UNIVERSITY OF LATVIA
FACULTY OF COMPUTING

MATĪSS RIKTERS

HYBRID MACHINE TRANSLATION BY COMBINING OUTPUT FROM MULTIPLE MACHINE TRANSLATION SYSTEMS

DOCTORAL THESIS
Submitted for the degree of Doctor of Philosophy in Computer Science (Dr. sc. comp.)

Field: Computer Science
Subfield: Software and Systems Engineering

Scientific supervisor:
Dr. sc. comp., Professor **Inguna Skadiņa**

RIGA 2019

ABSTRACT

This thesis aims to research methods and develop tools that allow to successfully combine output from various machine translation (MT) systems so that the overall translation quality of the source text would increase. An applicability of the developed methods for small, morphologically rich and under-resourced languages is evaluated, especially Latvian and Estonian. Existing methods have been analysed and several combinations of methods have been proposed. The proposed methods have been implemented and evaluated using automatic and human evaluation. During this research novel methods have been created that structure source language sentences into linguistically motivated fragments and combine them using a character level neural language model; combine neural machine translation output by employing source-translation attention alignments; use a multi-pass approach to produce additional incrementally improving training data. The key results of this research are new state-of-the-art machine translation systems for English ↔ Estonian; approaches for utilising neural MT generated attention alignments for MT combination and comprehension of resulting translations; MT combination systems for combining output from English → Latvian statistical MT. A practical application of the methods is implemented and described.

Keywords: Machine Translation, Hybrid Machine Translation, Machine Translation System Combination, Multi-System Machine Translation

ACKNOWLEDGEMENTS

The author wishes to thank his supervisor, Dr. sc. comp., Professor Inguna Skadiņa, for guidance through the research process, constructive criticism towards the authors research decisions, for providing important feedback for the thesis and the involved publications.

A big part of the research in this thesis would not have been possible without help from Mark Fishel and the Institute of Computer Science of University of Tartu, as well as Ondřej Bojar and the Institute of Formal and Applied Linguistics of Charles University. Both welcomed the author as an exchange student and offered guidance during the exchange, as well as further collaboration after the exchange was completed.

Special thanks have to be expressed to Tilde and colleagues in Tilde - Mārcis Pinnis, Raivis Skadiņš, Rihards Krišlauks, Valters Šics and others for giving advice from their professional experience and also constructively criticising questionable decisions made along the research and development process of the methods described in the thesis.

The author also wishes to thank Lelde Bērziņa for designing multiple posters for conferences as well as suggesting design changes for many self-designed posters. Additional gratitude needs to be expressed to the Institute of Electronics and Computer Science and Microsoft Azure for Research for providing GPU computing resources and the Institute of Mathematics and Computer Science of the University of Latvia for computing resources.

CONTENTS

1. INTRODUCTION

This chapter gives a brief introduction to the research area, motivates the research and describes the aims of the research. Further, several key results of the thesis are listed along with a specification of the author's contribution for each one. The end of this section outlines the structure of the remainder of the thesis.

The structure of this chapter is as following: section 1.1 introduces the research area of the work. Section 1.2 describes the motivation and section 1.3 – the aims of the research. Section 1.4 outlines the main results of experiments conducted during the research, section 1.5 describes several practical use-cases that have been developed in the course of this research. Finally, section 1.6 lists the main publications and presentations that are related to the research and section 1.7 outlines the structure of the thesis.

1.1 RESEARCH AREA

The research area is focused on one of the primary use-cases for the modern computer – machine translation (MT). Literature (Hutchkins, 2005) states that the first ideas of MT originated in the mid-1930s, however real research on the subject began only after the first computers were invented. In 1949, the "Translation memorandum" was proposed by Warren Weaver to apply methods from the field of communication theory, such as cryptographic and statistical techniques, to solve the text translation problem. Although references to the subject can be found as early as the 17th century. One of the earliest recorded MT projects was the Georgetown experiment (Dostert, 1954) in 1954, which involved successful fully automatic translation of more than sixty Russian sentences into English.

Nevertheless, early efforts in the field of MT were not overly convincing that automatic translation of adequate quality was actually possible in principle. The ALPAC (Automatic Language Processing Advisory Committee) report in 1966 concluded that for the last 10 years MT research had not fulfilled the expectations of the Georgetown experiment, dramatically reducing funding for MT research at that time (Pierce and Carroll, 1966). Thus, switching the focus towards tools for aiding human translators instead of fully automated translation.

Later, rule-based MT (RBMT) systems started dominating the field of MT, typically through some variety of intermediary linguistic representation involving morphological, syntactic, and semantic analysis. These systems required a high amount of manual work, as they utilised hand-crafted dictionaries, rules, patterns and exceptions to translate texts.

The next big phase of MT started in the late 1980s and early 1990s when computational power increased and became less expensive more interest started to grow towards statistical MT (SMT). In 1993, researchers from IBM introduced the IBM Models (Brown et al. 1993) – a set of five statistical models for MT. IBM models are the foundation for modern phrase-based SMT.

These days most commercial MT systems are built using a variety of statistical approaches and the most recent - neural network-based neural MT (NMT) approaches. Starting from 2015 NMT systems slowly began outperforming SMT in particular shared tasks for MT (Bojar et al., 2015, Bojar et al., 2016). In 2016 industry giants like Google (Johnson et al., 2016) and Systran (Crego et al., 2016) introduced their commercial NMT systems as well as first NMT for smaller languages by Tilde (Pinnis, 2016). Although some of the historically first rule-based MT systems are still in use today or added as a part of some hybrid MT (HMT) setup, most of the modern MT systems are built using corpus-based approaches (neural network and statistical methods).

Currently MT has not yet reached a level of quality where it can fully replace a human translator, and it probably will not reach this level in any near future. However, MT has become a highly useful utility in scenarios such as providing a starting translation for post-editing or extracting information from texts in foreign languages. For the world to become ever more multicultural, the demand for faster and cheaper translation has breed many commercial products (e.g. IBM WebSphere Translation Server, Systran, SDL BeGlobal) and multiple translation services are freely available on the web or as mobile applications (e.g. Google Translate[1], Bing Translator[2], Yandex.Translate[3], Baidu Translate[4], Tilde Translator[5]), demonstrating high translation quality for a wide variety of languages.

A lot of current research focuses on MT for the widely-used languages, like English, Chinese, Spanish, Portuguese, French, Arabic, Japanese and Russian, as well as languages that appear in competition shared tasks, like Czech, Finnish and Turkish. Much less work is being done in the area of hybrid methods, for instance, combining multiple different paradigms to utilise their strengths and cover weaker points. Smaller languages like the Baltic three - Estonian, Latvian and Lithuanian are far less resourced in available MT services, or even language technologies in general, which lack sophistication due to little available linguistic resources and technological approaches that enable development of cost-effective MT services for new language pairs. This has caused a technological gap to emerge between the two groups of languages.

Some systems like Google Translate, Bing Translator, Yandex Translator and Baidu Translate are freely available as online services and broaden the set of inter-translatable language pairs, even incorporating the Baltic languages as well as many other less resourced languages. Typically, these online translation services are employed to translate short texts by occasional users. Another common use-case is the translation of websites and, most recently, social media posts.

1.2 MOTIVATION OF THE RESEARCH

Even though research in the field of machine translation has been going on for more than a half of a century and the number of different MT engines is ever growing, the initial goal of replacing human translators is far from being met. The current systems are not yet fully able to produce translations of the same quality as human translators (Hutchins, 2006).

Rule-based, statistical and neural MT methods all have both stronger points as well as some noticeable weaknesses.

Rule-based MT (RBMT) systems can achieve a high-quality translation if they have a full set of the knowledge necessary. While this can be done for narrow domain texts and very specific MT use-cases, a fully general RBMT system is not possible. RBMT typically handles specific language phenomena like word agreements, inflections, long distancer reordering, and long-distance dependency, etc. better. The output of RBMT systems is predictable and therefore more consistent, making it easy to locate and correct the cause of translation errors. Unfortunately, real-world human languages are complex with many ambiguities and exceptions, as well as always changing as time moves forward. While it is completely possible to advance RBMT, it soon becomes too complex and labour-intensive due to linguistic expertise and domain knowledge needed to create RBMT systems. The RBMT

[1] https://translate.google.com
[2] https://www.bing.com/translator
[3] https://translate.yandex.com
[4] http://translate.baidu.com
[5] https://translate.tilde.com

knowledge of a system for one specific language pair in one specific domain typically is not reusable in another language pair or domain.

In contrast to RBMT, SMT systems do not need manually written knowledge sets like dictionaries and rules. Most SMT systems usually consist of subcomponents that are trained and optimized for usage separately, but with the same sets of data. The knowledge is automatically learned by training statistical models on large datasets. This makes improving the systems as well as adapting them to other language pairs more flexible as all they require is more data. Training SMT models from large amounts of data used to be computationally expensive, but that is no longer the case. Learning from data is challenging for highly inflectional languages that have too many word forms, cases, etc. for all possible word form and sentence construction variants to appear in the training data. Therefore, SMT still struggles with word agreements, inflections, long distance reordering, and long-distance dependencies. A large high-quality parallel corpus is essential for corpus-based MT, but it is often unavailable for small and less popular languages.

Similar to SMT, NMT is also trained on a large amount of parallel data. It is significantly more computationally expensive than SMT for both training the models and using them to translate texts. Another big difference is that neural systems are usually trained end-to-end without any subcomponents. Some drawbacks of NMT include struggles in rare word translation and sometimes even a complete failure to translate all given source sentence words. In addition, since some NMT systems do translation in the character level and not the word level, they have a tendency to make up new words that may almost look real but in fact, do not exist. However, the advantages definitely are in generalization and handling inflections.

Given that all of the MT methods have their given advantages and drawbacks, it is reasonable to try to combine results from different MT systems to fix the mistranslations produced by one system with the help of the other systems. In addition, given that the Latvian language is small, has a complex grammar, rich morphology and limited amount of qualitative data, pure data-driven methods may not be sufficient. The complex grammar makes using pure knowledge-based methods difficult as well. Combining results from several approaches has the potential to produce a better final result.

1.3 AIM OF THE RESEARCH

The focus of this research is the problem of combining output from multiple different machine translation systems to acquire one superior final translation. This is an area that, when perfected, can achieve ever better results with every other single MT method (used here as a component) that improves upon itself.

This thesis describes problematic areas related to machine translation, limitations of current MT methods and provides suggestions on how to combine translations to achieve better overall quality of MT.

The main goal is to assemble a set of methods that would be able to improve the quality of MT output for the Baltic languages that are small, have a rich morphology and little resources available. These characteristics currently make them rather difficult to translate with the tools that are currently available.

The research primarily focuses on solving MT problems that are related to translating from and into Latvian. Nevertheless, the aim is to find such methods that may be applied other languages as well.

For his research, the author has suggested the following **hypothesis:**

Combining output from multiple different MT systems makes it possible to produce higher quality translations for the Baltic languages than the output that is produced by each component system individually.

The goal of this research is to create a method for combining output from multiple MT systems that provides a higher overall translation quality. This goal encompasses all of the following major aspects:

- An analysis of RBMT, SMT and NMT methods as well as existing HMT and multi-system MT (MSMT) methods;
- Experiments with different methods for combining translations;
- MT quality evaluation;
- Applicability of methods for Estonian, Latvian, Lithuanian, and other less resourced languages;
- Practical applications of MT combining.

1.4 RESEARCH METHODS

The following research methods were used in this thesis:

- Literature review – in order to identify modern state-of-the-art methods related to the thesis, publications from the leading natural language processing (NLP) conferences and workshops were analysed, as well as publication preprints and open-source implementations of relevant toolkits;
- Iterative development – most of the solutions described in this thesis are also implemented as open-source software and iteratively improved during the course of the research;
- Controlled experiments – to empirically verify the performance of the described methods and compare them to the corresponding baselines and related work, one or several controlled experiments were executed;
- Automatic evaluation – in order to quickly verify experiment results, automatic evaluation was performed, often by comparing experiment results against manually prepared resources;
- Manual evaluation – in order to fully verify experiment results, manual evaluation was performed where applicable and possible to complement automatic evaluation;
- Error analysis – in order to identify areas for further improvement, manual analysis and classification of systematic errors was performed where applicable and necessary.

1.5 KEY RESULTS OF THE RESEARCH

The main contributions of the thesis are as follows:

- Research has been conducted on all modern existing MT techniques with a focus on ways to combine them for increased MT quality;
- Several methods for combining translations have been implemented and evaluated:
    - Multi-System machine translation using online APIs for English-Latvian (Rikters, 2015);

- Syntax-based Multi-System Machine Translation (Rikters and Skadiņa, 2016a);
- Combining machine translated sentence chunks from multiple MT systems (Rikters and Skadiņa, 2016b);
- Interactive Multi-System Machine Translation with Neural Language Models (Rikters, 2016a);
- Combining Neural Machine Translation output using attention alignments (Rikters and Fishel, 2017)
- Incrementally augmenting training data for NMT (Pinnis et al., 2018)

- The method that improves MT quality the most is the application of neural network language models for candidate scoring in multi-system MT (MSMT) (Rikters, 2016d). This method was able to outperform baseline English-Latvian MT systems in both - automatic evaluation as well as all other methods for combining translations. The method was also tested on the English-Estonian language pair and may be applied for translation into other morphologically rich languages.
- The method that achieves the highest overall MT quality is using a multi-pass approach to incrementally augment training data for NMT (Pinnis et al., 2018). It outperformed most of the competition in the annual international WMT news translation competition for English-Estonian, reaching 3rd place according to automated evaluation. The method has also been applied to English-Lithuanian, English-Latvian and other morphologically-rich less resourced languages.

1.6 PRACTICAL IMPLEMENTATION OF THE RESEARCH

Most of the code that has been developed during this research has been made publicly available on the authors' private GitHub page[6]. Each of the separate projects is located in their respective repositories. Additionally, some systems have live demos available online[7]. Several implementations are published in Tilde's GitHub page[8].

The main practical implementations are 1) a toolkit for visualizing and debugging neural machine translations (described in detail in section 5.2); and 2) a toolkit for cleaning corpora (described in detail in section 5.3).

1.7 AUTHOR'S PUBLICATIONS RELATED TO THE RESEARCH

The thesis is based on the author's contributions to the following 17 publications:

- 10 publications in peer-reviewed conference proceedings recognised by the Latvian Council of Science:
  - Rikters, M., Pinnis, M. (2018, December) Debugging Translations of Transformer-based Neural Machine Translation Systems. Baltic Journal of Modern Computing. The author's contribution to the paper is 85%.

[6] M4t1ss on GitHub - https://github.com/M4t1ss
[7] NLP project demos - http://nlp.lielakeda.lv
[8] Tilde on GitHub - https://github.com/tilde-nlp

- Rikters, M. (2018, September) Impact of Corpora Quality on Neural Machine Translation. In The 8th Conference on Human Language Technologies - the Baltic Perspective (Baltic HLT 2018). The author's contribution to the paper is 100%.
- Rikters, M., Pinnis, M., Rozis, R., Krišlauks, R. (2018, September) Advancing Estonian Machine Translation. In The 8th Conference on Human Language Technologies - the Baltic Perspective (Baltic HLT 2018). The author's contribution to the paper is 75%.
- Rikters, M. (2018, July). Debugging Neural Machine Translations. In The 13th International Baltic Conference on Databases and Information Systems. The author's contribution to the paper is 100%.
- Rikters, M., Pinnis, M., Krišlauks, R. (2018a, May). Training and Adapting Multilingual NMT for Less-resourced and Morphologically Rich Languages. In Proceedings of The 11th International Conference on Language Resources and Evaluation (LREC 2018). Paris, France: European Language Resources Association (ELRA). The author's contribution to the paper is 65%.
- Rikters, M. (2016, October). Searching for the Best Translation Combination Across All Possible Variants. The 7th Conference on Human Language Technologies - the Baltic Perspective (Baltic HLT 2016) (pp. 92-96). The author's contribution to the paper is 100%.
- Rikters, M. (2016, September). Interactive multi-system machine translation with neural language models. In Frontiers in Artificial Intelligence and Applications. The author's contribution to the paper is 100%.
- Rikters, M. (2016, July). K-Translate-Interactive Multi-System Machine Translation. In The 12th International Baltic Conference on Databases and Information Systems (pp. 304-318). Springer International Publishing. The author's contribution to the paper is 100%.
- Rikters, M., Skadiņa, I. (2016, may). Syntax-based multi-system machine translation. In N. C. C. Chair) et al. (Eds.), Proceedings of the tenth international conference on language resources and evaluation (LREC 2016). Paris, France: European Language Resources Association (ELRA). The author's contribution to the paper is approximately 70%.
- Rikters, M., Skadiņa, I. (2016, April) Combining machine translated sentence chunks from multiple MT systems. The 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing'2016). 2016. The author's contribution to the paper is approximately 70%.

- 7 publications in other peer-reviewed conference proceedings
  - Pinnis, M., Rikters, M., Krišlauks, R. (2018, October). Tilde's Machine Translation Systems for WMT2018. In the proceedings of The 3rd Conference on Machine Translation. The author's contribution to the paper is 40%.
  - Rikters, M., Fishel, M. (2017, September). Confidence Trough Attention. In the proceedings of the 16th Machine Translation Summit. The author's contribution to the paper is 70%.
  - Rikters, M., Bojar, O. (2017, September). Paying Attention to Multi-word Expressions in Neural Machine Translation. In the proceedings of the 16th Machine Translation Summit. The author's contribution to the paper is 80%.

- Rikters, M., Amrhein, C., Del, M., Fishel, M. (2017, September). C-3MA: Tartu-Riga-Zurich Translation Systems for WMT17. In the proceedings of the 2nd Conference on Machine Translation. The author's contribution to the paper is 50%.
- Rikters, M., Fishel, M., Bojar, O. (2017, August). Visualizing Neural Machine Translation Attention and Confidence. In the Prague Bulletin For Mathematical Linguistics issue 109. The author's contribution to the paper is 70%.
- Rikters, M. (2016, December). Neural Network Language Models for Candidate Scoring in Hybrid Multi-System Machine Translation. In CoLing 2016, 6th Workshop on Hybrid Approaches to Translation. The author's contribution to the paper is 100%.
- Rikters, M. (2015, July). Multi-system machine translation using online APIs for English-Latvian. ACL-IJCNLP 2015, 4th Workshop on Hybrid Approaches to Translation. The author's contribution to the paper is 100%.

The author has presented the results of the research at 13 international conferences and workshops:

- The 3rd Conference on Machine Translation, Brussels, Belgium, October 2018;
- The 8th Baltic Conference "Human Language Technologies – the Baltic Perspective", Tartu, Estonia, September 2018;
- The 13th International Baltic Conference on Databases and Information Systems, Trakai, Lithuania, July 2018;
- The 11th edition of the Language Resources and Evaluation Conference, Miyazaki, Japan, May 2018;
- The 16th Machine Translation Summit, Nagoya, Japan, September 2017;
- The 2nd Conference on Machine Translation, Copenhagen, Denmark, September 2017;
- The 12th Machine Translation Marathon, Lisbon, Portugal, August 2017;
- The 6th Workshop on Hybrid Approaches to Translation, Osaka, Japan, December 2016;
- The 7th Baltic Conference "Human Language Technologies – the Baltic Perspective", Riga, Latvia, October 2016;
- The 12th International Baltic Conference on Databases and Information Systems, Riga, Latvia, July 2016;
- The 17th International Conference on Intelligent Text Processing and Computational Linguistics, Konya, Turkey, April 2016;
- The 10th edition of the Language Resources and Evaluation Conference, Portorož, Slovenia, May 2016;
- The 4th Workshop on Hybrid Approaches to Translation, Beijing, China, July, 2015;

and two local conferences:

- The 76th conference of the University of Latvia, computational linguistics section, Riga, Latvia, February 2018;

- The 74th conference of the University of Latvia, computational linguistics section, Riga, Latvia, February 2016.

Research results are reported in the 16 papers published in the proceedings of the international conferences (see list of author's publications on the author's publications page).

1.8 OUTLINE OF THE THESIS

The remainder of this document is structured as follows:

- Chapter 2 summarizes existing machine translation methods and outlines advantages and disadvantages for each approach, especially detailing related work in the area of hybrid MT and existing combinations MT approaches.
- Chapter 3 introduces the methods for combining translations of from multiple statistical MT engines. For each method, an overview and relevance to the aims of this research is given, following by a description of evaluation methods used, as well as a detailed description of the experiments made.
- Chapter 4 gives an insight into combining translations from neural MT engines. The structure is similar to the previous chapter.
- Chapter 5 introduces several practical implementations that incorporate the previously mentioned translation combination methods.
- Chapter 6 sums up conclusions of this research.

2. BACKGROUND AND RELATED WORK

Since the very first appearances of MT in the mid-20th century, there have been several main paradigms that have shifted from one to the next over the years. The focus of MT research started mainly with a dominance of rule-based approaches that were later accompanied by statistical ones like corpus-based MT and example-based MT. In the past couple of decades, there have also been several hybrid approaches to MT, using combinations of different approaches or parallel running systems. In the most recent years, neural network MT is rapidly starting to outperform other methods in specific use-cases.

At first, it may have seemed that rule-based approaches can solve all MT problems with just the right amount of linguistic knowledge about source and target languages like grammars and lexicons and rules for syntactic analysis, lexical transfer, syntactic generation, morphology, etc. RBMT systems were the focus of MT research and the industry standard for commercial MT systems for over 25 years with large scale projects like EUROTRA (Johnson et al., 1985) and SYSTRAN (Toma, 1977), that is still active today.

With the introduction of the first IBM model (Brown et al., 1988) and the increasing availability of large corpora of monolingual and parallel texts, the corpus-based methods and SMT approaches finally started to produce acceptable quality translations in the last decade of the 20th century. Although statistical methods were common in the early periods of MT, results back then were very poor. Since then, the field of MT has changed dramatically several times - first, with the introduction of free online MTs in the late 1990s and open source MT tool platforms in the early 2000s (Hutchkins, 2012).

As the expansion of methodologies grew further, many researchers saw that there are obvious limitations of adopting one single approach to MT. This gave way to various attempts of combining the best qualities of both rule-based and statistical worlds resulting in hybrid MT configurations. Other extensions of HMT involve running multiple MT systems in parallel or employing automatic post-editing after the initial translation has been produced.

In the most recent years (Kalchbrenner and Blunsom, 2013) neural network translation methods have been attracting interest of both MT researchers and industry professionals. Although first appearing in 1997 (Castañ and Casacuberta, Forcada and Ñeco), at that time the size of the neural networks required to train an efficient NMT system was prohibitive due to the high amount of time and computing resources required to efficiently train them. Currently some NMT systems can outperform state-of-the-art SMT systems either on their own (Sennrich et al., 2016) or as a part of a HMT setup (Peter et al., 2016).

This chapter describes four of the general MT paradigms in the order of increasing interest by researchers and enterprise users over the course of history. Section 2.1 gives an insight on how MT is evaluated, section 2.2 covers rule-based, section 2.3– corpus-based, section 2.4 – hybrid, and section 2.5 – neural approaches to MT.

2.1 MT EVALUATION

To understand if an automatic translation is good or not, it must be compared to what a human translator would be able to produce, given the same source. The solution is not so trivial, since many different translations for the same source sentence are acceptable. Manual human evaluation is by far the best for such a task, especially when done by professional translators, but it is very expensive and impractical for performing on large amounts of texts on a regular basis. This reason creates a high demand for automatic evaluation metrics of MT quality that have a good correlation with human judgments. Among the first, successful and most popular metrics are BLEU (Papineni et al., 2002), TER

(Snover et al., 2006) and METEOR (Banerjee and Lavie, 2005). These three are also the most commonly used among related papers mentioned in this thesis.

2.1.1 BLEU

The bilingual evaluation understudy (BLEU) is currently the most widely used and most cited MT evaluation metric and was one of the first to report a high correlation with human judgment. The main idea of BLEU (3) is to reward MT outputs that have many overlapping n-grams (where n ranges from 1 to 4) with professional human translations (n-gram precision - (1), where Count $_{clip}$ (n–gram) is the count of n-gram matches between a candidate translation and a reference truncated to not exceed the largest count of that n-gram that is observed in the reference and Count(n-gram') is the total number of n-grams in the test corpus), while penalizing translations that are shorter than the human reference (brevity penalty - (2) , where *c* is the length of the candidate translation and *r* is the length of the reference). BLEU scores (3) are usually computed using 4-gram precision where N=4 and weights $w_n=\frac{1}{N}$. BLEU scores are represented on a scale of 0.00 to 1.00, where 1.00 is the best and 0.00 – the worst, and the final results are typically multiplied by 100. The current state-of-the-art MT systems tend to achieve between 20 and 40 BLEU points, depending on the language pair and translation direction in question. Unless stated otherwise, all BLEU scores reported in this thesis will be calculated using the *multi-bleu.perl* script from the Moses toolkit (Koehn et al., 2007).

$$p_n = \frac{\sum_{C \in \{Candidates\}} \sum_{n-gram \in C} Count_{clip}(n-gram)}{\sum_{C' \in \{Candidates\}} \sum_{n-gram' \in C'} Count(n-gram')} \quad (1)$$

$$BP = \begin{cases} 1 & if\ c > r \\ \exp(1 - r/c) & if\ c \leq r \end{cases} \quad (2)$$

$$BLEU = BP \cdot \exp(\sum_{n=1}^{N} w_n \log p_n) \quad (3)$$

2.1.2 TER

The Translation Edit Rate (TER) aims to measure the amount of editing that a human would have to perform to change MT output to exactly match the reference translation. TER allows to have multiple human references that may be of different lengths. A formal representation of TER is shown in (4). TER scores are traditionally represented on scale of 0.00 to 1.00, where 1.00 is the best and 0.00 – the worst. For state-of-the-art MT systems TER scores should be between 0.50 and 0.70.

$$TER = \frac{edit\ count}{average(reference\ word\ count)} \quad (4)$$

2.1.3 METEOR

The Metric for Evaluation of Translation with Explicit Ordering (METEOR - (9)) is based on the harmonic mean of unigram precision and recall (7), where recall is weighted higher than precision. What distinguishes METEOR from other metrics is that it also considers synonyms and performs stemming instead of just exact word matching (8). While it does report a higher correlation with human judgment than many other metrics, one downside is that it needs additional data and tuning for the optimal results and has extended support for only a handful of languages. Just like TER, METEOR is also expressed on scale of 0.00 to 1.00, where 1.00 is the best and 0.00 – the worst. High-quality MT systems should have METEOR scores between 0.40 and 0.80.

$$unigram\ precision = \frac{\text{number of candidate translation unigrams that are also found in the reference}}{\text{number of unigrams in the candidate translation}} \quad (5)$$

$$unigram\ recall = \frac{\text{number of candidate translation unigrams that are also found in the reference}}{\text{unigrams in the reference}} \quad (6)$$

$$Fmean = \frac{10 * unigram\ precision * unigram\ recall}{unigram\ recall + 9 * unigram\ precision} \quad (7)$$

$$Penalty = 0.5 * \left(\frac{number\ of\ chunks}{number\ of\ matched\ unigrams}\right) \quad (8)$$

$$METEOR = Fmean * (1 - Penalty) \quad (9)$$

2.2 RULE-BASED MT

RBMT is often denoted as the classical approach to MT. It mainly relies on the semantic, syntactic and morphological rules of the source and target languages as well as large monolingual dictionaries for each language and a bilingual dictionary for the actual translation between words. Most of this linguistic information is not learned automatically and needs to be composed by expert linguists. That is one of the main disadvantages of RBMT, making it more expensive to build and expand if necessary. On the other hand, the advantages of RBMT are complete control and ease of debugging, no need of large parallel corpora of texts, domain independence in many cases, and a certain level of reusability, for instance when using the same source language to translate into new target languages.

There are three main types of RBMT that are illustrated in Figure 1 (Vauquois, 1968) – dictionary-based, transfer based and interlingua. The first of which is the simplest one, translating from word to word, usually with no deeper analysis or generation. The transfer-based approach adds some analysis of the source sentence, which is then transferred for generation of the target language sentence. The interlingua approach takes this process one step further by creating an internal representation that is independent of the source and target languages. This section gives more detail on each of these types.

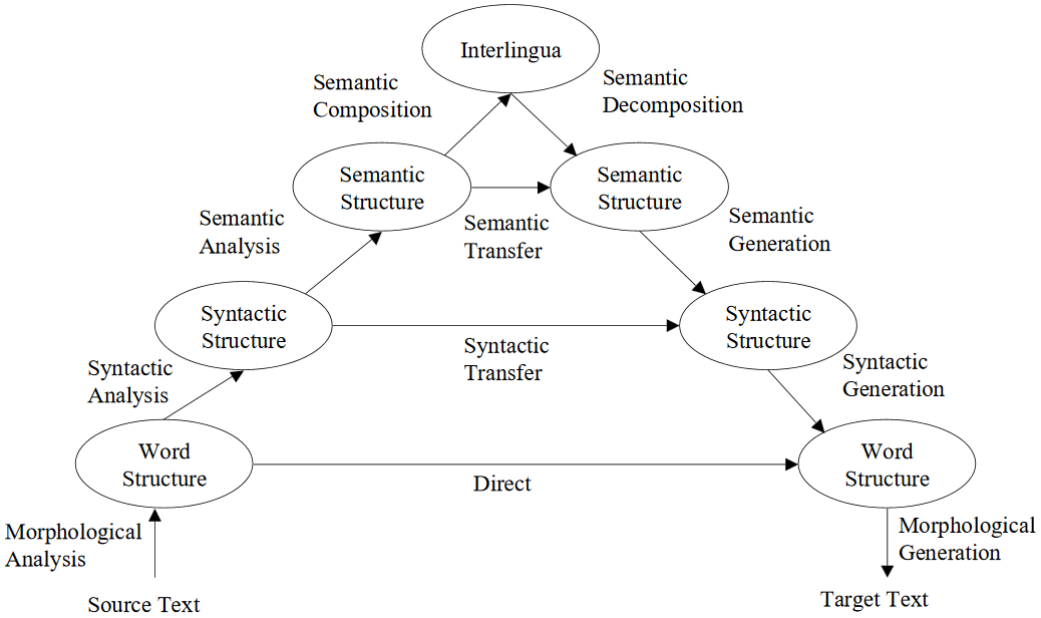


Figure 1: The Vauquois Triangle for MT (Vauquois, 1968)

2.2.1 Dictionary-based MT

Dictionary-based machine translation (DBMT) is the simplest and least complex. Texts are translated word-by-word without morphological analysis or lemmatisation. DBMT's main use-cases

were translating long lists of context independent words or short phrases and assistance for human translators who are fluent in the target language and can correct syntax and grammar, if required. In 1990, researchers from IBM introduced a DBMT system for translation from English to German called LMT (Neff and McCord, 1990), which utilised multiple machine-readable dictionaries for acquiring lexical information. Another viable use of DBMT is the translation between very close languages (Hajič et al., 2000).

2.2.2 Transfer-based MT

Unlike DBMT, in Transfer-based MT (TBMT) the process is separated into three steps – source text analysis, structural transfer of the analysis result to a structure suited for the target language, and target language text generation. The transfer rules depend on the language pair selected for translation. That is also the main difference between TBMT and Interlingua, which adds an internal representation that is independent of the language pair.

The first step includes morphological and lexical analysis. Morphology is analysed by obtaining a part-of-speech (POS) tag (e.g. noun, verb, etc.) and sub-category (number, gender, tense, etc.), along with the lemma of the word. Lexical analysis inspects the context of a word to determine the correct meaning in the context of its surrounding words. The transfer step has two parts – lexical and structural. The former is the same as DBMT and the latter deals with reordering of words or phrases. The structural transfer can be conducted on one of two levels, depending on how close the translatable language pair is. For closely related languages like Spanish and Catalan, the syntactic level of transfer would be sufficient. While for more distant languages like Spanish and English, a deeper level of transfer is required, capturing the semantic differences between the languages. In the last step, a target language phrases are generated in the adequate morphological forms from the output of the structural transfer stage.

Unlike DBMT that mostly relies on bilingual dictionaries, TBMT requires more hand crafting of human knowledge to build. At the very least there need to be rules to structure the source language texts, rules for the syntactic transfer and rules for generating the target language texts. A more complex solution will require semantic transfer rules as well.

In the commercial MT world, one of the best-known TBMT systems was the one made by SYSTRAN. Founded in 1968, they have been in the industry for almost 50 years and most of that time has been devoted to RBMT and especially TBMT. Most recently, SYSTRAN has stepped into the fields of Hybrid MT[9] and Neural MT[10]. Out of the open-source TBMT projects Apertium (Forcada et al., 2011) is one of the most popular. It is currently capable of translating between approximately 40 different language pairs either online[11] or downloadable for using locally.

2.2.3 Interlingua MT

The main idea of interlingua MT is similar to TBMT, but instead off transferring source language lexical and structural information to the target language, it is instead used to generate an intermediate abstract language-independent representation, called the interlingua. The target language text is then synthesized from the interlingua. This is intended to work similarly to a human translator, who reads words from the source sentence, comprehends the meaning, and is capable to produce a target language sentence of the same meaning. Interlingua methods are also often referred to as knowledge-based, due

[9] SYSTRAN Hybrid Technology - http://www.systransoft.com/systran/translation-technology/systran-hybrid-technology/
[10] Pure Neural™ Machine Translation (PNMT™) Beta Test - http://blog.systransoft.com/neural-machine-translation-nmt-beta-test/
[11] Apertium - A free/open-source machine translation platform - https://www.apertium.org

to the necessity of extensive knowledge resources (lexicons, grammar rules, in-domain knowledge) to transform words into meaning representations.

