University of Groningen

Faculty of Science and Engineering

Morphological Segmentation of Polysynthetic Languages for Neural Machine Translation: The Case of Inuktitut

Author:

Christian Roest

Artificial Intelligence MSc

Internal supervisor:

dr. J. K. Spenader

Faculty of Science and Engineering

External supervisor:

dr. A. Toral Ruiz

Faculty of Arts

Groningen, 2020

Abstract

Segmentation of words into sub-word units is a crucial preprocessing step of modern machine translation systems, which allows the translation of unseen and rare words. It is still largely unknown how existing methods perform on a category of languages with a much higher degree of morphological complexity, called polysynthetic languages. Characteristic for polysynthetic languages are long sentence-words, that consist of many morphemes. These long words are a result inflection and agglutination, which allow words to carry a much more detailed meaning than words in most other languages can. We hypothesise that, to deal with such complex languages, translation systems require a robust segmentation method to isolate meaningful parts of a word accurately and consistently. The current state-of-the-art of language-agnostic segmenters were not designed with polysynthetic languages in mind, which begs the question whether they can provide adequate performance for these languages.

In this thesis, various methods are compared on their ability to generate linguistically correct segmentations, and their ability to produce quality translations as a part of a state-of-the-art neural machine translation system for the low-resource polysynthetic language Inuktitut. With the results we determine whether linguistically correct segmentation is beneficial for the translation performance, and how it affects the translation model's generalization ability.

Contents

1 Introduction **3**

1.1 Research questions 5

2 Morphological segmentation **8**

2.1 Challenges of polysynthetic languages 9

2.2 Inuktitut 11

2.2.1 Nunavut Hansard 13

2.3 Segmentation methods 14

2.3.1 Morfessor Baseline 14

2.3.2 Morfessor Categories-MAP 15

2.3.3 Morfessor FlatCat 17

2.3.4 Byte-Pair Encoding 17

2.3.5 Linguistically-Motivated Vocabulary Reduction 19

2.3.6 Neural segmentation 19

3 Intrinsic evaluation **22**

3.1 Experimental setup 22

3.2 Segmentation methods 24

3.3 Results 29

4 Machine Translation **38**

4.1 Rule-based Machine Translation 38

4.2 Statistical Machine Translation 40

4.3 Neural Machine Translation 41

4.3.1 Encoder-decoder models 42

4.3.2 Attention mechanism 44

4.3.3 Transformer architecture 46

4.4 Segmentation for MT 48
4.5 Low-resource NMT 50
4.5.1 Monolingual data 50
4.5.2 Domain adaptation 51
4.6 Evaluation metrics for MT 54
4.6.1 BLEU 55
4.6.2 Character-F 56

5 Extrinsic Evaluation **57**
5.1 Preprocessing 57
5.1.1 Normalization of Inuktitut syllabics 57
5.1.2 Romanization 58
5.1.3 Further preprocessing 58
5.2 Experimental setup 58
5.3 Results 60
5.3.1 Inuktitut to English 60
5.3.2 English to Inuktitut 65

6 Conclusion **68**

Bibliography **72**

List of Figures **78**

Chapter 1

Introduction

The objective in Machine Translation (MT) is to translate text between two languages automatically without the need for human assistance.

Segmentation of words into smaller parts called *sub-words* is a crucial step in the preprocessing pipeline of a modern MT system. In essence, this step aims to 'reverse' the word formation processes by which morphemes are combined to form a word. The derived segmentation provides the model with morphological information, that is, information about the meaningful parts of which a word consists. It allows the translation model to learn meaning at a lower level than words, thereby enabling it to infer the meaning of unseen words that consist of segments that were present in training data.

The difficulty to model a language computationally has been shown to correlate with the morphological complexity of a language, suggesting that translation models may benefit from improved segmentation methods that provide the model with more effective morphological information (Mielke et al., 2019). Segmentation has received a lot of attention from researchers in recent years, as it is one of the more tangible preprocessing optimizations that can have a large impact on the performance of MT systems. Past studies have made comparisons between segmentation models for a variety of languages, but it is still unknown how these methods hold up against a category of languages with a much higher degree of morphological complexity, called

polysynthetic languages.

Characteristic for polysynthetic languages are long words, that consist of many morphemes. These long words are a result inflection and agglutination, which allow words to carry a much more detailed meaning than words in most other languages can. In some cases single words can be equivalent in meaning to a full sentence in English. To deal with such languages, MT systems require a robust segmentation method to isolate meaningful parts of the word accurately and consistently. The current state-of-the-art of language-agnostic segmentation approaches were not designed with polysynthetic languages in mind, which begs the question if they can deliver an adequate performance for this challenging category of languages.

In recent years, the focus of MT research has shifted from statistical models (SMT) to systems leveraging large neural networks and deep learning architectures that can automatically extract linguistic rules from large amounts of data, Neural Machine Translation (NMT) (Kalchbrenner & Blunsom, 2013; Vaswani et al., 2017). Because NMT and SMT models do not have explicitly programmed rules, they are entirely dependent on training data for deriving implicit rules. Training such models requires millions of quality sentence pairs in order to reach a good translation quality and coverage. For a select group of language pairs, such as English–German, plenty of parallel data is available. But for polysynthetic languages, and many others, this is not the case. Such a low resource setting increases the importance of careful data preprocessing to provide the little available data in the optimal way, to make it easier for the model to recognize patterns.

Another important factor that impacts the performance of an MT system is the *domain* of the training data, which includes features such as the topic, writing style, and source of a dataset. The domain of the data on which a system is trained affects the ability of such a system to generate quality translations for a different domain. Preferably, the data used to train the system is of the same domain as the data which the system will be applied to. The domain of the training data can also introduce a bias to segmentation methods, as they are usually trained on the same data used for training the translation system.

This work aims to shed light on the challenges for existing segmentation methods with polysynthetic languages. The methods are compared on their ability to produce linguistically correct segmentations, and their ability to produce quality translations as a part of an end-to-end NMT system. Additionally, we investigate whether the choice of segmentation influences the ability of the MT system to generalize to a different domain. We also introduce a novel method that is not restricted by the assumptions that hold for less morphologically rich languages.

We compare the methods on Inuktitut, a polysynthetic language spoken by approximately 263,840 people in the Canadian territory of Nunavut. In 2020 the Nunavut Hansard Parallel Corpus 3.0 was released, which contains approximately 1.3 million aligned sentence pairs of English (EN) and Inuktitut (IU). As a polysynthetic language, Inuktitut is morphologically much more complex than other languages for which substantial corpora are available. As such, Inuktitut provides a challenge for existing segmentation methods, as well as for current MT architectures.

1.1 Research questions

This study focuses on the performance of segmentation approaches, both as a component of an IU↔EN NMT system (extrinsic evaluation), and as a separate task where the correctness of the segmentation itself is assessed (intrinsic evaluation). By comparing various methods on the Nunavut Hansard corpus we try to derive which segmentation methods are effective for polysynthetic languages. With the obtained results, this work aims to find answers to the following research questions (RQs):

1. Which approaches yield the most linguistically correct segmentation (intrinsic evaluation)?
2. Which segmentation approaches lead to the best translation quality (extrinsic evaluation)?
3. Will a more linguistically correct segmentation lead to better translation quality?

4. Is there a difference in the effect of the segmentation between IU→EN and EN→IU?

5. How does the choice of segmentation method influence the system's ability to generalize to data of a different domain?

For RQ1, we perform intrinsic evaluation on the morphological segmentations resulting from various methods based on an existing set of gold-standard segmentations, to assess the quality of the generated segmentation. In line with Kann et al. (2018) we compare methods on the word accuracy and Border-F1, in addition to the boundary precision and recall. By answering this RQ1, we try to derive which segmentation approaches are suitable for dealing with the complex morphology of polysynthetic languages like Inuktitut.

For RQ2 we compare the different segmentation approaches as a component of a translation model EN↔IU. The performance of each system is measured using BLEU (Papineni et al., 2002) and chrF (Popović, 2015) scores. The model parameters and preprocessing steps are kept identical between comparisons. By answering this research question we learn which method provides the most effective morphological information to the translation model.

With the results obtained from experiments performed for RQ1 and RQ2, we will be able to address RQ3. Depending on the correlation between the results of the intrinsic and extrinsic evaluation, we learn whether translation models for polysynthetic languages benefit from more linguistically accurate segmentation. Although it seems intuitively likely that the more linguistically correct segmentation will give the model more accurate information, it is possible that there are other characteristics of a segmentation method that is more beneficial to a translation model than the intrinsic quality of the segmentation, such as the vocabulary size (Ataman et al., 2017).

To derive an answer to RQ4, we assess the performance when segmentation is applied when Inuktitut is the source language, versus when Inuktitut is the target language. We train and rank NMT models for both directions to determine whether certain

approaches are more effective to use on the source side than on the target side of a translation model.

To answer RQ5, we compare each of the NMT systems trained for extrinsic evaluation on a test set originating from the Hansard corpus and on a test set originating from an Inuktitut news website. The news data is *out-of-domain* with respect to the training data, which allows us to determine whether there is an effect of the segmentation method on the generalization ability of an NMT system.

For each of the segmentation methods and translation directions we also train an additional system with a small amount of news data in addition to the Hansard training data. This allows us to determine to which degree the choice of segmentation method influences the system's domain adaptation capability.

Chapter 2

Morphological segmentation

Morphological segmentation aims to break down words into meaningful *sub-word units* or *morphs*. Morphs are the surface forms of morphemes, which refers to the smallest meaningful units of a word. The research field of morphological segmentation is a sub-field of morphology, which studies how morphemes in a language are combined to form words, and the processes that influence word formation.

The optimal solution for this task would be to perfectly segment the word into its constituent parts. The earliest work on morphological segmentation was by Harris (1955) who devised a procedure to identify likely morpheme candidates in words by using their phonemic representations. The morphemes derived using this procedure could then be used to perform morphological analysis. In line with Harris (1955) subsequent research on morphological segmentation has mainly focused on developing *unsupervised* procedures, meaning that they are derived solely based on their occurrences in text, without the use of labelled data or by providing known morphemes (Ruokolainen et al., 2016).

The advantage of approaches that work in an unsupervised manner is that they can be applied to any language for which text data is available–especially because reference data can be difficult to obtain for languages with few speakers. Another aspect of unsupervised procedures for morphological segmentation that makes it interesting for linguistic research is the parallel to language acquisition in children.

Morphological segmentation algorithms have been applied successfully as an NLP tool for a variety of tasks over the years, such as speech recognition, information retrieval, and machine translation (Ruokolainen et al., 2016).

The remainder of this chapter serves as an overview for morphological segmentation with a focus on polysynthetic languages. The challenges that polysynthetic languages pose for the task of morphological segmentation are presented in Section 2.1. The language of Inuktitut, the polysynthetic language used in the experiments in this work, is described in Section 2.2. Finally, the various segmentation methods that are compared in this work, are discussed in Section 2.3.

2.1 Challenges of polysynthetic languages

Polysynthetic languages are languages that are highly synthetic, which means that these languages exhibit a high degree of morphological compexity. The morphological processes involved in word formation of a language determine the rules for combining morphemes into words, and how information can be added or altered in words by changing the form of the word.

The morphological complexity of polysynthetic languages can be attributed to two main morphological processes, agglutination and inflection. *Agglutination* is an example of a morphological process that involves the concatenation of morphemes to alter the meaning of a word, or to add information. The concatenated morphemes each have with an isolated meaning, such that there is a one-to-one relation between morpheme and meaning. Morphemes that are concatenated by agglutination have morpheme boundaries that are easy to identify. *Inflection* on the other hand alters the meaning of a word by changing the surface form of a morpheme or word, which makes it difficult to identify the boundaries between morphemes in a word (Ruokolainen et al., 2016). Inflectional changes can affect the meaning of a morpheme significantly, for example relative to tense, gender, or mode of action. Irregular inflectional changes can be especially challenging for segmentation methods, because they deviate from the regular rules which a segmentation method attempts to adapt

English	Yimas	Decomposition
a) I saw them.	pukatay	pu + ka + tay
b) They saw me.	puNatay	pu + Na + tay

Table 2.1: An example of inflectional changes influencing the meaning of a word in Yimas, a polysynthetic language spoken in Papua New Guinea. The inflectional change of the second morpheme changes the case from the singular nominative case in example a, to a singlular accusative case in example b. Source: Phillips, 1994, example 2.

to.

While many other languages also incorporate agglutination or inflection, polysynthetic languages exhibit both of these processes to a much higher degree. The combination results in very long words, often consisting of five or more morphemes. By our estimation, Inuktitut has an average of around 4.39 morphemes per word[1]. Greenlandic, another polysynthetic language similar to Inuktitut, has an average 3.72 morphemes per word (J. Greenberg conducted in 1960 (as cited by Mithun, 2009)). In turn, this means that morphological segmentation of such languages involves the correct placement of four or more segmentation boundaries, which is inherently a more challenging task. The words also simply consist of more characters, which leads to more possible positions where segmentation boundaries can be placed, which makes the hypothesis space for the task much larger for polysynthetic languages.

As a result of the mentioned word-formation processes, single words carry a lot of information, and can be used to express a significantly more detailed concept than words in other languages can (Mager et al., 2018). Table 2.2 shows an example of how polysynthetic languages can express detailed concepts with single words.

A further complicating factor in the morphology of polysynthetic languages is that the surface form of a morpheme can change based on the presence of other morphemes as a result of euphonic inflection, or by the application of morphophonolog-

[1]Based on rule-based segmentation of words in the Hansard corpus.

Yupik	tuntussurqatarniksaitengqiggteuq
Decomposition	tuntu + ssur + qatar + ni + ksaite + ngqiggte + uq
English	He had not yet said again that he was going to hunt reindeer.

Table 2.2: As a result of the combination of word formation processes in the polysynthetic language Yupik, single words can be used to express concepts that would require a full sentence in English.

	English	**Inuktitut**	**Decomposition**
a)	I have an ulu	uluqaqtunga	ulu + qaq + ulu
b)	Does he have a beard?	umiqaqqa	umi<u>k</u> + qaq + qa
c)	He doesn't have a beard.	umiqanngittuq	umi<u>k</u> + qa<u>q</u> + nngit + tuq

Table 2.3: Example of subsequent morphemes influencing the surface form of the previous morpheme in Inuktitut. The underlined characters get removed because both 'qaq' and 'nngit' delete any consonant that appears immediately before them[2].

ical rules, as illustrated in Table 2.3. While this again is not exclusive to polysynthetic languages, it does affect polysynthetic languages more strongly due to the high morpheme-to-word ratio.

2.2 Inuktitut

Inuktitut is a polysynthetic language that is spoken by Inuit in the Nunavut region of Canada. The Nunavut territory is the largest territory of Canada. At the time of writing, it has a population of 35,944. Inuktitut is recognized as an official language in the region of Nunavut alongside English, French and Inuinnaqtun. The language is a member of the Inuit-Yupik-Unangan language family (Joanis et al., 2020). More specifically, it is included in the group of dialects that are spoken in the Nunavut territory called Inuktut (not to be confused with Inuktitut), which also includes Inuinnaqtun.

