North Sámi morphological segmentation with low-resource semi-supervised sequence labeling

Stig-Arne Grönroos
stig-arne.gronroos@aalto.fi
Aalto University, Finland

Sami Virpioja
sami.virpioja@aalto.fi
Aalto University, Finland
Utopia Analytics, Finland

Mikko Kurimo
mikko.kurimo@aalto.fi
Aalto University, Finland

December 14, 2018

Abstract

Semi-supervised sequence labeling is an effective way to train a low-resource morphological segmentation system. We show that a feature set augmentation approach, which combines the strengths of generative and discriminative models, is suitable both for graphical models like conditional random field (CRF) and sequence-to-sequence neural models. We perform a comparative evaluation between three existing and one novel semi-supervised segmentation methods. All four systems are language-independent and have open-source implementations. We improve on previous best results for North Sámi morphological segmentation. We see a relative improvement in morph boundary F_1-score of 8.6% compared to using the generative Morfessor FlatCat model directly and 2.4% compared to a seq2seq baseline. Our neural sequence tagging system reaches almost the same performance as the CRF topline.

Tiivistelmä

Puoliohjattu sekvenssiluokitus on tehokas tapa opettaa morfologinen pilkontajärjestelmä kielelle, jolle on saatavilla niukasti lingvistisiä resursseja. Osoitamme, että generatiivisen mallin tuottamien piirteiden käyttäminen soveltuu paitsi graafisille malleille kuten ehdollinen satunnaiskenttä (CRF), myös sekvenssistä-sekvenssiin (seq2seq) -neuroverkkomalleille. Vertailemme kolmea olemassaolevaa ja yhtä uutta puoliohjattua menetelmää. Kaikki menetelmät ovat kieliriippumattomia, ja niille on avoimen lähdekoodin toteutus. Parannamme aikaisempia tuloksia pohjoissaamen morfologisen pilkonnan tehtävässä. Suhteelliset parannukset morfirajojen osumien F_1-mittaan ovat 8.6% verrattuna generatiiviseen Morfessor FlatCat -malliin ja 2.4% verrattuna seq2seq-verrokkimalliin. Ehdottamamme uusi neuroverkkomalli saavuttaa lähes saman tason kuin paras CRF-malli.

Proceedings of the fifth Workshop on Computational Linguistics for Uralic Languages, pages 15–26,
Tartu, Estonia, January 7 - January 8, 2019.

1 Introduction

Subword models have enjoyed recent success in many natural language processing (NLP) tasks, such as machine translation (Sennrich et al., 2015) and automatic speech recognition (Smit et al., 2017). Uralic languages have rich morphological structure, making morphological segmentation particularly useful for these languages. While rule-based morphological segmentation systems can achieve high quality, the large amount of human effort needed makes the approach problematic for low-resource languages. As a fast, cheap and effective alternative, data-driven segmentation can be learned based on a very small amount of human annotator effort. Using active learning, as little as some hundreds of annotated word types can be enough (Grönroos et al., 2016).

Adopting neural methods has lead to a large performance gain for many NLP tasks. However, neural networks are typically data-hungry, reducing their applicability to low-resource languages. Most research has focused on high-resource languages and large data sets, while the search for new approaches to make neural methods applicable to small data has only recently gained attention. For example, the workshop Deep Learning Approaches for Low-Resource NLP (DeepLo[1]) was arranged first time in the year of writing. Neural methods have met with success in high-resource morphological segmentation (e.g. Wang et al., 2016). We are interested to see if data-hungry neural network models are applicable to segmentation in low-resource settings, in this case for the Uralic language North Sámi.

Neural sequence-to-sequence (seq2seq) models are a very versatile tool for NLP, and are used in state of the art methods for a wide variety of tasks, such as text summarization (Nallapati et al., 2016) and speech synthesis (Wang et al., 2017). Seq2seq methods are easy to apply, as you can often take e.g. existing neural machine translation software and train it with appropriately preprocessed data. Kann et al. (2018) apply the seq2seq model for low-resource morphological segmentation.