Some of the obvious advantages of interlingua are that it takes fewer components to add a new language for translation. For instance, Figure 2 illustrates how translation between four languages requires 12 sets of transfer rules and dictionaries for the TBMT approach on the left, while the interlingua approach on the right requires only 8 sets. This makes it more efficient to build multilingual MT systems. On the other hand, the main disadvantage is the difficulty to maintain the same meaning of texts with each new language added. This includes the loss of stylistic elements.

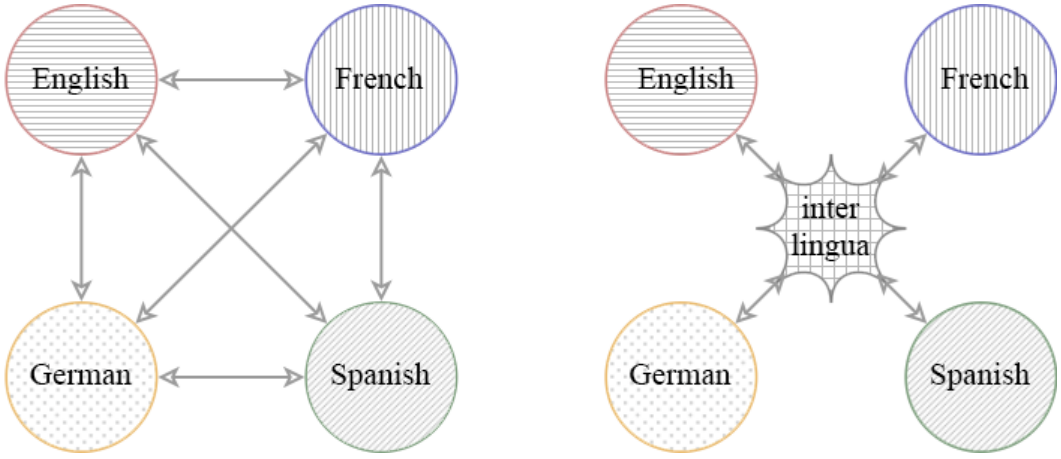


Figure 2: Transfer-Based MT vs. Interlingua MT

One of the successful commercial interlingua MT applications is the KANT project (Nyberg and Mitamura, 1992). It was designed for translation of technical documents written in simplified technical English (no pronouns, conjunctions, etc.) to French, Spanish, and German. Another approach to interlingua MT is the MOLTO project (Ranta et al., 2010) that is based on the Grammatical Framework (Mäenpää and Ranta, 1999) – an open-source toolkit for multilingual grammar implementations.

2.3 CORPUS-BASED MT

Corpus-based MT (CBMT), also known as data-driven MT, uses large bilingual parallel text corpora as its main resource. These corpora are used to train models for translation. Usually, the same setup can be used to train MT systems for multiple language pairs just by changing the training dataset. Thereby attempting to eliminate one of the general shortcomings of RBMT by limiting the necessity of high amounts of manual labour for linguistic analysis and various rule composition. One of the drawbacks of CBMT is that while for the big and widely used languages the necessary corpora can be found in sufficient quantities, for smaller, lesser-used languages, these corpora are often limited in size or non-existent at all.

Corpus based methods can be divided into two types – Statistical MT and Example-based MT. In SMT, translations are generated using statistical models whose parameters are derived from the analysis of bilingual text corpora. Currently it is the most widely studied MT method by far. EBMT employs the idea of translation by analogy, where sentences are decomposed into smaller phrases, translated and recomposed back into the full length.

2.3.1 Statistical MT

The main idea of SMT comes from information theory. A translation is produced according to the probability distribution of sentences in the target language (i.e. English) are translations of sentences in

the source language (i.e. French). One approach to modelling this probability distribution is to apply the Bayes Theorem, where the translation model calculates the probability that the target sentence is the translation of the source sentence, and the language model (LM) calculates the probability of seeing that sentence appear in the target language. Using these two models, a decoder performs the actual translation process.

Regarding the translation model – there are three main types (Gao, 2011) – word-based, phrase-based and syntax-based. Word-based models were proposed in 1993 (Brown et al., 1993) and now are known as the pioneering characteristic of SMT. These models use words as the fundamental unit of translation, making them difficult to use in cases where multiple words need to be translated into fewer words or a single word. Phrase-based models (Kohen et al., 2003) tackle this restriction by abstracting from using words as translation units to using sequences of words or phrases. A comparison of word-based and phrase-based approaches is illustrated in Figure 3. Here the word-based model aligns each word of the Latvian sentence "Kaķis sēdēja uz paklāja" to one or more words in the English "The cat sat on the mat", while the phrase-based model allocates five English words into three phrases and translates them into three Latvian phrases consistent of four words in total. Nevertheless, phrases in phrase-based models are not necessarily linguistically motivated. To incorporate linguistic information, many methods for syntax-based models have been introduced that incorporate parsing on the source sentence (Huang et al., 2006), target sentence (Galley et al., 2006), or both (Zhang and Gildea, 2008). A comparison of word-based and syntax-based alignments is illustrated in Figure 4.

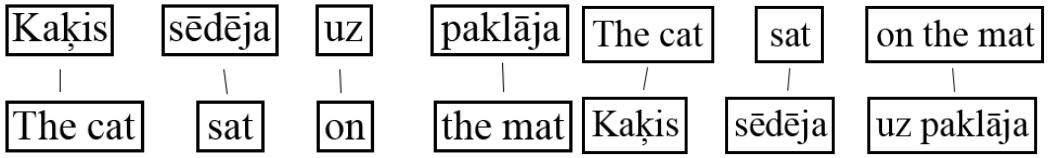


Figure 3: Word-based model on the left (IBM model 4) vs. phrase-based model on the right.

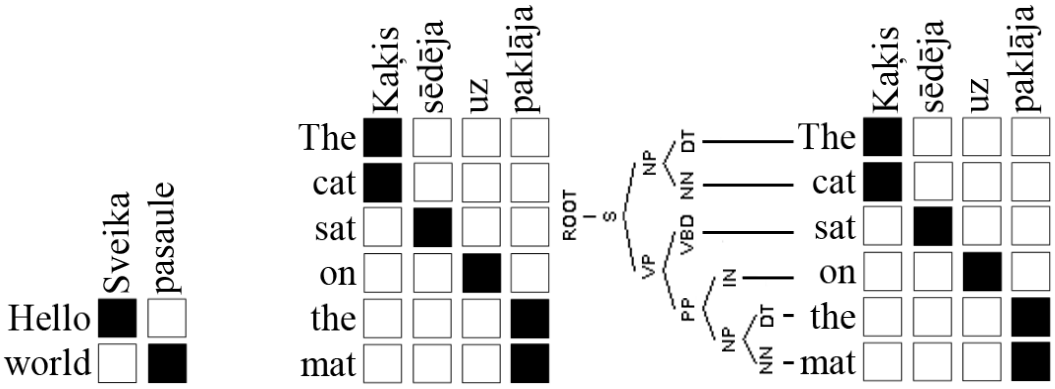


Figure 4: Two word-based alignments and a syntactic alignment on the right.

SMT has found many viable use-cases throughout the years. SMT was used as the main engine in Google Translate for over 10 years before they switched to NMT in November of 2016 and is still used by other industry leaders - Yandex.Translate and Bing Translator. The open-source SMT toolkit world is dominated by Moses (Koehn et al., 2007), but also has a place for others like Jane (Vilar et al., 2010), Joshua (Li et al., 2009), and others.

2.3.2 Example-based MT

EBMT or translation by analogy made its first appearance (Nagao, 1984) almost just before the re-emergence of SMT. EBMT is trained similarly to phrase-based SMT, using large bilingual parallel corpora. The core difference is how the translation process is executed. When given an input text of the

source language, EBMT looks for similar phrases in the source language training data and retrieves the equivalent phrases from the target language training data as a partial translation. The example in Table 1 shows how two sentences differ by one element. This helps an EBMT system learn that "The **X** sat on the mat" in English corresponds to "**X** sēdēja uz paklāja" in Latvian. When that is clear, all that needs to be done is to translate the X and compose the final output.

Table 1: Example of EBMT parallel corpus.

English	Latvian
The **cat** sat on the mat	**Kaķis** sēdēja uz paklāja
The **rat** sat on the mat	**Žurka** sēdēja uz paklāja

In the area of commercial systems there has been some interest from Microsoft (Richardson et al., 2001), but there are no known successful implementations. Some examples of open-source toolkits for EBMT are Cunei (Phillips and Brown, 2009) and OpenMaTrEx (Dandapat et al., 2010).

2.4 HYBRID MT

HMT describes a subset of MT where different MT approaches are used in the same system to complement each other's weaknesses in order to boost the accuracy level of the translations. Some of the best-known types of HMT include modifying SMT systems with RBMT generated output and generating rules for RBMT systems with the help of SMT. These systems would be categorized under the statistical rule generation subset of HMT. The other big subsets are multi-pass, where a sentence is fully translated with one MT system and the output is passed on as input for another MT system, and multi-system MT, where multiple translations of one sentence are generated in parallel.

2.4.1 Statistical rule generation

This is basically an RBMT approach with the main difference being that the necessary lexical and syntactic rules are generated from data. Thereby attempting to avoid the difficult and time-consuming manual labour of creating comprehensive and fine-grained linguistic rules by extracting them from the training corpus.

2.4.2 Multi-pass

The main idea of multi-pass systems is the processing of an input sentence multiple times in a row. An example could be firstly pre-processing a sentence with an RBMT system and then using that output as input for an SMT system that produces the final translation output. Such an approach can bring a balance between the amount of work required for composing rules for RBMT and parallel data and processing power required for SMT.

A multi-pass MT framework has been used to translate from Chinese to English (Chen et al., 2008), achieving competitive results in terms of BLEU score and METEOR. Omniscien[12] (formerly known as Asia Online) has been using multi-pass as a component in a larger hybrid setup.

2.4.3 Multi-System MT

MSMT involves usage of multiple MT systems in parallel and combining their output with the aim to produce a superior result to each of the individual systems. There is a vast variety of methods for

[12] https://omniscien.com

accomplishing such combination and therefore this review was conducted. MSMT is a relatively new branch of MT and interest from researchers has emerged more widely in only the past 15 years or so. In addition, even now such systems mostly live as experiments in lab environments instead of real, live, functional MT systems. Since no single system can be truly perfect and many have advantages over others a good combination must lead towards better overall translations.

SMT + RBMT

This is one of the most common methods for combining MT systems. Ahsan and Kolachina (2010) described a way of combining SMT and RBMT systems. The combination was done in five experimental setups where each one had input from the SMT system added in a different phase of the RBMT system - source analysis, local reordering, long-distance reordering, local + long-distance reordering and generation. The highest scoring variant proved to be the addition of SMT in the generation phase of RBMT. Authors did not use automatic evaluation but employed human subjective evaluation. They reported a slight improvement of an English – Hindi MT system compared to the SMT baseline.

Eisele et al. (2008a) describe an MSMT architecture for combining RBMT with SMT. They experimented with two different approaches - one provides enriched lexical resources to the SMT decoder with the help of the rule-based engines, the other uses parts from the SMT infrastructure to aid the rule-based system. The authors of this paper did not provide any results in the form of BLEU or other scoring methods only stated that a comparison revealed a significant increase in lexical coverage using their model and also that their system has already been put to practical use by translating a vast amount of documents.

Chen et al. (2007) described an architecture, which is much like the previous one – using SMT to align translations from multiple RBMT systems, extract phrases from the alignments and incorporate them into the phrase table of the SMT system. The authors report that their method increased the BLEU score by 1.69 - 3.32 points for a German – English MT over the baseline system.

The MT system described in the other publication of Eisele et al. (2008b) shared many similarities with the previous two. The system has multiple RBMT engines integrated with SMT. Tuning was also used to find the best configuration for the SMT system combined with 6 RBMT systems. This combination of systems was tested on two different corpora for six different language pairs and the highest result was achieved for English – Spanish scoring 7.85 BLEU more than the baseline. This was the highest increase in the reviewed papers.

Confusion network

Feng et al. (2009) introduce a lattice-based system combination model that is similar to confusion network system combination and consists of six steps. The first step is to collect hypotheses from the candidate systems, after that a backbone is chosen among them using a sentence-level Minimum Bayes Risk (MBR) method. Further steps involve aligning of the backbone and hypothesis pairs and normalisation. The final steps are constructing and decoding the lattice. The main difference from a confusion network-based system is the ability to express n-to-n mappings between the words in candidate translations instead of only 1-to-1 mappings. They used an IHMM-based system combination model as confusion network system to compare with their lattice-based system and the results slightly favoured the lattice-based one. Evaluation was also done on Chinese – English corpora and the best result increased the BLEU score by 3.92 (10.5% better than the baseline).

Barrault (2010) describes a MT system combination method where he combines multiple confusion networks of 1-best hypotheses from MT systems into one lattice and uses a language model for decoding the lattice to generate the best hypothesis. This system has been made freely available

online for download[13]. This system also uses tuning for performance improvement. The author reported a BLEU score increase by 2.26 points for Arabic – English and 1.61 for Chinese - English comparing to the best standalone system.

Confusion network + improvements

He and Toutanova (2009) combine multiple MT systems where the systems cooperate in making word alignment, ordering, and lexical selection decisions according to a set of feature functions combined in a single log-linear model. Each of the features models either the alignment, ordering, or lexical selection sub-problems. For decoding they use a beam search algorithm similar to Heafield and Lavie (2010). The evaluation was performed on a Chinese – English MT system and the best system grew the BLEU score by 5.17, which is 13.5% better than the baseline. This result ranks above the average between other MSMT systems.

Zhao and He (2009) establish two new methods for improving MT system combination performance for confusion network-based systems. The methods are based on a language model using n-gram fractional counts and n-gram voting scores for modifying the confidence scores of hypotheses in a confusion network. Both methods combined proved to provide the highest results. For evaluation purposes, they employ Chinese – English corpora. With that, they succeeded to improve the baseline BLEU score by 0.84, which is a noticeable increase, but mostly insignificant when compared to other results discussed in this review.

Other methods

Heafield and Lavie (2010) describe an open source MSMT system with the download link[14] provided. The system itself consists of four components – hypothesis alignment (with METEOR aligner), definition of a search space on top of the alignments, definition of features for scoring hypotheses and a beam search decoder. They were the only ones who used tuning to improve the results of the system. Evaluation was done on six different language pairs and in three metrics – BLEU, TER and METEOR. The highest scoring results were achieved for the Arabic – English language translation, reporting a BLEU score increase by 6.67 points, using the combo system.

Xuan et al. (2012) provided a general overview of the different approaches to hybrid MT which of course included MSMT or as they call it - *parallel coupling*. Although no specific systems were mentioned, they state that parallel coupling can only perform as well as the best of the component systems or in some cases lead to a 2-3 BLEU improvement. Apart from parallel coupling, other hybrid methods like serial coupling and a three-dimensional MT space model were also described.

Eisele (2005) tried out two different methods for the selection of the most promising translation variant from multiple SMT systems. One heuristic method used a set of features for each translation and the other – a statistical method based on probabilities from the language model for the target language. They reported increase of the BLEU score only by 0.2 points comparing to the best standalone baseline system for the French – English language pair and even less for other languages. Among all MSMT systems described in the reviewed publications, this is the weakest improvement.

The paper of Mellebeek et al. (2006) distinguished itself by describing one of the rare systems out of all others mentioned in this review that utilized online MT engines for MSMT. They introduce a system that at first attempts to split sentences into smaller parts for easier translation by the means of

[13] MANY: Open Source Tool for Machine Translation System Combination - https://code.google.com/p/many/
[14] Open Source Code Machine Translation - http://kheafield.com/code/

syntactic analysis, then translate each part with each individual MT system while also providing some context, and finally recompose the output from the best-scored translations of each part (they use three heuristics for selecting the best translation). For testing, they translated English – Spanish data of 800 sentences. Three different syntactic analysis methods and three MT engines were used. The results compared to the best baseline system had improved by 1.55 BLEU points which in the described case was about 5%.

Jayaraman and Lavie (2005) utilize a method for combining the best parts of individual MT system outputs to produce a unique superior translation. The system has three main steps – alignment of the words from the component MT systems, generation of synthetic sentence hypothesis translations using the alignments and scoring of the hypotheses based on the alignment information, the confidence of the individual systems, and a language model. Results were provided using the METEOR score, beating the best standalone system by 0.0778 METEOR points for Arabic – English MT systems. Due to such a selection of preferred metric, it is problematic to compare these results with other reviewed MSMT systems.

Santanu et al. (2014) describe a hybrid MT system that firstly involves pre-processing of data, e.g. cleaning and aligning named entities (NE) and then combining SMT, EBMT, translation memory (TM), and NE. The authors ran experiments on Bengali – Hindi corpora of multiple domains and the highest BLEU score increase over the baseline system was 0.38 for health-related data.

2.5 NEURAL MT

NMT is the newest architecture for getting machines to learn to translate. Despite its age, NMT has already shown promising results, achieving state-of-the-art performance for various language pairs (Sennrich et al., 2016). One of the main differences when compared to other SMT methods, which consist of many small sub-components that are tuned separately, is that in NMT only the one fully end-to-end model is trained and jointly tuned to maximize the translation performance.

Some drawbacks of NMT include a rather poor performance for long sentences, production of multiple repeated translations of a phrase and most notably – dealing with unknown words. These troubles have been addressed by shifting from word level translation to sub-word (or byte-pair) level or even character level translation, which introduced a new problem – the occasional production of new, non-existing words in the output translation.

Before the appearance of fully end-to-end NMT, there were some methods in using neural networks as components for traditional SMT in the form of neural language models (Schwenk, 2006) and neural translation models (Son et al., 2012). The first pure neural MT was introduced with encoder-decoder models and later enhanced by adding attention.

2.5.1 Neural language models

As mentioned before, an LM is responsible for estimating the probability of words, phrases or sentences appearing in a specific natural language. The main use-case for LMs in MT is ensuring fluent output in the target language during decoding time. In the early 2000s, when SMT was still in the lead performance wise, the n-gram language modelling tools in use were also based on statistics.

One major problem of n-gram LMs is that it is difficult to estimate a reliable probability of n-grams that have appeared only a few times in the training data, due to no information about similarities between words. Neural language models (Bengio et al., 2003) address this issue by representing words in a high-dimensional vector space where similar words would end up closer to one another. While in later implementations (Mikolov et al., 2011) these models started outperforming state-of-the-art n-gram

LMs, using them in real-world settings was still far too computationally expensive to handle. Only after applying many optimizations neural LMs became useful (Vaswani et al., 2013) in SMT.

Current implementations of neural LMs go even further than vector space representations of just words, by resorting to the character-level (Kim et al., 2016) and using convolutional networks (Dauphin et al., 2016). They are also being used in a broader spectrum of tasks – aside from more traditional ones like MT, speech recognition and generation, neural network specific applications such as image and video captioning.

2.5.2 Encoder-decoder models

Encoder-decoder models can be considered as two different neural language models that have mapping between the encoder – the language model for the source part, and the decoder – the language model for the target part. In other words – the encoder converts a source sentence into a vector representation and the decoder uses that representation to generate the output target sentence. To learn this mapping, these models need to be trained jointly.

At first encoder-decoder models were used as a component to improve phrase-based SMT (Cho et al., 2014a). Nevertheless, in little time this approach became one of defining cornerstones (Cho et al., 2014b) of the neural machine translation that we know today. While these initial NMT systems reported to achieve a comparable level of quality to the current state-of-the-art systems, they still failed to outperform them.

2.5.3 Attentional models

The obvious bottleneck of the initial encoder-decoder NMT model architecture was that encoding increasingly longer sentences into fixed sized vectors lead to loss of information. Sutskever et al. (2014) attempted to mend this problem by reversing the order of words in the source sentences. This was eventually solved by Bahdanau et al. (2015), who introduced the attentional NMT model. It enables the model to find parts of a source sentence that are relevant to predicting a target word (pay attention), without the need to form these parts as a hard segment explicitly. This allowed for higher quality NMT systems to be trained due to a decreased number of trainable parameters for the neural network.

Using the attentional model to decode sentences resulted in a useful by-product – soft alignments between tokens of source and target sentences. These soft alignments (Figure 5) resemble alignments from SMT (first image of Figure 4), although giving no guarantee that the attention corresponds to alignments. Nevertheless, they serve a good purpose in visualizing attention and can also be used to replace unknown words with back-off translations from a dictionary (Jean et al., 2015). Further use-cases for the attention alignments include penalizing output that accumulates an overly high amount of attention during decoding, and also scoring produced translations via various attention-based metrics (Rikters and Fishel, 2017).

Figure 5: Example of attention-based soft alignments. White areas represent stronger attention and grey/black areas – weaker attention.

2.5.4 Fully convolutional models

Gehring et al. (2017) built upon the recurrent translation models, which used a combination of CNN and RNN, and introduced a fully convolutional NMT architecture. Historically, CNNs have been most successful in machine learning tasks like image recognition (Krizhevsky et al., 2012) and video analysis (Baccouche et al., 2011), while RNNs dominated textual applications, such as machine translation. CNNs are highly parallelizable, because unlike RNNs, they do not depend on computations of the previous word to compute the next one. To maintain the context of each word, a convolution encodes it together with its left and right context, in a limited window. Such windows can be computed independently, making the CNNs more efficient for parallel computing. To increase the size of the effective context of the network several convolutional layers are stacked on top of each other. The main differences of the networks are depicted in Figure 6.

First competitive results (Gehring et al., 2016) of using a CNN as an encoder for NMT showed that the architecture is capable of producing translations of state-of-the-art quality, while doing the process much faster than the strong LSTM baseline. Later results (Gehring et al., 2017) demonstrated that a deep fully convolutional (CNN as both – the encoder and the decoder) NMT architecture achieves a new state-of-the-art on several public translation benchmark datasets, outperforming previous results by 1.6-1.9 BLEU.

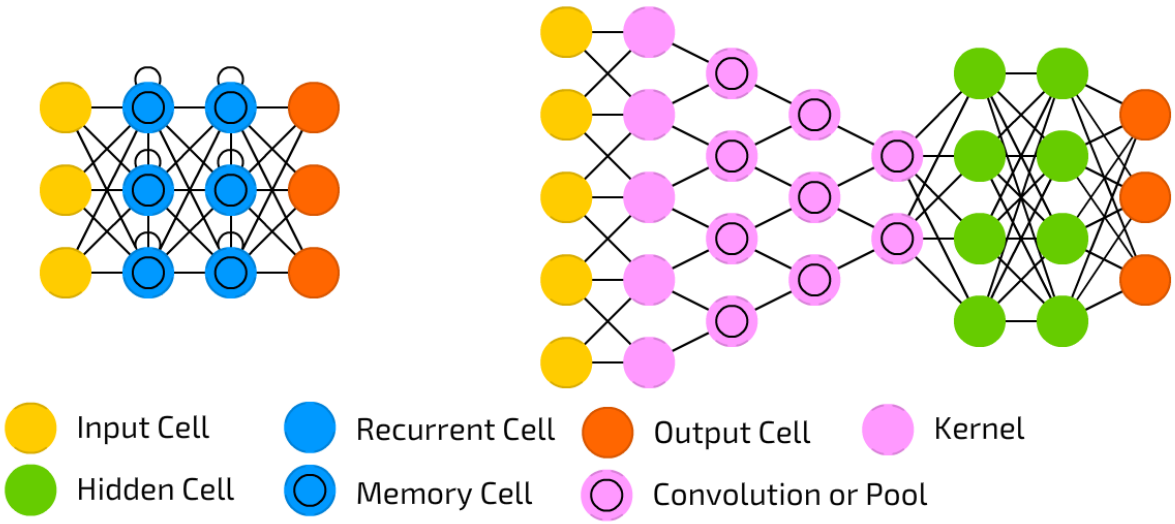


Figure 6: Neural network architecture differences[15] of long short-term memory (LSTM) and convolutional neural network (CNN).

2.5.5 Self-attentional models

Vaswani et al. (2017) proposed a new neural network architecture, the Transformer, which relies only on the attention mechanism to draw global dependencies between input and output. It has an encoder-decoder structure using multiple stacked self-attention and point-wise, fully connected layers for both the encoder and decode. Like CNNs, self-attentional models are also highly parallelizable, as they do not employ the recurrent connections of RNNs.

The models outperform previous architectures in both – translation quality and speed. Most recently they have been widely adapted by research groups around the world and were the most used architecture in the WMT 2018 news machine translation shared task (Bojar et al., 2018).

[15] The Neural Network Zoo - www.asimovinstitute.org/neural-network-zoo

3. COMBINING STATISTICAL MACHINE TRANSLATION OUTPUT

3.1 COMBINING FULL SENTENCE TRANSLATIONS

This section describes an HMT method that employs several online MT system APIs, forming a Multi-System Machine Translation (MSMT) approach. The goal is to improve the automated translation of English – Latvian texts over each of the individual MT APIs. The selection of the best hypothesis translation is done by calculating the perplexity for each hypothesis. Experiment results show a slight improvement of BLEU score and WER (word error rate). This section is based on the paper of Rikters (2015). The author's contribution to this work is 100%.

3.1.1 Introduction

MSMT is a subset of HMT where multiple MT systems are combined in a single system to complement each other's weaknesses to boost the accuracy level of the translations. It involves usage of multiple MT systems in parallel and combining their output with the aim to produce better result as for each of the individual systems. It is a relatively new branch of MT and interest from researchers has emerged more widely during the last 10 years. And even now such systems mostly live as experiments in lab environments instead of real, live, functional MT systems. Since no single system can be perfect and different systems have different advantages over others, a good combination must lead towards better overall translations.

There are several recent experiments that use MSMT, described in more detail in section 2.4.3. Most of the research is done English – Hindi, Arabic – English and English – Spanish language pairs in their experiments. Where it concerns English - Latvian machine translation, no such experiments have been conducted.

This section presents a first attempt in using an MSMT approach for the under-resourced English-Latvian language pair. Furthermore, the first results of this hybrid system are analysed and compared with human evaluation. The experiments described use multiple combinations of outputs from two MT systems and one experiment uses three different MT systems.

3.1.2 System description

The main system consists of three major constituents – tokenization of the source text, the acquisition of a translation via online APIs and the selection of the best translation from the candidate hypotheses. A visualized workflow of the system is presented in Figure 7.

Currently the system uses three translation APIs (Google Translate[16], Bing Translator[17] and LetsMT[18]), but it is designed to be flexible and adding more translation APIs has been made simple. Also, it is initially set to translate from English into Latvian, but the source and target languages can also be changed to any language pair supported by the APIs.

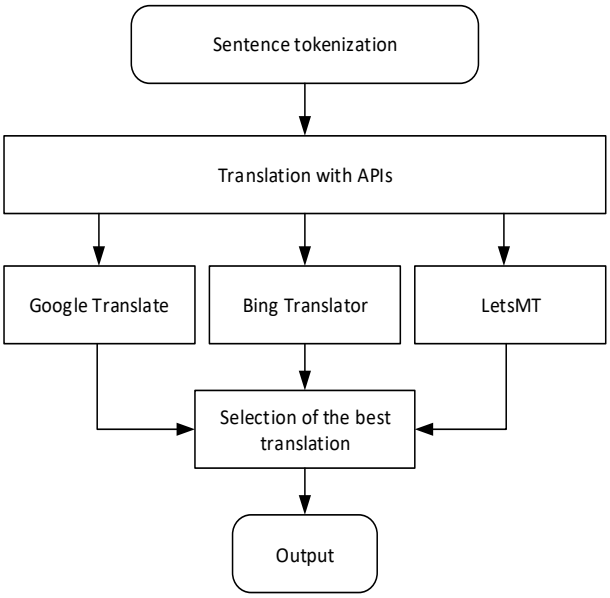


Figure 7: General workflow of the translation process

API description

Currently there are three online translation APIs included in the project – Google Translate, Bing Translator and LetsMT. These specific APIs were chosen for their public availability and descriptive documents as well as the wide range of languages that they offer. One of the main criteria when searching for translation APIs was the option to translate from English to Latvian.

Selection of the final translation

The selection of the best translation is done by calculating the perplexity of each hypothesis translation using KenLM (Heafield, 2011). First, a language model (LM) must be created using a preferably large set of training sentences. Then for each machine-translated sentence a perplexity score represents the probability of the specific sequence of words

[16] Google Translate API - https://cloud.google.com/translate/
[17] Bing Translator Control - http://www.bing.com/dev/en-us/translator
[18] LetsMT! Open Translation API - https://www.letsmt.eu/Integration.aspx

appearing in the training corpus used to create the LM. Sentence perplexity has been proven to correlate with human judgments close to the BLEU score and is a good evaluation method for MT without reference translations (Gamon et al., 2005). It has been also used in other previous attempts of MSMT to score output from different MT engines as mentioned by Callison-Burch et al. (2011) and Akiba et al. (2002).

Perplexity on a test set is calculated using the language model as the inverse probability (P) of that test set, which is normalized by the number of words (N) (Jurafsky and Martin, 2014). For a test set W = w_1, w_2, ..., w_N:

$$perplexity(W) = \mathrm{P}(w_1, w_2, \dots, w_N)^{-\frac{1}{N}} = \sqrt[N]{\frac{1}{\mathrm{P}(w_1, w_2, \dots, w_N)}} \tag{10}$$

Perplexity can also be defined as the exponential function of the cross-entropy:

$$H(W) = -\frac{1}{N}\log \mathrm{P}(w_1, w_2, \dots, w_N) \tag{11}$$

$$perplexity(W) = 2^{H(W)} \tag{12}$$

3.1.3 Experiments

The first experiments were conducted on the English – Latvian part of the JRC-Acquis corpus version 3.0 (JRC) (Steinberger et al., 2006) from which both the language model and the test data were retrieved. The test data contained 1581 randomly selected sentences. The language model was created using KenLM with order 5.

Translations were obtained from each API individually, combining each two APIs and lastly combining all three APIs. Thereby forming 7 different variants of translations. Google Translate and Bing Translator APIs were used with the default configuration and the LetsMT API used the configuration of TB2013 EN-LV v03[19].

Evaluation on each of the seven outputs was done with three scoring methods – BLEU, TER (translation edit rate) and WER (Klakow and Peters, 2002). The resulting translations were inspected with a modified iBLEU tool (Madnani, 2011) that allowed to determine which system from the hybrid setups was chosen to get the specific translation for each sentence.

The results of the first translation experiment are summarized in Table 3. Surprisingly all hybrid systems that include the LetsMT API produce lower results than the baseline LetsMT system. However, the combination of Google Translate and Bing Translator shows improvements in BLEU score and WER compared to each of the baseline systems.

[19] https://www.letsmt.eu/TranslateText.aspx?id=smt-e3080087-866f-498b-977d-63ea391ba61e

The table also shows the percentage of translations from each API for the hybrid systems. Although per scores the LetsMT system was by far better than the other two, it seems that the language model was reluctant to favour its translations.

Since the systems themselves are more of a general domain and the first test was conducted on a legal domain corpus, a second experiment was conducted on a smaller dataset containing 512 sentences of a general domain (Skadiņa et al., 2010). In this experiment, only the BLEU score was calculated as it is shown in Table 2.

Table 2: Second experiment results.

System	**BLEU**
Google Translate	24.73
Bing Translator	22.07
LetsMT	32.01
Google + Bing	23.75
Google + LetsMT	28.94
LetsMT + Bing	27.44
Google + Bing + LetsMT	26.74

Table 3: First experiment results

System	**BLEU**	**TER**	**WER**	**Translations selected**			**Equal %**
				Google %	**Bing %**	**LetsMT %**	
Google Translate	16.92	47.68	58.55	100	-	-	-
Bing Translator	17.16	49.66	58.40	-	100	-	-
LetsMT	28.27	36.19	42.89	-	-	100	-
Google + Bing	**17.28**	48.30	58.15	50.09	45.03	-	4.88
Google + LetsMT	22.89	41.38	50.31	46.17	-	48.39	5.44
LetsMT + Bing	22.83	42.92	50.62	-	45.35	49.84	4.81
Google + Bing + LetsMT	21.08	44.12	52.99	28.93	34.31	33.98	2.78

Table 4: Native speaker evaluation results

System	User 1	User 2	User 3	User 4	User 5	AVG user	Hybrid	BLEU
Bing	21,88%	53,13%	28,13%	25,00%	31,25%	31,88%	28,93%	16.92
Google	28,13%	25,00%	25,00%	28,13%	46,88%	30,63%	34,31%	17.16
LetsMT	50,00%	21,88%	46,88%	46,88%	21,88%	37,50%	33,98%	28.27

3.1.4 Human evaluation

A random 2% (32 sentences) of the translations from the first experiment were given to five native Latvian speakers with an instruction to choose the best translation (just like the hybrid system should). The results are shown in Table 4. Comparing the evaluation results to the BLEU scores and the selections made by the hybrid MT a tendency towards the LetsMT translation can be observed among the user ratings and BLEU score that is not visible from the selection of the hybrid method.