There are two official scripts for Inuit languages. It can be written either using Latin script, or using Inuktitut syllabics. Each symbol in the syllabics script represents a combination of *consonant + vowel*. The consonants that can be included in these syllables are g, j, k, l, m, n, p, q, r, s, t, v, ng, or ł. And the vowel part of the syllable can be either a, i, u, or ai. The vowel or consonant part of the syllable can also be absent (Figure 2.1). The syllabics script does not contain any capitalized versions of the characters, while the Latin script version handles capitalization similar to English. The two scripts can be transliterated between without loss of information, although certain equivalent spelling choices in Latin scripts can get lost when transliterating to the syllabics script. Inuktitut syllabics script is incorporated in the Unicode standard since version 3.0, in the Unified Canadian Aboriginal Syllabics Unicode block. This means that Inuktitut syllabics characters can be encoded using it.

Figure 2.1: The Inuktitut syllabics alphabet. Syllabics characters with a small dot above them represent the long version of the syllable, whereas the characters without it represent the short version of the syllables. The displayed Latin transcriptions correspond to the short version of the vowel, for long versions the vowel is doubled. The syllabics in the right-most column correspond to the syllables with an absent consonant or vowel.

The Inuktitut language involves a high degree of morphological complexity. Words are formed by appending a number of affixes to roots. As a result of the complex

word formation processes, the morphemes can take on many different surface forms, and are therefore difficult to identify.

2.2.1 Nunavut Hansard

All transcripts of the Nunavut Legislative Assembly, the deliberative authority in the Canadian territory of Nunavut, are made available in both English and Inuktitut by the regional government. The newly released Nunavut Hansard parallel corpus 3.0[3] is the third installment of parallel corpora based upon the proceedings of the Legislative Assembly of Nunavut. The corpus contains 1.3 million aligned sentence pairs of Inuktitut and English, making it the largest parallel corpus available for any polysynthetic language at the time of writing. The details of the text extraction and sentence alignment process are described in Joanis et al. (2020). The proceedings are publicly available on the website of the assembly[4], in the form of individual PDF documents per session and language.

The content of the corpus covers the proceedings of 687 sessions from April 1999 to June 2017. The policy of the editors of the Nunavut Hansard "is to provide a verbatim transcript with a minimum of editing and without any alteration of the meaning of a Member's speech."[5]

The Inuktitut part of the corpus is written in Inuktitut syllabics script, and contains approximately 1.5 million word types (unique words), across 8,067,977 words. This gives a type-to-token ratio (TTR) of 0.18. On the English side, there are approximately 50,000 word types, across approximately 17,330,271 total words, resulting in a TTR of only 0.003.

[3]https://doi.org/10.4224/40001819
[4]https://assembly.nu.ca/hansard
[5]https://assembly.nu.ca/hansard

2.3 Segmentation methods

A variety of segmentation methods have been developed which are based upon different theories, and optimize different criteria. The available methods include supervised, unsupervised, semi-supervised, and rule-based methods. Unsupervised methods are trained using only unannotated data, whereas supervised methods are trained using only annotated data. In the context of morphological segmentation, annotated data refers to already segmented words, and unannotated data refers to unsegmented words. Semi-supervised methods on the other hand can learn from both unannotated data and annotated data. Usually these methods still rely mostly on unannotated data, because annotated data is difficult to obtain in large volumes (Ruokolainen et al., 2016). In particular for polysynthetic languages this can be challenging, because they have very few speakers and are largely undocumented (Littell et al., 2018). Rule-based methods are not trained, and instead produce segmentations by following a pre-programmed set of rules.

The choice of segmentation method is determining for the vocabulary of the translation model, the number of tokens per line, and the frequencies of morphs in the training data, each of which can have a big impact on the performance of the model. Most trainable methods therefore offer a number of configurable parameters to directly or indirectly influence the mentioned properties.

2.3.1 Morfessor Baseline

Morfessor Baseline (Creutz & Lagus, 2002) is a segmentation approach that performs unsupervised segmentation by building a lexicon of morphs, which are described as elements that comprise words. While the authors mention that it is not the intention for morphs to exactly replicate the morphemes of a word, morphemes naturally tend to become incorporated by Morfessor models as morphs as a result of training. This method aims to minimize a two-part cost function based on the Minimum Description Length criterion (Rissanen, 1978), which is in essence a criterion that rewards both simple and descriptive hypotheses. In the context of segmentation, it

tries to keep the codebook of morphs as small as possible, while also minimizing the amount of morphs needed to code the training corpus. The search algorithm starts with all words unsegmented. During each epoch, it considers all possible splits for each input word. For each word, the best split is selected and performed depending on the associated cost. When not further splitting a word yields the best cost, then the word is not split. Training finishes when the decrease in cost between epochs is smaller than a provided threshold value. It is also possible to provide a maximum number of epochs.

The initial implementation of the Morfessor Baseline algorithm (Creutz & Lagus, 2002) supported only unsupervised training, but later an extension was made to support semi-supervised training (Kohonen et al., 2010). The addition of the semi-supervised option was intended to alleviate bias problems that arise when training segmentation models with unsupervised training. Unsupervised methods may for example be biased towards language types that have a higher or lower amount of morphemes per word, leading to over- or under-segmentation. By making use of the additional annotated data, the model can be better adjusted to prevent this. We suspect that over- and under-segmentation could be a problem for polysynthetic languages, because the words consist of many characters, which increases the possibility space for segmentation. For most languages, annotated segmentation data is not available in large amounts, but often a small amount can be obtained by manual decomposition by a natural speaker, or from existing systems. The cost function of the semi-supervised implementation contains an additional component for the likelihood of the labelled data, with an associated weight β. By continually adjusting the weights of each component of the cost function based on a separate validation set during training, the model is able to learn to apply the right amount of segmentation from little labelled data relative to the unlabelled data.

2.3.2 Morfessor Categories-MAP

The Morfessor Categories-MAP (Creutz & Lagus, 2005) algorithm also attempts to optimize the morph lexicon by keeping it as concise as possible, while also repre-

senting the corpus by as few morphs as possible. It is an extension to the Morfessor Baseline algorithm (Section 2.3.1) that uses maximum a posteriori (MAP) estimation to optimize the structure in terms of accuracy and model complexity. In a comparison study between Categories-MAP and three other algorithms, Creutz and Lagus (2005) show Morfessor Categories-MAP to perform very well on English and Finnish, another highly inflectional language, outperforming Morfessor Baseline.

The Morfessor Categories-MAP algorithm reanalyzes segmentations produced by the Morfessor Baseline algorithm by representing each word type present in the output as a Hidden Markov Model (HMM) of morph categories. Although the categories assigned to the morphs are not directly relevant for the task of morphological segmentation, the resulting constraints are hypothesized to help with identifying the right morphs. The four possible morph categories are *prefix*, *suffix*, *stem* and *non-morph*, which impose certain morphotactical constraints on the sequence; prefixes cannot occur at the end of a word, and suffixes cannot occur at the start of a word. An additional constraint is that a suffix cannot follow a prefix directly, but requires a stem in between.

In Inuktitut words are exclusively formed by appending affixes to a single root[6], which is a structure that is supported by Morfessor Categories-MAP's category constraints. However, misclassification of a morph category could be problematic, because the constraints could prevent the generation of the correct subsequent morphs.

The algorithm is designed to deal with under and oversegmentation. In the case of undersegmentation, word types are segmented by identifying morphs in the morph lexicon that can be combined to form the word. The new segmentation for that word is then made using those existing morphs. To combat oversegmentation, words consisting of too many short morphs are identified, and its morphs become categorized under the the *non-morph* category, which will be joined with neighbouring morphs to create longer morphs.

For the morphological segmentation of polysynthetic languages this could be problematic, because there is a large variety in the lengths of morphs. Morphemes

[6] tusaalanga.ca

in Inuktitut commonly consist of between 2 and 8 characters. I anticipate that Morfessor Categories-MAP's strategy to combat oversegmentation could potentially lead to undersegmentation of short morphemes and oversegmentation of long morphemes.

2.3.3 Morfessor FlatCat

Morfessor FlatCat (Grönroos et al., 2014) is another variant of the Morfessor algorithm which uses a flat lexicon as opposed to the hierarchical lexicon used by Morfessor Categories-MAP. The basic approach of Morfessor FlatCat is similar to Morfessor Categories-MAP; it models words as a Hidden Markov Model (HMM) of morphs that fall in one the four categories *prefix*, *suffix*, *stem* and *non-morph*. In contrast to Morfessor Categories-MAP, it is possible to use annotated data with FlatCat for semi-supervised learning similar to the semi-supervised extension to Morfessor Baseline (Kohonen et al., 2010).

The approach has been found to perform well on information retrieval tasks, which can be attributed to the high level of consistency in the segmentations and relatively small morph lexicon as a result of a more structured segmentation (Grönroos et al., 2014).

2.3.4 Byte-Pair Encoding

Byte-Pair Encoding (BPE) (Sennrich et al., 2016b) is an unsupervised segmentation method that was developed to improve translation using NMT models, and is currently widely used in state-of-the-art systems for this purpose. The method uses a straightforward approach which contrary to the previously described methods puts no emphasis on producing linguistically correct morphological segmentations.

The algorithm starts by splitting each word in the training data into characters, which become the initial morphs. The most frequent pair of neighbouring morphs is then identified and merged into a single morph in each word where it occurs.

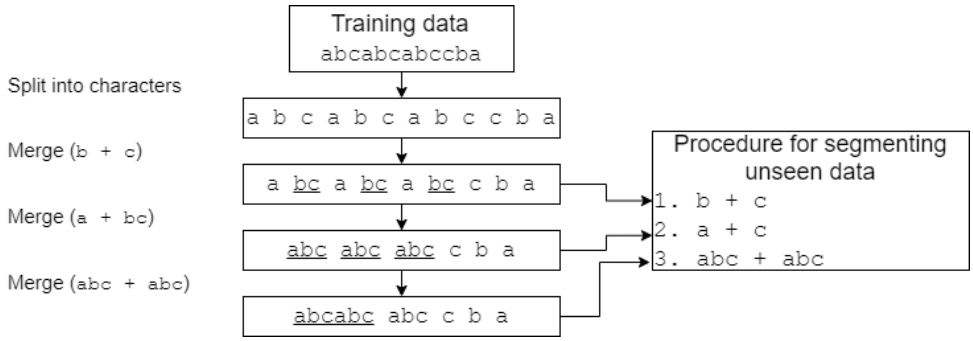

Figure 2.2: An example of how a BPE model is trained on the string "abcabcabccba". The string is initially split into characters, after which the most frequent pair of morphs is merged. The order in which the merges are done is stored as a procedure to be applied to unseen data.

This newly formed morph is added to the morph lexicon, while also keeping the morphs it consists of. With the data in which the pair is now merged, the next most frequent pair of neighbouring morphs is identified and merged into a single morph. This process repeats for a number of iterations specified by the user, or until there are no more pairs that occur more than once. The order in which the morphs are merged based on the training data is stored and can then be used as a procedure for segmenting new data in the same way. A simple example of a BPE model creating such a procedure on a training set of a single string can be seen in Figure 2.2. This allows the user to train a BPE model on training data, and apply it to unseen data to produce a segmentation that only contains morphs that are present in the segmented training data.

A big advantage of the approach for its implementation in NMT systems is that it offers the user full control over the size of the resulting vocabulary. It also prevents the production of morphs that are not present in the training data (*open-vocabulary*), because rare or unknown words will get segmented into smaller known morphs.

Because BPE takes the frequencies of words into account, it encodes very frequent words into single tokens, while less frequent words become more segmented. By encoding frequent words into single tokens, the resulting average sentence length in tokens for BPE encoded data is shorter than for other approaches.

2.3.5 Linguistically-Motivated Vocabulary Reduction

Linguistically-Motivated Vocabulary Reduction (Ataman et al., 2017) (LMVR) is an extension to the Morfessor FlatCat which allows control over the resulting vocabulary size.

LMVR was developed with the aim to improve the usability of existing morphological segmentation methods for NMT purposes. Reducing the size of the vocabulary is beneficial for the application of a segmentation method for the purpose of building an NMT system, because a reduced vocabulary size reduces the model's complexity, and increases the frequency and consistency of the individual morphs in the data, thus improving the NMT model's capability for open-vocabulary translation.

It was proposed as an alternative approach to BPE (Section 2.3.4), which while offering control over the resulting vocabulary size, lacks consideration of the linguistic properties of morphs in its procedure. LMVR on the other hand does take into account such linguistic properties by adding an additional weight to the MAP estimate of the model. By modifying this weight, a preference for the complexity of the morph lexicon can be supplied to the model.

In experiments with NMT models based on Bahdanau et al. (2014), reduced-vocabulary segmentation with LMVR resulted in an improvement over the BPE baseline for translation of the agglutinative language Turkish into English (Ataman et al., 2017). Segmentation with LMVR also yielded an improvement over BPE for the translation of the low-resource agglutinative language Kazakh into English (Toral et al., 2019).

2.3.6 Neural segmentation

The use of character based encoder-decoder models for morphological segmentation (Section 4.3.1) was proposed by Kann et al. (2016). They trained a neural encoder-decoder based on Bahdanau et al. (2014) for the related task of *canonical segmentation*, which means that the morphs are not only segmented, but are also

restored to a standardized form. For example, the canonical segmentation of the English word 'realization' would be 'realize + ation'. On this task, the encoder-decoder model achieved a significant improvement over the previous state-of-the-art on English, German and Indonesian with an accuracy margin of up to 21%.

Although the approach showed a big increase in performance over existing methods, some new problems arose because of the nature of the model and task. Contrary to previous methods, the sequence-to-sequence model generates the output character-by-character instead of combining subsets of the input, or a selection of morphs from a dictionary. This lead to the generation of non-existing morphs, incorrectly split words, failure to split words, and missing characters (Kann et al., 2016).

Kann et al. (2018) explored the use of neural segmentation models for the task of morphological segmentation on several low-resource Mexican polysynthetic languages. Their RNN-based sequence-to-sequence model was able to improve over several baselines set by existing methods using only minimal training data. Evidently, neural approaches are well-suited for dealing with the complex morphology of polysynthetic languages.

Qiu et al. (2019) approach the related task of Chinese Word Segmentation with a Transformer (Section 4.3.3) model. Their proposed model achieves a clear improvement over the previous state-of-the-art, which uses LSTM-based models. They hypothesize that the self-attention layers included in the Transformer architecture (Section 4.3.2) help to alleviate the problem that RNN and convolutional approaches have with long-term dependencies.

The architecture lends itself well to multitask learning (Caruana, 1997) for improving the performance on a specific criterion or domain, because of the possibility to prime the model for a task using *tags* (Section 4.5.2). Qiu et al. (2019) exploit this by training the model for a collection of corpora with different criteria, with the data of each corpus prepended with a unique tag. This allows the model to adapt its output to a specified target corpus at inference, while learning from all corpora during training.

In the following chapter, all of the segmentation methods described in this chapter are applied to Inuktitut data. They are then assessed on their ability to perform morphological segmentation for this language by comparing the generated segmentations to reference data. The results of this analysis are used to determine which methods are most capable of dealing with the complex morphology of polysynthetic languages.

Chapter 3

Intrinsic evaluation

In this chapter the segmentation methods introduced in Chapter 2 are compared based on their ability to produce morphological segmentations for the polysynthetic language Inuktitut. We discuss the details of the experimental setup (Section 3.1), and the configurations of the compared segmentation methods (Section 3.2). The methods are assessed quantitatively, which we try to correlate to observed tendencies in the generated segmentations (Section 3.3).

