segmentation, agglutinative languages, polysynthetic languages, evaluation, language modelling, predictive text.

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 store 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 stored in the model vocabulary.

The segmentation task is not new, there are many algorithms with BPE [5] being the most known and used to do the 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. Though this task is not the only one that allows to compare morphological and statistical segmentation, we decided to choose it for being the most evident.

We also 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 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 reason we decided to work with Indigenous languages is that there is a need for new researches, instruments and experiments with Indigenous languages [10]. The amount of works in language technology for such languages increases each year, yet we find ourselves willing to have a hand in the development of this field. We suppose that one of the main problems is that there is not much Indigenous languages data or instruments to work with which leads to people not wanting to work with them, and that leads to having not much data and instruments. We believe that if we find tools which will work best with Indigenous languages, which are often agglutinative or polysynthetic, it will pave the way for creation of new and better ones as the baseline will be already described and made.

As most of the papers devoted to low-resource languages provide results of experiments on a set of 1 to 3 languages, we wanted to include more languages in our research in order to be able to compare the used methods better and provide more generalized results. We suppose that developing a universal pipeline for many languages can be useful for other researchers, thus we tried to conduct the experiments for all of the languages as similar as we could, excluding morphological segmentation.

The remainder of the paper is laid out as follows: in Section 2 we overview the languages we experiment on, in Section 3 we speak about the works that were an inspiration for this paper, in Section 4 we describe the experiments, in Section 5 we review used segmentation methods and do the segmentation, in Section 6 we provide results of language modelling, in Section 7 we speak about predictive text evaluation, 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. While English is neither an agglutinative nor polysynthetic language, we give examples in English in order for the reader to better understand them.

2 Languages

As we wanted to make the research more generalized, we conducted our experiments using five languages – four agglutinative:

— Bashkir (ISO-639: `bak`), a Turkic language belonging to the Kipchak branch, co-official with Russian in Bashkortostan, Russian Federation;

— Chuvash (ISO-639: `chv`), a Turkic language spoken in European Russia, primarily in the Chuvash Republic and adjacent areas;

— K'iche' (ISO-639: `quc`), a Mayan language of the central highlands in Guatemala and Mexico;

— Mari (ISO-639: `chm`), formerly known as the Cheremiss language, a Uralic language, spoken primarily in the Mari Republic of the Russian Federation;

and one polysynthetic:

— Chukchi (ISO-639: `ckt`), a Chukotko-Kamchatkan language of Siberia.

Both of these types are characterised by words consisting of a large number of individual morphs, surface representations of morphemes.

The following example in K'iche' (1) demonstrate this tendency.[1]

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

[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'.

2.1 Data

As we work with low-resource languages, the availability of large corpora is limited. We used both unannotated text and annotated text for morphological segments.

For Bashkir we used a parallel Bashkir-Russian corpus [25], collected from translated books and internet pages. As there was no annotated data available, we used Hunspell [18] files to produce the annotated data.

For Chuvash, we used the Corpus of the Chuvash language[2], collected by activites from "Chuvash language laboratory". This corpus includes texts of different types – publicism, prose, laws, etc. As there were no annotated data available, we used Hunspell files to produce the annotated data.

For Mari we used a parallel Mari-Russian corpus [3], collected from translated books and internet pages. As there is no annotated data, we used Hunspell files to produce the annotated data.

For K'iche' we used annotated and unannotated texts. The annotated texts consist of a hand-segmented set of sentences from a range of sources including grammar-book and dictionary examples, stories and legal texts, which were used in constructing a morphologically and syntactically annotated corpus of K'iche'. The second, unannotated, portion of the data was obtained from the *An Crúbadán* project [21] that collects corpora from the web for Indigenous and marginalised languages.

For Chukchi, the annotated data came from the ChukLang[3] 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.

As the amount of data in the full corpora available for Bashkir, Chuvash and Mari largely exceeded the amount of data available for K'iche' and Chuckhi, we decided to use only 10 per cent of the available data for those three languages.