3.1.5 Conclusions

This section described a machine translation system combination approach using public online MT system APIs. The focus was to gather and utilize only the publicly available APIs that support translation for the under-resourced English-Latvian language pair.

One of the test cases showed an improvement in BLEU score and WER over the best baseline.

In all hybrid systems that included the LetsMT API a decline in overall translation quality was observed. This can be explained by scale of the engines - the Bing and Google systems are more general, designed for many language pairs, whereas the MT system in LetsMT was specifically optimized for English – Latvian translations. This problem could potentially be resolved by creating a language model using a larger training corpus and a higher order for more precision.

3.2 COMBINING SENTENCE FRAGMENT TRANSLATIONS - SIMPLE FRAGMENTING

This section describes a hybrid machine translation system that explores a parser to acquire syntactic chunks of a source sentence, translates the chunks with multiple online MT system APIs and creates output by combining translated chunks to obtain the best possible translation. The selection of the best translation hypothesis is performed by calculating the perplexity for each translated chunk. The goal of this approach is to enhance the baseline multi-system hybrid translation (MHyT – described in 3.1) system that uses only a language model to select best translation from translations obtained with different APIs and to improve overall English – Latvian machine translation quality over each of the individual MT APIs. The

presented syntax-based multi-system translation (SyMHyT) system demonstrates an improvement in terms of BLEU and NIST scores compared to the baseline system. Improvements reach from 1.74 up to 2.54 BLEU points. This section is based on the paper of Rikters and Skadiņa (2016a). The author's contribution to this work is 75%.

3.2.1 Introduction

Different approaches of MSMT have been appearing lately (more detail in Section 2.4.3). Traditional MSMT (Hildebrand and Vogel, 2009) selects the best translation from a list of possible candidate translations generated by different MT engines using n-gram approach. Improvement has been reported when translated from French (+1.6 BLEU), German (+1.95 BLEU) or Hungarian (+1 BLEU) into English. However, application of a similar approach for English-Latvian MT (described in Section 3.1) has resulted in insignificant improvement by only +0.12 BLEU points (Rikters, 2015).

Freitag et al. (2015) presented a novel system combination approach that enhances the traditional confusion network system combination approach (Heafield et al., 2009) with an additional model trained by a neural network. The proposed approach yielded in translation improvement from up to +0.9 points in BLEU and -0.5 points in TER for Chinese-English and Arabic-English.

This section presents a method that allows improving the MMT approach by incorporating syntactic information. These experiments were inspired by analysis of typical errors produced by statistical MT engines when translation is performed into a morphologically rich language with rather free order – Latvian (Skadiņa et al., 2012). This error analysis showed that the main type of errors is wrong inflection, which is usually caused by ignoring syntax rules. Our hypothesis is that translation of smaller, linguistically motivated chunks can improve this situation.

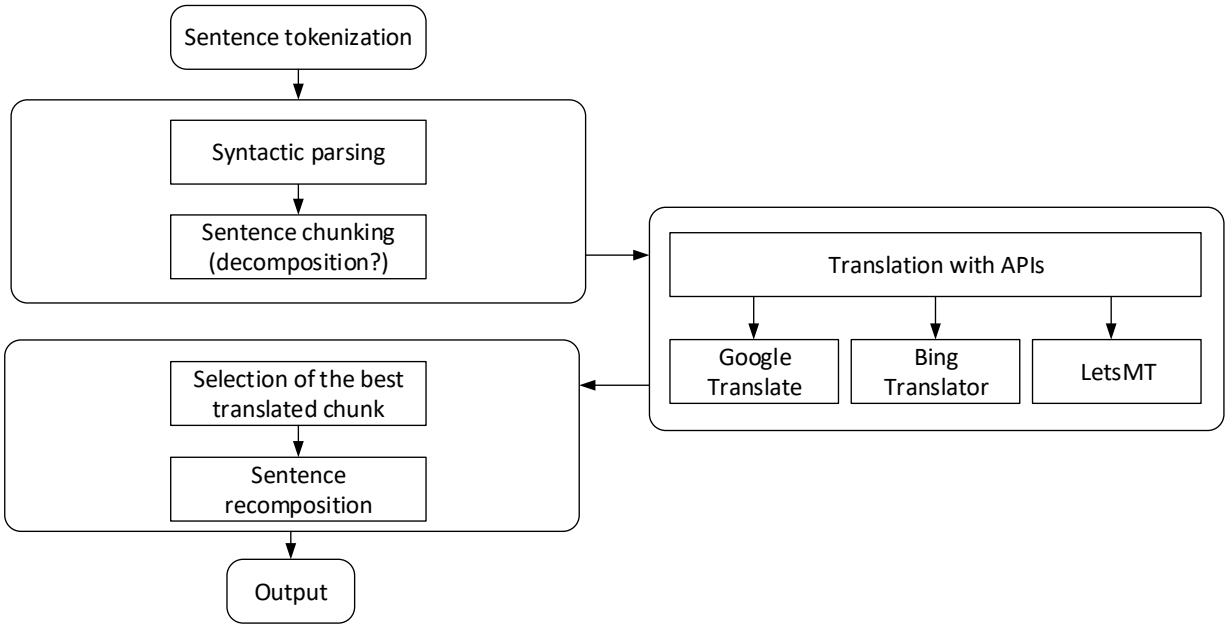


Figure 8: General workflow of the translation process

The experiments described in this section use multiple combinations of outputs from the same English-Latvian MT systems as described in the previous section (3.1). We believe that the syntax-based combination of two MT systems from companies that have access to enormous language resources with an MT system which is tailored for the under resourced Latvian language allows to improve translation quality. In the section, we analyse combination of all three MT systems as well as combinations of system pairs. The automatic evaluation results obtained with this hybrid system are analysed and compared with human evaluation results.

The framework developed within this work allows the application of proposed strategy to other language pairs for which MT APIs are available. The developed *SyMHyT* framework is freely available on GitHub[20].

3.2.2 System description

The hybrid system described in this section consists of similar components to the previous one – 1) pre-processing of the source sentences, 2) the acquisition of a translations and 3) post-processing - the selection of the best translation of chunks and generation of MT output. A visualized workflow of the system is presented in Figure 8.

For translation three translation APIs are used. Each translation API in our system is defined with a function that has source and target language identifiers and the source chunk as

[20] Syntax-based Multi-System Hybrid Translator is available at: https://github.com/M4t1ss/Multi-System-Hybrid-Translator/tree/Syntactic-Multi-System-Hybrid-Translator

input parameter and the target chunk as the only output. This makes the system's architecture flexible allowing to integrate more translation APIs easily.

Although the system is configured to translate from English into Latvian, the source and target languages could also be changed to other language pairs that are supported by the MT APIs. Changing source language involves need for a parser that is compliant with the Berkeley Parser (Petrov et al., 2006).

Pre-processing

The aim of the pre-processing step is to divide sentences into linguistically motivated chunks that will be then translated with the online APIs. For this task, the Berkeley Parser is used.

The parse tree of each sentence is then processed by the chunk extractor to obtain the top-level sub-trees (noun phrases, verb phrases, prepositional phrases, etc.). This step relies only on source language parser and does not consider properties of the target language, i.e., it is independent from the target language.

Translation with the APIs

In the scope of the section, three online translation APIs were used – *Google Translate*[21], *Bing Translator*[22] and *LetsMT*![23]. The less known LetsMT! (Vasiļjevs et al., 2012) is full-service platform that gathers public and user-provided MT training data and allows users to create custom MT systems by combining and prioritising this data. The training and translation facilities of LetsMT! are based on the open source toolkit Moses (Koehn et al., 2007). LetsMT! also provides access to a wide range of MT systems for different language pairs. These systems can be accessed using LetsMT! API for MT integration.

These specific APIs were selected because of their public availability and descriptive documentation as well as the wide range of languages that they support. One of the main criteria when searching for translation APIs was the possibility to translate from English into Latvian.

Selection of the best translated chunk

The selection of the best-translated chunk is performed as described in section 3.1.2 for whole sentences with the only difference being that chunks are shorter than whole sentences. When the best translation for each chunk is selected, the translation of the full sentence is generated by concatenation of chunks.

---

[21] Google Translate API is available online at: https://cloud.google.com/translate/
[22] Bing Translator Control is available online at: http://www.bing.com/dev/en-us/translator
[23] LetsMT! Open Translation API is available online at: https://www.letsmt.eu/Integration.aspx

Illustration of translation process

An example translation of a sentence using the syntax-based multi-system MT approach is illustrated in Figure 9. At first, the sentence "3. the list referred to in paragraph 1 and all amendments thereto shall be published in the official journal of the european communities." is parsed with Berkeley Parser. In the next step, the parsed sentence is divided into 3 chunks: "3. the list referred to in paragraph 1 and all amendments thereto", "shall be published in the official journal of the european communities" and ".". Each chunk is then translated with online APIs. Obtained three translations for each chunk are then evaluated and the best translation for the chunk is selected. Finally, the output is generated.

3.2.3 Experiments

This section describes the experiments performed to test the proposed syntax-based multi-system translation approach.

Data

The experiments were conducted using the same LM and test dataset as mentioned in section 3.1.3.

System combination

The proposed method was applied to all combinations of two and then all three APIs. Thus, seven different translations for each source sentence were obtained. *Google Translate* and *Bing Translator* APIs were used with the default configuration and the LetsMT! API used the configuration of TB2013 EN-LV v03.

Automatic evaluation

Output of each system was evaluated with two scoring methods – BLEU and NIST (Doddington, 2002). The results of the automatic evaluation are summarized in Table 5. The evaluation results clearly show an improvement over the baseline hybrid system (MHyT) that does not have a syntactic pre-processing step and thus selects the best translation from translations of full sentences.

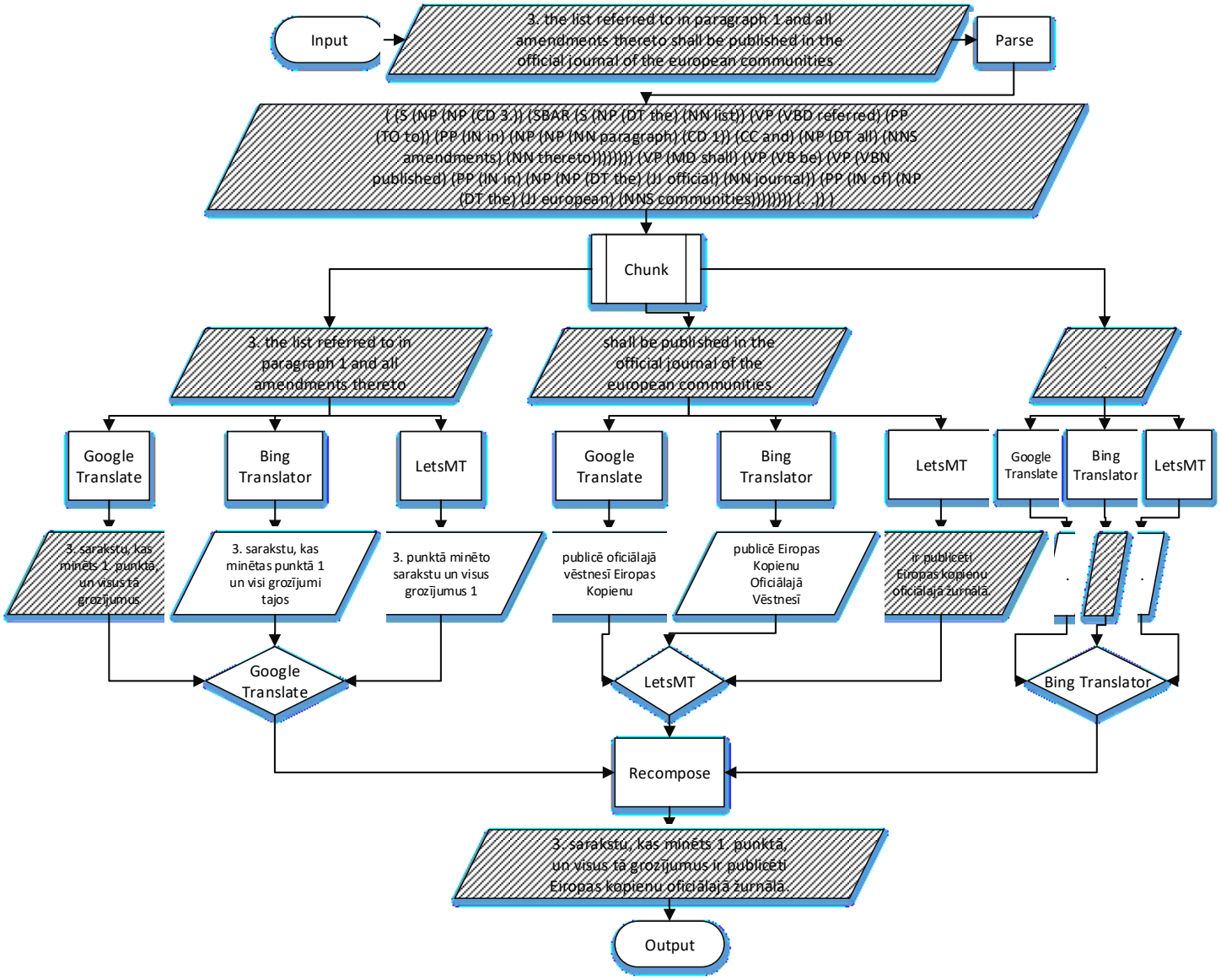


Figure 9: Illustration of the syntax-based multi-system translation approach

The combination of *Google Translate* and *Bing Translator* shows about +2 BLEU improvement compared to each of the baseline systems.

Surprisingly, all hybrid systems that include the LetsMT! API produce lower results than the baseline LetsMT! system. Thus, resulting translations were inspected with the Web-based

MT evaluation platform MT-ComparEval (Klejch et al., 2015) to determine, which system from the hybrid setups was selected to get the specific translation for chunk. Table 6 shows the percentage of translations from each API for the hybrid systems.

Table 5: Evaluation results: MHyT – baseline hybrid system, SyMHyT – syntax-based hybrid system

System	BLEU		NIST	
	MHyT	SyMHyT	MHyT	SyMHyT
Google Translate	18.09		8.37	
Bing Translator	18.87		8.09	
LetsMT!	30.28		9.45	
Google + Bing	18.73	21.27	7.76	8.30
Google + LetsMT	24.50	26.24	9.60	9.09
LetsMT! + Bing	24.66	26.63	9.47	8.97
Google + Bing + LetsMT!	22.69	24.72	8.57	8.24

Table 6: Distribution of selected chunks from different MT APIs

System	Google	Bing	LetsMT
Google Translate	100%	-	-
Bing Translator	-	100%	-
LetsMT	-	-	100%
Google + Bing	74%	26%	-
Google + LetsMT	25%	-	74%
LetsMT+ Bing	-	24%	76%
Google + Bing + LetsMT	17%	18%	65%

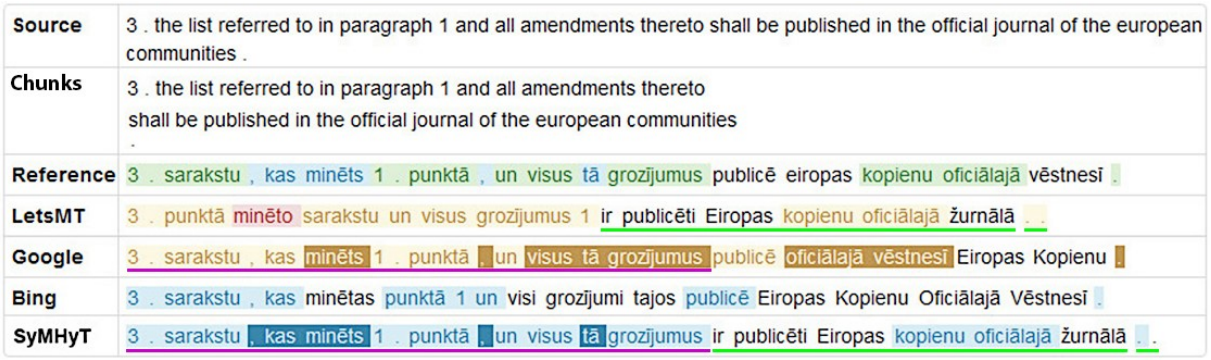


Figure 10: Comparison of translations of a sentence with the different systems with MT-ComparEval

Contrary to the baseline hybrid system (Google - 28.93%, Bing - 34.31%, LetsMT! - 33.98%, equal - 2.78%) the SyMHyT system tends to use more chunks from LetsMT. This resulted in increase of the BLEU score by +1.7 - 2.03 points over the baseline hybrid solution.

Figure 10 shows an example of the source sentence, extracted chunks, reference sentence, and all system translations, including the hybrid SyMHyT, with the differences highlighted. The purple line highlights the chunk selected from *Google Translate* and the green line – the chunks from LetsMT. It can be seen that the hybrid system (SyMHyT) used the first chunk from *Google's* output and the second chunk from LetsMT.

This illustration also shows weakness of the proposed approach – selected chunks are very long and are independent from the target language. Our hypothesis is that this is the reason why the hybrid approach did not perform better as LetsMT system.

Experiments with different language models

To evaluate the influence of language model size on the chunk selection process we trained two 12-gram language models – one on the JRC corpus (section 3.1.3) and another one on the DGT-Translation Memory (DGT-TM) corpus (Steinberger et al., 2012). The results of this experiment are presented in Table 7.

Table 7: Influence of different language models

LM	**Size (sentences)**	**BLEU**
5-gram JRC	1.4 million	**24.72**
12-gram JRC	1.4 million	24.70
12-gram DGT-TM	3.1 million	24.04

For this approach the higher order language model did not show improvement. Some additional experiments described in section 3.3, using 6-gram, 9-gram and 12-gram LMs resulted in slightly higher BLEU score but the change was not statistically significant.

Application of random chunks

To justify that our approach that uses the linguistically motivated chunks are much better as just cutting sentences into random chunks we performed three experiments. The sentence was split into 5-grams in one experiment (+ one shorter n-gram, if the last one is made up of less tokens), random 1-grams to 4-grams in the second experiment, random 1-grams to 6-grams in the third, and finally random 6-grams to n-grams of sentence length in the last experiment. We used the 5-gram LM as in section 3.1.3 for best translation selection. Results of these experiments (Table 8) fully confirmed our hypothesis of advantage of linguistically motivated chunks.

Table 8: Influence of different chunk selection strategies on MT output

Chunks	BLEU
SyMHyT chunks	**24.72**
5-grams	11.85
Random 1-4 grams	7.33
Random 1-6 grams	10.25
Random 6-max grams	20.94

3.2.4 Human evaluation

A random 2% (32 sentences) of the translations from the experiment were given to 10 native Latvian speakers with instructions to evaluate fluency and adequacy. The MT-EQuAl tool (Girardi et al., 2014) was used for evaluation. The three baseline systems were compared with the syntax-based hybrid system that combines all three baselines. Evaluators were instructed to mark each sentence with one of the following labels: "most fluent translation", "most precise translation", "neither most fluent, nor most precise", or "both most fluent and most precise". In case, if a translation is marked as most fluent and adequate, then all others alternatives needed to be marked as "neither most fluent, nor most precise".

The results of evaluation are summarized in Table 9. The free-marginal kappa (Randolph, 2005) for these annotations is 0.335 that indicates substantial agreement between the annotators.

Table 9: Manual evaluation results

System	Fluency AVG	Accuracy AVG	SyMHyT selection	BLEU
Google	35.29%	34.93%	16.83%	18.09
Bing	23.53%	23.97%	17.94%	18.87
LetsMT	20.00%	21.92%	65.23%	30.28
SyMHyT	21.18%	19.18%	-	24.72

As it can be seen from the table, about 1/3 of translations recognized by annotators as most fluent and most adequate are translations from *Google Translate* system. This contradicts with the automatic evaluation results and the selections made by the syntax-based hybrid MT, where a tendency towards the LetsMT! translation is observed.

Inspecting the annotations closer, we performed a broader analysis of this result. Our hypothesis is that LetsMT! was chosen less often by the annotators because of failure to translate dates or numbers in specific sentences while the rest of the sentence was very similar to the reference, hence scoring more BLEU points. Closer inspection revealed that three sentences from LetsMT! contained "βNUMβ" tag, which appears to be an error in the named entity processor during time of experiments. There were also five sentences that contained untranslated dates, e.g., "31 december 1992" or "february 1995." These errors account for

LetsMT! not be selected by annotators in 25% cases of the evaluation dataset, while in case of BLEU score, their influence was not so significant.

3.2.5 Conclusions

This section described an improved machine translation system combination approach for public online MT system APIs that uses syntactic and statistical features. All test cases showed an improvement in BLEU score when compared to the baseline hybrid system and improvement in NIST score in one case. When used only with *Google Translate* and *Bing Translator,* the SyMHyT approach resulted in +2.4 BLEU points compared to the best individual API.

For hybrid systems that included the LetsMT! API a decrease in BLEU was observed. This can be explained by the scale of the engines - the *Bing* and *Google* systems are more general, designed for many language pairs, whereas the MT system in LetsMT! is customized for English – Latvian translations.

The proposed method for chunking is very straightforward and easily accomplishable. In later experiments (Rikters and Skadiņa, 2016), we used a more sophisticated chunker that is slightly more dependent on the source language, as it includes additional rules for chunk selection.

In the described approach, the chunker splits sentences in top-level chunks without analysis of sub-chunks or cases when a chunk is single token. However, the larger chunks should be split in smaller sub-chunks and the single-word chunks should be combined with the neighbouring longer chunks. Better results could be achieved if sentence is divided into certain types of phrases, e.g. noun phrases and verb phrases, but not prepositional phrases, infinitive phrases, etc. These ideas lead to the improvements described in the next section (3.3).

3.3 COMBINING SENTENCE FRAGMENT TRANSLATIONS - ADVANCED FRAGMENTING

This section presents a hybrid machine translation (HMT) system that pursues syntactic analysis to acquire phrases of source sentences, translates the phrases using multiple online machine translation (MT) system application program interfaces (APIs) and generates output by combining translated chunks to obtain the best possible translation. The aim of this study is to improve translation quality of English – Latvian texts over each of the individual MT APIs. The selection of the best translation hypothesis is done by calculating the perplexity for each hypothesis using an n-gram language model. The result is a phrase-based multi-system machine translation system that allows to improve MT output compared to individual online MT systems. The proposed approach show improvement up to +1.48 points in BLEU and -0.015 in TER scores compared to the baselines and related research. This section is based on the paper of Rikters and Skadiņa (2016b). The author's contribution to this work is 75%.

3.3.1 Introduction

Although MT has been researched for many decades and there are many online MT systems available, the output of MT systems in many cases still has low quality. The problem of translation quality into under-resourced languages has been also recognized by EU H2020 programme and addressed in QT21 project. The QT21 project investigates novel methods, e.g. hybrid MT, neural network MT, etc. to improve MT output for morphologically rich under-resourced languages.

To address this issue, the MSMT approach can be used, boosting the accuracy and fluency of the translations (Costa-Jussa and Fonollosa, 2015). Our hypothesis is that quality of MT output for under-resourced languages can be increased by applying MSMT that combines outputs of MT systems developed by global players, who have access to large linguistic data, with MT systems developed by MT developers, who pay more attention to particular language and domain.

This section presents several methods how to enrich an MSMT system with linguistic knowledge. The experiments described use multiple combinations of outputs from two, three or four online MT systems. The automatic evaluation results obtained with this hybrid system are analysed and compared with each other. Our approach allowed to increase output by 1.48 BLEU points when translating general domain texts. It is a continuation of an experiment series that started as syntax-based multi-system machine translation (Rikters and Skadiņa 2016a).

3.3.2 Related work

In the last decades, the statistical machine translation has been the dominant research direction in machine translation. However, the quality of the output with state-of-art traditional methods is insufficient in many cases. This has been a reason, why new techniques, including hybrid solutions become more and more popular.

In 2014 the EU-BRIDGE project reported that they achieved significantly better translation performance with gains of up to +1.6 points in BLEU and -1.0 points in TER by combining up to nine different machine translation systems for translation between German and English (Freitag et al., 2014). Recently Freitag et al. (2015) presented novel system combination approach that enhance traditional confusion network system combination approach with an additional model trained by a neural network. Experiments were performed with high-quality input systems for Chinese-English and Arabic-English. The proposed approach yielded in translation improvement from up to +0.9 points in BLEU and -0.5 points in TER.

A more detailed summary of related work can be found in Section 2.4.3.

3.3.3 System description

The major components of the system are the same as in the previous section (3.2.2), and the general workflow is very similar to what was shown in Figure 8.

Pre-processing

The pre-processing step is performed similar to what was described in section 3.2.2, using the Berkeley Parser to obtain initial chunks, and then processing them with the chunk extractor to obtain the parts of the sentence that will be individually translated.

It must be stressed that when translation is performed into morphologically rich language, a simple chunk translation approach will not lead to a better translation. For example, when small chunks are translated into Latvian, they usually will be in canonical form that correspond to subject of sentence but will be incorrect for object. On the other hand, if long chunks are translated, then translation usually breaks agreement rules or translation has wrong word order. Thus, several approaches how to select best chunks for translation have been investigated.

Translation with the APIs

In addition to the APIs used in the previous setup in section 3.2.2, here we added *Yandex Translate*[24] and *Hugo*[25], but no longer used LetsMT! *Yandex Translate* was added due to its recent update adding support for Latvian[26], and LetsMT! was replaced by *Hugo* because it was a newer creation by the same developer team.

Selection of the best translated chunk

Selection of best translation from all possible chunk translations is done by calculating perplexity for each translation as described in section 3.2.2. If two or more translations are identical, the translation is selected as the best. When the best translation for each chunk is selected, the translation of sentence is generated.

Post-processing

The post-processing step is necessary to correct some common mistakes of the translation engines and remove duplicate punctuation marks that result by concatenating chunks into full sentences.

3.3.4 Experiments

Setup

Experiments were conducted on the English – Latvian language pair. Two legal domain corpora – JRC and DGT-TM – were used for language modelling.

---

[24] Yandex Translate API - https://tech.yandex.com/translate/
[25] Latvian public administration machine translation service API - http://hugo.lv/TranslationAPI
[26] https://twitter.com/TranslateYandex/status/624533549765521408

For evaluation two different test sets were used:

- The legal domain test set from section 3.1.3;
- ACCURAT balanced evaluation corpus consisting of 512 sentences (Skadiņa et al., 2010).

Translations were automatically evaluated with two scoring methods – BLEU and TER - and manually inspected in web-based MT evaluation platforms MT-ComparEval and iBLEU to determine, which system from the hybrid setups was selected to get the specific translation of chunk and inspect the differences in translations.

Baseline systems

As the baseline, we used full translations from each individual online API and simple MSMT system (Rikters 2015) that uses only perplexity to select the best translation from outputs of the online APIs. BLEU and TER scores for the baseline systems are presented in Table 10. As it was expected, systems developed by global MT developers show better results for general domain translation, while Latvian public administration MT system is better for translation of legal texts. The baseline MSMT system (using a 6-gram JRC LM) demonstrates lower results as individual systems in legal domain, while for general domain results are close to the best individual system.

Table 10: Automatic evaluation results for baseline systems

Test Set	**JRC**		**ACCURAT**	
System	**BLEU**	**TER**	**BLEU**	**TER**
Bing	16.99	0.695	17.43	0.765
Google	16.19	0.682	17.73	0.749
Hugo	20.27	0.708	17.14	0.764
Yandex	19.75	0.696	16.04	0.776
MSMT - BG	16.38	0.689	17.70	0.755
MSMT - BGH	17.89	0.694	17.63	0.756

Syntax based MSMT systems

We evaluated two approaches in chunk translation – translation of top-level chunks and translation of smaller chunks that are selected based on their properties in sentence.

Simple chunks (SyMHyT)

In first experiment, a parse tree of each sentence is processed by the chunk extractor to obtain the top-level sub-trees (noun phrases, verb phrases, prepositional phrases, etc.). The chunk extractor uses regular expressions to identify sub-trees. When sub-trees are identified,

they are translated with online APIs. Finally, the translation of the sentence is generated by combination of translation hypothesis of sub-trees as it is described in section 3.3.

We evaluated this approach for two SyMHyT systems: *Bing* + *Google* (BG) and *Bing* + *Google* + *Hugo* (BGH). Similar to the baseline MSMT system, SyMHyT also used a 6-gram LM trained on JRC corpus for selection of the best chunk. Evaluation results of this approach are summarized in Table 11. The SyMHyT approach allowed to increase translation quality for combination of *Bing* and *Google* APIs by +0.37 BLEU points to compare with the best baseline (*Bing* 16.99 BLEU points) on legal domain texts. When applied to general domain balanced corpus +0.22 BLEU points are obtained to compare with best baseline (*Google* 17.73 BLEU points).

Table 11. Automatic evaluation results for simple chunk baseline system

Test Set	**JRC**		**ACCURAT**	
System	**BLEU**	**TER**	**BLEU**	**TER**
Bing + Google	17.36	0.672	17.95	0.825
Bing + Google + Hugo	19.50	0.661	17.30	0.817

However, when three APIs were combined, the decrease of BLEU points is observed. To understand why combination of three systems did not improve translation, we analysed translation selection process. Figure 11 shows proportion of translated chunks of different APIs selected by SyMHyT system. When translations of all three systems are used to generate MT output, most of fragments are selected from translations produced by *Hugo.lv.* Since for general domain translation *hugo.lv* showed the worse result, it influenced SyMHyT output and decrease of -0.43 BLEU is observed. In case of legal domain *hugo.lv* showed the best result (+3 BLEU to compare with other baselines), however, since only 63% of fragments were selected from this system, it was insufficient to beat the baseline (-0.77 BLEU).

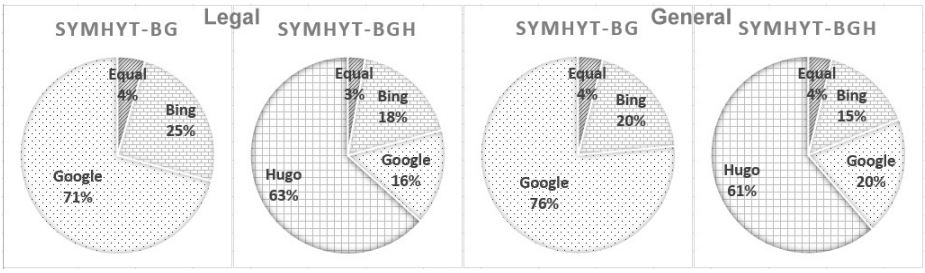


Figure 11: Proportion of chunks selected from translations different APIs

Linguistically motivated chunks (ChunkMT)

Although proposed SyMHyT approach demonstrated some improvement for general domain translation, the analysis of selected translated chunks revealed discrepancy between BLEU score evaluation results and preferences of selection module. In addition, we observed

some obvious flaws, e.g. one-word chunks, one-symbol chunks or very long chunks. This motivated us to investigate more complex algorithm for chunk extraction.

The proposed chunk extractor reads output of the Berkeley Parser and places it in a tree data structure. During this process, each node of the tree is initialised with its phrase (NP, VP, ADVP, etc.), word (if it has one) and a chunk consisting of the chunks from its child nodes. To obtain the final chunks for translation the resulting tree is traversed bottom-up post-order (left to right). A chunk is combined with the previous one, if it is a) non-alphabetical, b) only one symbol, or c) contains genitive phrase. If a chunk is very long (length of chunk > sentence length / 4 in the first chunking iteration), an attempt to break it into smaller chunks is made. Figure 12 illustrates chunk extraction result of both MSMT systems.

SyMHyT	**ChunkMT**
Recently there has been an increased interest in the automated discovery of equivalent expressions in different languages .	Recently there has been an increased interest in the automated discovery of equivalent expressions in different languages .

Figure 12: Examples of chunks extracted by SyMHyT and ChunkMT

The improved MSMT system was evaluated on legal domain and general domain test corpora. For selection of best hypothesis 6-gram and 12-gram language models were used. In almost all cases better results are obtained with higher order language model.

Table 12. Evaluation results for legal domain (JRC test corpus)

	12-gram		**6-gram**	
System	**BLEU**	**TER**	**BLEU**	**TER**
JRC LMs				
BG	17.67	**0.671**	16.70	0.686
HY	**21.38**	**0.681**	20.18	0.703
HG	19.44	**0.677**	-	-
All	**20.33**	**0.668**	18.47	0.698
DGT-TM LMs				
BG	17.61	**0.675**	16.81	0.688
HY	**21.39**	0.684	20.36	0.699
HG	19.86	**0.677**	-	-
All	20.01	**0.667**	17.98	0.699

For legal domain (Table 12), the best result (+1.11 BLEU) is obtained by combining *Yandex* (19.75 BLEU) and *hugo.lv* (20.27 BLEU) systems (HY). Similar to the previous experiments, inclusion of MT systems with significantly lower BLEU scores, produce output which in BLEU points did not exceed the best baseline.

Analysis of selected chunks (Table 13) revealed interesting phenomenon which needs further investigations – when all systems are combined, translations from the best baseline system is used only in 33% of cases, but from the second-best system only in 16.59% of cases.