However, arbitrary length sequence-to-sequence transduction is not the optimal formulation for the task of morphological surface segmentation. We return to formulating it as a a sequence tagging problem instead, and show that this can be implemented with minor modifications to an open source translation system.

Moreover, we show that the semi-supervised training approach of Ruokolainen et al. (2014) using feature set augmentation can also be applied to neural networks to effectively leverage large unannotated data.

2 Morphological processing tasks

There are several related morphological tasks that can be described as mapping from one sequence to another. Morphological segmentation is the task of splitting words into morphemes, meaning-bearing sub-word units. In morphological *surface* segmentation, the word w is segmented into a sequence of surface morphs, substrings whose concatenation is the word w.

$$\text{e.g.} \quad achievability \mapsto achiev \circ abil \circ ity$$

Canonical morphological segmentation (Kann et al., 2016) instead yields a sequence of standardized segments. The aim is to undo morphological processes that result in

[1] https://sites.google.com/view/deeplo18/home

allomorphs, i.e. different surface morphs corresponding to the same meaning.

$$w \mapsto y; \quad w \in \Sigma^*, y \in (\Sigma \cup \{\circ\})^*$$

$$\text{e.g.} \quad achievability \mapsto achieve \circ able \circ ity$$

where Σ is the alphabet of the language, and $\circ$ is the boundary marker.

Morphological analysis yields the lemma and tags representing the morphological properties of a word.

$$w \mapsto yt; \quad w, y \in \Sigma^*, t \in \tau^*$$

$$\text{e.g.} \quad took \mapsto take \text{ PAST}$$

where τ is the set of morphological tags.

Two related morphological tasks are reinflection and lemmatization. In *morphological reinflection* (see e.g. Cotterell et al., 2016), one or more inflected forms are given to identify the lexeme, together with the tags identifying the desired inflection. The task is to produce the correctly inflected surface form of the lexeme.

$$wt \mapsto y; \quad w, y \in \Sigma^*, t \in \tau^*$$

$$\text{e.g.} \quad taken \text{ PAST} \mapsto took$$

In *lemmatization*, the input is an inflected form and the output is the lemma.

$$w \mapsto y; \quad w, y \in \Sigma^*$$

$$\text{e.g.} \quad better \mapsto good$$

Morphological surface segmentation can be formulated in the same way as canonical segmentation, by just allowing the mapping to canonical segments to be the identity. However, this formulation fails to capture the fact that the segments must concatenate back to the surface form. The model is allowed to predict any symbol from its output vocabulary, although only two symbols are valid at any given timestep: the boundary symbol or the actual next character. If the labeled set for supervised training is small, the model may struggle with learning to copy the correct characters. Kann et al. (2018) address this problem by a multi-task training approach where the auxiliary task consists of reconstructing strings in a sequence auto-encoder setting. The strings to be reconstructed can be actual words or even random noise.

Surface segmentation can alternatively be formulated as structured classification