Table 1 describes the language data we used.

[2] https://en.corpus.chv.su/content/about.html

[3] https://chuklang.ru/

2.2 Preprocessing

In order to segment the raw data for K'iche' and Chuckhi using morphological segmentation the annotated data was split into two disjoint subsets: train (50 per cent) and test (50 per cent). This ratio was chosen due to low annotated data volume – we suppose that a choice of a disbalanced ratio like 80 per cent/20 per cent can lead to unreliable results.

In order to segment Bashkir, Chuvash and Mari we used all the corpus.

3 Related Work

Being one of the latest works [23] 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 usage is a good segmentation method performing well for lots of languages [14] [6] and there are even ones for Bashkir, K'iche' [20] and Chukchi [1], we decided that we will not use them because the coverage for Chukchi is too low, it is hard to do disambiguation with FST because it requires a huge tagged corpus and we did not have FST for Chuvash and Mari. Unfortunately, the authors could not do the end-task evaluation of the trained models but suggested doing predictive text.

The comparison of morphological segmentation and statistical segmentation was also described in previous researches. For example, in a paper devoted to machine translation for polysynthetic languages [11] the authors compared BPE [24] and several morphological segmentation models. They showed that morphological segmentation can outperform BPE for low-resource languages; though the task of machine tranlation differs from predictive text, it still was an inspiration for us to do our own research.

Another work [2] 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

Table 1. Dataset sizes for the five languages measured in sentences and words. Unannotated and annotated datasets do not intersect. Annotation for K'iche' and Chukchi was done manually. Annotated texts for Bashkir, Chuvash and Mari where not available to use

	Unannotated		Annotated	
	Sentences	Words	Sentences	Words
baskhir	109321	1299112	-	-
chuvash	158447	1561636	-	-
k'iche'	24,254	275,265	1,299	8,789
mari	200305	1549616	-	-
chukchi	33,322	151,585	1,006	4,417

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 [26, 31] 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. Additionally, the Bahasa paper described the idea of combining several technics in order to improve word prediction *system*. The technics which are described in the paper are: ranking words based on their input frequency; probability tables to store predefined phrases; n-grams to predict next words based on the previous one; syntactic using grammar, which predicts words using syntactic relations between words. Though some of the ideas can be applied later while doing end-user evaluation, currently we omitted them. The Korean paper [31] suggests making special embeddings derived from syllables and morphemes instead of doing subword tokenization. Moreover, the proposed method was already commercialized being achieving state-of-the-art performance. Even though that does seem like a good idea to change the way the embeddings are built, in this paper we decided to stick to subword tokenization as a way to solve OOV problem.

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 [8] 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 three 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.

Segmentation We use morphological segmentation models and statitical segmentation models to annotate the data described in Table 1.

Language modelling We use the segmentated data in order to do language modelling. We do 10-fold cross-validation in order to train models for end-task evaluation. The evaluation metric is normalized 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 evaluate the ability of the model to predict the user input. We look through top-3 predictions. The evaluation measure is keystroke savings rate. As in the previous task, we use the cross-validation models for this one.

Keystroke savings rate is measured as:

$$\mathrm{KSR} = \frac{\mathrm{keys}_{\mathrm{normal}} - \mathrm{keys}_{\mathrm{prediction}}}{\mathrm{keys}_{\mathrm{normal}}} \times 100. \quad (1)$$

5 Segmentation

In this paper we used three segmentation models for statistical segmantation: Byte-pair encoding [5] (BPE), which was popularised by [24], Unigram [7] and WordPiece [22].

For morphological segmentation for K'iche' and Chukchi we used NCRF++ [29] as we already had experience with it.

For morphological segmentation for Bashkir, Chuvash and Mari we used Hunspell [18] as we had dictionary and affix files for these languages while having no annotated data which could be used as gold standard.

As an output format, we decided to use one of the used in the mentioned work [19]: 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 $.