Table 13. Best results using test data and LM from JRC and selected chunk percentages

System	**BLEU**	**Equal**	**Bing**	**Google**	**Hugo**	**Yandex**
BLEU	-	-	16.99	16.19	20.27	19.75
MSMT - BG	16.38	4.88%	45.03%	50.09%	-	-
MSMT - BGH	17.89	2.78%	34.31%	28.93%	33.98%	-
SyMHyT - BG	17.36	4.59%	24.61%	70.80%	-	-
SyMHyT - BGH	19.50	2.88%	18.01%	15.71%	63.40%	-
ChunkMT - BG	17.67	15.23%	41.14%	43.63%	-	-
ChunkMT - HY	21.38	9.15%	-	-	44.79%	46.06%
ChunkMT - all	20.33	2.94%	27.80%	19.67%	33.00%	16.59%

For general domain data (Table 14), the best result (+1.48 BLEU) is obtained by combining output from all four MT systems. Just like for the legal domain, results of two system combination are better, when better baseline systems are combined. Increase by 0.56 BLEU points is observed when *Bing* and *Google* systems are combined (BG).

Table 14. Evaluation results on ACCURAT balanced test corpus

	12-gram		6-gram	
System	**BLEU**	**TER**	**BLEU**	**TER**
JRC LMs				
BG	17.34	0.757	17.30	0.757
HY	15.72	0.774	15.78	0.775
All	-	-	15.88	0.774
DGT LMs				
BG	**18.29**	0.753	17.81	0.760
HY	17.72	0.757	16.49	0.768
HG	18.06	**0.747**	-	-
All	**19.21**	**0.745**	16.36	0.776

Table 15 presents distribution of selected translated chunks between different MT engines. Most of translations come from *hugo.lv,* which can be explained with choice of legal domain language model, while *Google* and *Bing* were the best baseline systems for general domain.

Table 15. Best results using balanced test data and DGT-TM LM and distribution of selected chunks

System	**BLEU**	**Equal**	**Bing**	**Google**	**Hugo**	**Yandex**
BLEU	-	-	17.43	17.73	17.14	16.04
MSMT - BG	17.70	7.25%	43.85%	48.90%	-	-
MSMT - BGH	17.63	3.55%	33.71%	30.76%	31.98%	-
SyMHyT - BG	17.95	4.11%	19.46%	76.43%	-	-
SyMHyT - BGH	17.30	3.88%	15.23%	19.48%	61.41%	-
ChunkMT - BG	18.29	22.75%	39.10%	38.15%	-	-
ChunkMT - all	19.21	7.36%	30.01%	19.47%	32.25%	10.91%

3.3.5 Conclusions

In this section, we described a machine translation system combination approach that uses syntactic features to extract source text fragments, applies public online MT system APIs for translation and selects translations using statistical features. The results show improvements in BLEU (up to +1.11 for legal domain and +1.48 for general domain) and TER (down to -0.015 for legal domain and -0.004 for general domain) scores compared to the baselines and related research projects.

Experiments described in this section were performed for the English-Latvian language pair, however the framework that realizes described MSMT approach can be applied for other language pairs as well and is freely downloadable from GitHub[27].

3.4 COMBINING SENTENCE FRAGMENT TRANSLATIONS BY EXHAUSTIVELY SEARCHING ACROSS POSSIBILITIES

This section presents an attempt to improve the baseline MSMT combination system described in the previous Section (3.3) by using brute force and searching through all hypotheses for the best-combined translation instead of incrementally building the translation piece by piece. The result is an improved phrase-based MSMT system that boosts the quality of the MT output compared to the baseline while taking much more time to produce the final output. The proposed approach shows improvement up to +3.34 points in BLEU score compared to the baselines and up to +3.61 BLEU compared to related research. This section is based on the paper of Rikters (2016c). The author's contribution to this work is 100%.

3.4.1 Introduction

The problem with the previous approaches is that they can potentially miss some certain combinations of chunks that only score a low perplexity when put together in a full sentence but not necessarily as individual chunks.

With this in mind, as well as the increasing availability of high performance software engineering techniques and computing resources for experimentation, it has become possible to not simply evaluate each individual translated chunk and combine them but also iterate through all variants of different combinations. Doing it this way allows for finding the best version of a specific sentence that only 'looks' good as a whole but not necessarily that good as individual chunks.

3.4.2 System design

The full search MT system combination (FuSCoMT) was developed based on ChunkMT (section 3.3). Therefore, the architecture is very similar to ChunkMT but with a few key differences. The workflow of the system can be decomposed into following steps: pre-processing of the source sentence, acquisition of a translations via online APIs, and generation of MT output, as it is shown in Figure 13. The main difference is in the last step - the manner of scoring chunks with the LM and selecting the best translation. The other big change is the utilisation of multi-threaded computing that allows to run the process on all available CPU cores in parallel.

---

[27] ChunkMT - https://github.com/M4t1ss/ChunkMT

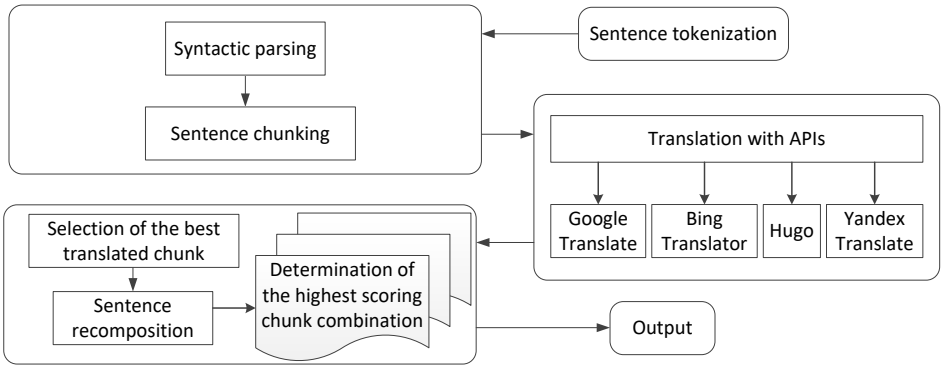


Figure 13. Workflow of the translation process

Translation selection

As opposed to ChunkMT, FuSCoMT firstly generates all unique sequential combinations of translations, using the given chunks. The amount of the combinations is calculated as n^r where n is the amount of different translation engines and r is the number of chunks. Since the translation engines in this case are the same four as in ChunkMT, the combination count will be 4^r.

After that comes the scoring of each full sentence perplexity, using the LM. Finally, when all full-sentence combinations have obtained a perplexity score, the lowest one is elected as the best candidate.

Multi-threaded computing

Since the original code of ChunkMT was written in PHP, the same environment was used for FuSCoMT with several slight additions. To be able to support multiple threads in PHP, the latest version that is PHP7[28] needed to be utilized. Also, for this, the PHP extension pthreads[29] is required.

3.4.3 Experiments

Experiment data

Experiments were conducted on English – Latvian data and three different corpora were used. The DGT-TM was used for training the LM. The legal domain test set as mentioned in section 3.1.3, and the general domain test set as mentioned in section 3.1.3, were used for testing.

[28] PHP7: New Features - http://php.net/manual/en/migration70.new-features.php
[29] PHP: Pthreads - http://php.net/manual/en/book.pthreads.php

Table 16 and Figure 14 outline the statistics of chunks obtained from the test data. The legal domain test data contained a large number of sentences that were split into six or more chunks. Since there are 4^9 or 262 144 different combinations possible for a sentence that is split into nine chunks, these experiments were computationally too expensive. Therefore, the maximum number of allowed chunks was limited to 9, although the chunker may have been able to produce more.

Table 16: Statistics of the test data

Chunks	**Combinations**	**Count**		**Percentage**	
		Legal	**General**	**Legal**	**General**
1	4	210	16	13.28%	3.13%
2	16	178	78	11.26%	15.23%
3	64	262	131	16.57%	25.59%
4	256	273	127	17.27%	24.80%
5	1024	275	94	17.39%	18.36%
6	4096	201	47	12.71%	9.18%
7	16384	96	11	6.07%	2.15%
8	65536	49	6	3.10%	1.17%
9	262144	37	2	2.34%	0.39%

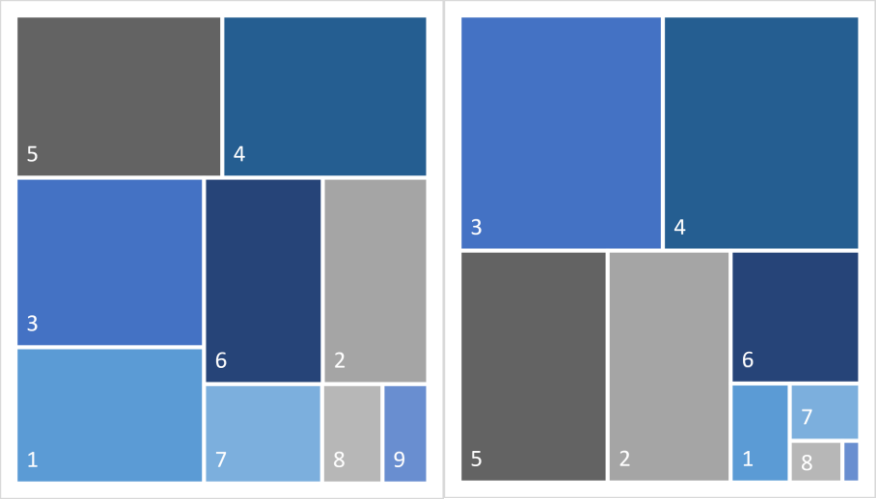


Figure 14. Chunk count visualization for the legal domain test set (left) and for the general domain test set (right).

Experiment results

To make the experiments comparable to the baseline MSMT system, the same corpora were used for both – training the LM and preparing test data. The translation quality results of the experiments are shown in Table 17. The time required to run these experiments was not measured but it was significantly higher than an unmodified version of ChunkMT.

Table 17: Experiment results

System	**BLEU**	
	Legal	**General**
Full-search	**23.61**	14.40
ChunkMT	20.00	17.27
Bing	16.99	17.43
Google	16.19	**17.72**
Hugo	20.27	17.13
Yandex	19.75	16.03

Example sentence analysis

In this section an analysis of one particular sentence is given in more detail to show the differences in how the full-search method compares to the single-best chunk selection that is used in ChunkMT. The sentence was split into three chunks by the chunker and each chunk was translated with the four MT APIs.

Table 18 shows the full sentence with the lowest perplexity score in comparison with a sentence that consists of the lowest perplexity scoring individual chunks and also some other possible sentence chunk combinations and their perplexities. Table 19 provides information on the perplexity scores of each chunk translated by each MT API. In both tables chunks and the sentence made up of chunks with the lowest perplexities are marked in bold whereas chunks and the sentence scoring best only when combined are marked in cursive.

Table 18: Example full sentence perplexities

System	Sentence / Chunk	Perplexity
Full-search	*Šis lēmums stājas spēkā tā publicēšanas dienā oficiālajā vēstnesī .*	*16.57*
ChunkMT	**šo lēmumu . stājas spēkā tās publicēšanas dienā , oficiālajā vēstnesī .**	**132.14**
Other possible variants	šo lēmumu lēmums stājas spēkā trešajā dienā pēc tās publicēšanas valsts oficiālajā vēstnesī.	54.31
	Šis lēmums lēmums stājas spēkā trešajā dienā pēc tās publicēšanas valsts oficiālajā vēstnesī .	68.82
	Šis lēmums stājas spēkā tās publicēšanas dienā Savienības Oficiālajā Vēstnesī .	21.79

Table 19: Example sentence individual chunk perplexities

System	Chunk / Perplexity		
Bing	*Šis lēmums*	*lēmums stājas spēkā trešajā dienā pēc tās publicēšanas*	Savienības Oficiālajā Vēstnesī.
	70.73	*33.21*	678.29
Google	šis lēmums	stājas spēkā tā publicēšanas dienā	oficiālajā vēstnesī.
	568.43	*64.58*	6858.23
Hugo	**šo lēmumu .**	**stājas spēkā tās publicēšanas dienā ,**	valsts oficiālajā vēstnesī.
	48.04	**23.91**	951.49
Yandex	šo lēmumu	stājas spēkā tās publicēšanas dienā	***oficiālajā vēstnesī .***
	760.09	61.66	***164.97***

3.4.4 Conclusions

The obtained results show that the purposed approach produces a higher quality output when the chunk counts of the input data is distributed more evenly like in the JRC legal domain test data. On the other hand, when more than a half of the sentences consist of three or four chunks, the baseline ChunkMT is still the best performer. It is also worth mentioning that due to the high number of perplexity scores that needed to be calculated for some sentences in the test data, the experiments took a rather high amount of time to perform – from a few days up to over a week.

3.5 COMBINING SENTENCE FRAGMENT TRANSLATIONS WITH NEURAL NETWORK LANGUAGE MODELS

This section presents the comparison of how using different neural network-based language modelling tools for selecting the best candidate fragments affects the final output translation quality in a hybrid multi-system machine translation setup. Experiments were conducted by comparing perplexity and BLEU scores on common test cases using the same training dataset. A 12-gram statistical language model was selected as a baseline to oppose three neural network-based models of different characteristics. The models were integrated in a hybrid system that depends on the perplexity score of a sentence fragment to produce the best fitting translations. The results show a correlation between language model perplexity and

BLEU scores as well as overall improvements in BLEU. This section is based on the paper of Rikters (2016d). The author's contribution to this work is 100%.

3.5.1 Introduction

Some recent open-source MSMT approaches tend to use statistical language models (LMs) for scoring and comparing candidate translations or translation fragments. It is understandable, because the statistical approaches have been dominant for the past decades. Whereas lately, neural networks (NNs) have been showing increasingly greater potential in modelling long distance dependencies in data when compared to state-of-the-art statistical models. Therefore, the aim of this research is to utilise this potential in combining translations.

Since LMs are probability distributions over sequences of words, they are a great tool for estimating the relative likelihood of whether some sequence of words belongs to a certain language. In the previous experiments (sections 3.1, 3.2 and 3.3), different order LMs were used in the described MSMT approaches. This last system that was presented in section 3.3 and the statistical model from KenLM that it uses will be treated as the baseline for further experiments.

This section presents an enrichment of the existing MSMT tool with the addition of neural language models. The experiments described use multiple combinations of outputs from online MT sources. Experiments described in this section are performed for English-Latvian. Translating from and to other languages is supported, but it has some limitations as described in the previous section. The code of the developed system is freely available at GitHub[30].

The structure of this section is as following: subsection 3.5.2 describes the architecture of the baseline system. Subsection 3.5.3 outlines the LM toolkits that are used in the experiments and subsection 3.5.4 provides the experiment setup and results.

3.5.2 System description

The core components of the system have not changed from the ones mentioned the previous sections (3.2.2, 3.3.3), and the general workflow is very similar to what was shown in Figure 8.

Going into more detail on the chunking part of the pre-processing step, Figure 15 represents the basic workflow for that. The syntax tree of a sentence is traversed bottom-up, right to left and combines smaller subtrees with bigger ones when possible thereby creating chunks that are no longer than a quarter of tokens or words in the sentence. This specific maximum length for chunks was chosen in previous experiments that showed a general decrease of translation quality or no changes at all for longer maximum chunks. However, if

[30] Machine translation system combination using neural network language models - https://github.com/M4t1ss/BatchChunkCombiner

the chunker returns a large number of chunks for a single sentence, this maximum ratio can be adjusted further. More details on the chunking can be found in sections 3.2 and 3.3.

For translation, the same online MT systems were used as in section 3.3.3. Source languages require compliance with Berkeley Parser parse grammars. The parser is able to learn new grammars from treebanks. Target languages require a language model that is compliant with either KenLM or one of the NN LM tools. New LMs can also be trained using monolingual plain text files as input.

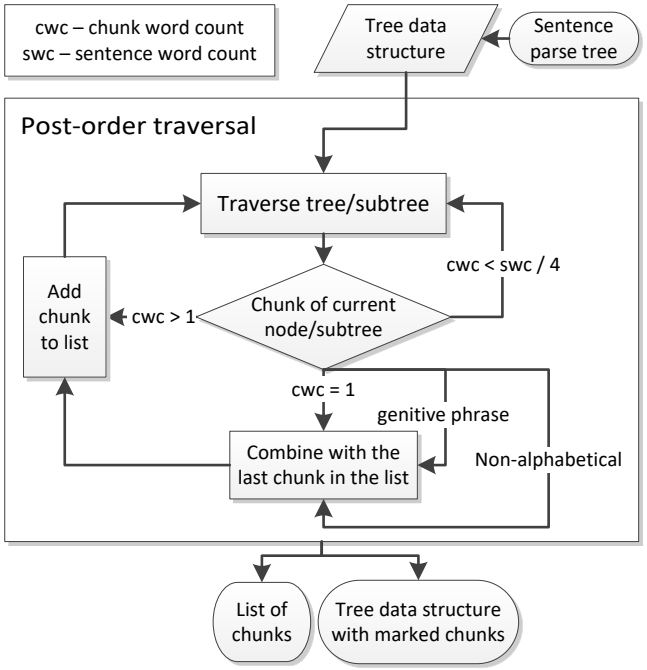


Figure 15. Illustration of how chunks are selected

3.5.3 Language models

Baseline

The baseline language model was trained with the statistical LM toolkit – KenLM - one of the most popular LM tools, integrated into many phrase-based MT systems like Moses (Koehn et al., 2007), cdec (Dyer et al., 2010), and Joshua (Li et al., 2009). It does the job quite efficiently, thus, it was included as the only LM option in the baseline system. For training, a large order of 12 was chosen for maximum quality.

RWTHLM

RWTHLM is a toolkit for training many different types of neural network language models (Sundermeyer et al., 2014). It has support for feed-forward, recurrent and long short-term memory NNs (Hochreiter and Schmidhuber, 1997; Gers et al., 2000). While training

different NN configurations, the best results were achieved with a model consisting of one feed-forward input layer with a 3-word history, followed by one linear layer of 200 neurons with sigmoid activation function.

MemN2N

MemN2N trains an end-to-end memory network (Sainbayar et al., 2015) model for language modelling. It is a neural network with a recurrent attention model over a possibly large external memory with architecture of a memory network. Because it is trained end-to-end, the approach requires significantly less supervision during training.

MemN2N requires Torch[31] scientific computing framework to be installed for running. Torch is an open source machine learning library that provides a wide range of algorithms for deep learning. For training, the default configuration was used with an internal state dimension of 150, linear part of the state 75 and number of hops set to six.

Char-RNN

Char-RNN[32] is a multi-layer recurrent neural network for training character-level language models. It has support for recurrent NNs, long short-term memory (LSTM) and gated recurrent units (GRUs).

To run Char-RNN on a CPU, a minimum installation of Torch is also required. Running on a GPU requires some additional Torch packages. The best scoring model was trained using 2 LSTM layers with 1024 neurons each and the dropout parameter set to 0.5.

Environment

The translation experiments were carried out on Ubuntu server with 16GB RAM and 4 cores. This was sufficient because querying the models requires far less computation power than training.

Experiments for LM training and perplexity evaluation were done on three desktop workstation machines with different configurations. The KenLM and RWTHLM models were trained on an 8-core CPU with 16GB of RAM. For training MemN2N a GeForce Titan X (12GB memory, 3072 CUDA cores) GPU with a 12-core CPU and 64GB RAM. The Char-RNN model was trained on a Radeon HD 7950 (3GB memory, 1792 cores) GPU with an 8-core CPU and 16GB RAM.

---

[31] A scientific computing framework for luajit - http://torch.ch
[32] Multi-layer Recurrent Neural Networks (LSTM, GRU, RNN) for character-level language models in Torch https://github.com/karpathy/char-rnn

3.5.4 Experiments

Data

To train the LMs, the Latvian monolingual part of the DGT-TM was used. In the case of training an LM with Char-RNN only the first half of this corpus (1.5 million sentences) was used in order to speed up the training process as well as because the character level model requires much less training data when compared with the others. When training all NN LMs evaluation and validation datasets were automatically derived from the training data with the proportion of 97% for training, 1.5% for validation and 1.5% for testing. The evaluation data consisted of 1134 sentences randomly selected out of a different legal domain corpus – JRC (section 3.1.3).

Test datasets were made up from the legal domain test set as mentioned in section 3.1.3, and the general domain test set as mentioned in section 3.1.3.

A 12-gram language model for the baseline was trained using KenLM.

Language modelling experiments

To justify using different language modelling approaches, different language models were trained with the same and similar (half of the corpus in one case) training data. Table 20 shows differences in perplexity evaluations that outline the superiority of NN LMs. It also shows that the statistical model is much faster to train on a CPU and that NN LMs train more efficiently on GPUs.

Table 20. Results of language model perplexity experiments.

System	Perplexity	Corpus size	Trained on	Training time	BLEU
KenLM	34.67	3.1M	CPU	1 hour	19.23
RWTHLM	136.47	3.1M	CPU	7 days	18.78
MemN2N	25.77	3.1M	GPU	4 days	18.81
Char-RNN	24.46	1.5M	GPU	2 days	19.53

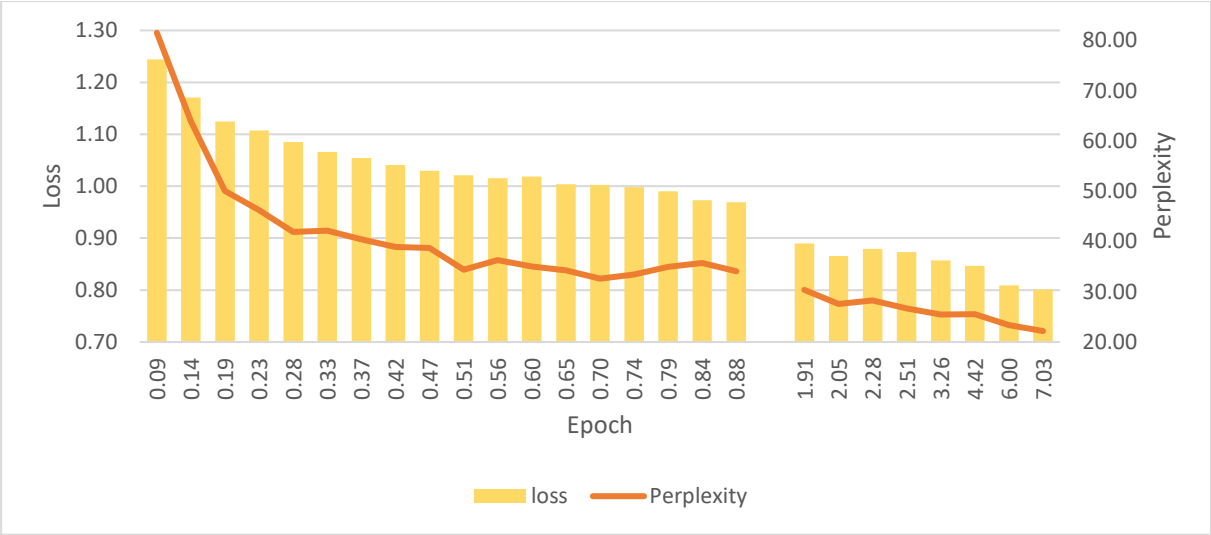


Figure 16. Changes of training loss and perplexity when training a two-layer Char-RNN with 512 neurons on 500 000 sentences.

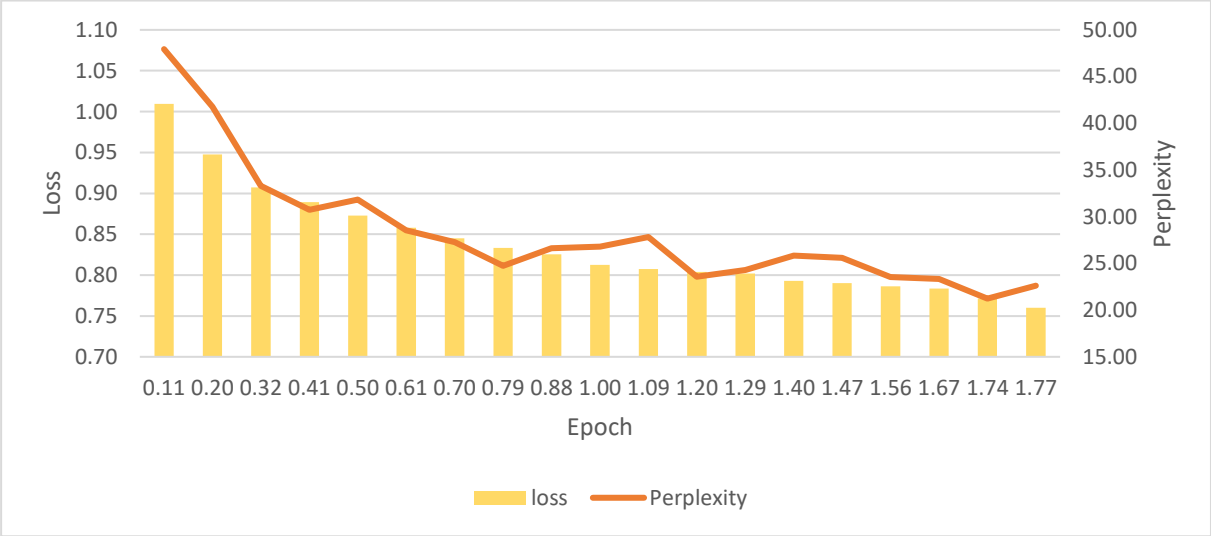


Figure 17. Changes of training loss and perplexity when training a three-layer Char-RNN with 1024 neurons on 1 million sentences

Since Char-RNN achieved the best results, several in-depth experiments were conducted using just this tool with varying training dataset sizes (for faster training) and NN layer combinations. Figure 16 shows how the network evolves in a setup with two 512-neuron layers. This experiment was conducted on a smaller dataset – only $1/6^{th}$ of the corpus – allowing it to run for more epochs without early stopping. The perplexity on test data gradually decreased, reaching a lowest score of 22.18.

Another variation for training a LM with Char-RNN is shown in Figure 17. Here 1/3rd of the corpus was used to train a 3-layer RNN with 1024 neurons per layer. The lowest achieved perplexity was 21.23 after training one day on a GPU.

Machine translation experiments

The last column of Table 20 shows BLEU scores for different NN LMs. Correlation between LM perplexity and the resulting BLEU score is visible as well as a slight improvement in the overall result. Again, due to the outstanding scores of Char-RNN models, they were inspected closer to see how BLEU changes along with perplexity.

The following charts show how perplexity correlates with BLEU in translation test cases on the general domain and legal domain test datasets. Figure 18 represents results from evaluation of a combination of Google and Bing (BG) online MT translations (denoted with darker blue colours) and a combination of Hugo and Yandex (HY) online MT (brighter blue colours) on the general domain test dataset. The trend lines (dotted) indicate that for this dataset the combination of BG stays mostly stable but the combination of HY gradually improves as the perplexity of the LM gets lower.

Figure 19 shows results of combining the same MT systems on the legal domain test dataset. In this case, while perplexity becomes lower at each step, the linear trend line for BLEU score of the BG hybrid system does not show a tendency towards climbing higher. As opposed to the BLEU score trend line for HY hybrid system, which showcase improvement along with perplexity.

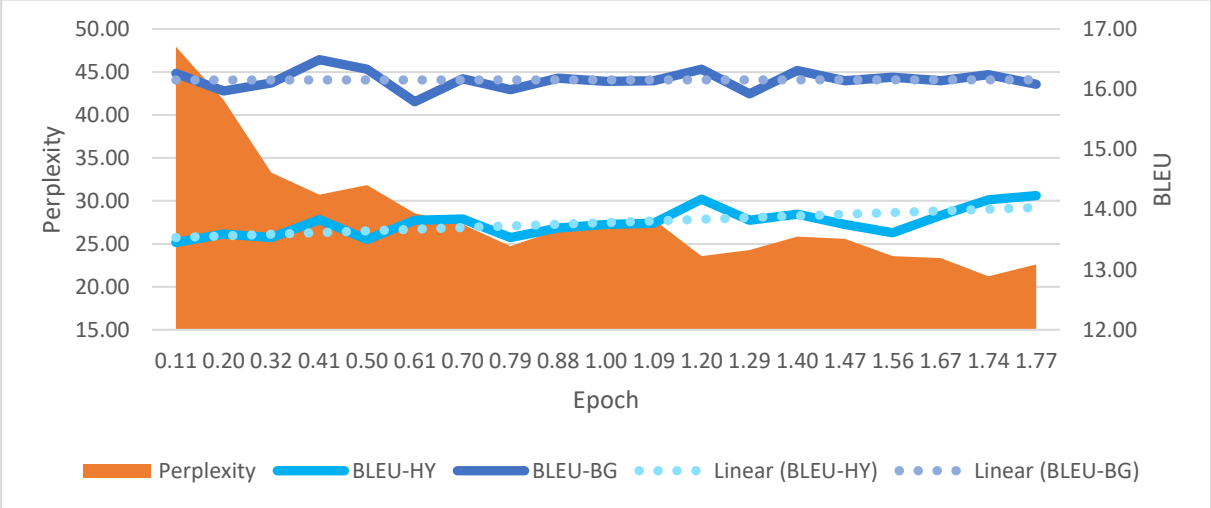


Figure 18. Changes of perplexity when training a three-layer Char-RNN with 1024 neurons on 1 million sentences and its effect on BLEU score when used in MSMT for combining Bing and Google (BG); Hugo and Yandex (HY) on the general domain test dataset.

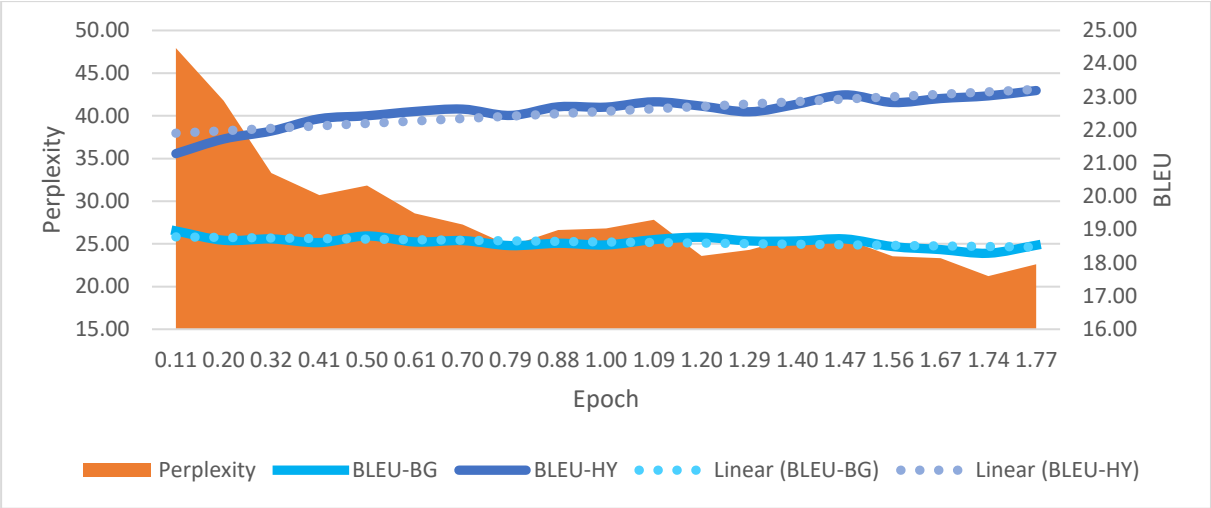


Figure 19. Changes of perplexity when training a three-layer Char-RNN with 1024 neurons on 1 million sentences and its effect on BLEU score when used in MSMT for combining Bing and Google (BG); Hugo and Yandex (HY) on the legal domain test dataset.

3.5.5 Conclusions

This section analysed ways to improve the baseline MSMT system with neural network language models. Test cases showed an improvement in BLEU score, when used only with Google and Bing, by 0.35 BLEU points.

In the detailed translation experiments where a BLEU score was obtained in every stage of the LM training there was only a steady correlation of BLEU and perplexity in the case of using Hugo and Yandex translations, which were very different (0.52 – 1.10 BLEU difference with each other) to begin with. In the case of combining Google and Bing translations where the difference was far less significant (0.3 – 0.8 BLEU difference with each other), the BLEU scores of the NN hybrid model were less uniform with perplexity. This indicates that out of very similar options, even the NN model fluctuates with its predictions but it gets more confident when the difference is more obvious.

4. COMBINING NEURAL MACHINE TRANSLATION OUTPUT

4.1 FINDING CORRELATION BETWEEN NEURAL NETWORK ATTENTION AND OUTPUT TRANSLATION QUALITY

Processing of multi-word expressions (MWEs) is a known problem for any natural language processing task. Even neural machine translation (NMT) struggles to overcome it. Since MWEs are often groups of words that have a specific meaning when viewed together, they make great subjects for exploring if NMT systems can learn to handle them as a union. This section presents results of experiments on investigating NMT attention allocation to the MWEs and improving automated translation of sentences that contain MWEs in English → Latvian and English → Czech NMT systems. Two improvement strategies were explored - (1) bilingual pairs of automatically extracted MWE candidates were added to the parallel corpus used to train the NMT system, and (2) full sentences containing the automatically extracted MWE candidates were added to the parallel corpus. Both approaches allowed to increase automated evaluation results. The best result - 0.99 BLEU point increase - has been reached with the first approach, while with the second approach minimal improvements achieved. We also provide open-source software and tools used for MWE extraction and alignment inspection.