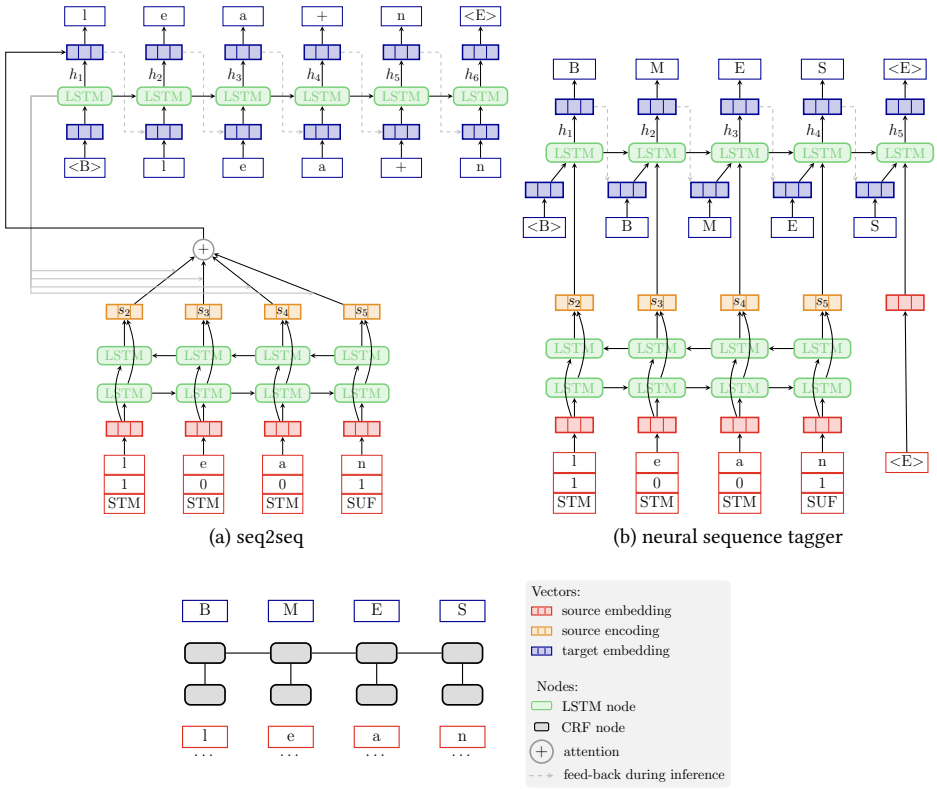
$$w \mapsto y; \quad w \in \Sigma^k, y \in \Omega^k, k \in \mathbb{N}$$

$$\text{e.g.} \quad uses \mapsto BMES$$

where Ω is the segmentation tag set. Note that there is no need to generate characters from the original alphabet, instead a small tag set Ω is used. The fact that the sequence of boundary decisions is of the same length k as the input has also been made explicit.

Different tag sets Ω can be used for segmentation. The minimal sets only include two labels: BM/ME (used e.g. by Green and DeNero, 2012). Either the beginning (B) or end (E) of segments is distinguished from non-boundary time-steps in the middle (M). A more fine-grained approach BMES[2] (used e.g. by Ruokolainen et al., 2014) uses

[2]Also known as BIES, where I stands for internal.



(a) seq2seq (b) neural sequence tagger

(c) conditional random field

Figure 1: Model architectures. To prevent the figure from becoming too large, the seq2seq model is drawn with only one LSTM layer in both encoder and decoder. The attention is only shown for the fist time step.

four labels. In addition to marking both beginning and end of segments, a special label is used for single-character (S) segments.

Morphological analysis or canonical segmentation resolve ambiguity, and are more informative than surface segmentation. Learning to resolve such ambiguity is a more challenging task to learn than surface segmentation. Surface segmentation may be preferred over the other tasks e.g. when used in an application that needs to generate text in a morphologically complex language, such as when it is the target language in machine translation. If surface segments are generated, the final surface form is easily recovered through concatenation.

To summarize, arbitrary-length sequence transduction is a formulation well suited for many morphological tasks. Morphological surface segmentation is an exception, being more appropriately formulated as sequence tagging.

3 Models for semi-supervised segmentation

Our semi-supervised training follows the approach of Ruokolainen et al. (2014). The training data consists of a large unlabeled set, and a smaller labeled training set. The labeled training set is further divided into two parts. A generative model, in our

Factor	Emb																	
character	350	b	i	e	b	m	o	r	á	h	k	a	d	e	a	m	i	s
boundary	10	0	0	0	0	0	0	1	0	0	0	0	0	1	0	0	0	1
category	10	M	M	M	M	M	M	M	M	M	M	M	M	f	f	f	f	f

Table 1: Example input factors with embedding dimension. The example word biebmoráhkadeamis is segmented as biebmo/STM ráhkad/STM eami/SUF s/SUF. Stem (STM) is abbreviated M, and suffix (SUF) is f.

case Morfessor FlatCat, is trained in a semi-supervised fashion using the first part of the labeled training set. The words in the second part of the labeled training set are segmented using the generative model. Now these words are associated with two segmentations: predicted and gold standard. A discriminative model is then trained on the second part of the labeled training set. The predictions of the generative model are fed into the discriminative model as augmented features. The gold standard segmentation is used as the target sequence.