3.1 Experimental setup

To evaluate the segmentation methods on how close they are to the linguistically optimal segmentation, we compare the performance of each of the methods with a collection of gold-standard segmentations from the Inuktitut Computing GitHub repository[7]. A researcher at the National Research Council of Canada (NRC) who worked on the Inuktitut Computing project, was able to provide us with information about how the data in the file was collected[8].

The file was created by manual decomposition of 1,096 words that occurred at least 100 times in Hansard documents dated between 1999 and 2002 . The original

[7]https://github.com/LowResourceLanguages/InuktitutComputing

[8]P.c. date of communication: April 28, 2020

purpose for the file was to serve as a unit test for the morphological analyser hosted in the same repository. Certain words in the file contain tags that were added to denote words that should be disregarded for analysis, such as surnames, place names, and words with spelling errors. Each of the methods is used to segment the list of unsegmented words, after which we compare the output to the reference file.

In line with previous studies Kann et al. (2018), we use the *Border F1* and *Token Accuracy* evaluation metrics. In addition we report the Precision and Recall for each method.

The token accuracy metric computes for each output segmentation whether it was predicted correct. This metric will only determine a segmentation as correct when it is an exact match with the reference segmentation. The final accuracy score for each segmentation method is computed as the average accuracy over all examples.

The Boundary F1 measure ignores the individual symbols in the output, and instead only looks at the indices of the token boundaries in the output. To compute this metric for a given gold standard set of boundaries B_{gold} and retrieved boundaries B_{retr}. First we calculate the precision and recall according to (3.1) and (3.2) respectively.

$$P = \frac{|B_{gold} \cap B_{retr}|}{|B_{retr}|} \tag{3.1}$$

$$R = \frac{|B_{gold} \cap B_{retr}|}{|B_{gold}|} \tag{3.2}$$

Then the F1-score is computed according to (3.3).

$$F_1 = 2 \cdot \frac{P \cdot R}{P + R} \tag{3.3}$$

	Segmentation	F1	Accuracy
Reference	aanniaq tu liri ji kkun nut	-	-
Example 1	aanniaq tu liri jik kun nut	1.000	1.000
Example 2	aanniaq tu liri jikkun nut	0.900	0.000
Example 3	aan niaq tu liri jik kun nut	0.916	0.000

Table 3.1: Example F1 scores / accuracy for different segmentations of an example word from the reference data.

3.2 Segmentation methods

Uqailaut Morphological Analyser

Uqailaut is a rule-based morphological analyser for Inuktitut that has been made available by the National Research Council of Canada. It comes in the form of an executable Java program, and can segment and analyse one word at a time. The program outputs the analyses in the following format:

```
{<surface form>:<deep form>/<morphological analysis information>}
```

An example output is shown below.

```
$ java -jar Uqaulaut.jar saqqitaujuq
> {saqqi:saqqik/1v}{ta:jaq/1vn}{u:u/1nv}{juq:juq/1vn}
> {saqqi:saqqik/1v}{ta:jaq/1vn}{u:u/1nv}{juq:juq/tv-ger-3s}
```

Often the analysis will return multiple analyses, meaning that there is ambiguity between multiple possibilities. The program in its unchanged form will perform recursive operations to decompose the suffixes until all consistent analyses have been found, and then returns them all.

To gather data for supervised methods, we apply the program to the Hansard training corpus, resulting in 626,894 successfully segmented word types out of 1,521,279

alphabetical word types in the corpus (41 percent). For the remaining words Uqailaut was unable to provide a segmentation.

We create a modified version of Uqailaut and use it to collect only unambiguous segmentations. Segmenting the training corpus resulted in 45,511 segmentations of unique words for which Uqailaut only offers a single segmentation possibility.

Morfessor Baseline

We make use of the semi-supervised variant of Morfessor Baseline (Section 2.3.1). The models used for the experiments are trained using the full Hansard training corpus as unlabelled data. We train one model on 45,000 unambiguous segmentations from the segmentations previously obtained with Uqailaut, and another model on 45,000 ambiguous segmentations. In both cases, we use 3,000 additional ambiguous segmentations as validation data for determining the corpus weights. The parameters are kept at their default values. The model is trained for a maximum of 25 epochs.

Morfessor Categories-MAP

The Morfessor Categories-MAP algorithm (Section 2.3.2) does not provide the option to train in a semi-supervised manner. We train the Categories-MAP model on the Hansard training corpus. The perplexity threshold is kept at the default value of 10.

Morfessor FlatCat

Similar to the Morfessor Baseline model, the Morfessor FlatCat model is trained on the Hansard corpus, with an additional 60,000 of the 626,894 segmentations that were obtained using Uqailaut used as annotated data.[9] An additional 3,000 annotated segmentations are used as validation data.

[9]Due to time constraints, separate models with ambiguous versus unambiguous annotated data were not trained for Morfessor FlatCat.

Byte-Pair Encoding

For the BPE models we make use of the scripts available in the *subword-nmt*[10] repository (Sennrich et al., 2016b). Two different methods for computing the BPE models were used. The first is the conventional way of training the model directly on the tokens in the corpus. This takes into account the frequency with which pairs of subword units occur in the corpus, resulting in frequent words being encoded into a single token, while infrequent words remain undersegmented.

We hypothesize that as a result of the repetitiveness[11] of the Hansard corpus, frequent words specific to the parliamentary domain of the corpus will become encoded into single tokens, whereas the remainder of the corpus will remain undersegmented, and that this will negatively impact both the intrinsic and extrinsic quality of the segmentation. In addition, this could lead to poor generalization of a translation model, as the model will not be able to learn from the morphological information of frequent words when they are undersegmented.

As an alternative we experiment with a BPE model trained on word types instead of tokens. In practice, this means that we derive a list consisting of all unique tokens types in the training corpus, and train a BPE model on this list. We hypothesise that by training on word types, the bias towards frequent words is reduced, and the model will end up with subword units that are closer to the actual Inuktitut morphemes, resulting in a better intrinsic evaluation score.

We refer to these methods as *Token BPE* and *Type BPE* respectively. For both training methods, we make a comparison between models with 1,000 and 5,000 merge operations. These parameters were selected because previously Joanis et al. (2020) found that 5,000 merge operations lead to the best results for translation into English. The 1,000 merge setting was added to the evaluation, because preliminary experiments showed that 5,000 merge operations resulted in an average undersegmentation. By decreasing the number merge operations to 1,000, they visually appeared more similar in length to Inuktitut morphemes.

[10] https://github.com/rsennrich/subword-nmt

[11] 40% of the sentence pairs in the training corpus are duplicates of other sentences.

LMVR

For training the LMVR model, we follow the example listed on the LMVR GitHub repository[12]. The Hansard training corpus is supplied as training data, and we set the maximum lexicon size to 20,000. The perplexity threshold is kept at the default value of 10.

Neural segmentation model

Using the 626 thousand segmented word types obtained using the rule-based segmenter, we train a Transformer model to replicate this segmentation. We implement the segmentation model using Marian[13] v1.7.6. On the source side, the unsegmented words are used as input data. The corresponding segmented words are used as target data. Each line of the source and target data contains a single word per line, and to make sure that the model trains at character level, we separate the characters of each word using spaces. On the target side, we denote the segmentation boundary by adding a *boundary token* '@', as shown in Example 3.2.

As segmentation is a less complex task than translation, the model used for this task has a reduced number of trainable parameters to prevent overfitting. In order to determine sensible values for the configuration of a Transformer segmentation model, we looked at similar studies on Chinese word segmentation (Qiu et al., 2019; Jiang & Tang, 2019). We derived that a configuration with the embedding dimensionality set to 128 appears to be enough, and decide on an encoder and decoder consisting of three layers. We randomly select 10 percent of the segmented word types and use it as validation data to determine the best model, the remaining 90 percent is used for training. Every 5,000 weight updates, the weights are stored, and the validation metrics for the current model are calculated.

Since the rule-based segmenter returns more than one possible segmentation for many words, we want to test whether the segmentation model benefits from using

[12]https://github.com/d-ataman/lmvr/blob/master/examples/example-train-segment.sh
[13]https://marian-nmt.github.io/

exclusively annotated data for which the rule-based segmenter returns only one possibility. We train two Transformer models, one of which is trained on 45,000 a random selection of the collected annotations, and the other trained on 45,000 single-possibility segmentations.

The models was trained until there was no improvement in the *perplexity* metric for five consecutive validation runs. The model with the best validation score was stored and chosen as the final model.

Source: H a k i r u m a n i q a t t i a q t u t i k
Target: H a k i @ r u m a @ n i @ q a t @ t i @ a q @ t u t i k

To apply the segmentation resulting from this model to the corpus, we perform the segmentation on the list of all 1.5 million unique alphabetic types of the Hansard corpus. By doing this, we end up with a list of pairs of the original word types and their segmented version. The list is then used to replace occurrences of those word types in the corpus with their segmented form.

Performing replacement of words with their segmented form in a large corpus, with a large dictionary, is a time consuming task using simple dictionary search and replacement. In order to overcome this we use the FlashText algorithm (Singh, 2017), which has the advantage that the time complexity is not dependent on the number of terms to be matched, allowing us to use the large replacement dictionary efficiently. This algorithm is optimized for the task of replacing exact matches of the replacement keys when they are surrounded by word boundaries, such as spaces and the start and ending of a line. Since we apply the segmentation to a tokenized version of the corpus, all words in the corpus are already surrounded by spaces, and therefore this algorithm is a fitting solution. Using the Python implementation of the algorithm it takes only one minute to segment the corpus on a single CPU.

3-step segmenter

As an attempt to make the best possible segmentation model that is usable for NMT, we created a custom segmentation method that combines the Uqailaut's rule-based

segmenter, the Transformer model trained on 45K unambiguous segmentations, and the Token BPE model with 5,000 merge operations. For each alphabetic word to be segmented, the method first attempts to find an unambiguous segmentation using Uqailaut. If this initial step fails, the word is segmented with the Transformer model. When also the Transformer model fails to segment the word, we apply the BPE model as a last resort. Tokens containing non-alphabetic characters are always segmented using the BPE model, because those characters cannot be segmented with Uqailaut, nor with the Transformer models. We refer to this combined model as *3-Step*.

Preliminary experiments showed that the resulting vocabulary size for this model was still over 80,000 unique entries, as a result of failed segmentations. Such a large vocabulary is unfavorable for use in an NMT system because some morphs become very infrequent, especially when taking into account that the dataset is not very large.

We extended this model by using LMVR (Section 2.3.5) to reduce the vocabulary size. Because LMVR requires a Morfessor Baseline text-based model as input, I segmented the Hansard training data with the existing Transformer model, and formatted the unique word segmentations and corresponding frequencies in the same format as the Morfessor Baseline text-based model. We applied LMVR to this spoofed Morfessor model and used the `lexicon-size` argument to specify a maximum lexicon size of 20,000 unique entries. We refer to this model as *3-Step + LMVR*.

3.3 Results

Morfessor Baseline

The segmentation model trained on unambiguous segmentations with Morfessor Baseline scored the highest on all metrics for the models in the Morfessor family. There is a fairly large margin between the models trained on ambiguous and unam-

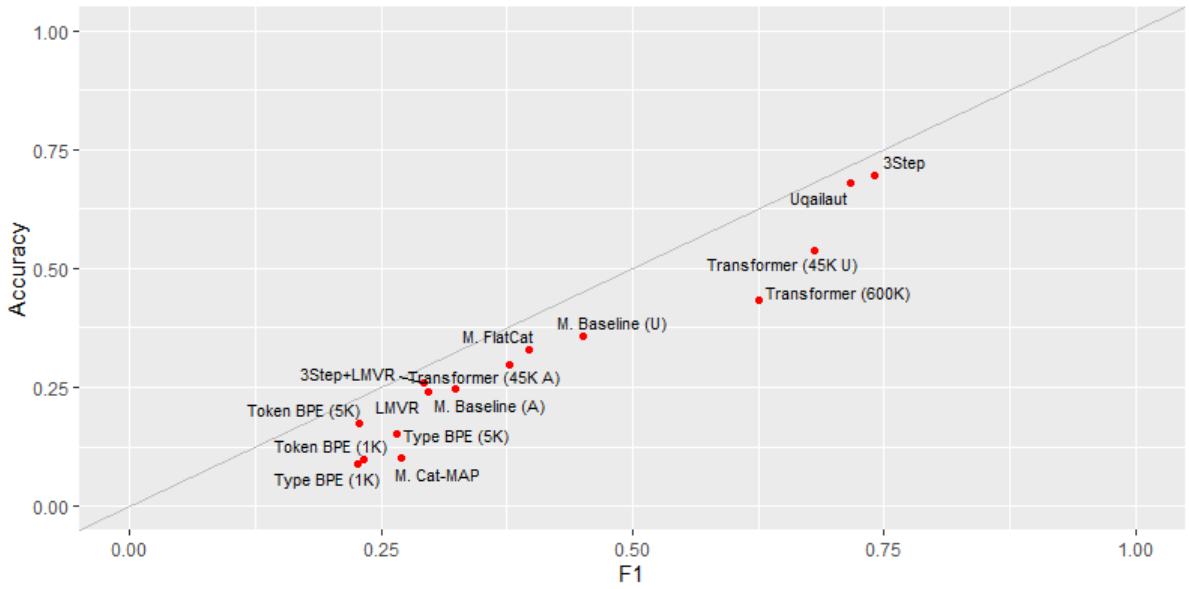

Figure 3.1: The F1 and accuracy of each segmentation approach in the intrinsic evaluation.

biguous data, indicating that the model benefits from consistently annotated data. Sample segmentations for both models are shown in Table 3.3.

Morfessor Categories-MAP

The Morfessor Categories-MAP model achieves the lowest scores of the Morfessor family across all metrics, besides the recall. The F1-score of 0.270 is comparable to the score of the Type BPE (5000) model, while the accuracy of 0.102 is in fact outperformed by the BPE model. Qualitative analysis of the produced segmentations show that while slightly oversegmenting most words, the segmentation boundaries are also often simply placed in the wrong position.

Inspection of the morph categories assigned by the model (Section 2.3.2), we see that many segmentations contain incorrectly classified morphs. This could be problematic for the consistency, for example because a suffix that is misclassified as a stem prevents it from occurring after a correctly classified stem. Or in the case of a stem misclassified as a prefix, it could prevent a correctly classified suffix from appearing after it.

Some example segmentations for the model are shown in Table 3.4.

Morfessor FlatCat

The Morfessor FlatCat model achieved reasonable scores across all metrics. The ratio of produced morphs versus the number of morphs present in the reference file indicates that on average FlatCat undersegments words, producing 11.6% less morphs than expected. In the data this is observed as a bias towards segmentations consisting of two morphs; words that are unsegmented in the reference data are often segmented into two morphs, and words consisting of more than two morphs in the reference data are often undersegmented. Examples of this are shown in Table 3.5.