The experiments described in this section helped the author comprehend possible use-cases for NMT attention alignments. The achieved results were essential to enable NMT system combination described in sections 4.2 and 4.3. This section is based on the paper of Rikters and Bojar (2017). The author's contribution to this work is 80%.

4.1.1 Introduction

It is a well-known fact that NMT has defined the new state-of-the-art in the last few years (Sennrich et al., 2016a; Wu et al., 2016), but the many specific aspects of NMT outputs are not yet explored. One of which is translation of multi-word units or multi-word expressions (MWEs). MWEs are defined by Baldwin and Kim (2010) as "lexical items that: (a) can be decomposed into multiple lexemes; and (b) display lexical, syntactic, semantic, pragmatic and/or statistical idiomaticity". MWEs have been a challenge for statistical machine translation (SMT). Even if standard phrase-based models can copy MWEs verbatim, they suffer in grammaticality. NMT, on the other hand, may struggle in memorizing and reproducing MWEs, because it represents the whole sentence in a high-dimensional vector, which can lose the specific meanings of the MWEs even in the more fine-grained attention model (Bahdanau et al., 2015), because MWEs may not appear frequently enough in the training data.

The goal of this research is to examine how MWEs are treated by NMT systems, compare that with related work in SMT, and find ways to improve MWE translation in NMT. We aimed to compare how NMT pays attention to MWEs during translation, using a test set particularly

targeted at handling of MWEs, and if that can be improved by populating the training data for the NMT systems with parallel corpora of MWEs.

The objective was to obtain a comparison of how NMT with regular training data and NMT with synthetic MWE data pays attention to MWEs during the translation process as well as to improve the final NMT output. To achieve this objective, it needed to be broken down into smaller sub-objectives:

- Train baseline NMT systems,
- Extract parallel MWE corpora from the training data,
- Train the NMT systems with synthetic MWE data, and
- Inspect alignments produced by the NMT.

The structure of this section is as follows: 4.1.2 summarizes related work in translating MWEs with SMT and NMT. 4.1.3 describes the architecture of the baseline system and outlines the process of extracting parallel MWE corpora from the training data. 4.1.4 provides the experiment setup and results. Finally, conclusions and aims for further directions of work are summarized in 4.1.5.

4.1.2 Related work

There have been several experiments with incorporating separate processing of MWEs in rule-based (Deksne et al., 2008) and statistical machine translation tasks (Bouamor et al., 2012; Skadiņa, 2016). However, there is little literature about similar integrations in NMT workflows so far.

Skadiņa (2016) performed a series of experiments on extracting MWE candidates and integrating them in SMT. The author experimented with several different methods for both the extraction of MWEs and integration of the extracted MWEs into the MT system. In terms of automatic MT evaluation, this allowed to achieve an increase of ~0.5 BLEU points for an English → Latvian SMT system.

Tang et al. (2016) introduce an NMT approach that uses a stored phrase memory in symbolic form. The main difference from traditional NMT is tagging candidate phrases in the representation of the source sentence and forcing the decoder to generate multiple words all at once for the target phrase. Although they do mention MWEs, no identification or extraction of MWEs is performed and the phrases they mainly focus on are dates, names, numbers, locations, and organizations, which are collected from multiple dictionaries. For Chinese → English they report a 3.45 BLEU point increase over baseline NMT.

Cohn et al. (2016) describe an extension of the traditional attentional NMT model with the inclusion of structural biases from word-based alignment models, such as positional bias, Markov conditioning, fertility and agreement over translation directions. They perform

experiments translating between English, Romanian, Estonian, Russian and Chinese and analyse the attention matrices of the output translations produced by running experiments using the different biases. Specific experiments targeting MWEs are not performed, but they do point out that using fertility, especially global fertility, can be useful for dealing with multi-word expressions. They report a statistically significant improvement of BLEU scores in almost all involved language pairs.

Chen et al. (2016) use a similar approach as we do. Their "bootstrapping" automatically extracts smaller parts of training segment pairs and adds them to the training data for NMT. The main difference is that they rely on automatic word alignment and punctuation in the sentence to identify matching sub-segments.

4.1.3 Data preparation and systems used

To measure changes introduced by adding synthetic MWE data to the training corpora, first, a baseline NMT system was trained for each language pair. The experiments were conducted on English → Czech and English → Latvian translation directions.

Baseline NMT system

To be able to compare the results with other MT systems, training and development corpora were used from the WMT shared tasks: data from the News Translation Task[33] for English → Latvian and data from the Neural MT Training Task[34] (Bojar et al., 2017a) for English → Czech. The English → Czech data consists of about 49 million parallel sentence pairs and the English → Latvian of about 4.5 million. The development corpora consist of 2003 sentences for English → Latvian and 6000 for English → Czech.

Neural Monkey (Helcl and Libovický, 2017), an open-source tool for sequence learning, was used to train the baseline NMT systems. Using the configuration provided by the WMT Neural MT Training Task organizers, the baseline reached 11.29 BLEU points for English → Latvian after having seen 23 million sentences in about 5 days and 13.71 BLEU points for English → Czech after having seen 18 million sentences in about 7 days.

Extraction of parallel MWEs

To extract MWEs, the corpora were first tagged with morphological taggers: UDPipe (Straka et al., 2016) for English and Czech, LV Tagger (Paikens et al., 2013) for Latvian. After that, the tagged corpora were processed with the Multi-word Expressions toolkit (Ramisch, 2012), and finally aligned with the MPAligner (Pinnis, 2013), intermittently pre-processing and post-processing with a set of custom tools. To extract MWEs from the corpora with the MWE

[33] http://www.statmt.org/wmt17/translation-task.html
[34] http://www.statmt.org/wmt17/nmt-training-task

Toolkit, patterns were required for each of the involved languages. Patterns from Skadiņa (2016) were used for Latvian (210 patterns) and English (57 patterns) languages and patterns from Majchrakova et al. (2012) and Pecina (2008) for Czech (23 patterns).

This workflow allowed to extract a parallel corpus of about 400 000 multi-word expressions for English → Czech and about 60 000 for English → Latvian. For an extension of this experiment, all sentences containing these MWEs were also extracted from the training corpus, serving as a separate parallel corpus.

4.1.4 Experiments

We experiment with two forms of the presentation of MWEs to the NMT system: (1) we add only the parallel MWEs themselves, each pair forming a new "sentence pair" in the parallel corpus, and (2) we use full sentences containing the MWEs. We denote the approaches *MWE phrases* and *MWE sents.* in the following.

Training corpus layout

Figure 20: Portions of the final training dataset for English → Czech

Figure 21: Portions of the final training dataset for English → Latvian

In both cases, we use the same corpus training corpus layout: we mix the baseline parallel corpus with synthetic data so that MWEs get more exposure to the neural network in training and hopefully allow NMT to learn to translate them better. Figure 20 and Figure 21 illustrate how the training data was divided into portions. The block 1xMWE corresponds to the full set of extracted MWEs (400K for En → Cs, 60K for En → Lv) and 2xMWE corresponds to two copies of the set (800K for En → Cs, 120K for En → Lv). For En → Lv the full corpus was used. For En →Cs we used only the first 15M sentences to be able to train multiple epochs on the available hardware. The MWEs get repeated five times in both language pairs. By doing this, the En → Cs dataset was reduced from 49M to 17M and the En → Lv dataset increased to 4.8M parallel sentences for one epoch of training.

While the experiments were running, early stopping of the training was executed and snapshots of the models for evaluation were taken in stages where the models already were starting to converge. For En → Lv this was after the networks had been trained on 25M sentences (i.e. 5.2~epochs of the mixed corpus), for En → Cs 27M sentences (i.e. 1.6~epochs).

Neural Monkey does not shuffle the training corpus between epochs. This is not a problem if the corpus is properly shuffled and the number of epochs is not very large compared to the size of the epochs. We shuffled only the baseline corpus and the interleaved it with (shuffled) sections for MWEs. This worked well when MWEs were provided in full sentences, but not with MWEs presented as expressions. In the latter case, the NMT started to produce only very short output, losing very much of its performance. We, therefore, shuffle the whole composed corpus for the *MWE phrases* runs, effectively discarding the interleaved composition of the training data.

Results

Table 21 shows the results for each approach on one language pair. Due to hardware constraints, we were not able to try out both approaches on both language pairs.

We evaluate all setups with BLEU on the full development set (distinct from the training set), as shown in the column *Dev*, and on a subset of 611 (En → Lv) and 112 (En → Cs) sentences containing the identified MWEs (column *MWE*).

Table 21: BLEU scores of experiments

Languages	**En → Cs**		**En → Lv**	
Dataset	**Dev**	**MWE**	**Dev**	**MWE**
Baseline	13.71	10.25	11.29	9.32
+MWE phrases	-	-	11.94	10.31
+MWE sents.	13.99	10.44	-	-

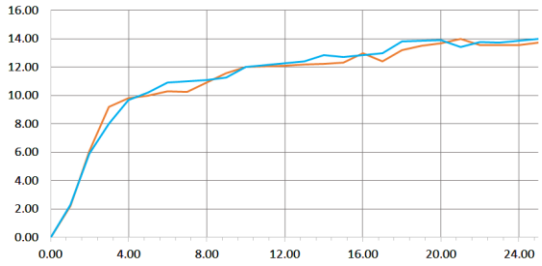


Figure 22: Automatic evaluation progression of En → Cs experiments on validation data. Orange - baseline; blue - baseline with added MWEs.

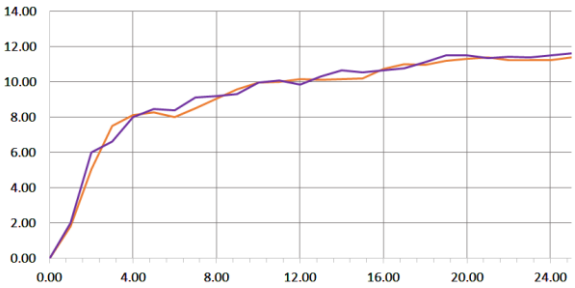


Figure 23: Automatic evaluation progression of En → Lv experiments on validation data. Orange - baseline; purple - baseline with added MWE sentences.

Figure 22 and Figure 23 illustrates the learning curves in terms of millions of sentences, as evaluated on the full development set.

We see that the difference on the whole development set is not very big for either of the languages, and that it fluctuates as the training progresses.

The improvement is more apparent when evaluated on the dedicated development dataset of sentences containing multi-word expressions. Even though the improvement for Latvian is 0.99 BLEU, it must be noted that the baseline performance of our system is not very high. Also, more runs should be carried out for a full confidence, but this was unfortunately out of our limits on computing resources.

Sentence	BLEU	Length ratio	Text
Source	-	-	Just like in a city bus or a tram .
Target	100.00	1.00	Stejně jako v městském autobuse či tramvaji .
Baseline	13.54	0.88	jako ve městě autobuse nebo tramvaji .
Improved NMT	41.11	1.00	jen jako v městském autobuse nebo tramvaji .

Figure 24: Differences in translation between baseline and improved NMT system. Improving n-grams are highlighted in green and worsening n-grams - in red.

Sentence	BLEU	Length ratio	Text
Source	-	-	He steps toward the electronic wall map depicting Australia and the surrounding sea areas .
Target	100.00	1.00	Přistoupí k nástěnné elektronické mapě , na níž je znázorněna Austrálie a přilehlé mořské oblasti .
Baseline	10.74	0.75	ukazuje na mapu elektronické stěny zobrazuje Austrálii a okolní mořské oblasti .
Improved NMT	13.87	0.81	vykročil směrem k elektronické mapě , které depicting Austrálie a okolní oblasti .

Figure 25: Differences in translation of a Czech sentence using baseline and improved NMT systems. Improving n-grams are highlighted in green and worsening n-grams - in red.

Sentence	BLEU	Length ratio	Text
Source	-	-	It should be noted that this is not the first time that Facebook has been actively involved in determining what network users see in their news feeds .
Target	100.00	1.00	Jāteic , ka šī nav pirmā reize , kad Facebook aktīvi iesaistās , nosakot , ko tīkla lietotāji redz savās jaunumu plūsmās .
Baseline	10.41	1.04	Jāatzīmē , ka šis nav pirmajā reizē , kad Facebook ir aktīvi iesaistīta , nosakot to , ko tīklā izmanto viņu ziņu pārraides .
Improved NMT	27.41	1.13	Ir jāatzīmē , ka šis ir pirmā reize , kad Facebook ir aktīvi iesaistījusies , nosakot to , ko tīkla lietotāji dara viņu ziņu formātā .

Source	It should be noted that this is not the first time that Facebook has been actively involved in determining what network users see in their news feeds.
Baseline	Jāatzīmē, ka šis nav pirmajā reizē, kad Facebook ir aktīvi iesaistīta, nosakot to, ko tīklā izmanto viņu ziņu pārraides.
Improved NMT	Ir jāatzīmē, ka šis ir pirmā reize, kad Facebook ir aktīvi iesaistījusies, nosakot to, ko tīkla lietotāji dara viņu ziņu formātā.
Reference	Jāteic, ka šī nav pirmā reize, kad Facebook aktīvi iesaistās, nosakot, ko tīkla lietotāji redz savās jaunumu plūsmās.

Figure 26: Differences in translation between baseline and improved NMT system. Improving n-grams are highlighted in green and worsening n-grams - in red.

Manual inspection

To find out whether changes in the results are due to the synthetic MWE corpora added, a subset of output sentences from the ones containing MWEs were selected for closer examination. For this task, we used the iBLEU tool.

In Figure 24, an improvement in the modified NMT translation is visible due to the treatment of the compound nominal “city bus” as a single expression. It seems that the baseline system translates “city” into “městě” and “bus” into “autobuse” individually, resulting in the wrong form of “city" in Czech (a noun used instead of an adjective). On the other hand, the improved NMT translates “city” into “městském” just like the target human translation. Attention alignments will be examined in the following section.

Figure 25 shows an example where the improved NMT scores higher in BLEU points and translates the MWE closer to the human but loses a part of it in the process. While translating the noun phrase “electronic wall map” the improved system generates a closer match to the human translation “elektronické mapě”, it does not translate the word “wall” that was translated into “stěny” by the baseline system. Upon closer inspection, we discovered that this error was caused by the MWE extractor and aligner because the identified English phrase “electronic wall map” was aligned to an identified Czech phrase “elektronické mapě” and the whole phrase “nástěnné elektronické mapě” was not identified by the MWE extractor at all.

Figure 26 illustrates translations of an example sentence by the En → Lv NMT systems. The MWE, in this case, is “network users” that is translated as “tīkla lietotāji” by the modified system and completely mistranslated by the baseline.

Alignment inspection

For inspecting the NMT attention alignments, we developed a tool (Rikters et al., 2017a) that takes data produced by Neural Monkey - a 3D array (tensor) filled with the alignment probabilities together with source and target subword units (Sennrich et al., 2016b) or byte pair encodings (BPEs)-as input and produces a soft alignment matrix (Figure 5) of the subword units that highlights all units, that get attention when translating a specific subword unit. The tool includes a web version that was adapted from Nematus (Sennrich et al., 2017) utilities and slightly modified. It allows to output the soft alignments in a different perspective, as connections between BPEs as visible in Figure 27 and Figure 28.

In these examples, the attention state of the previously mentioned MWE from En → Lv translations ("network users") is visible. The alignment inspection tool allows to see that the baseline NMT in Figure 27 has multiple faded alignment lines for both words "network" and "users", which outlines that the neural network is unsure and looking all around for traces to the correct translation. However, in Figure 28, it is visible that both these words have strong alignment lines to the words "tīkla lietotāji", that were also identified by the MWE Toolkit as an MWE candidate.

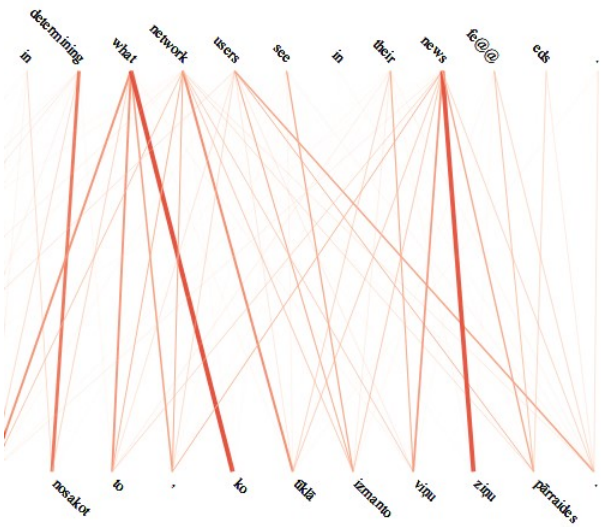


Figure 27: Fragment of soft alignments of the example sentence from the baseline NMT system.

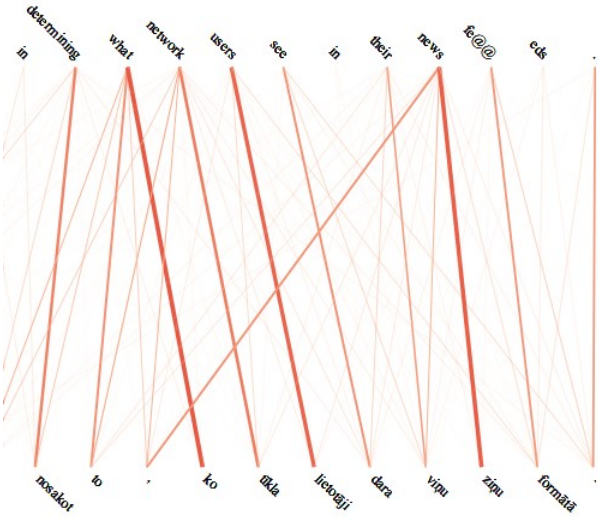
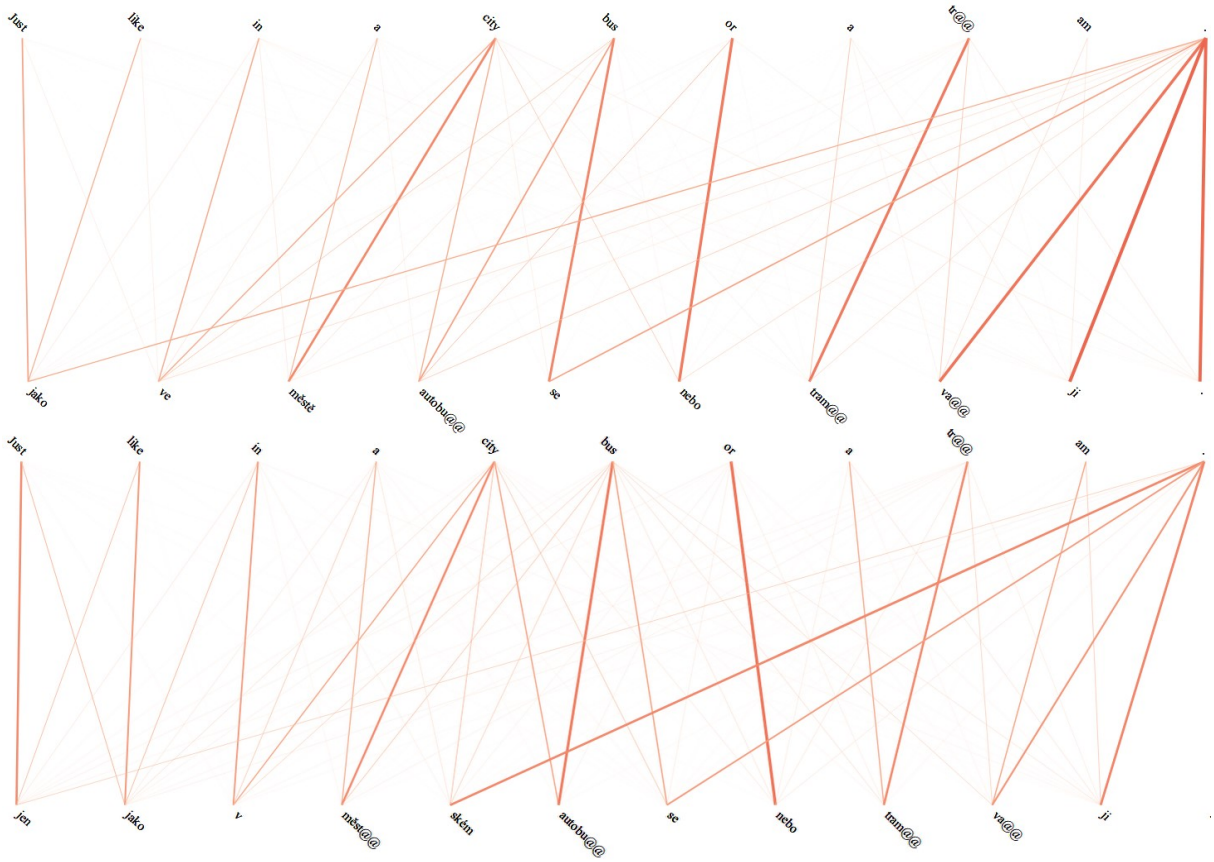


Figure 28: Fragment of soft alignments of the example sentence from the improved NMT system.



Source	Just like in a city bus or a tram.
Baseline	Jako ve městě autobuse nebo tramvaji.
Improved NMT	Jen jako v městském autobuse nebo tramvaji.
Reference	Stejně jako v městském autobuse či tramvaji.

Figure 29: Soft alignment example visualizations from translating an English sentence into Czech from the baseline (top, hypothesis 1) and improved (bottom, hypothesis 2) NMT systems.

Figure 29 shows one of the previously mentioned En → Cs translation examples. Here it is clear that in the baseline alignment no attention goes to the word “městě” or the subword units “autobu@@” and “se” when translating “city”. In the modified version, on the other hand, some attention from “city” goes into all closely related subword units: “měst@@”, “ském”, “autobu@@”, and “se”. It is also visible that in this example, the translation of “bus" gets attention from not only “autobu@@” and “se”, but also the ending subword unit of “city”, i.e. the token “ském”.

4.1.5 Conclusions

In this section, we described the first experiments with handling multi-word expressions in neural machine translation systems. Details on identifying and extracting MWEs from parallel corpora, as well as aligning them and building corpora of parallel MWEs were provided. We explored two methods of integrating MWEs in training data for NMT and

examined the output translations of the trained NMT systems with custom built tools for alignment inspection.

In addition to the methods described in this section, we also released open-source scripts for a complete workflow of identifying, extracting and integrating MWEs into the NMT training and translation workflow.

While the experiments did not show outstanding improvements on the general development dataset, an increase of 0.99 BLEU was observed when using an MWE specific test dataset. Manual inspection of the output translations confirmed that translations of specific MWEs were improving after populating the training data with synthetic MWE data.

4.2 SIMPLE SYSTEM COMBINATION USING NEURAL NETWORK ATTENTION

This section describes the NMT systems of the combined effort of the University of Latvia, University of Zurich and University of Tartu. We participated in the WMT 2017 shared task on news translation by building systems for two language pairs: English ↔ German and English ↔ Latvian. Our systems are based on an attentional encoder-decoder, using BPE subword segmentation. We identified several common mistakes that our baseline systems seemed to make repeatedly, like not being able to produce sentences that look like news due to a very limited amount in-domain (news) training data, mistranslating named entities, and occasionally producing a translation that is completely unrelated to the source. To counter these problems, we experimented with back-translating monolingual news corpora and filtering out the best translations as additional training data, enforcing named entity translation from a dictionary of parallel named entities, and combining output from multiple NMT systems with SMT. The described methods give 0.7 - 1.8 BLEU point improvements over our baseline systems. This section is based on the paper of Rikters et al. (2017a). The author's contribution to this work is 65%.

4.2.1 Introduction

The NMT systems are based on an attentional encoder-decoder (Bahdanau et al., 2015), using BPE subword segmentation for open-vocabulary translation with a fixed vocabulary (Sennrich et al., 2016). This section is organized as follows: In subsection 4.2.2 we describe our translation software and baseline setups. Subsection 4.2.3 describes our contributions for improving the baseline translations. Results of our experiments are summarized in subsection 4.2.4. Finally, the section is concluded in subsection 4.2.5.

4.2.2 Baseline systems

Our baseline systems were trained with two NMT and one statistical machine translation (SMT) framework. For English ↔ German we only trained NMT systems, for which we used Nematus (NT). For English ↔ Latvian, apart from NT systems, we additionally trained NMT systems with Neural Monkey (NM) (Helcl, 2017) and SMT systems with LetsMT! (LMT) (Vasiljevs et al., 2012).

In all of our NMT experiments we used a shared subword unit vocabulary (Sennrich et al., 2016c) of 35000 tokens. We clipped the gradient norm to 1.0 (Pascanu, 2013) and used a

dropout of 0.2. Our models were trained with Adadelta (Zeiler, 2012) and after 7 days of training we performed early stopping.

For training the NT models we used a maximum sentence length of 50, word embeddings of size 512, and hidden layers of size 1000. For decoding with NT, we used beam search with a beam size of 12.

For training the NM models we used a maximum sentence length of 70, word embeddings and hidden layers of size 600. For decoding with NM, a greedy decoder was used. Unfortunately, at the time when we performed our experiments the beam search decoder for NM was still under development and we could not reliably use it.

4.2.3 Experimental settings

Filtered synthetic training data

Increasing the training data with synthetic back-translated corpora has proven to be useful in previous work (Sennrich, et al., 2016). The method consists of training the initial NMT systems on clean parallel data, then using them to translate monolingual data in the opposite direction and generate a supplementary parallel corpus with synthetic input and human-created output sentences. Nevertheless, more is not always better, as reported by Pinnis et al. (2017), where they stated that using some amount of back-translated data gives an improvement, but using double the amount gives lower results, while still better than not using any at all.

We used each of our NMT systems to back-translate 4.5 million sentences of the monolingual news corpora in each translation direction. First, we removed any translations that contained at least one <*unk*> symbol. We trained an LM using CharRNN with 4 million sentences from the monolingual news corpora of the target languages, resulting in three character-level RNN language models - English, German and Latvian. We used these language models to get perplexity scores for all remaining translations. The translations were then ordered by perplexity and the best (lowest) scoring 50% were used together with the sources as sources and references respectively for the additional filtered synthetic in-domain corpus. We chose scoring sentences with an LM instead of relying on neural network weights because 1) it is fast, reliable and ready to use without having to modify both NMT frameworks, and 2) it is an unbiased approach to score sentences when compared to having the system score its output by itself.

Table 22: Human judgment matches with LM perplexity for filtering on 200 random sentences from the newsdev2017 dataset.

En → De	**De → En**	**En → Lv**	**Lv → En**
55%	56%	58%	56%

To verify that the perplexity score resembles human judgments, we took a small subset of the development sets and asked manual evaluators to rate each translation from 1 to 5. We sorted the translations by manual evaluation scores and automatically obtained perplexities and calculated the overlap between the better halves of each. Results from this manual evaluation in Table 22 show that the LM perplexity score is good enough to separate the worst from the best translations, even though the correlation with human judgments is low.

Table 23: Example sentences translated from Latvian into English that were filtered out from the back-translated news data.

Source	Hypothesis	Perplexity
šodien , 21 : 16	Sheodiennial	70455722055883
lai izdzīvotu , nepieciešams aizpildīt ap 65 % , bet valsts apmaksā 10 %	it is necessary to fill around 65th and the state is paid to the population .	86070783032565
potenciāli zaudētie mūzā gadi ir gadi , kurus cilvēks būtu nodzīvojis līdz kādam noteiktam vecumam , ja nebūtu miris nelaimes gadījumā , kādas slimības vai cita iemesla dēl (līdz 64 gadu vecumam) .	potential annualised annuity is a year that would have survived to a particular old age if it is not dead in an accident or for another reason to be in the age of 64 years old .	73076722556165
tiekoties ar cilvēkiem Latvijā , " veiksmes stāsts " neesot jūtams .	" we are talking about the people of Europe , " he said .	3.0285224517174
liela daļa Latvijas iedzīvotāju ir piederīgi tā saucamajai " krievu pasaulei " , vai vismaz Krievija viņus saredz kā tai piederīgus - tie ir ne tikai Krievijas pilsoņi , bet arī krievvalodīgie , un tie kuriem ir pievilcīgā Krievija un tas vērtības .	a part of the Latvian population is a small and Russian world , or at least Russia sees them as being belonging to them - it is not only Russia ' civil , but also Russian and well known to live in the Russian civil society .	3.0276750775676

Some extreme examples of sentences translated from Latvian into English are listed in Table 23. The first one is just gibberish, the second is English, but makes little sense, the third one demonstrates unusual constructions like *annualised annuity*. The last two examples have a good perplexity score because they seem like good English, but when looking at the source, it is clear that in the fourth example there are some parts that are omitted.

As a result, the filtering approach brought an improvement of 1.1 - 4.9 BLEU on development sets and 1.5 - 2.8 BLEU on test sets when compared to using the full back-translated news corpora.

Named entity forcing

For our experiments with English ↔ German we enforced the translation of named entities (NE) using a dictionary which we built on the training data distributed for WMT 2017.

First, we performed named entity recognition (NER) using spaCy[35] for German and NLTK[36] for English. We only considered NEs of type “person”, “organisation” and “geographic location” for our dictionary. We aligned the recognised entities with GIZA++ (Och and Ney, 2003), using the default parameters and created an entry in our translation dictionary

[35] Industrial-Strength Natural Language Processing in Python - https://spacy.io
[36] Natural Language Toolkit - http://www.nltk.org

for every pair of aligned (multi-word) NEs. Since there was still a lot of noise in the resulting dictionary, we decided to filter it automatically by removing entries that:

- did not contain alphabetical characters

  e.g. filtering out "$^{2}/_{3}$" aligned to "June"

- started with a dash

  e.g. filtering out "-Munich" aligned to "Hamburg"

- were longer than 70 characters or five tokens

  e.g. filtering out "Parliament's Committee on Economic and Monetary Affairs and Industrial Policy" aligned to "EU"

- differed from each other in length by more than 15 characters or two tokens

  e.g. filtering out "Georg" aligned to "Georg von Holtzbrinck"

When translating, we identified all NEs in the source text using the same tools as for the training data, looking up the most likely aligned translations by our systems via the attention matrix for every source NE expression. For every NE, we checked whether there was a translation in our NE dictionary and swapped the identified aligned translation with the one from the dictionary. If it was not in the dictionary, we copied the verbatim NE expression from the source sentence to the target sentence.

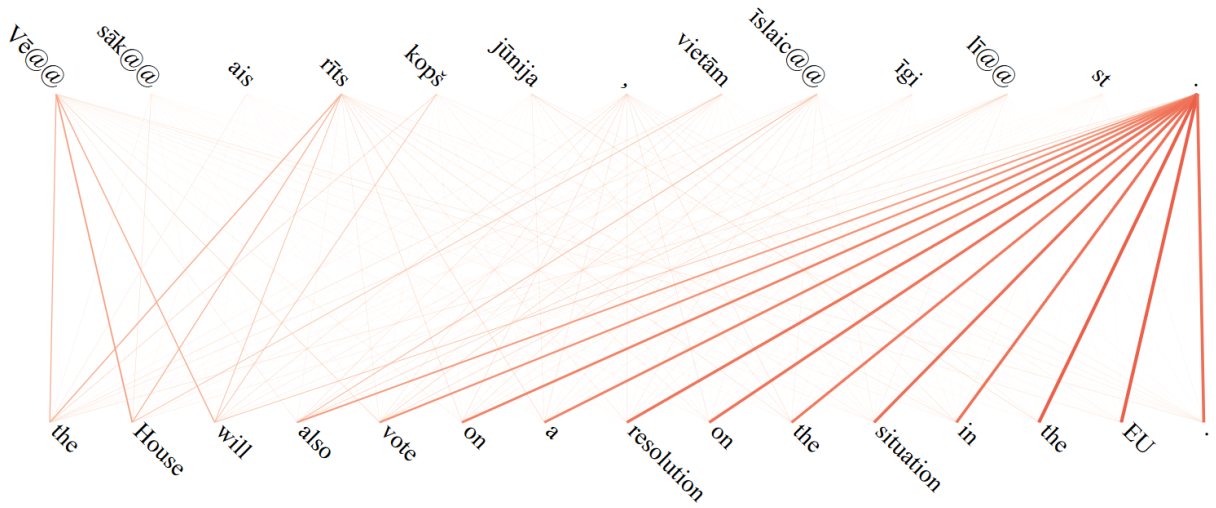


Figure 30: Attention alignment visualization of a translation, in which the strongest alignments are connected with the final token. Reference translation: the coldest morning since June , brief local showers ., hypothesis translation: the House will also vote on a resolution on the situation in the EU .

Hybrid system combination

For translating between English ↔ Latvian we used all 3 systems in each direction and obtained the attention alignments from the NMT systems. For each direction, we chose one main NMT system to provide the final translation for each sentence and, judging by the attention alignment distribution, tried to automatically identify unsuccessful translations. Two

main types of unsuccessful translations that we noticed were when the majority of alignments are connected to only one token (example in Figure 30) or when all tokens strongly align one-to-one, hinting that the source may not have been translated at all (example in Figure 31). In the case of an unsuccessful translation, the hybrid setup checks the attention alignment distribution from the second NMT system and outputs either the sentence of that or performs a final back-off to the SMT output. This approach gave a BLEU score improvement of 0.1 - 0.3.