In the following subsections we describe the chosen segmentation models. In order to make the comparison between morphological and statistical segmentation better for statistical segmentation we chose not only BPE but also Unigram and Wordpiece, which are proposed as current alternatives for BPE [9]. As for morphologial segmentation, we initially wanted to compare several models but after long consideration we decided that would be out of scope of this research.

BPE BPE was originally created in the context of data compression, then it turned out to perform well solving the problem of low frequency words and OOV problem. BPE was later slightly modified [24] so that frequently occuring subwords are not replaced by another token but are merged together, which later became the standard. BPE model has both characters and subwords in its vocabulary which makes it capable of managing large corpus data without introducing a token representing unique subwords.

Unigram While sharing the same idea as BPE, Unigram [7] is more flexible as it is based on a probabilistic language model. It works in such a way that it iteratively makes a vocabulary, optimizes the probabilities of subwords in it and keeps the subwords having the largest loss. Single-character subwords are always kept in the vocabulary in order to avoid OOV.

WordPiece This algorithm [22] is based on an idea that the most important subwords will increase the likelikood of the model to predict the given training data. The initial model is built using only the characters and then the subwords are iteratively chosen in order to improve the metric.

NCRF++ NCRF++ [29] is a toolkit built to combine both neural and statistical approaches. Opposing the NMS authors, NCRF++ creators decided to use CRF alongside neural networks, which seems to be a good idea while conditional random fields were successfully used at morphological segmentation of different languages – German [28], Korean [16], Nguni [15] and others. NCRF++ consists of character sequence layers, word sequence layers and the inference layer: the sequence layers are usually RNN or CNN, the inference is being done using softmax or CRF. The most beneficial feature of NCRF++ is that it is fully configurable, allowing making user-defined

layers, and can easily be used for any task, while the existing alternatives are mostly focused on a specific architecture and tasks. The results of their benchmarks show that their solution competes well against other known ones.

Hunspell Hunspell [18] is a spell checker and morphological analyser designed for languages with rich morphology and complex word compounding and character encoding, originally designed for the Hungarian language. It is based on MySpell and is backward-compatible with MySpell dictionaries. While MySpell uses a single-byte character encoding, Hunspell can use Unicode UTF-8-encoded dictionaries. Even though Hunspell returns stem and morphological features of morphemes as a result of analyses it can be easily modified to also return the morphemes themselves so that Hunspell can easily segment sentences.

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 [12]. While not being trained in the original paper on Indigenous language data, this model was applied [23] to several Indigenous languages, including Chukchi, and showed good performance. This model trains fast, can be trained both on character level and word level, and also is good dealing with overfitting, which is essential while working with low-resource languages. In order to do cross-validation and to handle `<unk>` token we have additionally modified the code.

The reason why we didn't choose BERT [4], although it was successfully used for low-resource languages [17, 27], is that BERT models usually have hundreds of millions of parameters , so they won't fit easily on mobile phones, while 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 per cent) and test (20 per cent) subsets. Then for the 10-fold cross-validation the modelling subset was split into train (75 per cent) and validation (25 per cent) subsets. The folds were made using ShuffleSplit[4] using 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.

We should also mention that perplexity scores for different segmentations can not be compared in a raw state 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 [13]. Basically, it is just a normalization of metrics in order to compare them. To do that, we computed the negative log-likelihood of the strings:

$$nll = \log ppl^{sw} * (C_{sw} + k), \tag{2}$$

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:

$$ppl^{w} = \exp \frac{nll}{C_w + k}, \tag{3}$$

$$ppl^{c} = \exp \frac{nll}{C_c + k}, \tag{4}$$

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.

The model training hyperparameters are included in the appendix.

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

Results As we can see in Table 2, the best performing models across all of the languages are the ones trained using Unigram segmented data. Morphological segmentation, on the other hand, loses to statistical segmentation with Wordpiece being the only one performing worse than morphological segmentation.