LMVR

The LMVR model scores similar to the Morfessor Categories-MAP model in terms of recall (0.284 vs. 0.287 respectively), while improving over Categories-MAP with a small margin in terms of precision and F1-score. In terms of accuracy, LMVR (0.240) outperforms Categories-MAP (0.102). Most words that the model gets entirely correct are words consisting of two morphs. It is clear from the data that, even more so than FlatCat, LMVR tends to segment words into two morphs. A difference between the segmentations of FlatCat and LMVR appears to be that LMVR prefers to create segmentations consisting of a long stem with a short suffix. Some examples of this are shown in Table 3.6.

Byte-Pair Encoding

The BPE models yield the lowest scores across all methods on the intrinsic evaluation. The low scores are expected, because there is no linguistic motivation behind the generated segmentations (Section 2.3.4). The best performing BPE model is the Type BPE (5000) model, which outperforms the other models in terms of precision, recall, and F1-score. It also achieves the best morph count ratio, with only a 2%

difference between the token count of the gold-standard. By manual inspection, we notice that the model fails to segment frequent suffixes from preceding characters that occur frequently together, as illustrated in Table 3.7, which is an expected consequence of the purely frequency based approach. Furthermore, it has the tendency to oversegment words consisting of infrequent letter combinations, which is also expected.

The highest accuracy among the BPE models was reached by the Token BPE (5000) model, which can be explained by the undersegmentation of frequent words, which makes it score well on words that remain single tokens in the reference data. Both BPE models with 1,000 merge operations perform poorly, with comparable results across all metrics.

Transformer segmentation models

The results of the Transformer-based segmentation models show a lot of variation between them. The best results of all methods tested besides the rule-based segmenter were obtained with the model trained on the 45K unambiguous segmentations. This model received an F1 score of 0.680, and token accuracy of 0.539, outperforming the second best Transformer segmentation model (*Transformer all 600K*) which scored 0.625 on the F1 metric, and 0.433 on the word accuracy. The Transformer model trained on 45K randomly selected ambiguous annotated data performed significantly worse on the test data, with an F1 score of 0.378 and an accuracy of 0.297.

The morph ratio shows that the Transformer models generate on average about 17 - 19% more morphs than there are present in the reference data.

Out of the Transformer models, the model trained on 45K randomly selected annotated data never failed to segment any of the words in the test data (Section 2.3.6). The model trained on 45K unambiguous annotated data failed on only 0.09% of the test data, and the model trained on all 600K segmentations failed on 0.55% of the words. Some example segmentations for these models are shown in Table 3.8.

Uqailaut's rule-based segmentation

The rule-based Uqailaut segmenter performed the best on the test-data, scoring the highest on all metrics with an F1-score of 0.800, and an accuracy of 0.765. The morph ratio of 0.996 which means that there are approximately as many morphs in the segmented test-data, as there were in the reference data. Qualitative analysis of the data reveals that the method performs well on most words, and there appears to be no clear bias towards one type of error.

3-Step segmenter

The custom 3-step segmenter combines the high performance of the rule-based segmenter, with added coverage from the Transformer and BPE models. Naturally this means that the obtained results are higher or equal to the performance of Uqailaut. When comparing the performance of the 3-step segmenter to the performance of just the rule-based segmenter, we see an increase of 0.025 for the F1 score, and 0.015 on the accuracy. While these increments seem relatively small, it should be taken into account that only the 11.5% of the segmentations for which Uqailaut failed were changed by adding the back-up models.

The back-up transformer model succeeds in improving the coverage of the words for which Uqailaut fails. Mistakes that still occur are generally minor, but still quite frequent. They include, for example, a slight oversegmentation of single morphs that elsewhere also occur further segmented. Another problem is that proper names in the reference file that are being segmented by the model, although none are segmented in the reference file. This is likely due to the fact that no examples of proper names occur in the training data, which makes the Transformer model treat it like a regular word.

The 3-step model that was extended with LMVR shows a strong decrease in performance compared to the regular 3-step model, with an F1 score of 0.292, and a token accuracy of 0.258. Compared to the regular LMVR model, it is comparable on the F1 score, but shows a slight increase in accuracy. Also similar to the regular

LMVR model, it also appears to prefer to segment words into a combination of two morphs, a relatively long stem, and a short suffix. We reason that the large drop in performance of this model is due to the fact that LMVR alters the existing morph vocabulary too much. Sample segmentations of both LMVR models are shown in Table 3.10.

Method	Prec.	Rec.	F1	Acc.	Ratio	Failed
Token BPE (1000)	0.230	0.237	0.233	0.098	1.14	-
Token BPE (5000)	0.239	0.218	0.228	**0.175**	0.664	-
Type BPE (1000)	0.218	0.235	0.226	0.089	1.338	-
Type BPE (5000)	**0.279**	**0.252**	**0.265**	0.151	0.980	-
Morfessor Categories-MAP	0.254	0.287	0.270	0.102	1.242	-
LMVR	0.309	0.284	0.296	0.240	0.809	-
Morfessor (45K unique)	**0.479**	**0.423**	**0.451**	**0.358**	0.816	-
Morfessor (45K ambig.)	0.344	0.304	0.324	0.246	0.920	-
Morfessor FlatCat	0.411	0.382	0.397	0.328	0.884	-
Transformer (45K unique)	**0.661**	**0.698**	**0.680**	**0.539**	1.198	0.09 %
Transformer (45K ambig.)	0.364	0.392	0.378	0.297	1.177	-
Transformer (all 600K)	0.594	0.656	0.625	0.433	1.171	0.55 %
3-Step	**0.736**	**0.746**	**0.741**	**0.696**	1.072	-
3-Step + LMVR	0.305	0.279	0.292	0.258	0.521	-
Uqailaut (Rule-based)	0.712	0.720	0.716	0.681	0.996	11.50 %

Table 3.2: The results of the intrinsic evaluation for each segmentation approach. The precision, recall, and F1 scores are calculated on segmentation boundaries, while the accuracy is calculated on the full segmentation. The *ratio* statistic shows the proportion of morphs that an approach generates as a fraction of the number of morphs present in the gold-standard. A number larger than 1 suggests oversegmentation, and lower than 1 suggests undersegmentation. The *failed* statistic indicates the proportion of words that the approach failed to reconstruct for the methods for which that can occur.

Reference	apiq qusi ksa it	ingirra ju liri nir mut
Morfessor (unique)	a piqqusi ksait	ingirraju lirinirmut
Morfessor (ambig.)	apiqqusi ksa it	ingirrajuliri nirmut

Table 3.3: Sample segmentations generated by the Morfessor Baseline method. While most segmentation boundaries are generally placed in the right location, under-segmentation is observed in most words.

Reference	tamatu mani	tukisi na liq tis sima juq
Morfessor Cat-MAP	ta matu mani	tukisi na li qti ssi ma juq

Table 3.4: Sample segmentations generated by the Morfessor Categories-MAP method.

	Undersegmentation	**Oversegmentation**
Reference	sivulliqpaa mit	taimannak
Morfesor FlatCat	sivu lliq paa mit	taima nnak

Table 3.5: Sample segmentations generated by the Morfessor FlatCat method.

Reference	iqqanaijaq tu liri ji kkun nit	asi nginnil lu
LMVR	iqqanaijaqtulirijikku nnit	asinginni llu
Morfesor FlatCat	iqqanaijaqtu lirijikkunnit	asi nginnillu

Table 3.6: Sample segmentations showing some characteristic mistakes of the LMVR and FlatCat models. The LMVR model is prone to generating long stems with a short suffix, whereas the FlatCat model also undersegments, but prefers a longer suffix.

Reference	iqalun nut	nunavum mi
Type BPE (5000)	iqalu nnut	nunavu mmi

Table 3.7: Example segmentations generated by the Type BPE (5000) model, showing the model's tendency to incorporate frequent endings of preceding morphs into the following morph.

Reference	ajji gii nngit tu nik	ikupig vi liri ji kkun nut	kati ma vim mi
Single 45K	ajji gii nngit tunik	ikupig vi liri ji kkun nut	kati ma vim mi
Random 45K	ajji gii nngit tu ni k	ikupig vi li ri ji kkun nut	kati ma vim mi u
All 600K	ajji gii nngit tu ni k	ikupi gvi liri ji kkunnut	kati ma vim mi

Table 3.8: Some sample segmentations from each of the regular Transformer segmentation models.

Reference	pula raq tu liri nir mut	mali ga liuq ti limaa nut	parna uti it
Uqailaut	pula raq tu li ri nir mut	mali ga li uq ti limaa nut	parna ut ii t

Table 3.9: Sample segmentations where the Uqailaut rule-based segmenter oversegments words.

Reference	kati ma ji ralaa ngu innaq tut	mali ga liuq ti u qati kka	nirtu nar tuq
3-step + LMVR	katimajiralaangu innaqtut	maligaliuqtiuqati kka	nirtu nar tuq
LMVR	katimajiralaanguinna qtut	maligaliuqtiuqati kka	nirt una r tuq

Table 3.10: Sample segmentations of the LMVR and 3-step + LMVR models, showing that the placement of the segmentation boundaries of the 3-step + LMVR is better than the regular LMVR model. Both models still clearly undersegment the words.

Chapter 4

Machine Translation

4.1 Rule-based Machine Translation

Although large neural networks are currently the standard for machine translation, this has not always been the case. Until the 1980's the field of machine translation was dominated by rule-based approaches (Hutchins, 2007). Rule-based Machine Translation encompasses translation models that accomplishes translation of a text by applying a collection of pre-programmed rules to a phrase.

Rule-based MT relied on linguistic experts to extract rules and formulate them, so that the system could apply them to parse phrases. Until 1966, MT research was dominated by systems based on the *direct translation* approach, meaning that rules are applied to translate a text directly from L_{source} to L_{target}.

By contrast, *interlingual MT*, an approach that did not gain much popularity in MT research until after 1967, involved translation of L_{source} into a common intermediate representation of the semantics of the source text that was subsequently translated into L_{target}. Interlingual MT lended itself well for translation between multiple languages, as the abstract intermediary representation was by definition independent of the choice of source and target language–the representation can in theory be derived from, and translated to any language, so long as the rules for the language exist (Figure 4.1).

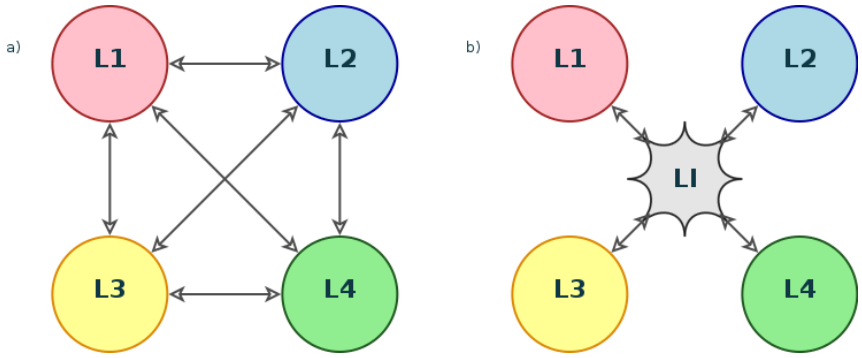

Figure 4.1: A translation graph[14] showing the required collections of rules for translating between four languages (represented by $L1, ..L4$). Subfigure A shows the translation graph for transfer-based and direct machine translation, where a new set of rules is needed for each combination of L_{source} and L_{target}. Subfigure B shows the graph for the interlingual approach, where LI represents the shared intermediary language.

A third approach called 'transfer-based MT' is another indirect method, that involves a general process consisting of three steps: analysis of the source text into an higher level representation, transfer of the source representation to the target representation, synthesis of the target representation into L_{target}. Contrary to interlingual MT, transfer-based MT involves two higher level representations–one for the source language, and one for L_{target}. The higher level representations of the input text are formed by performing analysis of the morphology and syntax on the text. The target representation is a mapping of the source representation that is suitable for generation of output in L_{target}.

Between 1976 and 1989 several operational transfer-based commercial translation systems were developed that produced translations with a decent quality for the time. While most systems were designed as general purpose translation systems, they often were biased to a certain domain for which the rules were robust. The systems also still required post-editting and carefully formatted input.

4.2 Statistical Machine Translation

After 1989, rule-based systems lost the interest of research groups in favor of *corpus-based* or *data-driven* methods. Machine translation, after decades of research, still required so many rules and exceptions to be written, that the idea to learn from examples became appealing. In the 1990's, IBM introduced a statistical approach for automatic translation of French into English named Candide (Berger et al., 1994). Aside from pre- and post-processing steps, this system did not make use of any pre-determined rules, and instead relied on probability models and training algorithms that could learn from examples (Berger et al., 1994). The translations that Candide was able to produce without the use of explicit rules was sufficient to drive statistical machine translation (SMT) to become the focus of research groups for the following period. Additional factors that were crucial for making SMT possible at the time were increases in computing power, storage capacity, and availability of text resources on the internet.

The statistical approach proposed by Berger et al. (1994) can be further categorized as *word-based SMT*, which means that the model rests on the assumption that words are the *atomic units*. As a result of this, each word in the source sentence can be mapped to at most one word in the target sentence, although the neighbouring words can be taken into account (Xiong & Zhang, 2016). This imposes a clear limitation on the approach, since in real translations it is common for a single word in the source language to translate to multiple words in the target language.

In order to overcome the limitation of word-based SMT, *phrase-based SMT* was introduced Koehn et al. (2003). Moving away from words as the atomic units, this approach takes phrasemes (expressions consisting of multiple words) as the atomic units, thereby granting the model the possibility to learn one-to-many and many-to-many mappings. The phrasemes did not need to be actual linguistic phrases, but were sequences of words that were extracted using statistical methods. Apart from the choice of atomic units, phrase-based SMT actually works the same as word-based SMT in that it creates one-to-one mappings between the phrasemes in the source language to phrasemes in the target language.

Another SMT approach that arose from the criticism of the word-based SMT model introduced by IBM, was *syntax-based SMT* (Yamada & Knight, 2001). Contrary to word-based SMT, syntax-based SMT allowed structural and syntactic aspects to be included in the model. Yamada and Knight (2001) proposed a model with a channel that accepts a parse-tree as input, and which allows reordering of child nodes, insertion of new words, and translation of leaf nodes. Because of the reordering operations, this approach was shown to be able to deal with language pairs that have different word orders.

The period between the 1990's until 2014 in which SMT was the leading MT framework, was characterized by a significant increase in the amount of machine translation research. This change was driven for a large part by the creation of open-source toolkits for SMT, lowering the barriers of entry for participation in SMT research, no longer limiting access to the field to expert research groups. Parallel corpora that had been collected by research groups were also being shared for use by others. Another factor was the start of MT evaluation workshops, such as the Workshop for Machine Translation in 2006, where the research community could compare findings and exchange ideas.

4.3 Neural Machine Translation

By 2013, deep learning had already achieved superhuman performance in the field of computer vision (Krizhevsky et al., 2012), but had not yet been successfully applied to machine translation. The fact that sentences could vary in length posed a problem for the application of deep neural networks, since they require the dimensions of the network to be known and fixed (Sutskever et al., 2014).

Although researches had conducted preliminary experimentation with neural machine translation as early as 1987 (Allen, 1987; Neco & Forcada, 1997), memory and processing power of computers was not be adequate for a successful implementation until 2013, when the first version of Neural Machine Translation (NMT) models were introduced with the Recurrent Continuous Translation Model in Kalchbrenner and

Blunsom (2013). It was the first implementation that used a CNN based encoder in order to map an input sentence to a fixed size vector, and an RNN based decoder model to the output sequence.