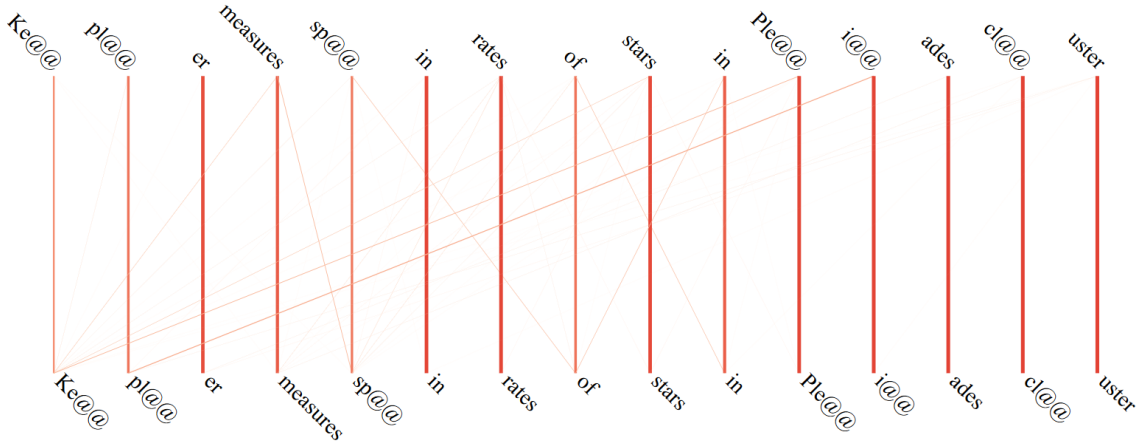


Figure 31: Attention alignment visualization of a translation, in which the all alignments are strong and mainly connected to only one-to-one. Reference translation: Keplers izmēra zvaigžņu griešanās ātrumu Plejādes zvaigznājā ., hypothesis translation: Kepler measures spin rates of stars in Pleiades cluster

Post-processing

In post-processing of translation output, we aimed to fix the most common mistakes that NMT systems tend to make. We used the output attention alignments from the NMT systems to replace any <*unk*> tokens with the source tokens that align to them with the highest weight. Any consecutive repeating n-grams were replaced with a single n-gram. The same was applied to repeating n-grams that have a preposition between them, e.g., *victim of the victim*. This approach gave a BLEU score improvement of 0.1 - 0.2.

Table 24: Experiment results for translating between English ↔ German. Submitted systems are in bold.

System	En → De		De → En	
Dataset	Dev	Test	Dev	Test
Baseline NT	27.4	21.0	31.9	27.2
+Filtered synthetic data	30.7	22.5	36.8	28.8
+NE forcing	30.9	**22.7**	36.9	**29.0**

4.2.4 Results

The results of our English ↔ German systems are summarized in Table 24 and the results of our English ↔ Latvian systems - in Table 25. As mentioned in section 4.2.3 - each implemented modification gives a little improvement in the automated evaluation. Some modifications gave either no improvement for one or both language pairs or lead to lower automated evaluation results. These were either used for only the language pair that did show improvements on the development data or not used at all in the final setup.

Table 25: Experiment results for translating between English ↔ Latvian on development (newsdev2017) and test (newstest2017). Submitted systems are in bold.

System	En → Lv		Lv → En	
Dataset	Dev	Test	Dev	Test
Baseline NM	11.9	11.9	14.6	12.8
Baseline NT	12.2	10.8	13.2	11.6
Baseline LMT	19.8	12.9	24.3	13.4
NM +filtered synthetic data	16.7	13.5	15.7	14.3
NT +filtered synthetic data	16.9	13.6	15.0	13.8
NM+NT+LMT	-	**13.6**	-	**14.3**

Shared task results

Table 26 shows how our systems were ranked in the WMT17 shared news translation task against other submitted primary systems in the constrained track (Bojar et al., 2017b). Since the human evaluation was performed by showing evaluators only the reference translation and not the source, the human evaluation rankings are the same as BLEU, which also considers only the reference translation. One exception is the ranking for En ↔ Lv, where an insufficient amount of evaluations was performed to cover all submitted systems, resulting in a tie for the 1st place across all but one submitted systems.

Table 26: Automatic (BLEU) and human ranking of our submitted systems (C-3MA) at the WMT17 shared news translation task, only considering primary constrained systems. Human rankings are shown by clusters according to Wilcoxon signed-rank test at p-level p<=0.05, and standardized mean DA score (Ave %).

System	Rank		
	BLEU	Human	
		Cluster	Ave %
De → En	6 of 7	6-7 of 7	7 of 7
En → De	10 of 11	9-11 of 11	9 of 11
En → Lv	11 of 12	1-11 of 12	11 of 12
Lv → En	5 of 6	4-5 of 6	4 of 6

4.2.5 Conclusions

In this section, we described our submissions to the WMT17 News Translation shared task. Even though none of our systems were on the top of the list by automated evaluation, each of the implemented methods did give measurable improvements over our baseline systems. To complement the system description, we release open-source software[37] and configuration examples that we used for our systems.

4.3 SYSTEM COMBINATION BY ESTIMATING CONFIDENCE FROM NEURAL NETWORK ATTENTION

Attention distributions of the generated translations are a useful bi-product of attention-based recurrent neural network translation models and can be treated as soft alignments between the input and output tokens. In this work, we use attention distributions as a confidence metric for output translations. We present two strategies of using the attention distributions: filtering out bad translations from a large back-translated corpus and selecting the best translation in a hybrid setup of two different translation systems. While manual evaluation indicated only a weak correlation between our confidence score and human judgments, the use-cases showed improvements of up to 2.22 BLEU points for filtering and 0.99 points for hybrid translation, tested on English → German and English → Latvian translation. This section is based on the paper of Rikters and Fishel (2017). The author's contribution to this work is 70%.

4.3.1 Introduction

The introduction of the attention mechanism (Bahdanau et al., 2015) that enables the model to find parts of a source sentence that are relevant to predicting a target word (pay attention), without the need to form these parts as a hard segment explicitly was one of the ground-breaking innovations in NMT. Decoding sentences with the attention-based model resulted in a useful by-product - soft alignments between tokens of source and target sentences. These can be used for many purposes, such as replacing unknown words with back-off

[37] Scripts for Tartu-Riga-Zurich Neural MT systems for WMT 17 - https://github.com/M4t1ss/C-3MA

translations from a dictionary (Jean et al., 2015) and visualizing the soft alignments (Rikters et al., 2017a).

In this section, we propose using the attention alignments as an indicator of the translation output quality and the confidence of the decoder. We define metrics of confidence that detect and penalize under-translation and over-translation (Tu et al., 2016) as well as input and output tokens with no clear alignment, assuming that all these cases most likely mean that the quality of the translation output is bad.

We apply these attention-based metrics to two use-cases: scoring translations of an NMT system and filtering out the seemingly unsuccessful ones, and comparing translations from two different NMT systems, in order to select the best one.

The structure of this section is as follows: subsection 4.3.2 summarizes related work in back-translating with NMT, machine translation combination approaches and confidence estimation. Subsection 4.3.3 introduces the problem of faulty attention distributions and a way to quantify it as a confidence score. Subsections 4.3.5 and 4.3.6 outline the two use-cases for this score - translation filtering and hybrid selections. Finally, conclusions are summarised in subsection 4.3.7.

4.3.2 Related work

Back-translation of monolingual data

One of the first uses of back-translation of monolingual data as an additional source of training data was reported by (Sennrich et al., 2016a) in their submission for the WMT16 news translation shared task. They translated target-language monolingual corpora into the source language of the respective language pair, and then used the resulting synthetic parallel corpus as additional training data. They performed experiments in ranges from 2 million to 10 million back-translated sentences and reported an increase of 2.2 - 7.7 BLEU for translating between English and Czech, German, Romanian and Russian. The authors also experimented with different amounts of back-translated data and found that adding more data gradually improves performance.

In a later paper Sennrich et al. (2016b) explored other methods of using monolingual data. They experimented with adding a large number of monolingual sentences as targets without any sources to the parallel corpus and compared that to performing back-translation on a part of the monolingual data. While both methods outperform using just parallel data, the back-translated synthetic parallel corpus is a much more powerful addition than the mono data alone.

Pinnis et al. (2017) experimented with using large and even larger amounts of back-translated data and came to a conclusion that any amount is an improvement, but using double the amount gives lower results, while still better than not using any at all. These results hint that it may be possible to get even better results when using only the part of the data selected with some criterion. One of the aims of our work is to provide one such criterion.

Machine translation system combination

Zhou et al. (2017) used attention to combine outputs from NMT and SMT systems. The authors first trained intermediate NMT, SMT and hierarchical SMT systems with one-half of

the training data. Afterwards, they used each system to translate the target side of the other half of the training data. Finally, the three translated parts as source sentence variants alongside the clean target sentence were used for training the combination neural network. This approach gave the network more choices of where to pay attention and which parts should be ignored in the training process. They perform experiments on Chinese → English and report BLEU score improvement by 5.3 points over the best single system and 3.4 points over traditional MT combination methods.

Peter et al. (2016) perform MT system combination in a more traditional manner - using confusion networks. They use 12 different SMT and NMT systems to generate hypothesis translations, align and reorder each hypothesis to match one skeleton hypothesis, creating a confusion network. For the final output is generated by finding the best path in the network. The authors report an improvement of 1.0 BLEU compared to the best single system, translating from English into Romanian.

Translation confidence metrics

Lately the idea of modelling coverage in NMT was introduced, for example, Tu et al. (2016) integrate it directly into the attention mechanism and report improved translation quality as a result. On the simpler side of things, Wu et al. (2016b) perform tests with a baseline attention that uses an additional coverage penalty at decoding time; they report no improvement compared to the common length normalisation. Our metrics are partially motivated by the coverage penalty, though we apply them at the post-translation stage to determine the confidence of the decoder and the quality of the already made translation, which makes it applicable regardless of which software or approach were used.

Another closely related task is quality estimation. The dominating approach there is collecting post-edits and training a machine learning model to predict the quality score or classify translations into usable/not, near-perfect/not, etc. (Bach et al., 2011; Felice and Specia, 2012). The main similarity between our work and quality estimation is their usage of glass-box features (i.e. information about the MT system or the decoder's internal parameters). While our approach does not cover all aspects of quality estimation, it requires no data or training and can be applied to any language and neural machine translation system.

4.3.3 Penalizing attention disorders

Before describing the confidence metrics based on attention weights, here is a brief overview of the NMT architecture where the attention weights come from.

Source of attention

Our work is built around the encoder-decoder machine translation approach (Sutskever et al., 2014; Cho et al., 2014) with an attention mechanism (Bahdanau et al., 2015). In this approach the source tokens are learned to be represented by an encoder, which consists of an embedding layer and a bi-directional LSTM or GRU layer (or 8, Wu et al., 2016b), the outputs of which serve as the learned representation.

There is also a decoder that consists of another layer (or 8, ibid.) of LSTM/GRU cells, with an output layer for predicting the softmax-encoded raw probability distribution of each

output word, one at a time. The state of the decoder layer(s) and thus the output distribution depends on the previous recurrent states, the previously produced output word and a weighted sum of the representations of the source sentence tokens. The weights in this sum are generated for every output word by the attention mechanism, which is a feed-forward neural network with the previous state of the decoder and each input word representation as input and the raw weight of that word for the next state as output. Finally, the attention weights are normalised (13),

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k=0}^{j} \exp(e_{ik})} \tag{13}$$

where e_{ij} is the raw predicted weight and α_{ij} - the final attention weight between the input token *j* and output token *i*.

Once the encoder-decoder network has been trained, it can be used to produce translations by predicting the probability for each next word, which can serve as the basis for sampling, greedy search or beam search (Sennrich et al., 2017). More detail on the attention mechanism is given in the paper by Bahdanau et al. (2015).

Together with the translation, it is also possible to save the attention values between the input tokens and each produced output token. These values can be interpreted as the influence of the input token on the output token, or the strength of the connection between them. Thus, weak or dispersed connections should intuitively indicate a translation with low confidence, while high values and strong connections between one or two tokens on both sides should indicate higher confidence. Next, we present our take at formalizing this intuition.

Measuring attention

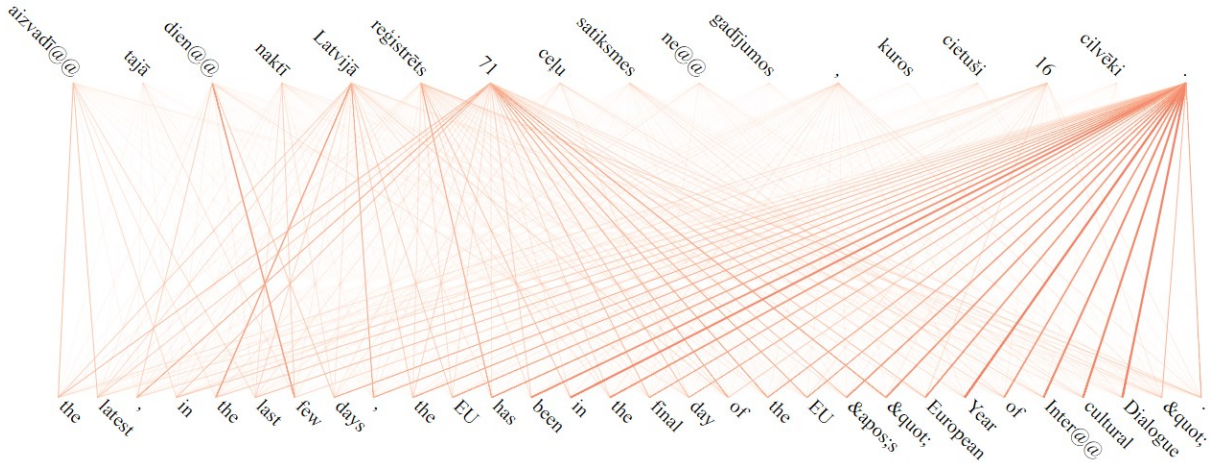


Figure 32: Attention alignment visualization of a bad translation. Reference translation: 71 traffic accidents in which 16 persons were injured have happened in Latvia during the last 24 hours., hypothesis translation: the latest , in the last few days , the EU has been in the final day of the EU 's " European Year of Intercultural Dialogue ". CDP = -0.900, AP_{out} = -2.809, AP_{in} = -2.137, Total = -5.846.

Figure 32 shows an example of a translation that has little or nothing to do with the input, a frequent occurrence in NMT. Besides the text of the translation, it is clear already by looking at the attention weights of this pair that the translation is weak:

- some input tokens (like the sentence-final full-stop) are most strongly connected to several unrelated output tokens, in other words their coverage is too high
- most of the input token attentions as well as some output token attentions are highly dispersed, without one or two clear associations on the counterpart.

On the other hand, a picture like Figure 33 intuitively corresponds to a good translation, with strongly focused alignments. It is this intuition that our metrics formalize: penalizing translations with tokens with a total coverage of not just below but much higher than 1.0, as well as tokens with a dispersed attention distribution.

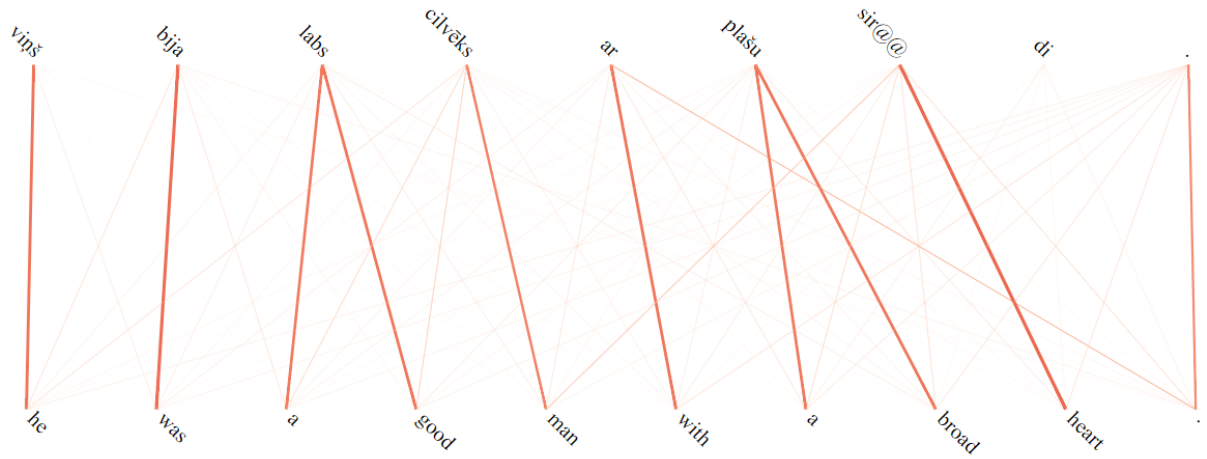


Figure 33: Attention alignment visualization of a good translation. Reference translation: He was a kind spirit with a big heart., hypothesis translation: he was a good man with a broad heart. CDP = -0.099, AP_{out} = -1.077, AP_{in} = -0.847, Total = -2.024.

Coverage deviation penalty

Previous work (Wu et al., 2016b) defines a coverage penalty, which is meant to punish translations for not paying enough attention to input tokens (14),

$$CP = \beta \sum_j \log\left(\min\left(\sum_i \alpha_{ji}, 1.0\right)\right) \quad (14)$$

where i is the output token index, j - the input token index, α – attention probability, β is used to control the influence of the metric and CP - the coverage penalty.

The first part of our metric draws inspiration from the coverage penalty; however, it penalizes not just lacking attention but also too much attention per input token. The aim is to penalize the sum of attentions per input token for going too far from 1.0, so tokens with total attention of 1.0 should get a score of 0.0 on the logarithmic scale, while tokens with less attention (like 0.2) or more attention (like 2.5) should get lower values. We thus define the coverage deviation penalty (15),

$$CDP = \frac{1}{L} \sum_j \log\left(1 + \left(\sum_i \alpha_{ji}\right)^2\right) \quad (15)$$

where L is the length of the input sentence, i is the output token index, j - the input token index, α – attention probability. The metric is on a logarithmic scale, and it is normalised by the length of the input sentence in order to avoid assigning higher scores to shorter sentences. This is not

required for choosing translations of the same sentence by the same system but is required in our experiments described in the next sections. See examples of the CDP metric's values on Figure 32 and Figure 33.

Absentmindedness penalty

However, it is not enough to simply cover the input, we conjecture that more confident output tokens will allocate most of their attention probability mass to one or a small number of input tokens. Thus, the second part of our metric is called the absentmindedness penalty (16) and targets scattered attention per output token, where the dispersion is evaluated via the entropy of the predicted attention distribution. Again, we want the penalty value to be 1.0 for the lowest entropy and head towards 0.0 for higher entropies.

$$AP_{out} = -\frac{1}{L}\sum_i \sum_j \alpha_{ji} \cdot log\alpha_{ji} \quad (16)$$

The values are again on the log-scale and normalised by the source sentence length L (i is the output token index, j - the input token index, α – attention probability).

The absentmindedness penalty can also be applied to the input tokens after normalising the distribution of attention per input token, resulting in the counter-part metric AP_{in}. This is based on the assumption that it is not enough to cover the input token, but rather the input token should be used to produce a small number of outputs. See examples of both metric's values in Figure 32 and Figure 33.

Finally, we combine the coverage deviation penalty with both the input and output absentmindedness penalties into a joint metric via summation (17).

$$confidence = CDP + AP_{out} + AP_{in} \quad (17)$$

Next, we evaluate the metrics directly against human judgments and indirectly by applying them to filtering translations and plugging them into a sentence-level hybrid translation scheme.

4.3.4 Human evaluation

It is clear that the defined metrics only paint a partial picture, since they rely on the attention weights only. For instance, they do not evaluate the lexical correspondence between the source and hypothesis, and more generally, being confident does not mean being right. We wanted to find out how much confidence in our case correlates with translation quality.

To do so we asked human volunteers to perform pairwise ranking of translations from two baseline NMT systems: one done with Nematus and the other - with Neural Monkey. The translations and measurements were done for English-Latvian and Latvian-English, using corpora from the news translation shared task of WMT'2017; further details can be found in section 4.3.5. We selected 200 random sentences for both translation directions and these were given to native Latvian speakers for evaluation. The MT-EQuAl (Girardi et al., 2014) tool was used for the evaluation task. The evaluators were shown one source sentence at a time along with the two different translations. They were instructed to assign one of five categories for each translation: "worst", "bad", "ok", "good" or "best", noting that both may be categorized as

equally "good" or "bad", etc. Differing judgments for the same sentence were averaged. All 200 sentences were annotated by at least one human annotator.

It makes more sense to treat the results as relative comparisons, not absolute scores, as the annotators only see two translations at a time. We use these comparisons to compute the Kendall rank correlation coefficient (Kendall, 1938) by only looking at the pairs where human scores differ. Since we only have comparisons for each pair and not between different sentences, the coefficient is computed as

$$\tau = \frac{pos-neg}{pos+neg} \tag{18}$$

where *pos* is the number of pairs where the metric agrees with the human judgment and *neg* is the number of pairs where they disagree.

The results are presented in Table 27, and as we can see they indicate weak correlation, with the absolute values of τ between 0.012 and 0.200.

Table 27: The Kendall's Tau correlation between human judgments and the confidence scores.

Language pair	CDP	AP_{in}	AP_{out}	Overall
En-Lv	0.099	0.074	0.123	0.086
Lv-En	-0.012	-0.153	-0.200	-0.153

Let us look closer at where the metrics disagree with human judgments. Figure 34 shows an example of a translation which was rated highly by human annotators but poorly with our metrics. While the sentence is a good translation, it does not follow the source word-by-word. Some subword units and functional words do not have a clear alignment, even though they are understood/generated correctly. This means that one problem with our metrics is that they might be over-penalizing translations that deviate from a direct literal translation.

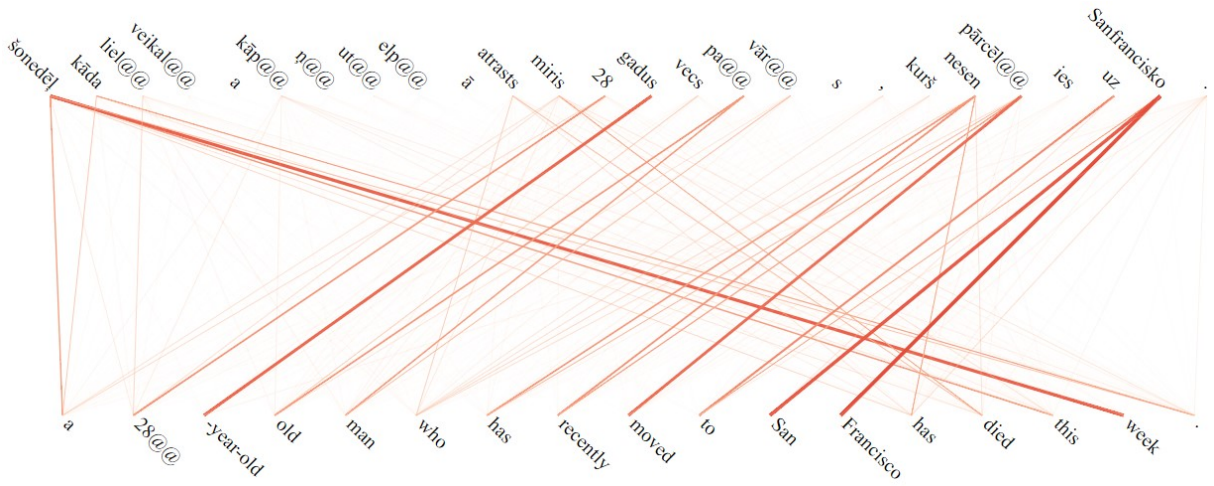


Figure 34: Attention alignment visualization of a bad translation. Reference translation: a 28-year-old chef who had recently moved to San Francisco was found dead in the stairwell of a local mall this week ., hypothesis translation: a 28-year-old old man who has recently moved

to San Francisco has died this week ., CDP = -0.250, AP_{out} = -1.740, AP_{in} = -1.46, Total = -3.45.

Next, we continue with the experiments of using our metrics to filter synthetic data and to select translations in a hybrid MT scenario.

4.3.5 Filtering back-translated data

Baseline systems and data

Our baseline systems were trained with two NMT frameworks - Nematus (NT) and Neural Monkey (NM). For all NMT models we used a shared subword unit vocabulary of 35000 tokens, clip the gradient norm to 1.0, dropout of 0.2, trained the models with Adadelta and performed early stopping after 7 days of training. For models with each NMT framework we used the default settings as mentioned in the frameworks documentation:

- For NT models, we used a maximum sentence length of 50, word embeddings of size 512, and hidden layers of size 1000. For decoding with NT, we used beam search with a beam size of 12.
- For NM models, we used a maximum sentence length of 70, word embeddings and hidden layers of size 600. For decoding with NM, a greedy decoder was used.

Training, development and test data for all systems in both language pairs and translation directions were used from the WMT17 news translation task[38]. For the baseline systems, we used all available parallel data, which is 5.8 million sentences for En ↔ De and 4.5 million sentences for En ↔ Lv.

Back-translating and filtering

We used our baseline En → Lv and Lv → En NM and NT systems to translate all available Latvian monolingual news domain data - 6.3 million sentences in total from *News Crawl: articles from 2014, 2015, 2016*, and the first 6 million sentences from the English *News Crawl 2016*. Much more monolingual data was available from other domains aside from news. Since the development and test data was of the news domain, we only used that, considering it as in-domain data for our systems.

For each translation, we used the attention provided from the NMT system to calculate our confidence score, sorted all translations according to the score and selected the top half of the translations along with the corresponding source sentences as the synthetic parallel corpus. We used only the full confidence score (combination of *CDP*, AP_{out} and AP_{in}) for filtering instead of each individual score due to its smoother overall correlation with human judgments. In between, we also removed any translation that contained any <*unk*> tokens.

To compare attention-based filtering with a different filtering method, we trained a CharRNN LM with 4 million sentences from news domain for each of the target languages. We used these LMs to get perplexity scores for all translations, order them and get the *better half*.

[38] EMNLP 2017 Second Conference on Machine Translation - http://www.statmt.org/wmt17

Table 28 summarizes how much human evaluation overlaps with each of the filtering methods. The final row indicates how much both filtering methods overlap with each other. While results from either approach don't look overly convincing, the LM-based approach has been proven to correlate with human judgments close to the BLEU score and is a good evaluation method for MT without reference translations (Gamon et al., 2005). Therefore, the attention-based approach that does not require training of an additional model and overlaps with human judgments to approximately the same level should be more desirable.

Table 28: Human judgment overlap results on 200 random sentences from the newsdev2017

Filtering method	**En → Lv**	**Lv → En**
LM-based overlap with human	58%	56%
Attention-based overlap with human	52%	60%
LM-based overlap with Attention-based	34%	22%

NMT with filtered synthetic data

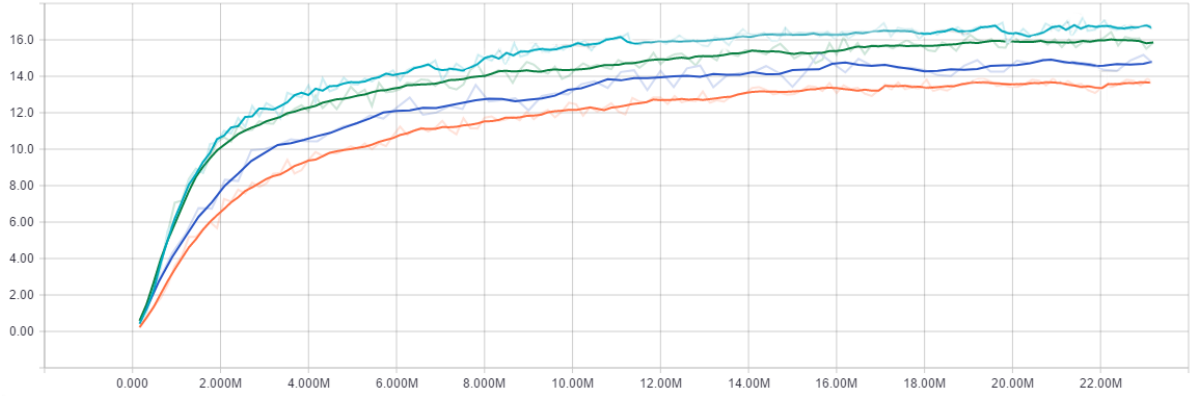


Figure 35: Automatic evaluation progression of Lv → En experiments on validation data. Orange - baseline; dark blue - with full back-translated data; green - with LM-filtered back-translated data; light blue - with attention-filtered back-translated data.

We shuffled each synthetic parallel corpus with the baseline parallel corpora and used them to train NMT systems. In addition to the baseline and two types of filtered BT synthetic data, we also trained a system with the full BT data for each translation direction. Figure 35 shows a combined training progress chart for Lv → En on the full *newsdev2017* dataset that was used as the development set for training. Here the differences between all four approaches are clearly visible. Further results on a subset of *newsdev2017* and the full *newstest2017* dataset are summarized in Table 29. While for Lv → En and En ↔ De the attention-based approach is the clear leader, for En → Lv it falls behind the LM filtered version. As expected, adding BT synthetic training data allows to get higher BLEU scores in all cases. It can be observed that filtering out half of the badly translated data and keeping only the best translations either does not decrease the final output quality in some cases or even further increase the quality in others, when using the LM. With filtering by attention, the results are more inconsistent - even higher in one direction while deterioration in the other. A reason for this could be that for Lv → En attention-based filtering the similarity with human judgments was higher than for En → Lv (Table 28), and it was also more different from the LM-based one. While for the other direction it is the other way around.

Table 29: Experiment results in BLEU for translating between English ↔ Latvian with different types of back-translated data using development (200 random sentences from *newsdev2017*) and test (*newstest2017*) datasets

	BLEU			
System	**En → Lv**		**Lv → En**	
Dataset	**Dev**	**Test**	**Dev**	**Test**
Baseline NM	8.36	11.90	8.64	12.40
NM + Full Synthetic	9.42	13.50	9.01	13.81
NM + LM-Filtered Synthetic	9.75	13.52	9.45	14.30
NM + Attention-Filtered Synthetic	8.99	12.76	11.23	14.83

4.3.6 Attention-based hybrid decisions

We translated the development set with both baseline systems for each language pair in each direction. The hybrid selection of the best translation was performed similarly to filtering, where we discarded the worst-scoring half of the translations. In the hybrid selection, we used the same score to compare both translations of a source sentence and choose the better one. Results of the hybrid selection experiments are summarized in Table 30. For translating between En ↔ Lv, where the difference between the baseline systems is not that high (0.06 and 1.55 BLEU), the hybrid method achieves some meaningful improvements. However, for En ↔ De, where differences between the baseline systems are bigger (3.46 and 4.46 BLEU), the hybrid drags both scores down.

Table 30: Hybrid selection experiment results in BLEU on the development dataset (200 random sentences from newsdev2017)

System	**En → De**	**De → En**	**En → Lv**	**Lv → En**
Neural Monkey	18.89	26.07	13.74	11.09
Nematus	22.35	30.53	13.80	12.64
Hybrid	20.19	27.06	14.79	12.65
Human	23.86	34.26	15.12	13.24

The last row of the results in Table 30 shows BLEU scores for the scenario when human annotator preferences were used to select each output sentence. An overview of human evaluator preferred translation selections is visible in Table 31. The results show that out of all translations the human evaluators deliberately prefer one or the other system. Aside from En - Lv, where a slight tendency towards Neural Monkey translations can be observed, all others look more or less equal. This highly contrasts with the BLEU scores from Table 30, where in both translation directions from English human evaluators prefer the lower-scoring system more often than the higher-scoring one. The final row of Table 31 shows how much our attention-based score matches the human judgments in selecting the best translation.

Table 31: Human evaluation results on 200 random sentences from the newsdev2017 dataset

System	**En → De**	**De → En**	**En → Lv**	**Lv → En**
Neural Monkey	54%	42%	61.5%	47%
Nematus	46%	58%	38.5%	53%
Overlaps with hybrid selection	57%	47%	62.5%	51%

4.3.7 Conclusions

In this section, we described how attentional data from neural machine translation systems can be useful for more than just visualizations or replacing specific tokens in the output. We introduced an attention-based confidence score that can be used for judging NMT output. Two applications of using attentional data were investigated and compared to similar approaches. We used a smaller dataset to perform manual evaluation and compared that to all automatically obtained results. Our experiments showed interesting results and some increases in automated evaluation, as well as a good correlation with human judgments.

In addition to the methods described in this section, we released open-source scripts[39] for (1) scoring, ordering and filtering NMT translations, (2) performing hybrid selections between two different NMT outputs of the same source, and (3) software for inspecting attention alignments[40] that the NMT systems produce in the translation process (used for Figure 32, Figure 33 and Figure 34). We also provide all development subsets that we used for manual evaluation with anonymized human annotations.