It is also worth mentioning that modelling results can be clustered by their performance into three groups with the first group including Bashkir, Chuivash and Mari, the second one including K'iche' and tthe third one including Chukchi. It may be that as the amount of data for Bachkir, Chuvash and Mari is higher, language modelling task becomes more complex, so the results are worse then K'iche' ones. Chukchi, on the other hand, is a polysynthetical language with the lowest dataset size (if measured in words), which can explain its poor performance. Interestingly, K'iche' scores are the best across all of the languages even though the dataset size is much lower than for Bashkir, Chuvach and Mari. This trend can be observed across all the used segmentation methods and the reason may be the level of language complexity, the used dataset volume, corpus content itself.

One way to investigate the impact of linguistical features on language modelling performance is to include more languages in the research and then compare their features with modelling results to find correlations. In order to check the impact of dataset volume another set of experiments should be done in which the datasets would be reduced to 90, 80, 70, etc. per cent of initial volume and we would check if this trend still exists. As for the corpus content, another set of experiments should be done in order to do language modelling using texts from one domain. This, however, may be hard for Chukchi and K'iche' as there is not much data available.

7 Predictive Text Input

In order to evaluate the models we did predictive text evaluation. The idea is that we emulate a person using a smart keyboard while it is offering some predictions. Though the simple evaluation could be just next token prediction when we would try to predict next segments based on the previous ones, we decided to make a closer-to-real-usage evaluation and emulate clicks and prediction choosing.

We have experimented before with different prediction algorithms and their options and came up with this algorithm with 2 Variations on second step:

1. Segment current user input – the current input is passed to a segmentation model;
2. Turn segments into tensor – the segments array is turned into a tensor with segments indexes and we decide on the 'prediction start'.
   (a) We use all of the segments as an input for the model except the ones coming after the latest space between words, which are joined to become the 'prediction start';
   (b) We use all of the segments as an input for the model except the last one (in case it is an unknown segment for the model), which is used as the 'prediction start';
3. Initialize an empty hidden – this decision was made based on our previous experiments as they show that this strategy is the best performing one compared to using the previous hidden or recomputing it for each prediction;
4. Make a prediction and get predictions to choose from – a prediction is considered a candidate if it is longer than 1 symbol (as such prediction is not performative), continues the sentence we are trying to predict and starts with the 'prediction start' (which can be an empty string). Then we look through 3 candidates to choose the longest one. Such method has some flaws such as choosing only those candidates that continue the sentence which improves the evaluation performance, however, we consider such method as an upper limit for end-user evaluation.

In order to illustrate the algorithm better, we provide examples in K'iche' for both evaluation variations in the appendix.

Table 2. Word level and character level normalized perplexities for the models (mean scores of the 10-fold cross-validation). We do not give subword level perplexities as they are not comparable

		Morph. segm	BPE	Unigram	Wordpiece
bashkir	word	102.9	98.9	82.2	147.0
	character	17.1	16.4	13.6	24.4
chuvash	word	130.7	106.0	88.0	179.6
	character	24.0	19.4	16.1	32.9
k'iche'	word	71.7	53.2	52.2	72.8
	character	18.5	13.7	13.5	18.8
mari	word	113.8	92.2	77.2	142.5
	character	20.8	16.9	14.1	26.1
chukchi	word	518.7	232.8	152.3	515.5
	character	79.6	35.7	23.3	79.1

It should be mentioned that we look through top-3 predictions due to the fact that the real predictive keyboard is limited in space, so typically three suggestions is the maximum to show. As we can see in the papers devoted to predictive text [30] [31], they all use the same count of predictions.

The language models were not trained on the evaluation data, so we can be sure that the evaluation is fair.

Results As we can see in Table 3, for Variation a of evaluation the best performing results are achieved using models trained on morphologically segmented data, while the best statistical segmentation method for all of the languages except Chukchi is Wordpiece. These results contrast with the results of language modelling, where morphological segmentation and Wordpiece performed the worst. As for the Variation b the best results are achieved with Unigram for Bashkir, Chuvash and Chukchi and BPE for K'iche' and Mari.