The next year, Sutskever et al. (2014) re-imagined the architecture, using an LSTM both for the encoder and decoder parts of the model, which had the benefit of keeping information about the word order on the source side.

4.3.1 Encoder-decoder models

The model introduced by Sutskever et al. (2014) is called an encoder-decoder model. As the name suggests, it consists of two main components, the encoder and the decoder. The task of the encoder is to encode the input sentence into a hidden state representation in the form of a fixed size vector, as shown in Figure 4.2.

Single RNNs (unidirectional RNN) can only take into account the context of the words that were processed before it. A set-up of two RNNs running in opposite directions is referred to as a bidirectional RNN. It consists of two Recurrent Neural Networks (RNN) instead of one, with one RNN running in the left-to-right direction, and the other processes from right to left. The use of a bidirectional RNN gives the encoder access to the context of the words both before and after it.

The decoder then converts this context vector into the output sentence word by word using another RNN to predict the probability distribution over the entire vocabulary of output words. When training a model, the probability distribution of the decoder is optimized to assign a higher probability to the correct output words, while keeping the probability mass for incorrect words small.

The way in which the decoder probability distribution is used to form the output sentence during inference is called a decoder algorithm. The most straight-forward choice for prediction of an output sentence is to choose the words with the highest probability at each step. This approach has the benefit that it is a very fast method with a simple implementation. But since each choice in the decoding phase influences the probability distribution for the next word, often this does not lead to the best

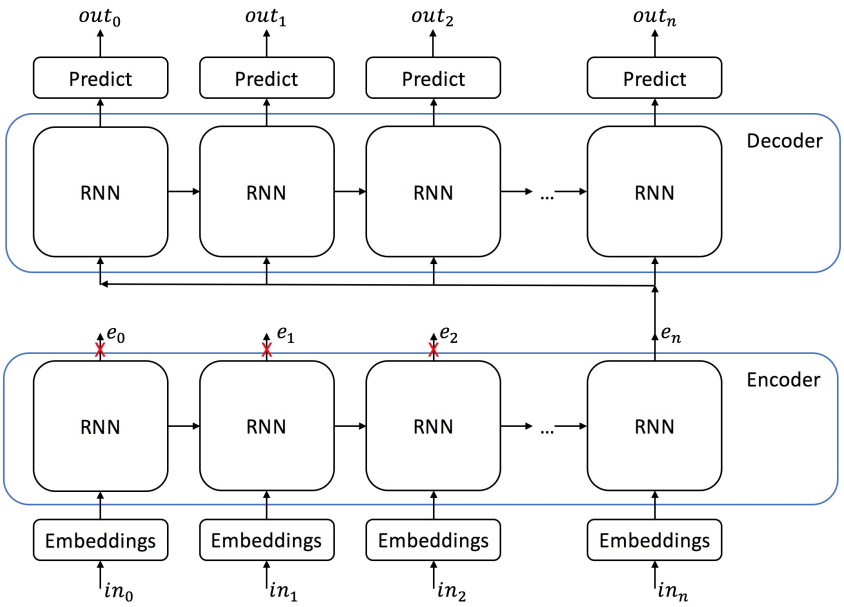

Figure 4.2: An schematic representation of the encoder-decoder model with unidirectional RNNs. The encoder builds a fixed-size vector representation (e_n)) of the one-hot encoded input sentence, which is used by the decoder's RNN at each step to predict the next word in the output sentence. Image sourced from: https://talbaumel.github.io/blog/attention/

possible output.

Beam search is a more involved decoder algorithm, which considers also options that do not have the highest probability at each step. It keeps track of a list of the k candidate sequences throughout the process of decoding. It starts with the k highest probability choices for the first word, then at each next step, the k best next words for each candidate become the new candidates, after which the list of candidates is again reduced to the top k. After all candidate sequences have been fully decoded, the best of the remaining candidates is chosen as the output sequence. Which candidate sequences are considered the 'best', is determined based on the likelihood of the entire sequences.

4.3.2 Attention mechanism

Although the encoder-decoder model in itself performed relatively well on short sentences, longer sentences were still problematic (Cho et al., 2014). The encoder needs to be able to compress all the information in the source sentence into a fixed-length vector, which becomes more challenging when the amount of information in the sentence increases. Bahdanau et al. (2014) introduced the self-attention mechanism for encoder / decoder models, which at each step attributes weights to parts in the source sentence that signify how important a specific source sequence element is for the target hidden state (Luong et al., 2015). This solves the problem of having to encode a large amount of information into a single fixed-length vector, by expanding the single embedding vector to a sequence of vectors for each token in the input sequence, called word annotations. The decoder then computes the embedding for each target word by *soft searching* for a set of relevant words in the source sentence.

The attention factor indicating the importance of input word j for producing output word i as a scalar value is computed according to Equation 4.1 (Bahdanau et al., 2014; Koehn, 2020).

$$a(e_{i-1}, h_j) = v_a^T tanh(w_1 e_{i-1} + w_2 h_j) \tag{4.1}$$

The final *context vector* that is used by the decoder, is computed as a weighted sum of the relevant annotation vectors of the source sentence Bahdanau et al. (2014) according to Equation 4.2.

$$c_i = \sum_{j=1}^{T_x} a_{ij} h_j \tag{4.2}$$

Where T is the length of the source sentence, a is the weight matrix, h is a vector of annotation vectors of the elements in the input sentence. Figure 4.3 shows how the attention is implemented into the encoder-decoder model.

The addition of attention to the existing encoder-decoder models allowed models to

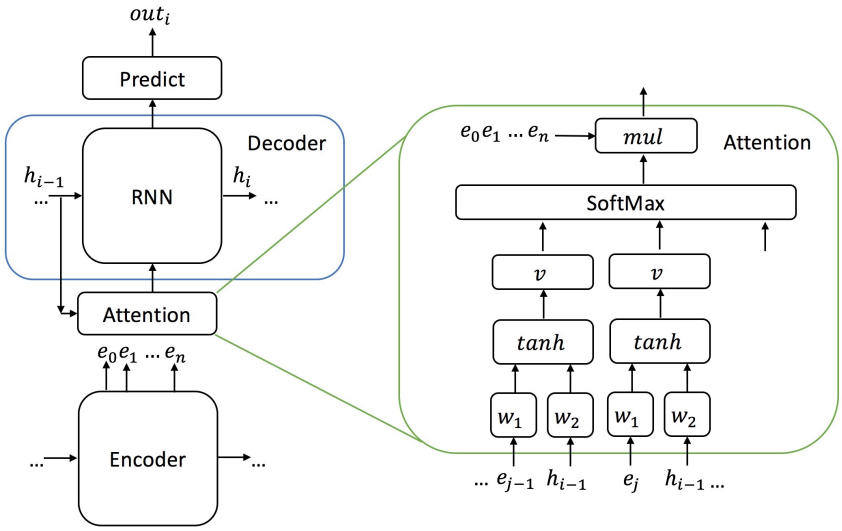

Figure 4.3: An encoder-decoder model with an added attention layer. The attention layer implements the steps described in Section 4.3.2 to provide the decoder with information about which input words are important for the prediction of the target words. Image sourced from: https://talbaumel.github.io/blog/attention/

achieve a better word-level alignment. It also helped models deal with non-monotonic alignments, as demonstrated in Figure 4.4, which is a problem that models without attention struggled with. The greater the difference between languages types, the higher proportion of non-monotonic alignments will be, which makes it an important development that benefits the translation between English and polysynthetic languages. An example of non-monotonic alignments between English and Inuktitut is presented in Table 4.1.

The attention mechanism introduced by Bahdanau et al. (2014) is not the only way

Qanniqlaunngikkalauqtuqlu
qanniq-lak-uq-nngit-galauq-tuq-lu
snow-a.little-frequently-NOT-although-3.IND.S-and
'And even though it's not snowing a great deal'

Table 4.1: An example of an Inuktitut word and its English translation, that illustrates that the word alignments between English and polysynthetic languages can be very non-monotonic. The order of words in the English sentence is not in alignment with the relevant morphemes in the Inuktitut word.

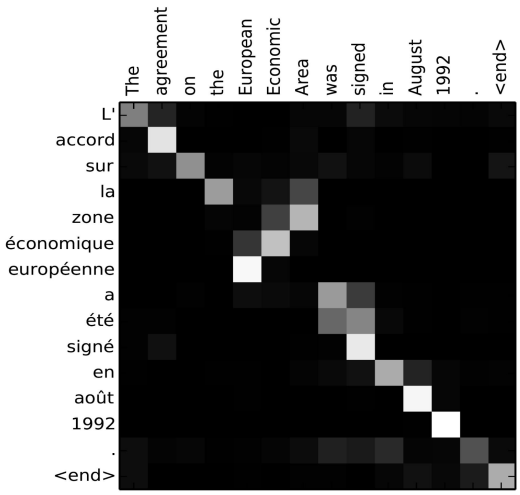

Figure 4.4: Alignment matrix between a French input phrase (y-axis), and its translation in English (x-axis). The matrix shows that attention mechanism (Bahdanau et al., 2014) helps the model translate the phrase "zone économique européenne" to "European Economic Area" successfully, despite the alignment between input and output words being reversed for this subphrase.

15

to compute the alignments. Luong et al. (2015) proposed *global* and *local attention models*, refering to whether the alignment model's scope is hard-limited to a specified window or not. They also experimented with other attention functions and found that using Dot-Product Attention lead to better results than the concat product attention function. In Vaswani et al. (2017), an improvement is made on dot-product attention by the addition of a scaling factor, which benefits the gradient of the softmax function during training.

4.3.3 Transformer architecture

The introduction of the Transformer architecture (Vaswani et al., 2017) has had a significant impact on the field of machine translation. The method was created with the aim to reduce the amount of sequential steps used in sequence-to-sequence models as a result of the use of RNNs. By relying solely on the self-attention mechanism, which was already being used in recurrent models, it succeeded in creating a parallelizable approach that was not dependent on convolution or recurrent layers.

This makes it possible to train models faster, and makes training deeper models realistic.

Self-attention is an adaptation of the attention mechanism discussed in Section 4.3.2, which instead of computing the alignments between input and output words, models the associations between input words. For each input word, the self-attention mechanism enriches the representation of that word with other relevant words in the input sentence that can help the model with determining the meaning (Koehn, 2020).

The Transformer architecture (Figure 4.5) also uses self-attention in the decoder, but contrary to the encoder, the decoder self-attention only provides connections to prior positions in the output sequence, so that predictions for an output word cannot depend on predictions for words that come after it.

An added advantage of the focus on the self-attention mechanism is that at each step in the processing of a sequence, the information flow from previous tokens is more direct, and not affected by the distance between the tokens. A self-attention layer creates paths of a constant number of operations between each of the positions in the input sentence, which makes Transformer models more resistant than RNN based-approaches against the problems with long sentences (Vaswani et al., 2017).

Because the Transformer architecture lacks recurrence, information about the position of words in the sentence is not available to the model by default. This problem is solved with a positional embedding, which is the positional information encoded in a vector of the same length as the input embedding, which is passed to the model by combining it directly with the input embeddings.

The removal of recurrent elements made it possible to train large language models significantly faster. Since its introduction it has been the primary model architecture used in language models, and consistently outperforms recurrent and statistical approaches (Barrault et al., 2019).

At the time of writing, neural approaches have largely superseded statistical approaches to machine translation, the Transformer being the most popular architec-

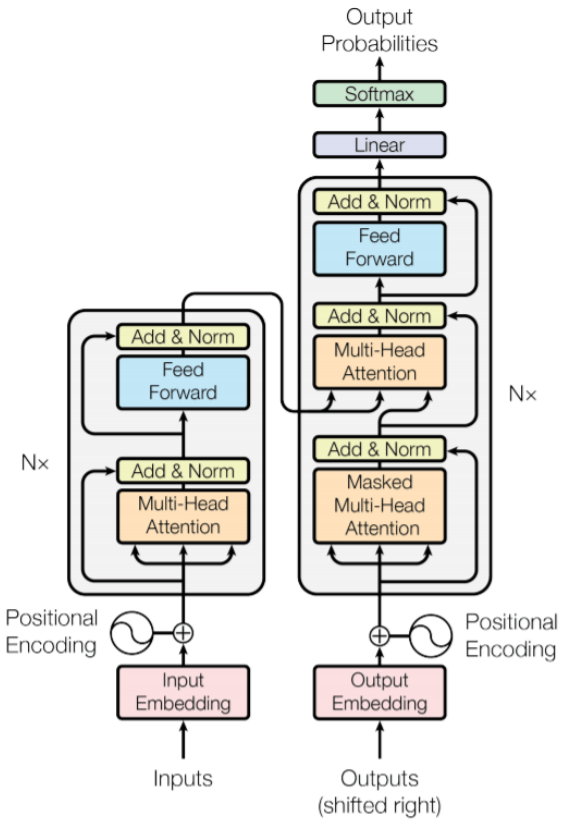

Figure 4.5: The Transformer architecture, as presented in Vaswani et al. (2017).

ture (Barrault et al., 2019).

4.4 Segmentation for MT

The pipeline of a translation system usually includes a number of preprocessing steps that make it easier for a neural language model to recognize patterns in the data. In recent years morphological segmentation has become an important part of the preprocessing steps for NMT models. Segmentation of words to sub-word units has been shown to improve the performance of translation models, and in particular it positively affects the model's ability to translate rare words Sennrich et al. (2016b), Vania and Lopez (2017). Even the translation of entirely unseen words improves from applying morphological segmentation, as shown in Figure 4.6.

The choice of segmentation method is determining for the vocabulary of the translation model, the number of tokens per line, and the frequencies of morphs in the

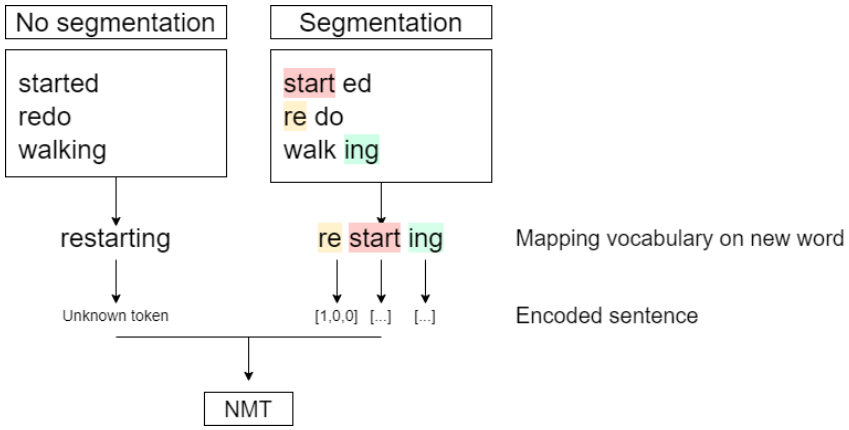

Figure 4.6: A simplified example illustrating the role that morphological segmentation plays in a translation pipeline for the translation of new or infrequent words. When segmentation is applied, the translation model is able to learn the meanings of the morphs in a word, instead of only learning from occurences of the full word. It can also make predictions about words that were not present in the training data, because it has seen the morphs it consists of before in the training data. When no segmentation is applied, and a new word occurs in a source text during inference, it will be replaced with *[unk]*, the token representing any word not present in the vocabulary of the model. In such a situation, the model will not be able to make any predictions based on the word itself, but can still make a guess by the context of the input and surrounding output words.

training data, each of which can have a big impact on the performance of the model. Most trainable methods therefore offer a number of configurable parameters to directly or indirectly influence the mentioned properties.