4.4 DATA COMBINATION FOR TRAINING MULTILINGUAL NEURAL MACHINE TRANSLATION SYSTEMS

This section presents results of employing multilingual and multi-way neural machine translation approaches for morphologically rich languages, such as Estonian and Russian. We experiment with different NMT architectures that allow achieving state-of-the-art translation quality and compare the multi-way model performance to one-way model performance. We report improvements of up to +3.27 BLEU points over our baseline results, when using a multi-way model trained using the transformer network architecture. We also provide open-source scripts used for shuffling and combining multiple parallel datasets for training of the multilingual systems. This section is based on the publications of Rikters et al. (2018a) and Rikters et al. (2018b). The author's contribution to this work is 80%.

4.4.1 Introduction

One of the major advantages of neural machine translation (NMT) is that unlike statistical machine translation (SMT), which was the previous industry standard (and is still actively used in commercial applications), NMT is trained and used jointly as a single end-to-end system without the need to optimize multiple independent models and relations between the models. However, training NMT systems for individual language pairs has shown to take significantly more time (e.g., two to three weeks or up to a week with newer platforms, such as Marian (Junczys-Dowmunt et al., 2016) or Google's Tensor2Tensor toolkit[41] than training of SMT systems (e.g., less than a day or up to several days for large systems). But even with this advantage, using the traditional approaches, one would still need to train a separate model for each translation direction. Since running a high amount of GPU-intensive NMT models in a production environment can quickly sum up to an enormous resource-usage cost, it has been

[39] Confidence Through Attention - https://github.com/M4t1ss/ConfidenceThroughAttention
[40] NMT Attention Alignment Visualizations - https://github.com/M4t1ss/SoftAlignments
[41] T2T: Tensor2Tensor Transformers - https://github.com/tensorflow/tensor2tensor

natural (as shown by related work in subsection 4.4.2) to look for solutions that allow compressing the models into an even more dense end-to-end solution that is able to handle multiple languages and language pairs simultaneously.

Another benefit of a single model for multiple translation directions could be the ability to learn not just from the training data of the language pair in question, but also from language pairs that include one of the languages. The advantages of learning from multiple translation directions at the same time can be (1) the ability for a model to learn how to translate language specific attributes that are common to multiple languages at the same time, and (2) to learn and generalize translations that may not occur in the parallel corpus of, e.g., A↔B, but do occur in parallel corpora of, e.g., A↔C and C↔B and therefore are deductible.

The structure of this section is as follows: subsection 4.4.2 summarizes related work in multilingual and multi-way NMT; subsection 4.4.3 introduces the setup of our experimental environment and subsection 4.4.4 - the data used; subsection 4.4.5 outlines the main results in translation quality as well as speed and resource usage, and in subsection 4.4.6 we look at several examples how translations produced by one-way systems differ from multi-way system translations.

4.4.2 Related work

Multilingual NMT has recently been investigated by several research groups. For instance, Firat et al. (2016) modify the current state-of-the-art attentional NMT approach by supplementing it with the ability to learn from multiple language pairs and multiple translation directions at the same time. They are able achieve this by creating a shared attention mechanism across the involved resources. The authors report improvements in translation quality over most individual baselines, using a single multilingual model trained on five language pairs in both directions. The authors especially highlight that by combining data from language pairs with many resources with data from a low-resource language pair, the quality gains for the low-resource language pair are higher.

Johnson et al. (2016) introduce a simple method for training a single-model multilingual NMT system, which does not require any modifications to the architecture of the system. They achieve this by adding a target language identifying token in the beginning of each source sentence of the training data. While they only report comparable and not outperforming results for models trained on high-resource language pairs, the biggest improvements are achieved in low-resource and even zero-shot translation. An interesting aspect of this approach is that, when trained on many translation directions at once, the same input sentence can be translated into any supported target language by changing only the target language identifying token.

Ha et al. (2016) use a similar approach to Johnson et al. (2016) by only modifying training data and using the same NMT system architecture. The main difference is that they add a language identifying token to each subword unit and apply this pre-processing to both - source and target sentences of the training data. Another difference is that they don't use particularly deep network architectures in their experiments. The authors describe two experiment scenarios where they train systems to translate from multiple source languages into one target language by (1) adding an additional parallel corpus and (2) adding a monolingual corpus as the additional source and target data. The achieved improvements reach up to 2.6 BLEU points for the first approach and up to 3.15 BLEU points for the second approach.

4.4.3 Experiment Setup

In our experiments, we mainly followed the path of Johnson et al. (2016) by not making any modifications to the network architecture and modifying only the data during training and inference. We did, however, experiment with different encoder and decoder cell types and add slight modifications to the data iterator module for it to automatically read the multilingual multi-way training data in equal batches for each translation direction and prepend the target language symbol at the beginning of each source sentence.

Our recurrent neural network NMT systems were trained with Nematus (Sennrich et al., 2017) using four main configurations. For training of the NMT systems with convolutional neural networks and transformer networks, we used Sockeye (Hieber et al., 2017). All SMT systems were trained using the Moses (Koehn et al., 2007) toolkit in the Tilde MT platform (Vasiļjevs et al., 2012). The details of the models are as follows:

- Recurrent neural network models
    - Maximum sentence length of 50;
    - Multiplicative long short-term memory (Krause et al., 2017) (MLSTM) shallow one-way (MLSTM-SU - the baseline model)
        - Encoder and decoder cell type – MLSTM (same as used by Pinnis et al. (2017));
        - A shared subword unit vocabulary (Sennrich et al., 2016) of 25,000 tokens;
    - Gated recurrent units (GRU)
        - Encoder and decoder cell type – GRU;
        - Shallow multilingual multi-way (GRU-SM)
            - 1-layer encoder and 1-layer decoder;
        - Deep - one-way (GRU-DU) and multilingual multi-way (GRU-DM)
            - 4-layer encoder and 4-layer decoder;
            - 2 GRU transition operations applied in the encoder layer; 4 GRU transition operations applied in the decoder layer; 2 GRU transition operations applied in decoder layers after the first layer;
            - Additional incremental training (Freitag and Al-Onaizan, 2016) after convergence of the GRU-DM model, using only parallel training and development data of a single translation direction;
- Fully convolutional neural network models - one-way (FConv-U) and multilingual multi-way (FConv-M)
    - Encoder and decoder cell type - convolutional neural network (CNN);
    - 15-layer encoder and 15-layer decoder;
    - Maximum sentence length of 128;
- Transformer neural network models - one-way (Transformer-U) and multilingual multi-way (Transformer-M)
    - Encoder and decoder cell type - transformer;
    - Maximum sentence length of 128;
    - 6-layer encoder with convolutional embeddings;
    - 6-layer transformer decoder;
    - Each block (self-attention or feed-forward network) is
        - Pre-processed with layer normalization;

    - Post-processed with dropout and a residual connection;
- SMT one-way models (SMT)
    - Word alignment performed using fast-align (Dyer et al., 2013);
    - 7-gram translation models and "*wbe-msd-bidirectional-fe-allff*" reordering models;
    - Language model trained with KenLM (Heafield, 2011);
    - Tuned using the improved MERT (Bertoldi et al., 2010).

Common parameters for all multilingual multi-way experiments:

- Multilingual training data was shuffled in equal batches per translation direction and with the target language identifier added before each sentence as described by Johnson et al. (2016).
- A shared subword unit vocabulary of 50 000 tokens was used.

For all one-way experiments we used a smaller shared subword unit vocabulary of 24 500 tokens.

All other parameters for the models were identical – we clip the gradient norm to 1.0 (Pascanu et al., 2013), use a dropout of 0.2 and trained the models with Adadelta (Zeiler, 2012). We used a word embedding of size of 500, and hidden layers of size 1024. All models were trained until they reached convergence on validation data.

4.4.4 Data

For training, we used English ↔ Russian, English ↔ Estonian, and Russian ↔ Estonian data. The one-way models were trained on English ↔ Estonian and Russian ↔ Estonian data while the multilingual multi-way models were trained on data from all three language pairs in both directions. The training corpora consist of multiple publicly available and proprietary datasets. Among the public datasets, the largest were the MultiUN (Chen and Eisele, 2012), DGT-TM (Steinberger et al., 2012), Open Subtitles (Tiedemann, 2009), Tilde MODEL (Rozis and Skadiņš, 2017).

The corpora were cleaned and filtered in order to reduce noise in the parallel training data. During filtering, we removed non-parallel sentence pairs, sentences with sentence splitting errors, and duplicate entries.

Data processing was performed in two steps - first, a low content overlap filter, which is based on the cross-lingual alignment tool *MPAligner* (Pinnis, 2013), was applied, followed by the standard data processing pipeline of the Tilde MT platform. For some corpora, the filtering resulted in an overall reduction of more than 50% of the original size. Corpora with content overlap below a certain threshold were manually examined and left out from the final dataset. The data filtering procedure is described in greater detail in the paper by Pinnis et al. (2017). An overview of the training data statistics before and after filtering for each language pair is given in Table 32.

Table 32: Training data sentence counts before and after filtering.

Language pair	**Before filtering (Total/Unique)**	**After filtering (Unique)**
English ↔ Estonian	62.5M / 24.3M	18.9M
English ↔ Russian	60.7M / 39.2M	29.4M
Russian ↔ Estonian	6.5M / 4.4M	3.5M

For Estonian ↔ Russian, we selected 2000 random sentences from the training data to be used as validation data. The validation datasets for all other translation directions were obtained from the ACCURAT development datasets (Skadiņa et al., 2010). In the multilingual multi-way model training scenarios, we concatenated $\frac{1}{6}^{\text{th}}$ of each 2000 sentence validation dataset, resulting in batches of 333 sentences from each translation direction, which we used as development data. As for evaluation data – we used the ACCURAT balanced evaluation corpus consisting of 512 sentences in each translation direction, for which the Russian version was prepared by in-house translators.

4.4.5 Results

In this section, we describe the results of our experiments. We evaluate MT system translation quality using BLEU (Papineni et al., 2002). We also analyse translation speed and GPU memory usage during translation, as well as training duration. While training models for multiple translation directions, we were mainly focused on improving the translation quality when translating between Russian and Estonian, because this specific language pair had the poorest performance among the baseline systems.

Translation Quality

Table 33 shows how each of the models that we described in the previous section compares to the baseline in terms of development and evaluation data translation quality.

When we compare the baseline one-way model (MLSTM-SU) to the other one-way models, the results show that the GRU-DU and FConv-U models reach lower translation quality on all development sets and all but one (for FConv-U) or two (for GRU-DU) evaluation sets. The GRU-DU model insignificantly out-performs the baseline model on the Estonian → Russian evaluation set (by 0.04 BLEU points) and the Estonian → English evaluation set (by 0.08 BLEU points). The FConv-U model shows slightly higher results (by 0.18 BLEU points) on the Estonian → English evaluation set. However, the results of the Transformer-U model are interesting. Although it got lower results on the Estonian ↔ Russian evaluation sets (by -1.15 and -2.01 BLEU points), it outperformed the baseline model on the Estonian ↔ Russian evaluation sets (by 2.29 and 3.3 BLEU points). A potential explanation of these results is that the Transformer-U model becomes more advantageous than the MLSTM-SU model when using larger datasets, however, for smaller datasets the MLSTM-SU model is still able to achieve state-of-the-art results.

Table 33: Translation quality results for all model architectures on development and evaluation data. The best results are in bold.

	Development				**Test**			
	Ru → Et	Et → Ru	En → Et	Et → En	Ru → Et	Et → Ru	En → Et	Et → En
SMT	27.74	25.48	17.99	25.89	9.88	7.27	21.44	29.69
MLSTM-SU	17.51	18.46	23.79	34.45	11.11	12.32	26.14	36.78
GRU-SM	13.7	13.71	17.95	27.84	10.66	11.17	19.22	27.85
GRU-DU	17.03	17.42	23.53	33.63	10.33	12.36	25.25	36.86
GRU-DM	17.07	17.93	23.37	33.52	13.75	14.57	25.76	36.93
FConv-U	15.24	16.17	21.63	33.84	7.56	8.83	24.87	36.96

FConv-M	14.92	15.80	18.99	30.25	10.65	10.99	21.65	31.79
Transf.-U	17.44	18.90	**25.27**	**37.12**	9.10	11.17	**28.43**	**40.08**
Transf.-M	**18.03**	**19.18**	23.99	35.15	**14.38**	**15.48**	25.56	37.97

Figure 36: Training progress for the deep multilingual multi-way model (GRU-DM).

Next, we look at whether the multi-way models allow increasing translation quality over one-way models. The results show that the GRU multi-way model outperforms the one-way models for all language pairs on all datasets. However, the convolutional and transformer models increase quality only for the low-resource language pairs. The quality improvement for the Estonian ↔ Russian language pairs ranges from 2.16 BLEU points (for the FConv-M model on the Estonian → Russian evaluation set) up to 5.28 BLEU points (for the Transformer-M model on the Russian → Estonian evaluation set). For the high-resource language pairs, on the other hand, both FConv-M and Transformer-M models show significantly lower translation quality than their respective one-way models. The quality decrease ranges from -2.11 BLEU points (for the Transformer-M model on the Estonian → English evaluation set) down to -5.17 BLEU points (for the FConv-M model on the Estonian → English evaluation set). This shows that the newer NMT architectures in multi-way scenarios are beneficial only to low-resource language pairs.

Finally, if we look at which models achieved the highest overall results on evaluation sets, it is evident that the transformer models performed the best. For the low-resource language pairs, the best results were achieved by the multi-way model. However, for the high-resource language pairs, the best results were achieved by the respective one-way models.

The reason why the results of the SMT system on the development set for Estonian ↔ Russian (underlined) are so much higher than for all other models may be due to the characteristic of SMT systems being good at memorizing similar sentences to what they have

already seen during training. As stated in the previous section, this was the only language pair for which the development dataset was derived from the training dataset. For all other language pairs, we used a separate dataset.

When the GRU-DM model had converged, we performed additional incremental training for two language pairs in both ways (English ↔ Estonian and Russian ↔ Estonian). Figure 36 illustrates the training progress of this model and the four individual incrementally trained models. The idea of the incremental training was to adapt the system to a specific domain, which in this case would be translation into a single language. Incremental training improved the translation quality of the multi-way GRU-DM model for the individual language pairs by up to 0.60 BLEU points.

Figure 37 shows the training progress for multiple variations of Russian ↔ Estonian models. The deep one-way models (Estonian ↔ Russian GRU-DU) reached the early stopping criterion very quickly but did not get as high as the other models over more time. The other RNN-based models converged after observing approximately 142 million sentences during training. The transformer models stand out the most by being the very first to stop training, as well as reaching the highest BLEU scores the quickest.

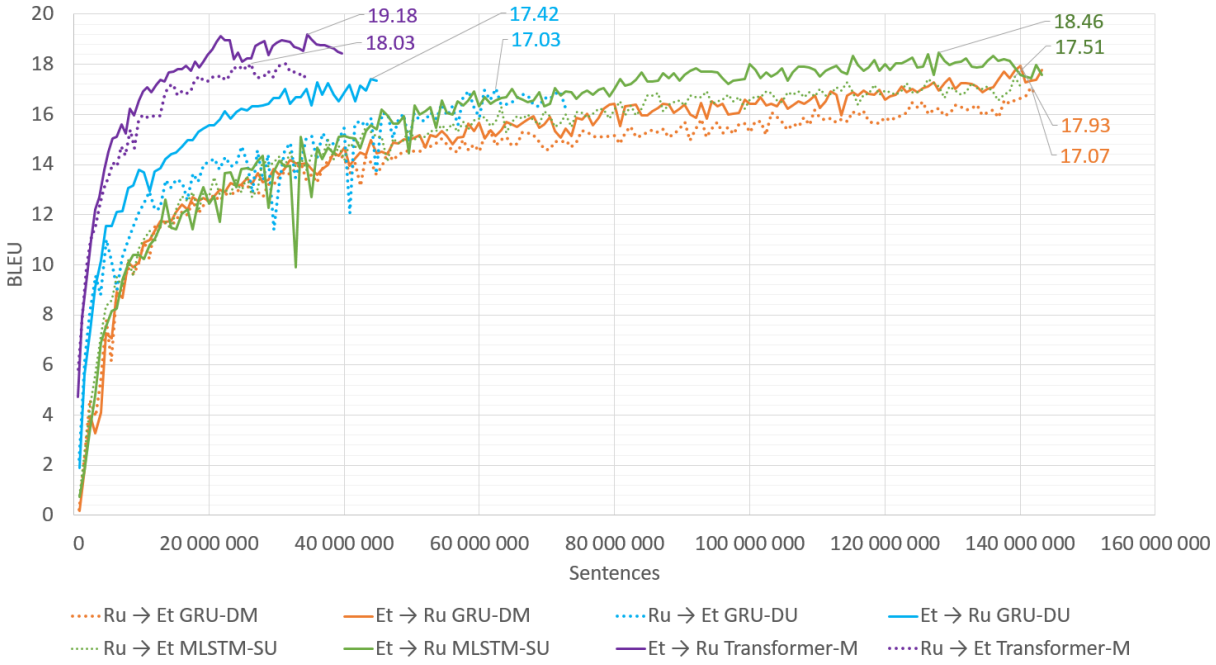


Figure 37: Training progress for Russian ↔ Estonian systems.

Table 34: Resource usage for all NMT model architectures during translation. The most efficient values are in bold. The final column shows the training time until the system converges.

	Seconds		Sentences per second	GPU RAM, MB	Train time, days
	Translation	Per sentence			
Theano-based Nematus					
MLSTM-SM	274.57	0.54	1.86	651	16.4
GRU-SM	211.51	0.41	2.42	**611**	8.5
GRU-DM	460.07	0.9	1.11	979	36.6
MXNet-based Sockeye					
FConv-M	177.19	**0.35**	**2.89**	971	4.5
Transformer-M	191.05	0.37	2.68	1391	**3.8**

Resource Usage During Translation

Training models with deeper architectures increases resource usage in both - training time and required computational power. The higher resource usage is present during translation as well. Table 34 shows a comparison of time and GPU RAM consumption when translating the evaluation dataset using the NMT systems with several architectures from our experiments. In the table, we isolate models trained with Nematus from models trained with Sockeye, as they are based on different deep learning frameworks, respectively, Theano (Theano Development Team, 2016) and MXNet (Chen et al., 2015).

The highest-scoring Transformer models are the quickest to train and also nearly the fastest during translation, but they consume more than twice the amount of GPU memory during translation. The GRU-DM model, which was the runner-up model for translating Estonian ↔

Russian uses 30% less GPU memory during translation, but takes 2.4 times longer to complete the job, and training also took ~50% longer.

All tests were performed on a machine with an NVIDIA Titan X (Pascal) GPU, Intel Core i7-6850K CPU @ 3.60GHz, 64GB of RAM, and 1TB SSD. We only used a single GPU for training and translating, even though the frameworks have support for multi-GPU training and translation.

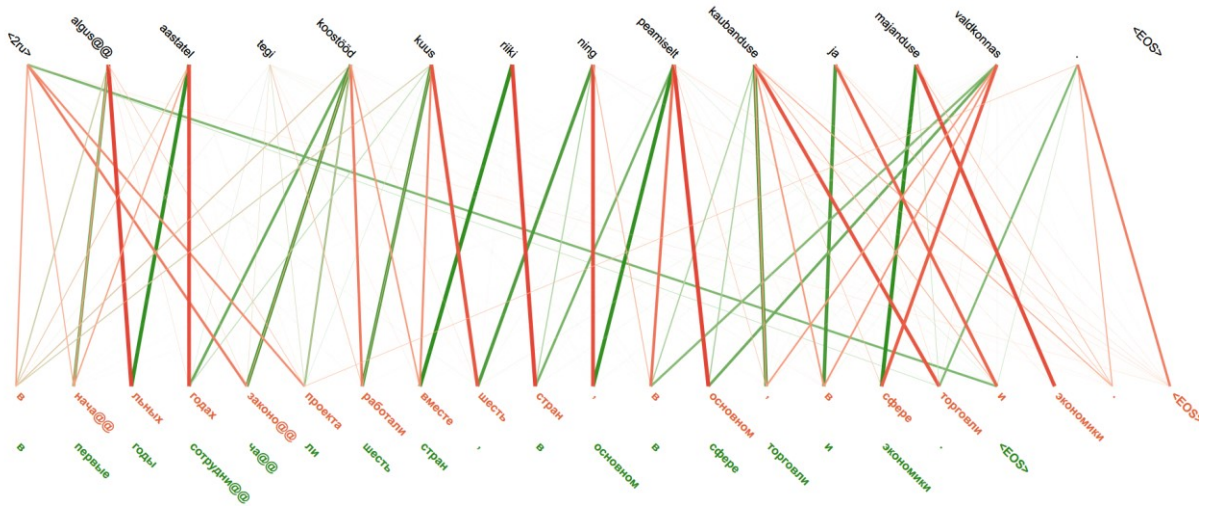
It is worth mentioning that while training all shallow RNN models – multi-way or one-way – the training time for a single model to converge did not change noticeably. The same can be said about CNN and Transformer models. In the case of deep RNN models, training time increased by about 2-3 times, reaching 3-4 weeks on a single GPU.

4.4.6 Translation Examples

Source:	Üle poole rahvastikust kasutab Internetti regulaarselt.
Transformer-U:	более половины населения практикуют работу с Интернетом.
(transl. into English):	More than half of the population practice working with the Internet.
Transformer-M:	более половины населения регулярно использует Интернет.
(transl. into English):	More than half of the population regularly uses the Internet.
Reference:	более половины жителей регулярно пользуются интернетом.
English Reference:	More than half the population are regular internet users.

Figure 38: Translation examples comparing the highest-scoring system (multi-way transformer) with its one-way counterpart. BLEU score of both - **15.62**.

In this section, we show three examples where we compare sentences from one-way and multi-way architectures (e.g. the deep GRU models or transformer models).



Source:	Algusaastatel tegi koostööd kuus riiki ning peamiselt kaubanduse ja majanduse valdkonnas.
GRU-DU:	в начальных годах законопроекта работали вместе шесть стран , в основном , в сфере торговли и экономики.
(transl. into English):	In the initial years of the bill project, six countries worked together, mainly in the sphere of trade and economy.
GRU-DM:	в первые годы сотрудничали шесть стран , в основном в сфере торговли и экономики.
(transl. into English):	In the first years, six countries cooperated, mainly in the sphere of trade and economy.
Reference:	в первый год сотрудничество вели шесть стран , в основном в сфере торговли и экономики.
English Reference:	In the early years , the cooperation was between six countries and mainly about trade and the economy.

Figure 39: Translation examples comparing the second highest-scoring system (deep multi-way GRU) with its one-way counterpart. BLEU scores - **47.63** (GRU-DU - orange alignments) and **67.04** (GRU-DM - green alignments).

Source:	Charles tõusis ja vaatas aknast välja.
Transformer-U:	Шарль встал и посмотрел в окно.
(transl. into English):	Charles stood up and looked out the window.
Transformer-M:	Шарль встал и оглянулся в окно.
(transl. into English):	Charles stood up and looked out the window.
Reference:	Чарльз поднялся и посмотрел в окно.
English Reference:	Charles rose and looked out of the window.

Figure 40: Translation examples comparing the highest-scoring system (multi-way transformer) with its one-way counterpart. BLEU scores - **61.48** (Transformer-U) and **26.27** (Transformer-M).

In Figure 38, we compare one of the poorest-scoring translations generated with both the overall highest-scoring multi-way system (Transformer-M) and its one-way counterpart. The BLEU score of both translations is identical, but while the translation of Transformer-M is almost perfect (with fluency issues in the last two words), the translation of Transformer-U features a more significant lexical choice mistake.

I.e., the words "kasutab}" (uses) and "regulaarselt}", which are correctly translated by the multi-way model as "использует}" (uses) and "регулярно}" (regularly), are mistranslated by the one-way model as "практикуют}" (practice) and "работу}" (work).

Figure 39 shows a comparison of a sentence that had one of the highest BLEU scores out of all GRU-DU translations compared with the same sentence translated using GRU-DM. There is a redundant word ("законопроекта}" - bill project or draft law) in the translation of the one-way model, which is not present in the source. It is also evident in the attention alignments (visualised using the toolkit by Rikters et al. (2017a)) that the sub-word units of this word are strongly aligned only to the target language tag at the beginning of the source sentence. This may mean that these are not translations of any specific sub-word units of the source sentence. The translation of the multi-way model does not exhibit such a problem in this example.

In Figure 40, we show the third example. Here the translation from the one-way transformer model scores higher according to BLEU than the multi-way model. The only difference between these two translations is how the Estonian word "vaatas}" (looked) is translated. The Transformer-U model produced the translation "посмотрел}" (looked), which matches the reference translation, but the Tranformer-M model produced the translation "оглянулся}" (looked back), which is the wrong lexical choice in the given context.

4.4.7 Conclusions

In this section, we described a wide range of experiments on training and evaluating multilingual and multi-way neural machine translation systems. Our results show that for low-resource language pairs, such as Estonian ↔ Russian, we can achieve a significant improvement in translation quality by adding data from other languages over using only one-way parallel data. Multi-way NMT systems in both directions improved translation quality (by 3.09 - 5.28 BLEU points for Russian → Estonian and 2.16 - 4.31 BLEU points for Estonian → Russian) for all three model architectures (deep GRU, convolutional, and transformer), for which we performed multi-way experiments. Our experiments also show that the largest improvements in BLEU scores, as well as the highest overall BLEU scores in the low-resource multi-way scenario were achieved by training systems with the Transformer model.

While the multilingual approach helped gaining improvements for the low-resource language pair, it did degrade the performance for the high-resource language pairs by several BLEU points. In almost all of our experiments the multilingual models showed a drop-in translation quality by 2.87 - 3.22 BLEU points for English → Estonian and 2.11 - 5.17 BLEU points for Estonian → English. However, the results showed that the most stable architecture for multi-way model training was the deep GRU model architecture. It showed improvements for both low-resource and high-resource language pairs on both development and evaluation datasets.

The results also showed that when training one-way systems for the low-resource language pairs, the newer convolutional and self-attention (i.e., transformer) models under-performed. The best results in these experiments were achieved by the MLSTM-based models (outperforming the convolutional models by up to 3.55 BLEU points and the transformer model by 2.01 BLEU points).

While manually analysing the evaluation sets, we noticed that there were several sentences translated perfectly by Transformer-M, but much worse by GRU-DM and vice versa. This suggests that further investigation may be required to find out whether a combination of the systems can lead to translations of even higher quality. There are many successful methods for MT system combination that could be utilized, for example, using confusion networks (Peter et al., 2017) to align hypotheses and pick the best parts of each as the final translation. A more neural network specific option for MT system combination by combining outputs according to the attention alignments produced by the neural networks (Rikters and Fishel, 2017) could also be used for this purpose.

Finally, we provide an update to Nematus[42] that allows training of multi-way models by providing multiple parallel corpora as input data. We also release a set of scripts[43] that can be used to prepare a multi-way corpus from multiple parallel corpora for training of multi-way NMT systems with other frameworks.

[42] Multilingual NMT iterator - https://git.io/vAgfv
[43] Multilingual NMT Corpora Tools - https://git.io/vAOoJ

5. PRACTICAL IMPLEMENTATIONS

5.1 INTERACTIVE MULTI-SYSTEM MACHINE TRANSLATION

The tool described in this section has been designed to help MT researchers to combine and evaluate various MT engine outputs through a web-based graphical user interface using syntactic analysis and language modelling. The tool supports user provided translations as well as translations from popular online MT system APIs. The selection of the best translation hypothesis is done by calculating the perplexity for each hypothesis. The evaluation panel provides sentence tree graphs and chunk statistics. The result is an interactive syntax-based multi-system translation tool. This section is based on the paper of Rikters (2016a). The author's contribution to this work is 100%.

5.1.1 Introduction

This section presents an attempt to enrich an MSMT approach with language specific information and a clean, self-explanatory user interface. The experiments described use multiple combinations of outputs from two, three or four MT systems. Experiments described in this section are performed for the English-Latvian language pair. Translating from English, French, and German to Latvian, English, French and German is currently supported, however the underlying framework developed within this work allows application of this strategy for other language pairs as well. The automatic evaluation results obtained with this hybrid system are analysed and compared with human evaluation. The code of the developed K-Translate system is freely available at GitHub[44]. A demo server[45] with data for combining English - Latvian translations is also available.

The structure of this section is as following: subsection 5.1.2 describes the back-end and the evaluation mechanism. Subsection 5.1.3 outlines the main functionality of the graphical interface and subsection 5.1.4 provides information about how the system performs under certain experiment. Finally, the section is summarised in subsection 5.1.5.

5.1.2 System description

For the back-end, the components described in section 3.3.3 were used (visualized workflow of the system is presented in Figure 8).

For translation, four translation APIs are used. However, the architecture of the system is flexible, allowing to integrate more translation APIs easily. The system is set to be able to translate from English, German or French into Latvian, German, English or French. Nevertheless, the source and target languages can also be changed to other language pairs that are supported by the APIs, Berkeley Parser parse grammars and KenLM language models. Each new source language requires a grammar that is compliant with the Berkeley Parser. The parser is able to learn new grammars from treebanks. Each new target language requires a language

[44] K-Translate on GitHub - https://github.com/M4t1ss/K-Translate
[45] K-Translate demo - http://k-translate.lielakeda.lv/

model that is compliant with KenLM. New language models can be trained using the *lmplz* program included in KenLM.

Pre-processing

The first step is to tokenize the input. The tokenizer uses the whitespace and punctuation tokenizer from the NlpTools *PHP* library[46] that is included in the system. Tokenization is essential for proper functioning of all subsequent steps – the syntactic parser can misclassify a word or a phrase and the translation APIs can issue an incorrect translation. For example, the parser will not correctly understand a word that has a dot, comma or a colon as the ending symbol.

After tokenization, it is necessary to divide sentences into linguistically motivated chunks that will be further given to the translation APIs. For this task the Berkeley Parser is used in conjunction with a chunk extractor (chunker). The parse tree of each sentence is processed by the chunker to obtain the parts of the sentence that will be individually translated and passed to the translation step.

Sentence chunking

The chunker reads output of the Berkeley Parser and places it in a tree data structure. During this process, each node of the tree is initialised with its phrase (NP, VP, ADVP, etc.), word (if it has one) and a chunk consisting of the chunks from its child nodes. To obtain the final chunks for translation the resulting tree is traversed bottom-up post-order and only the top-level subtrees are used as the resulting chunks. The chunking consists of steps shown in Figure 41.

Figure 42 shows an example of output generated by the Berkeley Parser for the English sentence “Characteristic specialities of Latvian cuisine are bacon pies and a refreshing, cold sour cream soup.” - the visualized parse tree with two chunks highlighted in green and purple colours.

Translation with online APIs

Support for the four online translation APIs that are described in section 3.3.3 are included in the project. Each translation API is defined with a function that has source and target language identifiers and the source chunk as input parameters and the target chunk as the only output. This makes adding new APIs very easy.

[46] Natural language processing tools - http://php-nlp-tools.com/documentation/tokenizers.html

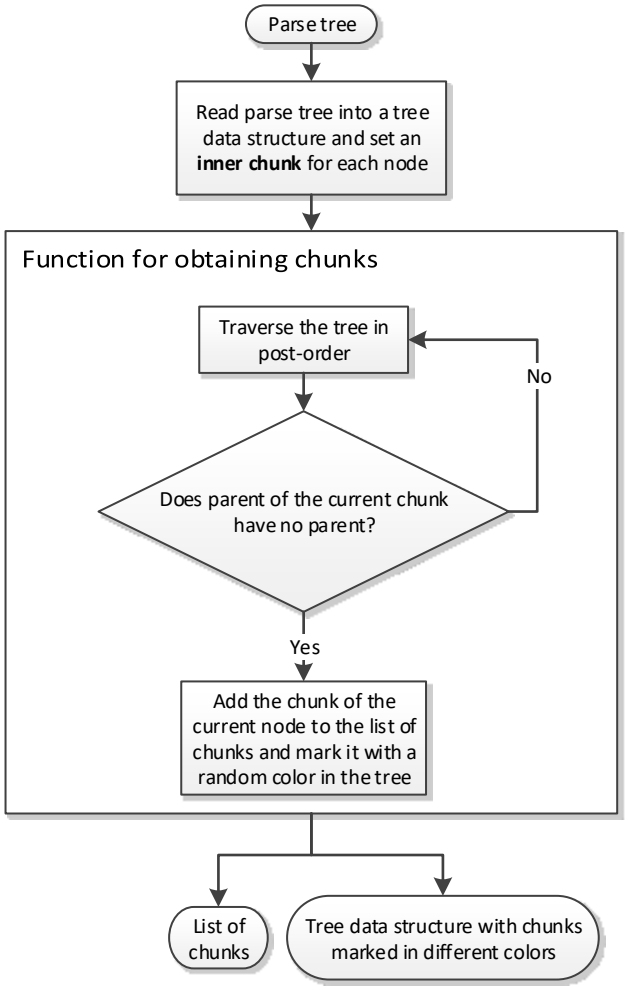


Figure 41: Chunking process flowchart

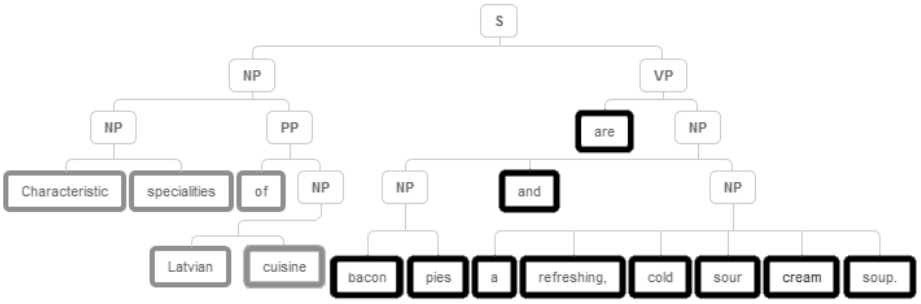


Figure 42: Visualised tree with marked chunks

```
( (S
 (NP
   (NP
     (JJ Characteristic)
     (NNS specialities) )
   (PP
     (IN of)
     (NP
       (JJ Latvian)
       (NN cuisine)
     ) ) )
 (VP
   (VBP are)
   (NP
     (NP
       (NN bacon)
       (NNS pies)  )
     (CC and)
     (NP
       (DT a)
       (JJ refreshing,)
       (JJ cold)
       (JJ sour)
       (NN cream)
       (NN soup.)
     ) ) ) ) )
```

Figure 43: Berkeley Parser parse tree output

Selection of the best translated chunk

The selection of the best translated chunk is performed exactly as described in section 3.1.2 - KenLM calculates perplexity as shown in (10), and used to compare the translated chunks.