At decoding time a two-step procedure is used: first the features for the desired words are produced using the generative model. The final segmentation can then be decoded from the discriminative model.

The idea is that the features from the generative model allow the statistical patterns found in the large unannotated data to be exploited. At the same time, the capacity of the discriminative model is freed for learning to determine when the generative model's predictions are reliable, in essence to only correct its mistakes.

3.1 Morfessor FlatCat

We produce the features for our semi-supervised training using Morfessor FlatCat (Grönroos et al., 2014). Morfessor FlatCat is a generative probabilistic method for learning morphological segmentations. It uses a prior over morph lexicons inspired by the Minimum Description Length principle (Rissanen, 1989). Morfessor FlatCat applies a simple Hidden Markov model for morphotactics, providing morph category tags (stem, prefix, suffix) in addition to the segmentation. The segmentations are more consistent compared to Morfessor Baseline, particularly when splitting compound words.

Morfessor FlatCat produces morph category labels in addition to the segmentation decisions. These labels can also be used as features. An example of the resulting 3-factor input is shown in Table 1.

3.2 Sequence-to-sequence

Our sequence-to-sequence (seq2seq) baseline model follows Kann et al. (2018) with some minor modifications. It is based on the encoder-decoder with attention (Bahdanau et al., 2014). The encoder is a 2-layer bidirectional Long Short-Term Memory (LSTM) layer (Hochreiter and Schmidhuber, 1997), while the decoder is a 2-layer LSTM. The model is trained on the character level.

Figure 1a shows the basic structure of the architecture. For simplicity a single layer is shown for both encoder and decoder.

3.3 Conditional random fields

Conditional random fields (CRF) are discriminative structured classification models for sequential tagging and segmentation (Lafferty et al., 2001). They are expressed as undirected probabilistic graphical models. Figure 1c shows the model structure. CRFs can be seen as generalizing the log-linear classifier to structured outputs. They bear a structural resemblance to hidden Markov models, while relaxing the assumption of the observations being conditionally independent given the labels.

We use the implementation of linear-chain CRFs by Ruokolainen et al. (2014)[3].

3.4 Neural sequence tagger

The encoder is a standard single-layer bidirectional LSTM. The decoder is a single-layer LSTM, which takes as input at time t the concatenation of the encoder output at time t and an embedding of the predicted label at $t-1$. There is no attention mechanism. However, the time-dependent connection to the encoder could be described as a hard-coded diagonal monotonic attention that always moves one step forward. The architecture can be seen in Figure 1b.

The most simple fixed-length decoding strategy is to forego structured prediction and instead make a prediction at each time-step based only on the encoder output s_t. The prediction at each time-step is then conditionally independent given the hidden states. We choose to instead feed the previous decision back in, causing a left-to-right dependence on previous decisions.

The proposed model has only 5% of the number of parameters of the seq2seq model (469 805 versus 8 820 037). The proposed model requires no attention mechanism, and the target vocabulary is much smaller. We also found that the optimal network size in terms of number of layers and vector dimensions was smaller.

We use factored input for the additional features. The FlatCat segmentation decision and morph category label are independently embedded. These factor embeddings are concatenated to the character embedding.

Because our human annotations include the category labels, we use a simple target-side multi-task setup to predict them in addition the the segmentation boundaries. The output vocabulary is extended to cover all combinations of segmentation decision and category label. Because our data set contains two morph categories, STM and SUF, this only increases the size of the output vocabulary from 5 (BMES + end symbol) to 10.

