Tensor Product of Representations in Service of Low-Resource Languages

G. Arshinov, S. Kosyak, D. Samsonova, F. Tyers, M. Voronov

June 2020

Abstract

Polysynthetic low-resource languages are poorly treated with standard language modeling approaches. In this paper a hypothesis that word-segment embeddings based on tensor product of representations show better performance for low-resource languages compared to conventional word- and char-based models is tested. In order to prove it a pipeline that allows to process low-resource polysynthetic languages was developed. Using Neural Sequence Labeling Toolkit (Yang and Zhang 2018) to train a segmenter on a Chukchi corpus, a raw Chuckchi corpus was segmented and the *iiksiin* (Schwartz, Haley, et al. 2019-2020) model was employed to create the embeddings. After that we tested them on the language modelling task and evaluated the results, which showed a notable increase in performance compared to regular approaches.

Tags: TPR, Chuckchi, language modeling, polysynthetic, low-resource, NLP

1 Introduction

Most traditional text vectorization approaches target languages that do not have much inflection (e.g. (Mikolov et al. 2013; Bojanowski et al. 2017; Pennington, Socher, and Manning 2014)). Such approaches treat *cat* and *cats* as individual words. It works reasonably well for analytic languages such as Chinese or English. However, there are polysynthetic languages that feature extensive morphology and cannot be efficiently processed this way.

In this work we will model the Chukchi language. Let us consider the following two examples (examples (1) and (2)). As you could see, the words have the same root but different inflectional affixes. Eventually, representing Chukchi tokens using one of the traditional approaches will inevitably fail to encode the complex meaning, that is built up out each and individual morpheme.

(1) weɬə-tko-ra-jpə-ŋ
goods-ITER-dwelling-ABL-AD
in the shop

(2) q-wełə-tko-ra-nta-ɣ-e=ʔəm
2.S/A.SUBJ-goods-iter-dwelling-GO.DO-IRR-2/3SG.S=EMPH
go to the shop!

Moreover, many polysynthetic languages are minority languages. For example, Chukchi is spoken in by approximately 5100 people and is marked as "threatened" in the *Ethnologue* database(*Ethnologue. Languages of the World. Chukchi* 2020). Therefore, there is very little language data available for Chukchi.

Due to the two problems mentioned it is impossible to apply traditional approaches, we decided our goal is to develop a pipeline that can process low-resource languages with extensive inflectional morphology.

2 Related work

The idea that some smaller segments can be used as representations was suggested in (Smolensky 1990). In this article Smolensky suggests "a formalization of the idea that a set of value variable pairs can be represented by accumulating activity in a collection of units each of which computes the product of a feature of a variable and a feature of its value" (Smolensky 1990, p. 159). He suggests using tensor product to accumulate representations of smaller structures into bigger ones.

This way, the word *cats* will be treated not only as a whole but also as a combination of two morphemes.

The idea to use tensor product of representations to process a natural language, was implemented in 2019 in a tool called *iiksiin* (Schwartz, Tyers, et al. 2020; Schwartz, Haley, et al. 2019-2020). It "constructs a sequence of morpheme tensors from a word using Tensor Product Representation" (Schwartz, Haley, et al. 2019-2020). We will further cover the way this tool functions.

3 Data

For our language model experiments we use Chukchi corpus (Tyers n.d.). The corpus consists of fiction and folklore texts in Cyrillic.

The corpus of Chukchi texts is very small (approximately 30 000 sentences) and hence if we manage to model Chukchi using this corpus we will prove that our pipeline is efficient for low-resource languages.

We also had some data serving as a segmentation standard, though it was written in Latin alphabet. Table 1 shows some statistics for the data.

	sentences	words
Corpus	33331	151667
Gold standard	1006	4417

Table 1: Preprocessed corpus statistics

To use the corpus, we had to deal with several issues:

- The Chukchi writing system allows for variation in the appearance of the following two letters: *н'* = *ӈ* and *к'* = *ӄ*. The latter two symbols were introduced in 1980s (Бурыкин 2000). We needed to unify these options, so we replaced *н'* and *к'* with *ӈ* and *ӄ* respectively.