4.5 Low-resource NMT

For many non-western European languages, parallel data is not available in large volumes. The data that is available can be only partially usable because of semantic discrepancies between both sides of sentence pairs as a result of poor translation, or misaligned sentences. A main challenge of machine translation in a low-resource setting is to make the most of the data that is available (Sennrich & Zhang, 2019), and clever methods have been developed to boost the performance of such systems. For the translation of polysynthetic languages this is especially important, because there is only minimal data available for these languages (Littell et al., 2018).

4.5.1 Monolingual data

Parallel data for most language pairs is difficult to obtain. On the other hand, monolingual data is often much easier to obtain. It can be collected from any text on the internet, from newspapers, or from books. Monolingual data has been of great importance for the performance of SMT systems (Section 4.2) ever since SMT was first introduced. Language models trained on monolingual data were responsible for correcting the output of SMT systems, because the independence assumptions of SMT models resulted in crude and unnatural output. In neural machine translation this role is already fulfilled by the decoder, which can be regarded as a language model in itself and is also conditioned on the input context (Sennrich et al., 2015). However, since monolingual data is still much more available than parallel data, a large portion of NMT research has focussed on finding ways to incorporate the linguistic information that in monolingual data in the most effective way.

A simple way that monolingual data has been shown to be able to improve the

quality of translation is by treating the monolingual data as sentences for the target language, with an empty or dummy source sentence. By using monolingual data in this way, the fluency of the resulting model can improve because the decoder learns from the word frequencies, positions and co-occurrences (Sennrich et al., 2015).

Back-translation (Sennrich et al., 2015) is a more involved technique that requires an existing translation system that translates in the opposite direction. That is, a system that can translate from the desired L_{target} to L_{source}. For this method, a dataset of monolingual target sentences is translated into the desired source language using the existing language model, in order to create a *synthetic* parallel dataset $\{L'_{source}, L_{target}\}$. This synthetic parallel data can then be used as additional training data.

Back-translation is a popular choice for increasing the amount of data, and depending on the amount of monolingual data and the quality of the existing translation system, it can lead to a strong increase in the quality of the translation (Sennrich et al., 2015).

4.5.2 Domain adaptation

In a low-resource setting, as is the case for the translation of polysynthetic languages, there is not much of a choice regarding the corpora to use for training an NMT system, and often the available corpora are composed of data obtained from a small number of sources. Inevitably this means that there is little variation in the writing style and tone.

For example, an NMT system trained on parallel data extracted from parliamentary documents will produce translations in an overly formal style, which is not desirable for a translation task involving news data (Koehn & Knowles, 2017). Furthermore, the topics discussed in such texts will be biased towards subjects that are within the scope of the institution that created it. Debate records of a local government will not have the same distribution and variety of words as global news texts.

When the domain of the parallel data is different than the target domain of the

translation task, there are several techniques that can be used to adapt the domain of an NMT model to the task (Chu & Wang, 2018).

When there are no in-domain parallel corpora available, in-domain monolingual data of the target language can be used to train the decoder on the frequencies and co-occurrences of the domain. Both backtranslation and target-side monolingual data with dummy codings (Section 4.5.1) among others can be used to adapt the domain to the domain of the monolingual data. For domain adaptation using monolingual data, ideally there is a large quantity of high-quality in-domain monolingual data available for the desired target language, or the performance of the model may not improve, or even decrease. If this is not the case, there are other options available that do not rely on monolingual data.

Fine tuning (Luong & Manning, 2015) is a widely used domain adaptation method, which involves a final training round on solely in-domain data, after pre-training the model on mixed-domain data. A significant benefit of fine-tuning versus training only on in-domain data is that the model is initialized with weights trained on a larger dataset, which has a regularizing effect (Dakwale & Monz, 2017). Furthermore, it exposes the model to word types and co-occurrences of words that may not be present in the in-domain data, especially if the in-domain dataset is very small.

Another way to influence the model to influence the model's domain bias is by increasing the influence that in-domain data has during training, and to decrease the effect of out-of-domain data.

One simple way to achieve this is by oversampling the in-domain data, which leads to more weight updates for the oversampled corpus. Especially when there is only little in-domain data available relative to the out-of-domain data, it will make the model pay more balanced attention to both in-domain and out-of-domain data.

Another method to adapt the domain of an NMT model is to assign weights to the sentences based on a score assigned by a separate domain classifier (Chen et al., 2017). By multiplying the cost function with the score assigned by the classifier, the magnitude of the weight updates associated with a sentence is increased or decreased.

As a result, sentences that receive a high score on the domain classification will have a large influence on the model, while sentences that receive a lower score While cost weighting is a sub-corpus approach, it can also be applied at corpus level. When cost-weighting is applied at corpus level by assigning a weight to an entire corpus, it is referred to as *corpus weighting*, which achieves an effect similar to oversampling.

While NMT models are sensitive to the quantity of the data on which they are trained (Luong & Manning, 2015), studies have reported positive domain adaptation effects by being selective about which sentences to include in the training data. *Data selection* is a technique that has previously been applied to SMT successfully (Axelrod et al., 2011), and has now also seen some fruitful NMT implementations for the purpose of domain adaptation. In essence it involves any technique that eliminates data from the training data based on a given statistic. Chen et al. (2017) apply data selection by using an external language model to score sentences, after which they retain only the top 10% sentences. The selected 10% of the data is then used to fine-tune the model after training on the full data-set.

Another data selection implementation proposed by Wang et al. (2017) does not require the use of an external language model, and instead uses the internal representation of the source-side sentences created by the model itself. The Euclidean distance between the internal representation of candidate sentences and sentences that are known to be in-domain is then used as the 'domain score' for ranking the candidate sentences.

When both in-domain and out-of-domain corpora are available, ideally we want to use both the in-domain and out-of-domain corpora to train the model without the out-of-domain data hurting the performance on in-domain tasks. A translation model that is trained on multiple domains and allows for control over the domain during inference is called a *multi-domain system*. A simple and popular way to achieve this is by priming the model to produce translations for a given domain using *tags* (Sennrich et al., 2016a). By prepending an additional token representing the domain to the input sentence, the model can learn from out-of-domain data while being able to differentiate between the domains. One could for example label every sentence from a news corpus and a parliamentary corpus with the tags "<news>"

and "<parl>" respectively. Then at inference, these tags can be used to control the domain bias of the model for the translation.

4.6 Evaluation metrics for MT

Evaluation metrics play an important role in MT research. They are used to score a set of translated sentences by comparing them to a set of reference sentences. This score can be compared to the translations of other models to determine which model performs best.

Human evaluation of translation output is time consuming and expensive, and for languages which have only a small number of speakers it can be hard to even find someone able to provide human evaluation at all. This illustrates the need for automated metrics of quality estimation of MT systems for validation and scoring of MT systems.

There are several properties that make a metric suitable. Ideally, such a metric would correlate highly with human evaluation, because the goal is to provide an estimate of how a human would rate the quality. Additionally, such a metric should be language agnostic, so that it could be applied to any language without changes or retraining. Lastly, it should be quick to compute, so that it can be used for on-the-fly validation of a model during training, in order to select the best performing model, or to determine whether a model is still increasing in performance.

Finding evaluation metrics that fulfill all of the above requirements has proven to be a challenging task. The state-of-the-art evaluation metrics lack the ability to compare two sentences semantically, and instead look purely at the syntactic similarity. Such methods are therefore very sensitive to the choice of words, which can lead to a large difference between the scores of semantically equivalent translations (Papineni et al., 2002).

Although current implementations of automated evaluation metrics have clear limitations, they still provide a useful heuristic for the comparison of MT models, which

allows researchers to test and compare new ideas more easily.

4.6.1 BLEU

The Bilingual Evaluation Understudy (Papineni et al., 2002) (BLEU) method is an automated evaluation metric for MT translations. It works by computing the *modified n-gram precision* between the translation output and reference sentence, which is a modification on the simple precision metric. In simple terms, the more n-grams the candidate sentence and reference have in common, the higher the resulting score for the candidate sentences.

There are, however, two main differences between the precision and the modified n-gram precision in Papineni et al. (2002). The first that it works on n-grams instead of individual words. Papineni et al. (2002) argues that single word matches between candidate and reference sentence correlate with the adequacy of a translation, whereas longer n-gram matches correlate better with fluency. The other modification is that n-grams in the reference sentence become exhausted after matching once with an n-gram in the candidate sentence. This prevents multiple occurrences of the same n-gram in a candidate sentence triggering multiple rewards for each occurrence.

Although candidate sentences that are too long are already being penalized by the n-gram precision implementation, sentences that are too short are not. Therefore an additional *brevity penalty* is introduced, which penalizes the BLEU score by a factor of $\exp(1 - r/c)$ if the sentence is shorter than the reference sentence, where r represents the length of the reference sentences, and c represents the length of the candidate sentences.

Because BLEU operates at word-level, it requires that words in the candidate sentence match exactly with the reference words to get rewarded. A slight difference in the spelling of a word means that no score will be gained for that word. This is not much of a problem for languages with relatively short words, where an MT system will often get the word entirely correct, or not at all. For agglutinative and

polysynthetic languages on the other hand, it could be argued that partially correct words should also receive some reward, as such words are usually longer, and are decoded in more separate parts than words in other languages (Section 2.1).

4.6.2 Character-F

The Character-F (chrF) method (Popović, 2015) on the other hand operates on the character level. Whereas BLEU is based on the precision, chrF is based on the n-gram F-score, which is a balanced metric between recall and precision. This balance can be influenced by adjusting the β parameter, to assign more or less importance to precision versus recall.

The method has two adjustable parameters, the n-gram length, and the β parameter. Popović (2015) found empirically that the optimal n-gram length is 6. They found the optimal β to be 3, meaning that the recall has three times more weight than the precision for the F-score.

When referring to the character-F metric, *chrF* is commonly used to refer to the 6-gram chrF metric with $\beta = 1$. If a different β value is used, it is specified in the name, such as *chrF3* to refer to the character-F with $\beta = 3$.

In comparisons between the BLEU and character-F metrics, the character-F showed comparable or better correlation with human evaluation, with chrF3 being the best performing metric, outperforming three other state-of-the-art metrics in all experiments (Popović, 2015). We use the chrF metric as the primary metric for the EN $\rightarrow$ IU experiments, because it has been shown to correlate better than BLEU with human evaluation when the target language is agglutinative, as is the case with polysynthetic languages.

Chapter 5

Extrinsic Evaluation

This chapter compares the extrinsic quality of the segmentation methods described in Section 2.3. Section 5.1 describes the preprocessing steps applied to the data before segmentation. In Section 5.2 the included segmentation methods, the NMT model parameters, evaluation metrics, and procedure for training the translation models are described. The results of the experiments are reported and interpreted in Section 5.3.

5.1 Preprocessing

5.1.1 Normalization of Inuktitut syllabics

Included with the Hansard corpus (Section 2.2.1) are scripts that standardize the spelling of Inuktitut words, used by Joanis et al., 2020 in their preprocessing pipeline. The scripts mainly regularize some choices in spelling that are equivalent in meaning, and have a recommended form on Tusaalanga[16], a website offering free learning material for Inuktitut. The goal of regularizing the data is to increase consistency in the data to benefit machine translation models. We apply the conservative script

[16]https://tusaalanga.ca/

(*normalize-spelling.pl*) to the dataset, consisting of five general rules that can be safely applied without introducing new errors into the data.

5.1.2 Romanization

The documentation included with the Hansard dataset also describes how the romanization of Inuktitut can be done using the Uniconv and Iconv application included with Yudit. We follow these instructions and convert the Inuktitut syllabics into romanized form with Uniconv. This initially leads to some output errors, which are repaired by running Iconv on the romanized Inuktitut.

5.1.3 Further preprocessing

For each of the remaining preprocessing operations, we use commonly used scripts from the *Moses*[17] MT toolkit. After romanization of the Inuktitut, we apply punctuation normalization and tokenization to both the English and Inuktitut. We then train a truecase model for the English part of the corpus using the training data, and apply it to all of the corpora. Finally the data is cleaned with a minimum sentence length of 1 token, a maximum length of 200 tokens, and a maximum sentence length ratio of 30.

5.2 Experimental setup

In the extrinsic evaluation experiments we exclude the BPE models with 1,000 merge operations, because the results of the intrinsic evaluation show that they perform worse than the models with 5,000 merge operations. Also excluded is the Morfessor Categories-MAP model, because the algorithm does not provide functionality for the segmentation of a text file with multiple words per line like the other methods in the Morfessor family. The rule-based segmenter Uqailaut and neural Transformer

[17] https://github.com/moses-smt/mosesdecoder

models are excluded from the analysis, because of the failure to segment certain words as described in Section 3.2 and Section 2.3.6, which make them unsuitable for application as system's sole segmentation method. The 3-Step and 3-Step + LMVR models (Section 3.2) are included in the analysis, because they do not have this problem.

The models are trained using Marian (Junczys-Dowmunt et al., 2018), which is a free Neural Machine Translation framework written in C++. It includes ready-to-use implementations of the Transformer (Vaswani et al., 2017) and Deep-RNN architectures. For our experiments we use Marian v1.7.6, and train the models on the Peregrine High-Powered Computing cluster [18] of the University of Groningen. Each model is trained on a GPU node equipped with an Nvidia V100 GPU card with CUDA v10.1.243 for GPU communication.

All translation models trained use the `transformer` model type, which is predefined in Marian. We use the `ce-mean-words` cost function. The embedding dimension size is set to 512, with a 6-layer encoder and a 6-layer decoder (Vaswani et al., 2017). A beam size of 6 is used, and a Transformer-dropout of 0.1 is applied during training. Label smoothing is applied with an epsilon of 0.1. The models are trained with a batch size of 2000.

Every 5,000 update steps, validation is performed where the `ce-mean-words`, and perplexity are calculated, as well as a translation score. For the IU→EN models, the translation score is the BLEU score (Section 4.6.1), which is computed using the `multi-bleu.pl` script included in *Moses*[19]. For the EN→IU models, we compute the chrF (Section 4.6.2) as an assessment of the translation quality. We use early stopping as a stopping criterion to prevent overfitting, which occurs when the cost function does not improve for 5 consecutive validation calculations.

For each segmentation method we select the model with the best translation score on the validation set. We compare the segmentation methods by BLEU and chrF scores on two different held-out test sets with the selected models. The primary test

[18] https://wiki.hpc.rug.nl/peregrine/

[19] https://github.com/moses-smt/mosesdecoder

set, referred to as the *News test* set, contains 568 sentences of news data, obtained from the Nunatsiaq website. The other test set, referred to as the *Hansard test* set, contains 3,602 sentences from the same source as the training and validation sets.

We perform the same experiments also with 1,859 additional training data from the news domain to determine if there are differences between the models' ability to adapt to a new domain. Because of the small size of the added data in comparison to the Hansard training data, we oversample it ten times. The dataset on which these models are validated is the same as for the other models, and consists of 3,028 sentences from the Hansard dataset.