Correlation between evaluation and language modelling results can not be observed and the results for the predictive text Variations also differ from each other. For this reason it can not be definitely said if one segmentation is better than the other. In order to compare morphological and statistical segmentation performance a more thorough research should be done including not only end-user evaluation of predictive text but also more evaluation tasks such as machine translation, speech-to-text, etc.

8 Discussion

As we can see, the results show that there is no incontestably best segmentation method. This, however, shows that morphological segmentation *at the very least* can compete with statistical one and in some cases outperform it. The best way to answer which segmentation method is the best is to include more languages and more NLP tasks as this will allow us to get a more generalized view.

Speaking of the languages presented in the paper, we could not find a strong correlation between the fact if a language is agglutinative or polysynthetic and tasks performance, however, the results for Chukchi at both language modelling and predictive text tasks are the worst across all the languages. This is yet to be thoroughly checked but we assume that the polysynthetic language complexity may be hindering the model from training. In the mentioned above paper [8] the authors also reported that polysynthetic languages have their special challenges such as high word length, complexity, etc.

Table 3. Predictive text keystroke savings rate 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

Variation a	No prediction	Morph. segm	BPE	Unigram	Wordpiece
bashkir	0.0	**11.33**	1.23	3.00	4.68
chuvash	0.0	**8.56**	1.56	2.53	2.64
k'iche'	0.0	**15.30**	11.56	12.91	14.08
mari	0.0	**8.17**	3.62	2.95	4.82
chukchi	0.0	**6.86**	4.02	3.93	1.60
Variation b	No prediction	Morph. segm	BPE	Unigram	Wordpiece
bashkir	0.0	2.29	2.16	**8.01**	7.80
chuvash	0.0	0.96	3.86	**6.12**	4.28
k'iche'	0.0	12.40	**16.08**	15.58	11.96
mari	0.0	0.80	**7.25**	5.58	5.58
chukchi	0.0	5.03	5.04	**6.29**	3.63

As we referenced the Mi'kmaq [2], it seems reasonable to compare the results of their experiments with the results of ours. The authors report that they considered using BPE and Unigram with different vocabulary sizes, and the best segmentation model appeared to be BPE with a vocabulary size of 2000. At the same time, our experiments show that BPE segmentation is worse than Unigram segmentation at language modelling and predictive text. Comparing the results, we can see that the best KSR score for Mi'kmaq is **3.81**, while the best score we could achieve for Ki'che' is **16.08**. At the same time, the metrics for other languages are not that high which means that there is room for further improvement.

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.

Another problem which is brought by existing predictive keyboards is that they predict whole words.

For example, if the user wants to input word *antidisestablishmentarianism*, the predictive keyboard due to the lack of space on the screen will show a prediction *antidises...rianism*.

This problem can be solved if language modelling is done using morphs but not whole words.

9 Future Work

We find it reasonable to experiment on more languages, for example, Turkish, Nahuatl and Yupik, and see if we can get similar results using them.

Another task to do is running an end-user evaluation and determine which units – morphologically justified or not, will be preferred by real people.

Acknowledgments

We thank Robert Pugh for his comments and suggestions on an earlier version of this manuscript. We also thank Aigiz Kunafin and Andrei Chemyshev for their help with obtaining language data for Bashkir, Chuvash and Mari.