- We then removed invalid characters and fixed the ones in wrong typeset such as *C* (U+0043) and *С* (U+0421).

- Finally, we fixed the 'ʔ' signs turning them into "'/ь/ъ" in accordance with Chukchi orthography (Dunn et al. 1999, p. 58) so that *ʔA* would be *A'*.

Fixing the segmentation standard also involved these steps, though at the beginning we had to transliterate it to Cyrillic alphabet. Unfortunately, we had to review some of the sentences manually to make sure the segmentation worked correctly. Table 2 shows the statistics for the post-processed corpus.

version	changes	sentences	words
v2	*н'* and *к'*	33331	151667
v3	invalid characters	33323	151585
v4	fixing *ʔ* sign	33323	151585

Table 2: Post-processed corpus statistics

The example of changes in the data can be seen in the Table 3.

version	changes
v1	ʔаачек>∅ эты н>ин>ив>к'ин
v2	ʔаачек>∅ эты н>ин>ив>ӄин
v3	ʔаачек>∅ эты н>ин>ив>ӄин
v4	а'ачек>∅ эты н>ин>ив>ӄин

Table 3: Changes in data

One of the questions we asked ourselves was if our data fixes may be incorrect and would significantly effect the quality of the segmentation model, so we ran it using different versions; the results will be later described.

Evidently, there are not many resources to use both for segmentation training and validation, so we decided to manually validate a piece of the output of the segmentation model in order to have more data to rely on. Subsequently, the corpus segmentation data had to be put into the tensor-making model; the output of the segmentation model had to be converted from BMES format to the segmented sentences with delimiters.

4 Segmentation

The TPR model requires moderately large dataset of texts segmented into morphemes for training. Initially, we had only 1000 segmented sentences in Chukchi and that was not sufficient enough for getting any meaningful training results. To extend our training set, we obtained an unsegmented Chukchi language corpus and segmented it automatically.

To achieve any satisfactory segmentation quality, we tested several different approaches varying from rule-based to neural net based solutions. At first, we tried using an LSTM sequence-to-sequence model. We used the OpenNMT library (Klein et al. 2017), that is suitable for solving various sequence-to-sequence tasks, mainly machine translation. We took a word-level tokenized sentence as an input sequence and an arrangement of morphemes and their respective glosses as an output sequence. We used 770 examples for the training and 130 ones for evaluation. The resulting accuracy of 0.33 was, obviously, not enough to rely on this model.

Later, we tried using a rule-based approach. We discovered an in-progress project (Andriyanets and Tyers 2018) that was based on finite state transducing. We tested this tool and got the accuracy of 76.2 %, that was still not satisfactory. After that we decided, that the rule-based approach is not the best possible way to achieve what we pursue, we reformulated the task: the main goal of the segmenter was to show where are the borders between morphemes, not identify them or gloss. Considering this fact, the task was restated as character-level sequence tagging. This allowed us to use the Neural Sequence Labeling toolkit (Yang and Zhang 2018), that leveraged convolutional neural network with conditional random field based output layer. We trained the model on the train sample of 1315 tokens and tested it on the remaining 146 tokens. The model was fed words without any context, these words were treated as "sentences". Each character was assigned one of the four labels: B-MORPH, M-MORPH, E-MORPH, which stand for beginning, middle and end of morpheme. One more label is S-MORPH, that stands for a single character morpheme. The output of the model is a sequence of the aforementioned tags. We trained over 1000 epochs, the 879th of which gave the most accurate results. This model showed 91% F-1 rate for morpheme segmentation.

The final evaluation metrics are shown in Table 4:

Accuracy	Precision	Recall	F1-measure
0.9577	0.9193	0.9131	0.9162

Table 4: Segmentation evaluation stats

5 Tensor Product Representation

Tensor product is defined as follows:

Definition 1 *Let V_1 and V_2 be two vector spaces. A space W furnished with a map $(x_1, x_2) \mapsto x_1 \cdot x_2$ of $V_1 \times V_2$ into W, is called the* ***tensor product*** *of V_1 and V_2 if the two following conditions are satisfied:*

i. $x_1 \cdot x_2$ is linear in each of the variables x_1 and x_2.

ii. If $(e_i 1)$ is a basis of V_1 and $(e_i 2)$ is a basis of V_2, the family of products $e_i 1 \cdot e_i 2$ is a basis of W.