5.3 Results

The results of the extrinsic experiments are shown in Table 5.1 – 5.6.

5.3.1 Inuktitut to English

The results for the Inuktitut to English experiments (Table 5.1) show the Token BPE model outperforming other models on the Hansard test set, but scoring lower than LMVR, Type BPE, and 3-step + LMVR on the news test set in terms of BLEU score. This can be explained by the fact that the Token BPE segmentation model is biased by the frequencies of tokens in the training dataset. The Hansard dataset contains tokens that are specific to the parliamentary domain, and are frequent only within the domain. The IU → EN translation models seem to benefit from a more well-rounded source-side segmentation method.

The best performing model for the IU → EN direction was the 3-step + LMVR model, which outperformed LMVR by 0.27 BLEU, Type BPE by 0.4 BLEU, and Token BPE 0.48 BLEU. In terms of chrF, the Token BPE model performed best, improving over the 3-step + LMVR model by 0.1 chrF.

The best validation score was reached by the Morfessor model that was trained

IU → EN	**Hans. valid**		**Hans. test**		**News test**	
Model	**BLEU**	**chrF**	**BLEU**	**chrF**	**BLEU**	**chrF**
Token BPE (5K)	34.11	53.59	**28.31**	**48.19**	14.77	**37.61**
Type BPE (5K)	33.57	53.43	28.15	48.04	14.85	36.52
Morfessor (unique)	**34.46**	**54.08**	27.19	47.00	14.35	37.14
Morfessor (ambig.)	33.36	53.44	27.50	47.43	14.57	37.25
FlatCat	32.50	51.80	26.49	45.60	12.86	34.36
LMVR	32.85	52.53	27.5	47.20	14.98	36.88
3-step	30.83	50.49	24.56	43.64	11.31	30.33
3-step + LMVR	33.80	53.6	28.06	47.95	**15.25**	37.51

Table 5.1: BLEU and chrF scores for the Inuktitut to English translation models for the 8 segmentation methods on three different datasets. The *Hans. valid* dataset is the deduplicated Hansard validation set which was used to validate the models during training. *Hans. test* is a held-out deduplicated Hansard test set, and the *News test* dataset is the held-out test set with sentences extracted from the Nunatsiaq news website. The highest scores for each dataset and metric are in bold.

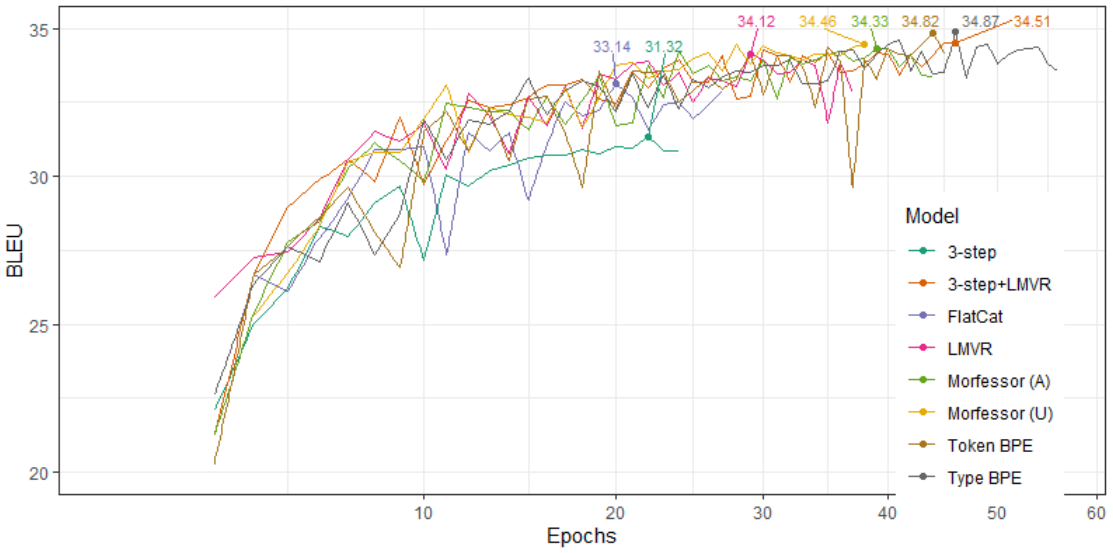

Figure 5.1: The progression of the validation BLEU score for each of the IU → EN translation models per training epoch. The best validation score is shown for each model as a dot. Training ended when a model's perplexity did not improve for subsequent validation calculations. Square-root scaling is applied to the X-axis for visual purposes.

IU → EN + News	**Hans. valid**		**Hans. test**		**News test**	
Model	**BLEU**	**chrF**	**BLEU**	**chrF**	**BLEU**	**chrF**
Token BPE (5K)	**37.79**	58.09	**28.20**	**48.04**	17.22	40.87
Type BPE (5K)	36.39	57.15	27.18	47.05	16.10	38.96
Morfessor (unique)	37.73	**58.33**	26.97	47.39	17.44	41.28
Morfessor (ambig.)	36.94	57.22	27.25	47.36	16.68	41.07
FlatCat	35.52	56.09	27.14	46.96	15.48	40.53
LMVR	36.67	57.31	27.83	47.74	**17.88**	**42.09**
3-Step	32.85	52.75	25.61	44.49	13.32	35.03
3-Step + LMVR	37.34	57.62	27.71	47.48	15.34	37.00

Table 5.2: BLEU and chrF scores for the Inuktitut to English translation models with additional training data from the Nunatsiaq news website for the 8 segmentation methods on three different datasets. The highest scores for each dataset and metric are in bold.

	Output
Source	maikul irngaut iglulikmiutaq tuktumi qukiqsilaursimangmat 2015-mi angunasuqujaunngitillugit
Reference	Michael Irngaut of Igloolik shot a caribou in 2015 while a hunting ban was in effect
Token BPE	Michael Irngaut from Igloolik shot a caribou moratorium in 2015.
Type BPE	Michael Irngaut of Igloolik's Tuktu shot in 2015 in order to hunt.
Morfessor (U)	Michael Irniq from Igloolik shot the caribou in 2015 without hunting.
Morfessor (A)	Michael Irniq from Igloolik caught a caribou hunting trip in 2015.
FlatCat	Michael Irniq from Igloolik, who shot caribou during the 2015 caribou hunt.
LMVR	Michael Irniq from Igloolik shot the caribou in 2015 when they were not allowed to go hunting.
3-step	Michael Irniq from Igloolik in Tuktu Caribou in 2015 to harvest caribou
3-step + LMVR	Michael Irniq of Igloolik shooted a caribou hunt in 2015 when he did not want to hunt.

Table 5.3: A sentence pair from the news test set and the respective translations generated by each of the IU → EN translation models. The displayed translations were produced by the systems trained without additional news training data.

on unambiguous segmentations. In general, the Morfessor Baseline models perform decent, but do not outperform the other methods on either of the test sets. We see that the Morfessor model trained on unambiguous segmentations outperforms the model trained on ambiguous segmentations by 0.76 BLEU on the news test set. There is no clear difference between the two models of BLEU scores on Hansard test set.

The lowest scores on the news and Hansard test sets were obtained by the 3-step model which scored 11.31 BLEU on the the news test set, 3.94 points lower than the 3-step + LMVR model. This can be explained by the large vocabulary size as a result of using the 3-step segmentation, and because it can produce out-of-

	Output
Ref	"What happens in one part of the planet has impact on us"
Token BPE	"If something happens in the world it affects us"
Type BPE	"Circumstances of the world affect us"
Morfessor (U)	"Any event in the world has an impact on all of us."
Morfessor (A)	"Any incident in the world would have an impact on all of us."
FlatCat	"In some parts of the world it affects all of us."
LMVR	"If something happens in the world it affects us"
3-step	"In some cases it affects all of us."
3-step + LMVR	"Whatever happens in the world is affecting us"

Table 5.4: Sample sentences for each of the IU→EN translation models. The displayed translationswere produced by the systems trained without additional news training data.

vocabulary morphs which lead to loss of information about the input sentence.

By including the news data in the training data, the scores on the Hansard validation are slightly increased, while the scores on the Hansard test set stay roughly the same. The BLEU scores on the news test set improve for all models. The best scores at 17.88 BLEU and 42.09 chrF is reached by the LMVR model, an improvement of 0.66 BLEU and 1.22 chrF over the Token BPE model. The 3-step + LMVR model appears to benefit the least from the addition of news data to the training set, with an increase of 0.09 BLEU and a decrease of 0.51 chrF.

The addition of news data strongly impacts the Token BPE model's relative performance, which after addition of the news data outperforms the Type BPE by 1.12 BLEU, and the 3-step + LMVR model by 1.88 BLEU. This can be explained by the fact that tokens that are less frequent in the Hansard data now also receive some activation because of the different distribution of token frequencies between the news and Hansard data. While the Token BPE model is biased towards the frequent tokens in the Hansard dataset and many of the most frequent tokens in the Hansard data are unfrequent in the News data, there is an overlap between the frequent tokens in both sets. Encoding those common frequent words into shorter

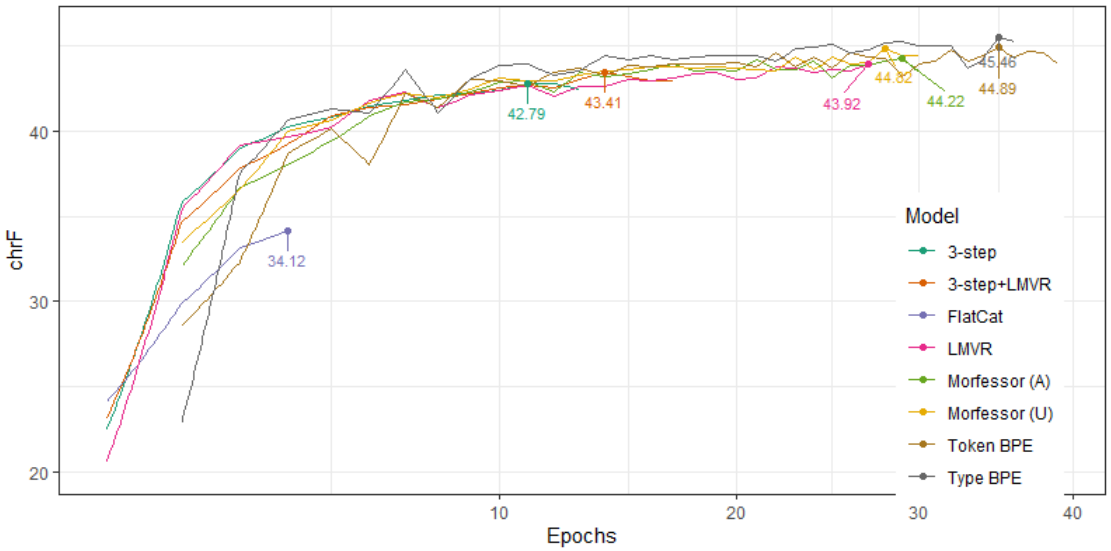

Figure 5.2: The progression of the validation BLEU score for each of the EN → IU translation models per training epoch. The best validation score is shown for each model as a dot. Training ended when a model's perplexity did not improve for subsequent validation calculations. Square-root scaling is applied to the X-axis for visual purposes.

sequences appears to still benefit the translation model.

5.3.2 English to Inuktitut

For the English to Inuktitut experiments, the results (Table 5.5) show the Type BPE model outperforming the other models on both the Hansard validation and test sets. The LMVR model was the best performing model on the news test set, with a chrF score of 34.84, and a BLEU score of 3.67. This is an improvement of 0.33 chrF and 0.08 BLEU over the 3-step + LMVR model, and 2.32 chrF and 0.32 BLEU over the Token BPE model. Both LMVR-based models appear to generalize better to the news test data than the BPE models, which indicates that LMVR is a good choice for segmentation on the Inuktitut target-side. The Type-BPE model does not appear to generalize well to the news test data. While its BLEU score is competitive with the other models, the chrF score is fairly low at 28.71.

The lowest scores are obtained by the FlatCat model. Although the validation chrF

EN → IU	**Hans. valid**		**Hans. test**		**News test**	
Model	**BLEU**	**chrF**	**BLEU**	**chrF**	**BLEU**	**chrF**
Token BPE (5K)	18.45	44.89	12.09	39.81	3.35	32.52
Type BPE (5K)	18.68	**45.46**	**14.64**	**41.82**	3.26	28.71
Morfessor (unique)	**18.76**	44.88	14.45	41.51	3.44	32.61
Morfessor (ambig.)	18.35	44.21	13.95	40.92	3.19	31.71
FlatCat	8.52	43.41	6.13	29.88	0.43	23.25
LMVR	16.82	43.92	13.10	41.25	**3.67**	**34.84**
3-step	17.32	42.79	13.29	39.33	2.64	31.34
3-step + LMVR	16.90	43.41	12.84	40.54	3.59	34.51

Table 5.5: BLEU and chrF scores for the English to Inuktitut translation models for the 8 segmentation methods on three different datasets. The highest scores for each dataset and metric are in bold.

EN → IU + News	**Hans. valid**		**Hans. test**		**News test**	
Model	**BLEU**	**chrF**	**BLEU**	**chrF**	**BLEU**	**chrF**
Token BPE (5K)	**21.64**	**48.21**	13.99	41.20	6.42	36.75
Type BPE (5K)	21.09	47.91	13.69	40.62	**6.86**	36.65
Morfessor (unique)	20.30	47.14	**14.45**	41.31	6.15	36.25
Morfessor (ambig.)	20.43	47.65	14.28	**41.58**	6.27	**38.96**
FlatCat	9.24	36.46	7.85	33.36	2.00	30.18
LMVR	18.31	45.40	12.77	40.68	4.45	34.49
3-step	17.43	41.75	13.23	38.59	3.70	34.36
3-step + LMVR	19.11	46.14	12.86	40.57	5.54	35.53

Table 5.6: BLEU and chrF scores for the English to Inuktitut translation models with additional training data from the Nunatsiaq news website for the 8 segmentation methods on three different datasets. The highest scores for each dataset and metric are in bold.

score is on par with other models, its BLEU score was significantly lower, as well as the chrF scores on both test sets. The poor performance of the FlatCat model suggests that the segmentation could be too inconsistent for the translation model to learn to combine the right morphs. The vocabulary size of the training data for this model supports this hypothesis, as it is the largest vocabulary size of all models at 167,404. The translation output shows that although the model performs decent on short sentences, it is prone to start repeating a single token for longer sentences. This suggests that the model may not have converged before early stopping, further evidenced by the relatively low number of epochs reached during training (Figure 5.2), likely also as a result of the large vocabulary size.

The results on the news test set show more variation between them than the results on the Hansard test set, suggesting that the choice of segmentation method has the biggest influence when applied to data outside of the domain of the training set.

The addition of news training data overall leads to increased performance on the news test data and higher validation scores. The Morfessor Baseline model trained on ambiguous data outperforms the other models with a margin, at 38.96 chrF. This is an increase of 7.25 chrF over the same model trained without the added news data. Both of the BPE models also benefit greatly from the additional data. The LMVR-based models are influenced to a lesser degree. While the BLEU of both models increases, the chrF score of the LMVR model decreases. The 3-step + LMVR model outperforms the regular LMVR model with over a point in terms of BLEU and chrF.