We use a modified beam search to ensure that the output sequence is of the correct length. This is achieved by manipulating the probability of the end symbol, setting it to zero if the sequence is still too short and to one when the correct length is reached.

The system is implemented by extending OpenNMT (Klein et al., 2017). Our implementation is open source[4].

4 North Sámi

North Sámi (davvisámegiella) is a Finno-Ugric language, spoken in the northern parts of Norway, Sweden, Finland and Russia. With around 20 000 speakers, it is biggest of the nine Sámi languages.

[3]Available from `http://users.ics.tkk.fi/tpruokol/software/crfs_morph.zip`

[4]Available from `https://github.com/Waino/OpenNMT-py/tree/same_length_decoder`

Purpose	Subset	Component	Word types	Labels
Training	Unlabeled	FlatCat	691190	No
Training	Feature train	FlatCat	200	Yes
	Main train	System	844	Yes
Development		Both	199	Yes
Testing			796	Yes

Table 2: Subdivision of data sets, with size in word types. The component column indicates which components use the data during training.

North Sámi is a morphologically complex language, featuring both rich inflection, derivation and productive compounding. It has complicated although regular morphophonological variation. Compounds are written together without an intermediary space. For example nállošalbmái ("into the eye of the needle"), could be segmented as nállo ○ šalbmá ○ i.

The morphology of Sámi languages has been modeled using finite state methods (Trosterud and Uibo, 2005; Lindén et al., 2009). The Giellatekno research lab[5] provides rule-based morphological analyzers both for individual word forms and running text, in addition to miscellaneous other resources such as wordlists and translation tools. A morphological analyzer is not a direct replacement for morphological segmentation, as there is no trivial way to map from analysis to segmentation. In addition to this, rule-based analyzers are always limited in their coverage of the vocabulary.

For an overview into the Giellatekno/Divvun and Apertium projects, including their work on Sámi languages, see Moshagen et al. (2014).

5 Data

We use version 2 of the data set collected by (Grönroos et al., 2015; Grönroos et al., 2016) as the labeled data, and as unlabeled data a word list extracted from *Den samiske tekstbanken* corpus[6].

The labeled data contains words annotated for morphological segmentation with morph category labels. The annotations were produced by a single Sámi scholar, who is not a native speaker of Sámi. In total 2311 annotated words were available. The development and test sets contain randomly selected words. The training set set of 1044 annotations is the union of 500 randomly selected words and and 597 using different active learning approaches. There was some overlap in the sets. Due to the active learning, it should be assumed that the data set is more informative than a randomly selected data set of the same size.

Table 2 shows how the data was subdivided. The unlabeled data, the development set and the test set are the same as in Grönroos et al. (2016). To produce the two labeled training sets, we first combined the labeled training data collected with different methods. From this set, 200 word types were randomly selected for semi-supervised training of Morfessor FlatCat, and the remaining 844 were used for training the dis-

[5] `http://giellatekno.uit.no/`

[6] Provided by UiT, The Arctic University of Norway.

criminative system. These two labeled data sets must be disjoint to avoid the system overestimating the reliability of the FlatCat output.

6 Training details

Tuning of FlatCat was performed following Grönroos et al. (2016). The corpus likelihood weight α was set to 1.4. The value for the annotation likelihood weight β was set using a heuristic formula optimized for Finnish:

$$\log \beta = 1.9 + 0.8 \log |\boldsymbol{D}| - 0.6 \log |\boldsymbol{A}|, \tag{1}$$

where $|\boldsymbol{D}|$ and $|\boldsymbol{A}|$ are the numbers of word types in the unannotated and annotated training data sets, respectively. Using this formula resulted in setting β to 13000. Perplexity threshold for suffixes was set to 40. For prefixes we used a high threshold (999999) to prevent the model from using them, as there are no prefixes in North Sámi.