(Serre 1977, p. 8)

Definition 2 ***Representation*** *is a piece of text data mapped to a tensor of real numbers.*

Now we provide the detailed explanation of how *iiksiin* works. The first step is to generate alphabet Σ for the Chukchi corpus Σ^* and the dictionary of morpheme tensors.

We generate tensors for each morpheme $m \in \Sigma^*$ in the following way: we sum the outer product of two one-hot vectors for each symbol s_i in a morpheme m. The length of the first one-hot vector is equal to the length of the alphabet Σ. The symbol index in the alphabet stands for its position in the vector. The length of the second vector equals to the length of the morpheme m. The symbol index in the morpheme stands for its position in the vector. This is shown in the Equation (1).

(1)
$$repr(m) = \sum_{i=1}^{n} \Big(oneHot(s_i, \Sigma) \otimes oneHot(r_i, m) \Big)$$

Where:

- $oneHot$ – one hot encoding function
- $\otimes$ – tensor product (in this case equals to the outer product)
- s – symbol in the morpheme
- r – role (index of a symbol within the morpheme)
- n – number of symbols in the morpheme

Here we provide an example:

(2)
$$\begin{aligned} repr(caab \in \{a, b, c, d\}^*) = &\begin{pmatrix}0 & 0 & 1 & 0\end{pmatrix} \otimes \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix} + \\ &+ \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix} \otimes \begin{pmatrix}0 & 1 & 0 & 0\end{pmatrix} + \\ &+ \begin{pmatrix}1 & 0 & 0 & 0\end{pmatrix} \otimes \begin{pmatrix}0 & 0 & 1 & 0\end{pmatrix} + \\ &+ \begin{pmatrix}0 & 1 & 0 & 0\end{pmatrix} \otimes \begin{pmatrix}0 & 0 & 0 & 1\end{pmatrix} = \\ = \begin{pmatrix}0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0\end{pmatrix} & \end{aligned}$$

The next step is to generate tensors for each word $w \in m^*$ in the following way: we sum the tensor product of the morpheme representation $repr(m)$ from the Equation (1) and the one-hot encoded position of the morpheme in the word. So, a tensor product of a 2D-matrix and a vector gives a 3D-matrix. You can find the formula in the Equation (3) and an example in the Equation (4).

$$repr(w) = \sum_{i=1}^{n} \Big(repr(m_i) \otimes oneHot(m_i, w) \Big) \tag{3}$$

Where n is a number of morphemes in a word.

Example:

$$\begin{aligned} & repr(\{caab, bd\}) = \\ & = \begin{pmatrix} 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \end{pmatrix} \otimes \begin{pmatrix} 1 & 0 \end{pmatrix} + \begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 0 & 0 \\ 0 & 1 \end{pmatrix} \otimes \begin{pmatrix} 0 & 1 \end{pmatrix} = \\ & = \left(\begin{pmatrix} 0 & 0 \\ 1 & 0 \\ 1 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 1 \\ 0 & 0 \\ 0 & 0 \\ 1 & 0 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \begin{pmatrix} 0 & 0 \\ 0 & 1 \\ 0 & 0 \\ 0 & 0 \end{pmatrix} \right) \end{aligned} \tag{4}$$

The resulting third-rank tensors are very sparse. So, they should be converted into first-rank tensors (vectors) with a neural network-based autoencoder. For a detailed explanation of it consult (Schwartz, Tyers, et al. 2020, pp. 47–50). As a result, we obtain a vector space which we call a tensor product of representations.

6 Evaluation

To evaluate the quality of the tensor representations of natural language we have decided to train an *awd-lstm-lm* (Merity et al. 2020) language model.

This language model was chosen due to the fact that for polysynthetic languages it gives results close to the state-of-the-art and its code is freely distributed and allowed to use.

The LSTM-model was trained on characters, words and segments (with tensor representation as pre-trained embeddings) and the perplexity of each language model was measured, the results are in Table 5.