References

1. **Andriyanets, V., Tyers, F. (2018).** A prototype finite-state morphological analyser for Chukchi. Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages, Association for Computational Linguistics, Santa Fe, New Mexico, USA, pp. 31–40.
2. **Boudreau, J., Patra, A., Suvarna, A., Cook, P. (2020).** Evaluating the impact of sub-word information and cross-lingual word embeddings on Mi'kmaq language modelling. Proceedings of the 12th Language Resources and Evaluation Conference, European Language Resources Association, Marseille, France, pp. 2736–2745.
3. **Chemyshev, A., Sabantsev, G., Timofeeva, N., Semenov, V. (2023).** Mari-Russian parallel corpora.
4. **Devlin, J., Chang, M.-W., Lee, K., Toutanova, K. (2019).** Bert: Pre-training of deep bidirectional transformers for language understanding.
5. **Gage, P. (1994).** A new algorithm for data compression. C Users J., Vol. 12, No. 2, pp. 23–38.
6. **Hlaing, T., Mikami, Y. (2014).** Automatic syllable segmentation of myanmar texts using finite state transducer. International Journal on Advances in ICT for Emerging Regions (ICTer), Vol. 6. DOI: 10.4038/icter.v6i2.7150.
7. **Kudo, T. (2018).** Subword regularization: Improving neural network translation models with multiple subword candidates.
8. **Lane, W., Bird, S. (2020).** Interactive word completion for morphologically complex languages. Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), pp. 4600–4611. DOI: 10.18653/v1/2020.coling-main.405.
9. **Ma, E. (2019).** 3 subword algorithms help to improve your nlp model performance.
10. **Mager, M., Gutierrez-Vasques, X., Sierra, G., Meza-Ruiz, I. (2018).** Challenges of language technologies for the indigenous languages of the Americas. Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics, Santa Fe, New Mexico, USA, pp. 55–69.
11. **Mager, M., Oncevay, A., Mager, E., Kann, K., Vu, N. T. (2022).** Bpe vs. morphological segmentation: A case study on machine translation of four polysynthetic languages.
12. **Merity, S., Keskar, N. S., Socher, R. (2017).** Regularizing and Optimizing LSTM Language Models. arXiv preprint arXiv:1708.02182.
13. **Mielke, S. J. (2019).** Can you compare perplexity across different segmentations?
14. **Mittal, V. (2010).** Automatic Sanskrit segmentizer using finite state transducers. pp. 85–90.
15. **Moeng, T., Reay, S., Daniels, A., Buys, J. (2021).** Canonical and surface morphological segmentation for nguni languages. .
16. **Na, S.-H. (2015).** Conditional random fields for Korean morpheme segmentation and POS tagging. ACM Transactions on Asian and Low-Resource Language Information Processing, Vol. 14. DOI: 10.1145/2700051.
17. **Ngoc Le, T., Sadat, F. (2020).** Revitalization of indigenous languages through pre-processing and neural machine translation: The case of Inuktitut. Proceedings of the 28th International Conference on Computational Linguistics, International Committee on Computational Linguistics, Barcelona, Spain (Online), pp. 4661–