The neural networks were trained using SGD with learning rate 1.0. Gradient norm was clipped to 5.0. Batch size was set to 64 words. Embeddings were dropped out with probability 0.3. Models were trained for at most 5000 steps, and evaluated for early stopping every 250 steps.

For the neural sequence tagger, the embedding size was 350 for characters and 10 for other input factors, and 10 for target embeddings. The encoder single bi-LSTM layer size was set to 150.

All neural network results are the average of 5 independent runs with different seeds.

7 Evaluation

The segmentations generated by the model are evaluated by comparison with annotated morph boundaries using *boundary precision*, *boundary recall*, and *boundary* F_1*-score* (see e.g., Virpioja et al., 2011). The boundary F_1-score equals the harmonic mean of precision (the percentage of correctly assigned boundaries with respect to all assigned boundaries) and recall (the percentage of correctly assigned boundaries with respect to the reference boundaries).

$$\text{Precision} = \frac{\#(\text{correct})}{\#(\text{proposed})}; \qquad \text{Recall} = \frac{\#(\text{correct})}{\#(\text{reference})} \tag{2}$$

Precision and recall are calculated using macro-averages over the words in the test set. In the case that a word has more than one annotated segmentation, we take the one that gives the highest score.

In order to evaluate boundary precision and recall, a valid segmentation is needed for all words in the test set. The seq2seq model can fail to output a valid segmentation, in which case we replace the output with the input without any segmentation boundaries. To include an evaluation without this source of error we also report word type level accuracy. A word in the test set is counted as correct if all boundary decisions are correct. Output that does not concatenate back to the input word is treated as incorrect.

system	Pre	Rec	F_1	w-acc
FlatCat (200 words)	78.20	77.60	77.90	57.20
Seq2seq (s)	86.94	78.62	82.54	64.60
NST (s)	83.26	83.92	83.58	69.12
CRF (s)	**87.70**	83.30	85.40	69.30
FlatCat (full)	74.30	84.10	78.90	61.80
Seq2seq (ss)	87.66	80.16	83.72	68.36
NST (ss)	84.28	**85.58**	84.94	71.02
CRF (ss)	86.30	85.20	**85.70**	**71.10**

Table 3: Results on the test set. Boundary precision (Pre), recall (Rec), and F_1-scores, together with word-type level accuracy (w-acc). NST is short for neural sequence tagger. FlatCat (200 words) shows the performance of the FlatCat system used to produce the input features. FlatCat (full) line shows FlatCat trained using the full training set. Fully supervised models, i.e. without using FlatCat features, are marked (s). Semi-supervised models are marked (ss).

	STM	STM + STM			STM + SUF			STM + SUF + SUF		
	Pre	Pre	Rec	F_1	Pre	Rec	F_1	Pre	Rec	F_1
FlatCat (200)	71.50	78.50	70.90	74.50	77.60	69.90	73.50	78.90	56.70	66.00
Seq2seq (s)	82.02	89.82	66.54	76.08	88.22	74.78	80.94	81.44	56.10	66.32
NST (s)	76.74	83.84	78.20	80.90	86.04	79.82	82.82	80.30	58.34	67.58
CRF (s)	79.80	95.50	60.00	73.70	**89.40**	**82.70**	**85.90**	83.30	62.70	71.60
FlatCat (full)	62.70	82.50	**92.70**	87.30	76.70	76.40	76.60	72.90	61.90	67.00
Seq2seq (ss)	**82.46**	91.08	68.00	77.80	88.46	76.46	82.00	84.74	57.60	68.56
NST (ss)	78.78	87.90	86.56	87.20	85.28	80.12	82.60	78.20	59.54	67.58
CRF (ss)	77.20	**96.40**	85.50	**90.60**	86.60	79.40	82.80	**88.60**	**67.90**	**76.90**

Table 4: Boundary precision (Pre), recall (Rec), and F_1-scores for different subsets of the evaluation data. NST stands for Neural sequence tagger. (s) stands for fully supervised, (ss) for semi-supervised.