Input format	Preplexity
Character	2677.94
Word	3930.33
Segment (with pertained embeddings)	623.53

Table 5: LSTM-model performance

According to the results, the tensor representation makes a significant improvement on the language model rate of perplexity.

Results are to be analyzed and described.

7 Conclusion

In this paper we test the hypothesis that word-segment embeddings based on tensor product of representations show better performance for low-resource languages compared to conventional word- and char-based models. To prove that we developed a pipeline that allows to process low-resource polysynthetic languages. Firstly, we used Neural Sequence Labeling Toolkit (Yang and Zhang 2018) to train a segmenter on a Chukchi corpus. Later, we segmented a raw Chuckchi corpus using it. Secondly, we used *iiksiin* (Schwartz, Haley, et al. 2019-2020) to create the embeddings. After that we tested them in action and evaluated the results, which showed a notable increase in language modeling performance.

Acknoledgements

We would like to show our appreciation to the HSE expeditions which visited Chukotka and collected the corpus of texts in Chukchi, which gave us an opportunity to test the hypothesis using the corpus they made. We would also like to mention that this research was supported in part through computational resources of HPC facilities at NRU HSE ™.

A List of abbreviations

- ITER – iterative aspect
- ABL – ablative case
- AD – archaic dative
- 2.S/A.SUBJ-...-IRR-2/3SG.S – nonimperfective subjunctive mood, subject is sigular, in second person
- EMPH – emphatic clitic
- LSTM – long short-term memory network
- TPR – tensor product representation = tensor product of representations

References

Andriyanets, Vasilisa and Francis Tyers (Aug. 2018). "A prototype finite-state morphological analyser for Chukchi". In: *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*. Santa Fe, New Mexico, USA: Association for Computational Linguistics, pp. 31–40. URL: https://www.aclweb.org/anthology/W18-4804.

Bojanowski, Piotr et al. (2017). "Enriching Word Vectors with Subword Information". In: *Transactions of the Association for Computational Linguistics* 5, pp. 135–146. ISSN: 2307-387X.

Dunn, Michael John et al. (1999). "A grammar of Chukchi". In:

Ethnologue. Languages of the World. Chukchi (2020). URL: https://www.ethnologue.com/language/ckt.

Klein, Guillaume et al. (July 2017). "OpenNMT: Open-Source Toolkit for Neural Machine Translation". In: *Proceedings of ACL 2017, System Demonstrations*. Vancouver, Canada: Association for Computational Linguistics, pp. 67–72. URL: https://www.aclweb.org/anthology/P17-4012.

Merity, Stephen et al. (2020). *awd-lstm*. URL: https://github.com/neural-polysynthetic-language-modelling/awd-lstm-lm.

Mikolov, Tomas et al. (2013). "Distributed Representations of Words and Phrases and Their Compositionality". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., pp. 3111–3119. URL: http://dl.acm.org/citation.cfm?id=2999792.2999959.

Pennington, Jeffrey, Richard Socher, and Christopher D Manning (2014). "Glove: Global vectors for word representation". In: *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543.

Schwartz, Lane, Coleman Haley, et al. (2019-2020). *iiksiin*. URL: https://github.com/neural-polysynthetic-language-modelling/iiksiin.

Schwartz, Lane, Francis Tyers, et al. (2020). *Neural Polysynthetic Language Modelling*. arXiv: 2005.05477 [cs.CL].

Serre, Jean-Pierre (1977). *Linear Representations of Finite Groups.* Trans. by Leonard L. Scott. Springer-Vertag New York, Inc.

Smolensky, Paul (1990). "Tensor Product Variable Binding and the Representation of Symbolic Structures in Connectionist Systems". In: *Artificial Intelligenc*, pp. 159–216.

Tyers, Francis (n.d.). *chukchicorpus.* URL: https://gitlab.com/itml_lab/UD_Chukchi-Amguema.

Yang, Jie 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.* URL: http://aclweb.org/anthology/P18-4013.

Бурыкин, А. А. (2000). "Изучение фонетики языков малочисленных народов Севера России и проблемы развития их письменности (обзор)". In: *Язык и речевая деятельность* 3, pp. 150–180.