4666. DOI: 10.18653/v1/2020.coling-main.410.

18. **Ooms, J. (2025).** hunspell: High-Performance Stemmer, Tokenizer, and Spell Checker. R package version 3.0.7.
19. **Pan, Y., Li, X., Yang, Y., Dong, R. (2020).** Morphological word segmentation on agglutinative languages for neural machine translation.
20. **Richardson, I., Tyers, F. M. (2021).** A morphological analyser for K'iche'.
21. **Scannell, K. (2007).** The Crúbadán Project: Corpus building for under-resourced languages. Proceedings of the 3rd Web as Corpus Workshop, pp. 5–15.
22. **Schuster, M., Nakajima, K. (2012).** Japanese and korean voice search. International Conference on Acoustics, Speech and Signal Processing, pp. 5149–5152.
23. **Schwartz, L., Tyers, F., Levin, L., Kirov, C., Littell, P., kiu Lo, C., Prud'hommeaux, E., Park, H. H., Steimel, K., Knowles, R., Micher, J., Strunk, L., Liu, H., Haley, C., Zhang, K. J., Jimmerson, R., Andriyanets, V., Muis, A. O., Otani, N., Park, J. H., Zhang, Z. (2020).** Neural polysynthetic language modelling.
24. **Sennrich, R., Haddow, B., Birch, A. (2016).** Neural machine translation of rare words with subword units.
25. **Shakirov, I., Kunafin, A. (2023).** Bashkir-Russian parallel corpora.
26. **Suhartono., D., Wong., G., Kusuma., P., Saputra., S. (2014).** Predictive text system for Bahasa with frequency, n-gram, probability table and syntactic using grammar. Proceedings of the 6th International Conference on Agents and Artificial Intelligence - Volume 1: ICAART,, INSTICC, SciTePress, pp. 305–311. DOI: 10.5220/0004756603050311.
27. **Wang, Z., K, K., Mayhew, S., Roth, D. (2020).** Extending multilingual bert to low-resource languages.
28. **Würzner, K.-M., Jurish, B. (2015).** Dsolve – morphological segmentation for german using conditional random fields. pp. 94–103. DOI: 10.1007/978-3-319-23980-4_6.
29. **Yang, J., Zhang, Y. (2018).** Ncrf++: An open-source neural sequence labeling toolkit. Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics.
30. **Yu, S., Kulkarni, N., Lee, H., Kim, J. (2017).** An embedded deep learning based word prediction.
31. **Yu, S., Kulkarni, N., Lee, H., Kim, J. (2017).** Syllable-level neural language model for agglutinative language.

Article received on 10/02/2025; accepted on 08/12/2025.
**Corresponding author is Sergey Kosyak.*

A Appendix: Hyperparameters

In this appendix, we provide hyperparameter values for the various models to aid in the reproduction of the results.

A.1 NCRF++

K'iche' morphological segmentation was done using hyperparameters presented in Table 1.

Table 4. NCRF++ hyperparameters (K'iche').

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

Chukchi morphological segmentation was done using hyperparameters presented in Table 2.

A.2 Language modeling

All the models were trained with hyperparameters presented in Table 3.

Table 5. NCRF++ hyperparameters (Chukchi).

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 6. Awd-lstm 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

B Evaluation

Below we provide evaluation examples.

Evaluation process example for sentence "Rumal rech che ri" (Variationa).

User input	Segments	'Prediction start'	Chosen prediction
	[<bos>]		
r	[<bos>]	r	rumal
rumal	[<bos>, rumal, <eow>]		
rumal r	[<bos>, rumal, <eow>]	r	rech
rumal rech	[<bos>, rumal, <eow>, rech, <eow>]		ch
rumal rech ch	[<bos>, rumal, <eow>, rech, <eow>]	ch	che
rumal rech che	[<bos>, rumal, <eow>, rech, <eow>, che, <eow>]		
rumal rech che r	[<bos>, rumal, <eow>, rech, <eow>, che, <eow>]	r	
rumal rech che ri	[<bos>, rumal, <eow>, rech, <eow>, che, <eow>, ri, <eow>]		

Evaluation process example for sentence "Rumal rech che ri" (Variation b).

User input	Segments	'Prediction start'	Chosen prediction
	[<bos>]		
r	[<bos>]	r	
ru	[<bos>, ru]		
rum	[<bos>, ru]	m	
ruma	[<bos>, rum]	a	al
rumal	[<bos>, rumal, <eow>]		
rumal r	[<bos>, rumal, <eow>]	r	
rumal re	[<bos>, rumal, <eow>]	re	
rumal rec	[<bos>, rumal, <eow>, re]	c	ch
rumal rech	[<bos>, rumal, <eow>, rech, <eow>]		ch
rumal rech ch	[<bos>, rumal, <eow>, rech, <eow>]	ch	che
rumal rech che	[<bos>, rumal, <eow>, rech, <eow>, che, <eow>]		
rumal rech che r	[<bos>, rumal, <eow>, rech, <eow>, che, <eow>]	r	
rumal rech che ri	[<bos>, rumal, <eow>, rech, <eow>, che, <eow>, ri, <eow>]		