8 Results

Table 3 shows results on the full test set. The semi-supervised CRF shows the best performance both according to F_1-score and word-type level accuracy. Semi-supervised seq2seq has high precision but low recall, indicating under-segmentation. The neural sequence tagger shows the opposite behavior, with the highest recall.

All semi-supervised methods improve on the quality of the semi-supervised FlatCat trained on 200 annotated words which is used as input features. All three discriminative methods also outperform FlatCat trained on the whole training set, on F_1 and accuracy. All three semi-supervised methods outperform their fully supervised variants. These results show that two-step training is preferable over using only Morfessor FlatCat or one of the discrinative methods.

The seq2seq model frequently fails to output a valid segmentation, either generating incorrect characters, stopping too early, or getting stuck repeating a pattern of characters. For 10.7% of the test set, the seq2seq output does not concatenate back to the input word.

Table 4 shows results for subsets of the evaluation data. The subsets include all words were the gold standard category labels follow a particular pattern: No internal structure (STM), uninflected compound (STM+STM), single-suffix inflected word (STM+SUF) and two-suffix inflected word (STM+SUF+SUF).

The seq2seq model has the best performance for the STM-pattern. This is only partly explained by the bias towards not segmenting at all caused by the replacement procedure for the invalid outputs.

The seq2seq model has high precision for all category patterns. Fully supervised CRF has superior precision and recall for the STM+SUF pattern, while semi-supervised CRF is superior for the STM+SUF+SUF pattern. CRF is good at modeling the boundaries of suffixes. Adding the FlatCat features improves the modeling of the boundary between multiple suffixes, while slightly deteriorating the modeling of the boundary between stem and suffix. The left-to-right decoding is a possible explanation for the weaker performance of the neural sequence tagger on the STM+SUF+SUF pattern. Fully supervised CRF is poor at splitting compound words, evidenced by the low recall for the STM+STM pattern. This deficiency is effectively alleviated by the addition of the FlatCat features.

The neural sequence tagger is good at modeling the ends of stems, indicated by high recall on the STM+STM and STM+SUF patterns.

9 Conclusions and future work

Semi-supervised sequence labeling is an effective way to train a low-resource morphological segmentation system. We recommend training a CRF sequence tagger using a Morfessor FlatCat-based feature set augmentation approach. This setup achieves a morph boundary F_1-score of 85.70, improving on previous best results for North Sámi morphological segmentation. Our neural sequence tagging system reaches almost the same word-type level accuracy as the CRF system, while having better morph boundary recall.

The bidirectional LSTM-CRF model (Huang et al., 2015) uses the power of a recurrent neural network to combine contextual features, and stacks a CRF on top for sequence level inference. The performance of this architecture on the North Sámi morphological segmentation task should be explored in future work.

Acknowledgments

This research has been supported by the European Union's Horizon 2020 Research and Innovation Programme under Grant Agreement No 780069. Computer resources within the Aalto University School of Science "Science-IT" project were used.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. In *ICLR15*.

Ryan Cotterell, Christo Kirov, John Sylak-Glassman, David Yarowsky, Jason Eisner, and Mans Hulden. 2016. The sigmorphon 2016 shared task—morphological reinflection. In *Proceedings of the 14th SIGMORPHON Workshop on Computational Research in Phonetics, Phonology, and Morphology*. pages 10–22.

Spence Green and John DeNero. 2012. A class-based agreement model for generating accurately inflected translations. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, pages 146–155.

Stig-Arne Grönroos, Katri Hiovain, Peter Smit, Ilona Erika Rauhala, Päivi Kristiina Jokinen, Mikko Kurimo, and Sami Virpioja. 2016. Low-resource active learning of morphological segmentation. *Northern European Journal of Language Technology* .