Chapter 6

Conclusion

With the results obtained from the intrinsic and extrinsic evaluation experiments, we can now reflect on the research questions. We asked:

- which segmentation approaches yield the most linguistically correct segmentations (RQ1);
- which segmentation approaches lead to the best performance of an NMT system (RQ2);
- whether more linguistically correct morphological segmentation correlates to a better performance of an NMT system (RQ3);
- whether there is a difference between the performance of segmentation approaches when Inuktitut is the source language, versus when it is the target language (RQ4);
- to which degree the segmentation approach impacts the system's ability to generalize to a different domain (RQ5).

From the results of the intrinsic evaluation (RQ1) it is clear that the long words and high morpheme-to-word ratio of polysynthetic languages are challenges for morphological segmentation. The results show large differences between the ability of the compared approaches to produce correct segmentations. Out of the existing

language-agnostic methods that were tested, semi-supervised Morfessor Baseline appears best able to handle Inuktitut's complex morphology when trained with unambiguous segmentations as annotated data. We reason that the semi-supervised training offered by Morfessor enable it to better adapt to the high morpheme-to-word ratio of a polysynthetic language. While FlatCat also offers semi-supervised training, its added morpheme categorization might be detrimental to the intrinsic performance on Inuktitut as a result of the miscategorization of morphs.

Our Transformer-based segmentation models outperformed existing trainable methods with a considerable margin. An advantage is that it is a neural approach, which allows it to adapt to a more complex morphology than existing language-agnostic methods based on morph vocabularies. Our results show that such models benefit from unambiguous training data, even when the quantity of *possibly ambiguous* data is an order of magnitude larger. A disadvantage of the method is that it only allows supervised training, thus requiring a large collection of annotated segmentations. To overcome the need for many annotated segmentations, we hypothesize that common techniques for low-resource NMT could be successfully adapted for segmentation, such as the use of auxiliary tasks and multi-lingual training, which could be investigated in future research.

The extrinsic results (RQ2) show that BPE models are the best choice for segmentation approach when the target domain is the same as the domain of the training data. For IU→EN, the best BPE model appears to be the Token BPE model, whereas the Type BPE model performs better for EN→IU. For both translation directions LMVR-based models produce the best results when the target domain is different than the trained domain. This supports the hypothesis that BPE models can be too specific to the domain of the training data, especially when domain-specific phrases are repeated often, as is the case with the Hansard corpus.

With the results of the intrinsic and extrinsic evaluation combined, we cannot conclude that a more linguistically correct segmentation leads to a better translation quality (RQ3). There are multiple factors that could explain the fact that the intrinsically better methods scored lower on the extrinsic evaluation. Most importantly, the sub-word vocabularies resulting from morphological segmentation become too large

for NMT models to properly converge. This is for a large part because polysynthetic morphemes can take many different surface forms as a result of inflection. Especially in a low-resource setting it is important that the vocabulary does not become too large, otherwise many sub-words will become very infrequent and the model will not learn much. Furthermore, linguistically correct segmentation of Inuktitut leads to long sequences of short morphs, which could become more difficult for the translation model to correlate. Another possible reason is that the short morphemes in Inuktitut individually can carry abstract semantic concepts, which could be too isolated for the NMT model to successfully derive their meaning from the training data.

Between the two translation directions there does not appear to be a clear difference in the preferred segmentation method (RQ4). While the performance of each model varies between translation EN→IU and IU→EN, generally the methods that perform well in one direction also perform well in the other direction.

Regarding RQ5, the relative performance of the translation models appears to change based on the target domain. Our results showed that for translation of data from a different domain than the training data, BPE is outperformed by other models including LMVR-based models and semi-supervised Morfessor Baseline. The low performance of the BPE models before the addition of news data could be because the BPE morphs that are more frequent in news data did not get much activation during training, whereas the activation of LMVR's less corpus biased morph vocabulary was better distributed before the addition of news data.

The effect that the addition of news data to the training data was different for each of the segmentation methods. LMVR appears the best suited for situations where a small amount of in-domain data is available for domain adaptation. We reason that the combination of a small vocabulary size and a reduced corpus bias for the segmentations make it the most suitable option for translation of data from a different domain.

Directions for future research include the development of segmentation methods that can deal with the complex morphology of polysynthetic languages. Existing methods that work for other languages, produce inconsistent segmentations for the

long morph sequences of polysynthetic languages. To improve translation quality for polysynthetic languages, it is important to allow some control over the vocabulary size.

Considering the good extrinsic performance of LMVR, future research could further investigate the relationship between translation performance and vocabulary size by making a comparison between LMVR models with various vocabulary sizes.

Lastly, while the fully supervised Transformer segmentation yielded promising results for the intrinsic evaluation, the method still has some clear disadvantages that prevent its application to NMT that can be further investigated in future research. If a solution can be found for the failure to segment certain words, it could be successful as a segmentation method for NMT systems for polysynthetic languages.

Bibliography

Mielke, S. J., Cotterell, R., Gorman, K., Roark, B., & Eisner, J. (2019). What Kind of Language Is Hard to Language-Model? *CoRR*, *abs/1906.04726* arXiv 1906.04726. http://arxiv.org/abs/1906.04726

Kalchbrenner, N., & Blunsom, P. (2013). Recurrent Continuous Translation Models, In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, Seattle, Washington, USA, Association for Computational Linguistics. https://www.aclweb.org/anthology/D13-1176

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need.

Kann, K., Mager Hois, J. M., Meza-Ruiz, I. V., & Schütze, H. (2018). Fortification of Neural Morphological Segmentation Models for Polysynthetic Minimal-Resource Languages, In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, New Orleans, Louisiana, Association for Computational Linguistics. https://doi.org/10.18653/v1/N18-1005

Papineni, K., Roukos, S., Ward, T., & Zhu, W.-J. (2002). BLEU: a method for automatic evaluation of machine translation, In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*.

Popović, M. (2015). chrF: character n-gram F-score for automatic MT evaluation, In *Proceedings of the Tenth Workshop on Statistical Machine Translation*.

Ataman, D., Negri, M., Turchi, M., & Federico, M. (2017). Linguistically motivated vocabulary reduction for neural machine translation from Turkish to English. *The Prague Bulletin of Mathematical Linguistics*, *108*(1), 331–342.

Harris, Z. S. (1955). From phoneme to morpheme, In *Papers in structural and transformational linguistics*. Springer.

Ruokolainen, T., Kohonen, O., Sirts, K., Grönroos, S.-A., Kurimo, M., & Virpioja, S. (2016). A comparative study of minimally supervised morphological segmentation. *Computational Linguistics*, *42*(1), 91–120.

Phillips, C. (1994). Verbal case and the nature of polysynthetic inflection. *Proceedings of CONSOLE II, Leiden. To Appear*.

Mager, M., Mager, E., Medina-Urrea, A., Meza Ruiz, I. V., & Kann, K. (2018). Lost in Translation: Analysis of Information Loss During Machine Translation Between Polysynthetic and Fusional Languages, In *Proceedings of the Workshop on Computational Modeling of Polysynthetic Languages*, Santa Fe, New Mexico, USA, Association for Computational Linguistics. https://www.aclweb.org/anthology/W18-4808

Joanis, E., Knowles, R., Kuhn, R., Larkin, S., Littell, P., kiu Lo, C., Stewart, D., & Micher, J. (2020). The Nunavut Hansard Inuktitut–English Parallel Corpus 3.0 with Preliminary Machine Translation Results, In *Proceedings of LREC-2020*, Marseille, France.

Littell, P., Kazantseva, A., Kuhn, R., Pine, A., Arppe, A., Cox, C., & Junker, M.-O. (2018). Indigenous language technologies in Canada: Assessment, challenges, and successes, In *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA, Association for Computational Linguistics. https://www.aclweb.org/anthology/C18-1222

Creutz, M., & Lagus, K. (2002). Unsupervised Discovery of Morphemes, In *Proceedings of the ACL-02 Workshop on Morphological and Phonological Learning*, Association for Computational Linguistics. https://doi.org/10.3115/1118647.1118650

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471.

Kohonen, O., Virpioja, S., & Lagus, K. (2010). Semi-Supervised Learning of Concatenative Morphology, In *Proceedings of the 11th Meeting of the ACL Special Interest Group on Computational Morphology and Phonology*,

Uppsala, Sweden, Association for Computational Linguistics. https://www.aclweb.org/anthology/W10-2210

Creutz, M., & Lagus, K. (2005). Inducing the Morphological Lexicon of a Natural Language from Unannotated Text, In *In Proceedings of the International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning (AKRR'05*.

Grönroos, S.-A., Virpioja, S., Smit, P., & Kurimo, M. (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: Technical Papers*.

Sennrich, R., Haddow, B., & Birch, A. (2016b). Neural Machine Translation of Rare Words with Subword Units, In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Berlin, Germany, Association for Computational Linguistics. https://doi.org/10.18653/v1/P16-1162

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural Machine Translation by Jointly Learning to Align and Translate.

Toral, A., Edman, L., Yeshmagambetova, G., & Spenader, J. (2019). Neural Machine Translation for English–Kazakh with Morphological Segmentation and Synthetic Data, In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*.

Kann, K., Cotterell, R., & Schütze, H. (2016). Neural morphological analysis: Encoding-decoding canonical segments, In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*.

Qiu, X., Pei, H., Yan, H., & Huang, X. (2019). Multi-Criteria Chinese Word Segmentation with Transformer. *CoRR*, *abs/1906.12035* arXiv 1906.12035. http://arxiv.org/abs/1906.12035

Caruana, R. (1997). Multitask learning. *Machine learning*, *28*(1), 41–75.

Jiang, W., & Tang, Y. (2019). A Seq-to-Seq Transformer Premised Temporal Convolutional Network for Chinese Word Segmentation. *CoRR*, *abs/1905.08454* arXiv 1905.08454. http://arxiv.org/abs/1905.08454

Singh, V. (2017). Replace or Retrieve Keywords In Documents at Scale. *ArXiv e-prints*, arXiv 1711.00046.

Hutchins, J. (2007). Machine translation: A concise history. *Computer aided translation: Theory and practice*, *13*(29-70), 11.

Berger, A. L., Brown, P. F., Della Pietra, S. A., Della Pietra, V. J., Gillett, J. R., Lafferty, J. D., Mercer, R. L., Printz, H., & Ures, L. (1994). The Candide System for Machine Translation, In *Human Language Technology: Proceedings of a Workshop held at Plainsboro, New Jersey, March 8-11, 1994*. https://www.aclweb.org/anthology/H94-1028

Xiong, D., & Zhang, M. (2016). *Linguistically Motivated Statistical Machine Translation*. Springer.

Koehn, P., Och, F. J., & Marcu, D. (2003). *Statistical phrase-based translation* (tech. rep.). UNIVERSITY OF SOUTHERN CALIFORNIA MARINA DEL REY INFORMATION SCIENCES INST.

Yamada, K., & Knight, K. (2001). A syntax-based statistical translation model, In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*.

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks, In *Advances in neural information processing systems*.

Sutskever, I., Vinyals, O., & Le, Q. V. (2014). Sequence to Sequence Learning with Neural Networks. *CoRR*, *abs/1409.3215*arXiv 1409.3215. http://arxiv.org/abs/1409.3215

Allen, R. B. (1987). Several studies on natural language and back-propagation, In *Proceedings of the IEEE First International Conference on Neural Networks*. IEEE Piscataway, NJ.

Neco, R. P., & Forcada, M. L. (1997). Asynchronous translations with recurrent neural nets, In *Proceedings of International Conference on Neural Networks (ICNN'97)*. IEEE.

Cho, K., van Merrienboer, B., Bahdanau, D., & Bengio, Y. (2014). On the Properties of Neural Machine Translation: Encoder-Decoder Approaches. *CoRR*, *abs/1409.1259*arXiv 1409.1259. http://arxiv.org/abs/1409.1259

Luong, M.-T., Pham, H., & Manning, C. D. (2015). Effective Approaches to Attention-based Neural Machine Translation.

Koehn, P. (2020). *Neural machine translation*. Cambridge University Press.

Barrault, L., Bojar, O., Costa-Jussà, M. R., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Koehn, P., Malmasi, S., Et al. (2019). Findings of the 2019 conference on machine translation (wmt19), In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*.

Vania, C., & Lopez, A. (2017). From Characters to Words to in Between: Do We Capture Morphology? *CoRR*, *abs/1704.08352*arXiv 1704.08352. http : //arxiv.org/abs/1704.08352

Sennrich, R., & Zhang, B. (2019). Revisiting Low-Resource Neural Machine Translation: A Case Study, In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, Florence, Italy, Association for Computational Linguistics. https://doi.org/10.18653/v1/P19-1021

Sennrich, R., Haddow, B., & Birch, A. (2015). Improving Neural Machine Translation Models with Monolingual Data.

Koehn, P., & Knowles, R. (2017). Six challenges for neural machine translation. *arXiv preprint arXiv:1706.03872*.

Chu, C., & Wang, R. (2018). A Survey of Domain Adaptation for Neural Machine Translation, In *Proceedings of the 27th International Conference on Computational Linguistics*, Santa Fe, New Mexico, USA, Association for Computational Linguistics. https://www.aclweb.org/anthology/C18-1111

Luong, M.-T., & Manning, C. D. (2015). Stanford neural machine translation systems for spoken language domains, In *Proceedings of the International Workshop on Spoken Language Translation*.

Dakwale, P., & Monz, C. (2017). Finetuning for neural machine translation with limited degradation across in-and out-of-domain data. *Proceedings of the XVI Machine Translation Summit*, *117*.

Chen, B., Cherry, C., Foster, G., & Larkin, S. (2017). Cost weighting for neural machine translation domain adaptation, In *Proceedings of the First Workshop on Neural Machine Translation*.

Axelrod, A., He, X., & Gao, J. (2011). Domain Adaptation via Pseudo In-Domain Data Selection, In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, UK., Association for Computational Linguistics. https://www.aclweb.org/anthology/D11-1033

Wang, R., Finch, A., Utiyama, M., & Sumita, E. (2017). Sentence embedding for neural machine translation domain adaptation, In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*.

Sennrich, R., Haddow, B., & Birch, A. (2016a). Controlling politeness in neural machine translation via side constraints, In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*.

Junczys-Dowmunt, M., Grundkiewicz, R., Dwojak, T., Hoang, H., Heafield, K., Neckermann, T., Seide, F., Germann, U., Fikri Aji, A., Bogoychev, N., Martins, A. F. T., & Birch, A. (2018). Marian: Fast Neural Machine Translation in C++, In *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia, Association for Computational Linguistics. http://www.aclweb.org/anthology/P18-4020

List of Figures

2.1 The Inuktitut syllabic alphabet. 12
2.2 An example of Byte Pair Encoding. 18

3.1 Overview of the F1 and accuracy results of intrinsic evaluation. . . . 30

4.1 Required rule-sets of interlangual versus transfer-base RBMT. 39
4.2 Encoder-decoder model. 43
4.3 Attention model. 45
4.4 Alignment matrix showing non-monotonous alignment. 46
4.5 The Transformer architecture. 48
4.6 The role of morphological segmentation in a translation pipeline. . . 49

5.1 Progression of validation scores for IU → EN translation models. . . . 62
5.2 Progression of validation scores for EN → IU translation models. . . . 65