Stig-Arne Grönroos, Kristiina Jokinen, Katri Hiovain, Mikko Kurimo, and Sami Virpioja. 2015. Low-resource active learning of North Sámi morphological segmentation. In *Proceedings of 1st International Workshop in Computational Linguistics for Uralic Languages*. pages 20–33.

Stig-Arne Grönroos, Sami Virpioja, Peter Smit, and Mikko Kurimo. 2014. Morfessor FlatCat: An HMM-based method for unsupervised and semi-supervised learning of morphology. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics*. Association for Computational Linguistics, Dublin, Ireland, pages 1177–1185.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation* 9(8):1735–1780.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF models for sequence tagging. *CoRR arXiv:1508.01991* .

Katharina Kann, Ryan Cotterell, and Hinrich Schütze. 2016. Neural morphological analysis: Encoding-decoding canonical segments. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 961–967.

Katharina Kann, Manuel Mager, Ivan Meza, and Hinrich Schütze. 2018. Fortification of neural morphological segmentation models for polysynthetic minimal-resource languages. In *Proceedings of NAACL 2018*. Association for Computational Linguistics, New Orleans, Louisiana, USA.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M. Rush. 2017. OpenNMT: open-source toolkit for neural machine translation. In *Proc. ACL*.

John Lafferty, Andrew McCallum, and Fernando CN Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of ICML*. pages 282–289.

Krister Lindén, Miikka Silfverberg, and Tommi Pirinen. 2009. HFST tools for morphology–an efficient open-source package for construction of morphological analyzers. In *State of the Art in Computational Morphology*, Springer, pages 28–47.

Sjur Moshagen, Jack Rueter, Tommi Pirinen, Trond Trosterud, and Francis M. Tyers. 2014. Open-source infrastructures for collaborative work on under-resourced languages. In *Collaboration and Computing for Under-Resourced Languages in the Linked Open Data Era. LREC 2014.*. pages 71–77.

Ramesh Nallapati, Bowen Zhou, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *in Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. pages 280–290.

Jorma Rissanen. 1989. *Stochastic Complexity in Statistical Inquiry*, volume 15. World Scientific Series in Computer Science, Singapore.

Teemu Ruokolainen, Oskar Kohonen, Sami Virpioja, and Mikko Kurimo. 2014. Painless semi-supervised morphological segmentation using conditional random fields. In *European Chapter of the Association for Computational Linguistics (EACL)*. Association for Computational Linguistics, Gothenburg, Sweden, pages 84–89.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2015. Neural machine translation of rare words with subword units. In *ACL16*.

Peter Smit, Sami Virpioja, Mikko Kurimo, et al. 2017. Improved subword modeling for WFST-based speech recognition. In *In INTERSPEECH 2017–18th Annual Conference of the International Speech Communication Association.*.

Trond Trosterud and Heli Uibo. 2005. Consonant gradation in Estonian and Sámi: two-level solution. In *Inquiries into Words, Constraints and Contexts–Festschrift for Kimmo Koskenniemi on his 60th Birthday*, Citeseer, page 136.

Sami Virpioja, Ville T Turunen, Sebastian Spiegler, Oskar Kohonen, and Mikko Kurimo. 2011. Empirical comparison of evaluation methods for unsupervised learning of morphology. *Traitement Automatique des Langues* 52(2):45–90.

Linlin Wang, Zhu Cao, Yu Xia, and Gerard de Melo. 2016. Morphological segmentation with window LSTM neural networks. In *AAAI*. pages 2842–2848.

Yuxuan Wang, RJ Skerry-Ryan, Daisy Stanton, Yonghui Wu, Ron J Weiss, Navdeep Jaitly, Zongheng Yang, Ying Xiao, Zhifeng Chen, Samy Bengio, Q Le, Y Agiomyrgiannakis, R Clark, and R. A. Saurous. 2017. Tacotron: Towards end-to-end speech synthesis. In *Proc. Interspeech*. pages 4006–4010.