Sentence recomposition

When the best translation for each chunk is selected, the translation of the full sentence is generated by concatenation of chunks. The chunks are recomposed in the same order as they were split up.

5.1.3 Translation combination panel

This section presents the translation combination panel which is the graphical front-end of K-Translate. Figure 44 shows a schematic overview of the options available. Each of the two ways of combining translations consists of all or most of the steps covered in the previous section. An exception is when the user choses to input their own translations – this process skips translation with online APIs.

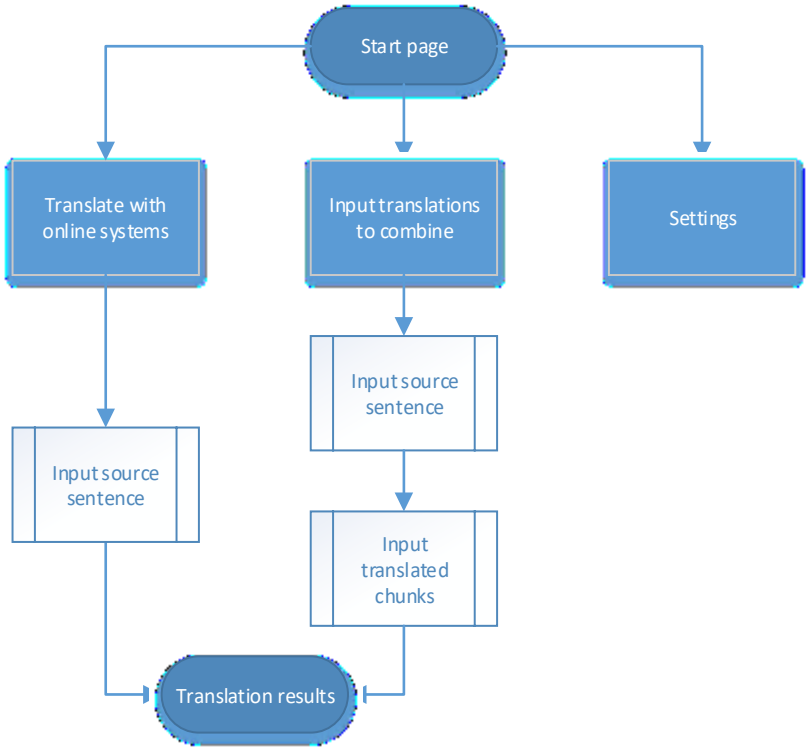


Figure 44: Architectural visualization of the translation combination panel

Translating with online systems

The start-up screen of the translation combination panel allows to fully automatically get translations from several online MT systems that have APIs available, combine them and output the best fitting hybrid translation. The source sentence input screen is shown in Figure 45 and the results look the same as when combining user provided translations (Figure 50) with the exception of showing the name of the used online system as the source instead of MT1, MT2, etc.

Combining multiple user provided translations

The second option of the translation combination panel is intended for the more experienced MT professionals who already have several (two or more) translations of the input sentence from different MT systems and just want to obtain the combined result. At first the

user must select source and target languages and input the sentence in a source language as shown in Figure 46.

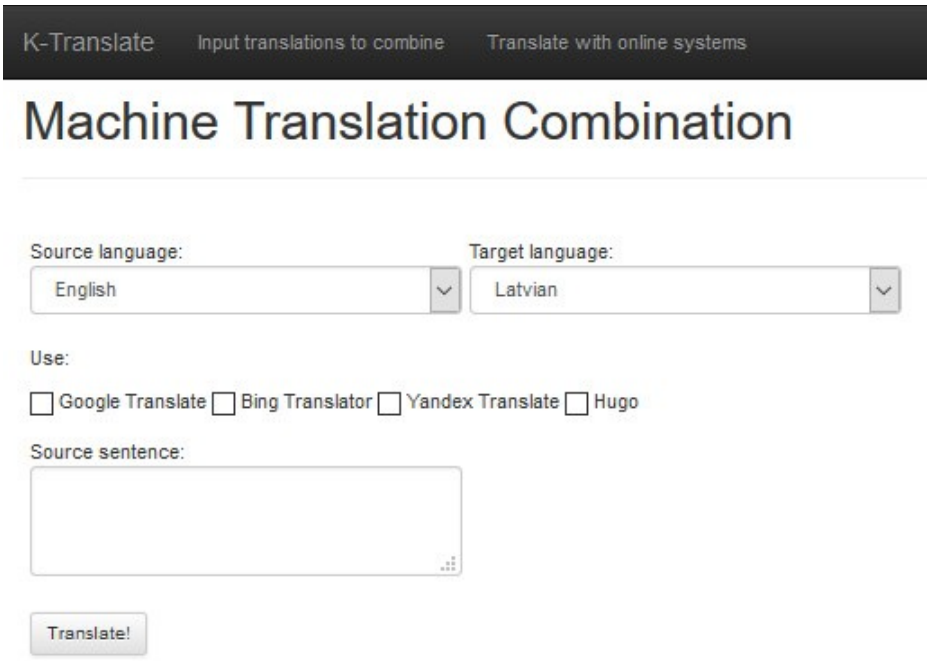


Figure 45: Translating with online APIs

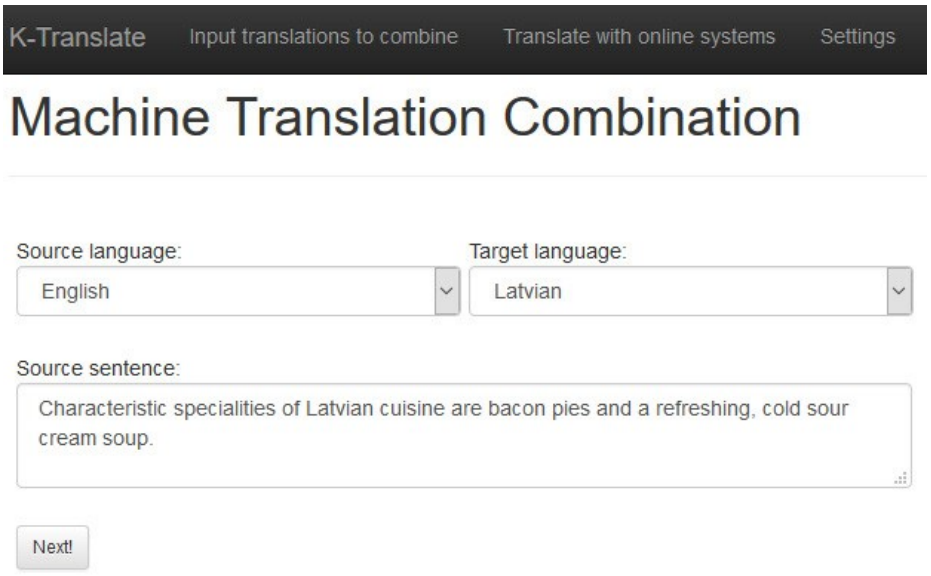


Figure 46: First step of combining of multiple user provided translations

Next, K-Translate will perform syntactic analysis on the input sentence and split it into chunks as shown in Figure 47[47]. The syntax tree with highlighted color-coded chunks will also be shown so that the user can better understand where and why the chunks have their boundaries (Figure 48). These chunks will be given in a text box each in a new line for the user to translate

[47] The process behind chunking is clarified in section 3.3.

with the chosen MT systems. Finally, the obtained translations must be pasted in the MT 1, MT 2, etc. text boxes (Figure 49) below each chunk per line to move on to the last step.

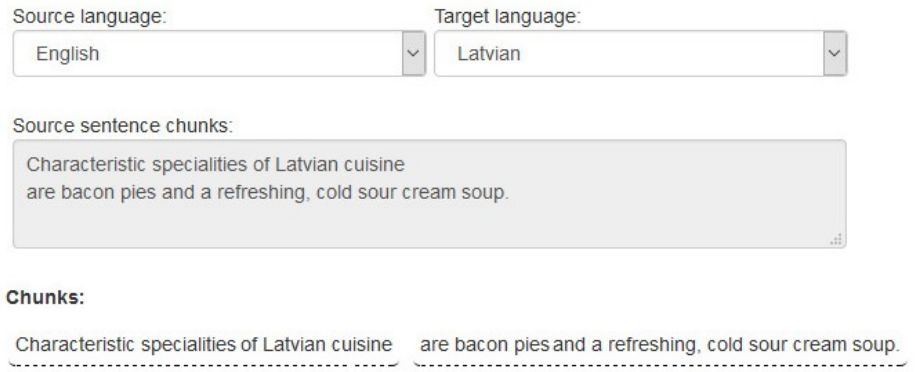


Figure 47: Second step of combining of multiple user provided translations – part 1

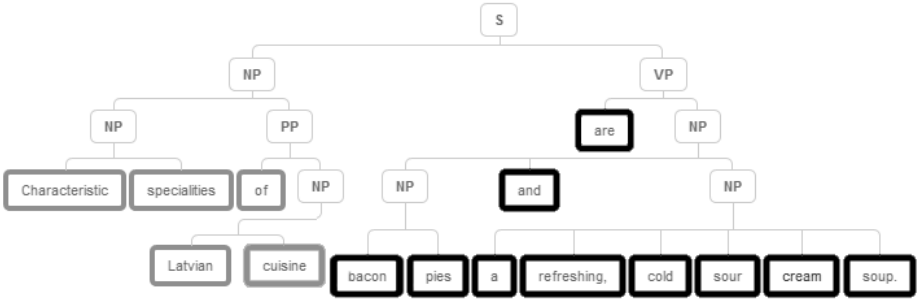


Figure 48: Second step of combining of multiple user provided translations – part 2 – a syntax tree visualization

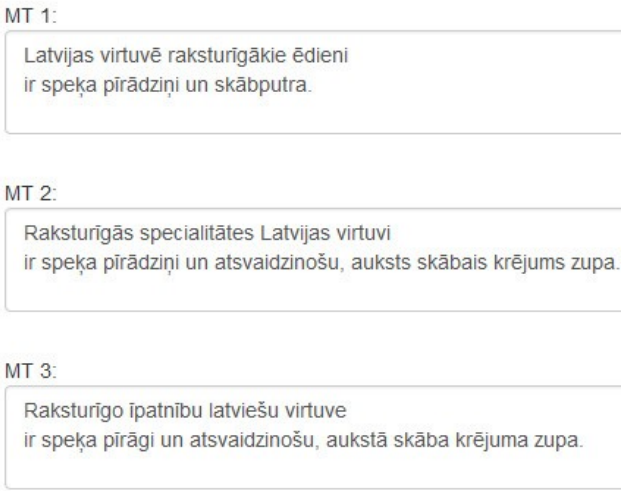


Figure 49: Second step of combining of multiple user provided translations – part 3 – input different translated chunks for source sentence chunks

In the last step (Figure 50) K-Translate will provide the best combined translation and highlight which chunks were used from which input. It also shows the source used for each

chunk and the confidence level of each selection. The confidence is calculated by comparing chunk perplexities to each other.

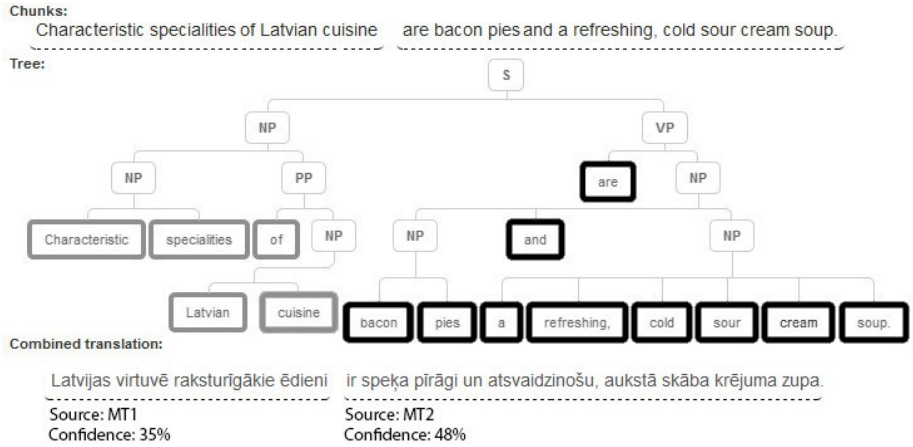


Figure 50: Translation combination results page

Settings

Before any work with K-Translate can be performed, one must first provide a Berkeley Parser compatible grammar file for each desired source language and a KenLM compatible language model file for each target language. Also, if usage of online APIs for translation is planned, the corresponding API settings are mandatory. The settings page allows for easy configuration of these values. The necessity of these requirements is explained in sections 0 and 0.

5.1.4 Experiments

This section describes the experiments performed to test the workflow of K-Translate. At first, details on the input data and experiment methodology are provided. Next, the results are summarized and interpreted. Finally, a human evaluation is performed showing how the results coincide with judgement of native speakers. For the purposes of the experiment a slightly similar hybrid MT system - Multi-System Hybrid Translator (Rikters 2015) was chosen as a baseline.

Experiment setup

The experiments were conducted on the English – Latvian part of the JRC corpus (Section 3.1.3) from which both the test data and data for training of the LM were retrieved. For testing, the test set from Section 3.1.3 was used, as well as the 5-gram LM.

The method was applied by combining all possible combinations of two and then also all three APIs. As a result, seven different translations for each source sentence were obtained. Google Translate, Hugo, Yandex and Bing Translator APIs were used with the default configuration.

Output of each system was evaluated with two scoring methods – BLEU and NIST. The resulting translations were inspected with the Web-based MT evaluation platforms MT-

ComparEval and iBLEU to determine, which system from the hybrid setups was selected to get the specific translation for each chunk and analyse differences in the resulting translations.

Experiment results and discussion

The results of the automatic evaluation are summarized in Table 35. Surprisingly all hybrid systems that include the Hugo API produce lower results than the baseline Hugo system. However, the combination of Google Translate and Bing Translator shows improvements in BLEU and NIST scores compared to each of the baseline systems. The results also clearly show an improvement over the baseline hybrid system that does not have a syntactic pre-processing step. Also, contrary to the baseline, the new system tends to use more chunks from Hugo, which, according to BLEU and NIST scores, is the better selection.

Table 35. Experiment results. B – Bing, G – Google, H – Hugo, L - LetsMT, Y – Yandex.

System	**BLEU**	**NIST**	**Hybrid selection**			
			G	**B**	**H/L**	**Y**
Google	16.19	8.37	100%	-	-	-
Bing	16.99	8.09	-	100%	-	-
Hugo	20.27	9.45	-	-	100%	-
LetsMT	20.55	9.48	-	-	100%	-
Yandex	19.75	9.30	-	-	-	100%
Baseline hybrid MT						
BG	17.09	8.41	56%	44%	-	-
GL	19.87	9.03	52%	-	48%	-
BGL	19.32	9.15	37%	29%	34%	-
K-Translate						
BG	**17.34**	8.54	74%	26%	-	-
GH	18.63	9.09	25%	-	74%	-
BH	18.98	8.97	-	24%	76%	-
HY	20.01	9.33	-	-	65%	35%
BGHY	18.33	8.67	17%	18%	35%	30%

The table also shows the percentage of translations from each API for the hybrid systems. Although, according to scores, the Hugo system was a little better than the other systems, it seems that the language model was eager to favour its translations.

Figure 51 shows an example of the source and reference sentences, and all system translations with the differences highlighted. Upon closer inspection, it can be seen that K-Translate used the first chunk from Google's output and the second chunk from Hugo. The baseline hybrid MT system would have only selected one full sentence as its output.

Figure 51: Comparison of a sentence translations with the different systems with MT-ComparEval

Human evaluation

A random 2% (32 sentences) of the translations from the experiment were given to 10 native Latvian speakers with instructions to identify the most fluent and the most adequate translation for each source sentence. The results are summarized in Table 36. Comparing the evaluation results to the BLEU scores and the selections made by the syntax-based hybrid MT, a tendency towards the Hugo translation can be observed for the BLEU score and the selection of the hybrid method, that is not visible from the user ratings. The free-marginal kappa (Randolph, 2005) for these annotations is 0.335 which indicates substantial agreement between the annotators.

Table 36. Human evaluation results

System	Fluency AVG	Accuracy AVG	K-Translate selection	BLEU
Google	35.29%	34.93%	16.83%	16.19
Bing	23.53%	23.97%	17.94%	16.99
Hugo	20.00%	21.92%	45.13%	20.27
Yandex	25.93%	27.07%	20.10%	19.75
K-Translate	21.18%	19.18%	-	18.33

The table shows that translations from the *Google Translate* system were recognized by annotators as most fluent and most adequate in 35% of cases. This contradicts with the automatic evaluation results and the selections made by K-Translate where a tendency towards the *Hugo* translation is observed.

A broader analysis of this result was performed. The hypothesis is that *Hugo* was chosen less often by the annotators because of failure to translate dates or numbers in specific sentences while the rest of the sentence was very similar to the reference, hence scoring more BLEU points. Closer inspection revealed that three sentences from *Hugo* contained "βNUMβ" tag, which appears to be an error in the named entity processor during time of experiments. There were also five sentences that contained untranslated dates, e.g., "31 december 1992" or "february 1995." These errors account for *Hugo* not be selected by annotators in 25% cases of the evaluation dataset, while in case of BLEU score, their influence was not so significant.

5.1.5 Conclusions

This section described an interactive MT system combination approach that uses syntactic and statistical features and visualizes the intermediate steps. The main goals were to provide

MT researchers with an intuitive and easy to use tool for combining translations and to improve translation quality over the selected baseline.

All test cases showed an improvement in BLEU and NIST scores when compared to the baseline system. When used only with Google and Bing, the K-Translate scores 0.35 BLEU points higher than the best individual translation provided by the APIs.

In all hybrid systems that included the Hugo API a decrease in overall translation quality was observed. This can be explained by the scale of the engines - the Bing and Google systems are more general, designed for many language pairs, whereas the MT system in Hugo was specifically optimized for English – Latvian translations.

5.2 VISUALIZING AND DEBUGGING NEURAL MACHINE TRANSLATIONS

In this section, a tool for visualizing the output and attention weights of neural machine translation systems and for estimating confidence about the output based on the attention is described. The aim is to help researchers and developers better understand the behaviour of their NMT systems without the need for any reference translations. Further in the section several specific use-cases for finding suspicious and faulty translation output with the help of this tool are provided. The tool includes command line and web-based interfaces that allow to systematically evaluate translation outputs from various engines and experiments. We also present a web demo[48] of our tool with examples of good and bad translations. This section is based on the papers of Rikters et al. (2017b) and Rikters (2018a). The author's contribution to this work is 85%.

5.2.1 Introduction

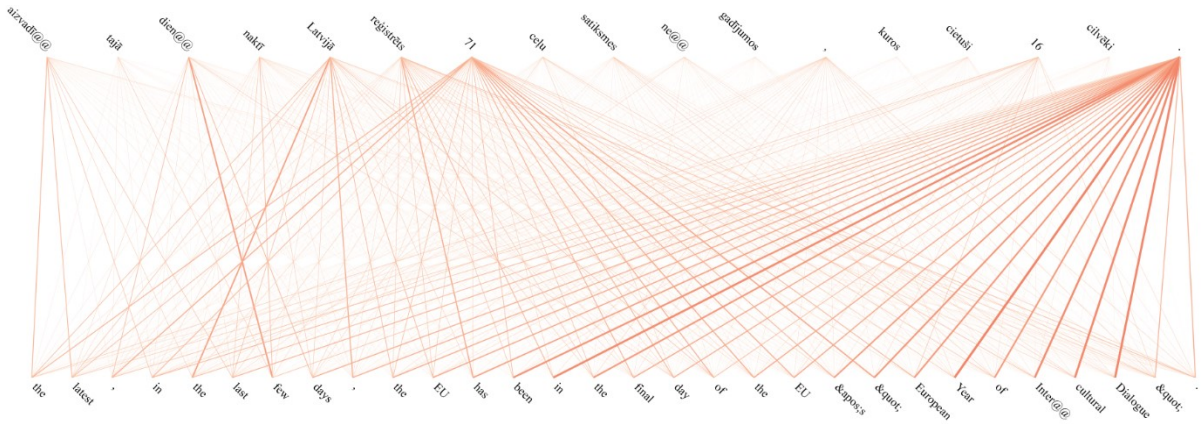
While the world of MT transitions from statistical (Koehn, 2009) to neural (e.g. Bahdanau et al., 2015), the systems themselves are slowly being replaced. The necessities behind analysing them largely remain the same, as do the tools built mostly for the older approaches.

In this section introduces a translation inspection tool that specifically targets NMT output. The tool uses the attention weights corresponding to specific token pairs during the decoding process, by turning them into one of several visual representations that can help humans better understand how the output translations were produced. The tool also uses the attention information to estimate the confidence in translation which allows to distinguish acceptable outputs from completely unreliable ones, no reference translations are required. A key difference from other similar tools is that to distinguish acceptable outputs from completely unreliable ones no reference translations are required; instead we rely on the visualized strength of the connection between the source text and the translation output; see Figure 52 for an example.

The section is structured as follows: subsection 5.2.2 summarizes related work on tools for inspecting translation outputs and alignments. Subsection 5.2.3 describes the proposed

[48] Attention visualization demo - http://ej.uz/nmt-attention

visualizations - both command line and web-based. Subsection 5.2.4 provides a look into the back-end of the system. Finally, conclusions of the section are in subsection 5.2.5.



Source	Aizvadītajā diennaktī Latvijā reģistrēts 71 ceļu satiksmes negadījumos, kuros cietuši 16 cilvēki.
Hypothesis	The latest, in the last few days, the EU has been in the final day of the EU's "European Year of Intercultural Dialogue".
Reference	71 traffic accidents in which 16 persons were injured have happened in Latvia during the last 24 hours.

Figure 52: A Latvian to English neural translation output that has no relation to the input. The weak connection is obvious from the visualized attention weights, even without knowing the source and target languages or seeing the input or output texts. Confidence: **18.11%**; CDP: 44.49%; AP_{out}: 67.41%; AP_{in}: 79.58%.

5.2.2 Related work

Zeman et al. (2011) describe Addicter - a set of command-line and simple web-based tools that can be useful for inspecting automatic translations and finding systematic errors among them. One of the tools in Addicter, *alitextview.pl*, is designed to convert SMT alignments from the typical alignment pair format (*source_token_id - target_token_id*}) to a table representation, making it more human-readable. Our command-line interface took much inspiration from this work while adapting to the specifics of the NMT counterpart of alignments.

Madnani (2011) introduces iBLEU - a web-based tool for visualizing BLEU scores. Unlike alignments between the source and the hypothesis, the calculation of BLEU requires a reference translation to which the hypothesis will be compared. On top of that, iBLEU also allows to add another file with hypotheses from another MT system for a direct comparison. Given these inputs, the tool highlights the differences between the translations and reference material. It also enables easy navigation through the set of sentences by representing the BLEU score of each sentence in a clickable bar chart. A quick jump to a specific sentence is possible by entering its number. The clickable chart and jumps seemed most desirable features for us, so we added similar capabilities to the web version of our tool.

Klejch et al. (2015) developed MT-ComparEval - a web-based translation visualization tool that seems to build upon iBLEU by adding many more fine-grained features. It also allows to compare differences between translations and references, other translations and the source

input. The main differences are that (1) MT-ComparEval stores all imported data as experiments for viewing at any time, where iBLEU forgets everything upon a page refresh; (2) for each of these experiments, one can add output from multiple systems (iBLEU can cope with only 2); (3) MT-ComparEval displays additional scores (precision, recall, F-measure); and (4) it shows various detailed sentence and n-gram level statistics with configurable highlighting of the differences. A noticeable shortcoming is that one cannot jump to a specific sentence in the set. While ordering by sentence ID is possible, to view the 1000th of 2000 one would have to scroll through the first 999.

Nematus includes a set of utilities for visualizing NMT attentions. The first one, *plot_heatmap.py* plots alignment matrices similar to the previously mentioned *alitextview.pl*, using Nematus output translations with alignments. The second tool, *visualize_probs.py* generates HTML for a web view that displays the output translation in a table with the background of each token shaded according to the attention weight. The final tool, consisting of *attention.js* and *attention_web.php*, connects source and target tokens with lines as thick as the corresponding attention weights between them. However, there is no tool included to generate the latter visualization for an arbitrary sentence - it is given only in the form of one set example. This last tool was a strong inspiration for building our tool. We reused parts of its code in the web version of our visualization.

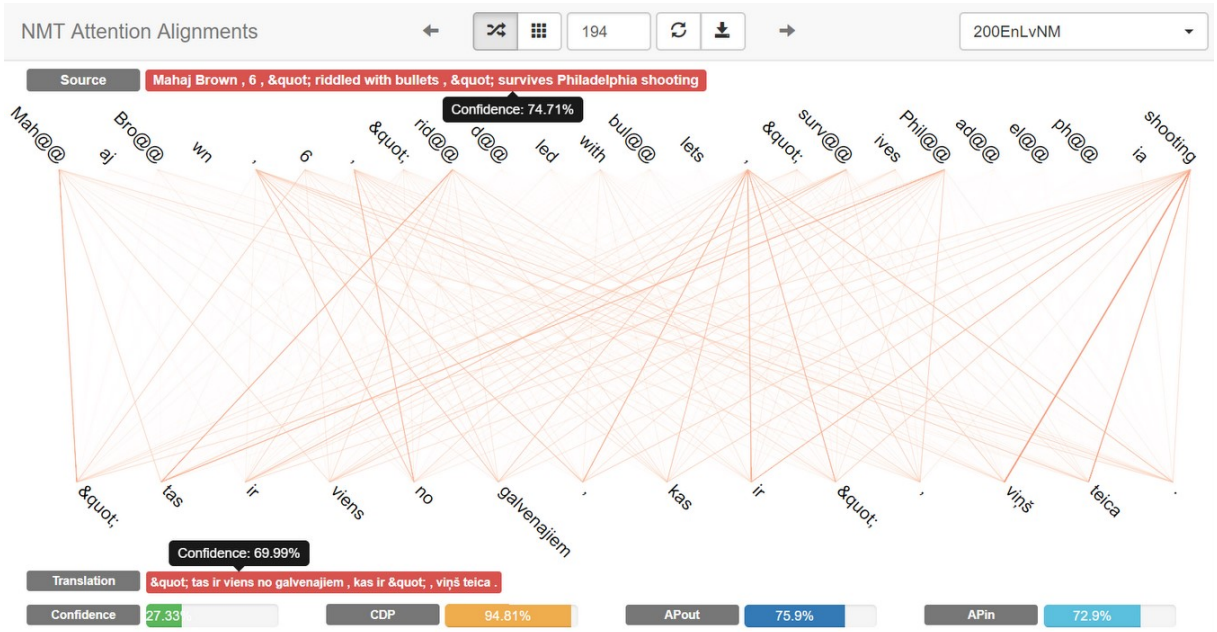
Neural Monkey provides several visualization tools for checking the training process that include visualizing attention as soft alignments. It can generate matrices similar to the previously mentioned *alitextview.pl* for each sentence in the first validation batch during the training process. A few drawbacks of this method are that the images are (1) of a static size (the predefined maximum input length * maximum output length) - if sentences are longer, the attention image gets cut off, if shorter, bottom rows of the matrix (representing the input) are left black and columns (representing the output) on the far right side are filled with "phantom" attention; (2) no input and output words, tokens or subword units are displayed, only the matrix; (3) there is no option to generate visualizations for a test set outside the system training process.

5.2.3 The tool from a users' perspective

The main goals of our tool are to provide multiple ways of visualizing NMT attention alignments, as well as to make it easy to navigate larger datasets and find specific examples. To accomplish these goals, we implemented two main variations of our tool, a textual command line visualization and a web-based visualization. This section provides an insight into the features of both of them and suggestions as to when they can be useful.

Web browser visualization

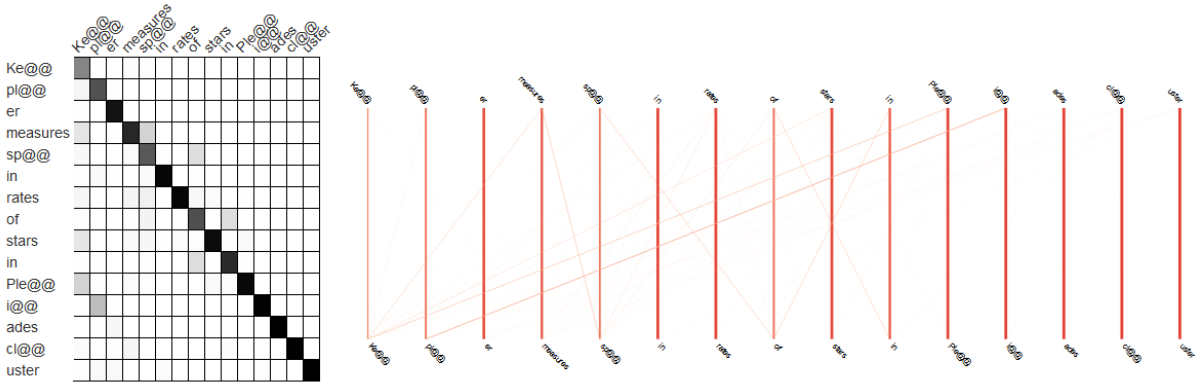
The web visualization is intended to provide an intuitive overview of one or multiple translated test sets. This is done by showing one sentence at a time, with navigation to other sentences by ID, length or multiple confidence measures. Switching between experiments (test sets) is also easy. For each individual sentence, four confidence metrics are shown, and a confidence score for each source and translated token (or subword unit). The tool also allows to export the alignment visualization of any selected sentence to a high-resolution PNG file with one click.



Source	Mahaj Brown , 6 , "riddled with bullets ," survives Philadelphia shooting
Hypothesis	"tas ir viens no galvenajiem , kas ir" , viņš teica.
Reference	6 gadus vecais Mahajs Brauns "ložu sacaurumots" izdzīvo apšaudē Filadelfijā.

Figure 53: An example of a translated sentence that exhibits a low confidence score. Confidence: **27.33%**; CDP: 94.81%; AP_{out}: 75.9%; AP_{in}: 72.9%.

The essential part of the visualization is represented in the following way: source tokens (at the top) are connected to translated tokens (at the bottom) via orange lines, ranging from completely faint to very thick, as visible in Figure 53 and Figure 54. A thicker line from a translated token to a source token means that the decoder paid more attention to that source token when generating the translation. Ideally, these lines should mostly be thick with some thinner ones in between. When they look chaotic, connecting everything to everything (Figure 53) or everything in the translation to mostly just one token in the source, that can be a well indication of an unsuccessful translation that will possibly have little to no relation with the source sentence. On the other hand, if all lines are thick, straight downwards, connected one-to-one (right part of Figure 54), that may point to nothing being translated at all. Additionally, the matrix style visualization is also available in the web version as shown on the left part of Figure 54.



Source	Kepler measures spin rates of stars in Pleiades cluster
Hypothesis	Kepler measures spin rates of stars in Pleiades cluster
Reference	Keplers izmēra zvaigžņu griešanās ātrumu Plejādes zvaigznājā.

Figure 54: An example of a translated sentence that exhibits a suspiciously high confidence score. The translation here is a verbatim rendition of the input. Matrix form visualization on the left, line form visualization on the right. Confidence: **95.44%**; CDP: 100.0%; AP_{out}: 98.84%; AP_{in}: 98.85%.

Confidence scores

To aid in locating suspicious and potentially bad translations, we introduced a set of confidence metrics (more details in 4.3.3). For each sentence, the tool displays an overall confidence score, coverage deviation penalty, and input and output absentmindedness penalties. The overall confidence score is also shown for each source token, indicating the amount of confidence that the token has been used to generate a correct translation, as well as for each translated token, indicating the amount of confidence that it is a correct translation. All of these scores are represented in percentages from 0 to 100 and can be used to navigate through the test set (Figure 55), making it easy to quickly find very good or very bad translations among hundreds. The selected sentence is highlighted simultaneously across all navigation charts and each chart can be sorted in either direction or reset to the order by sentence ID.

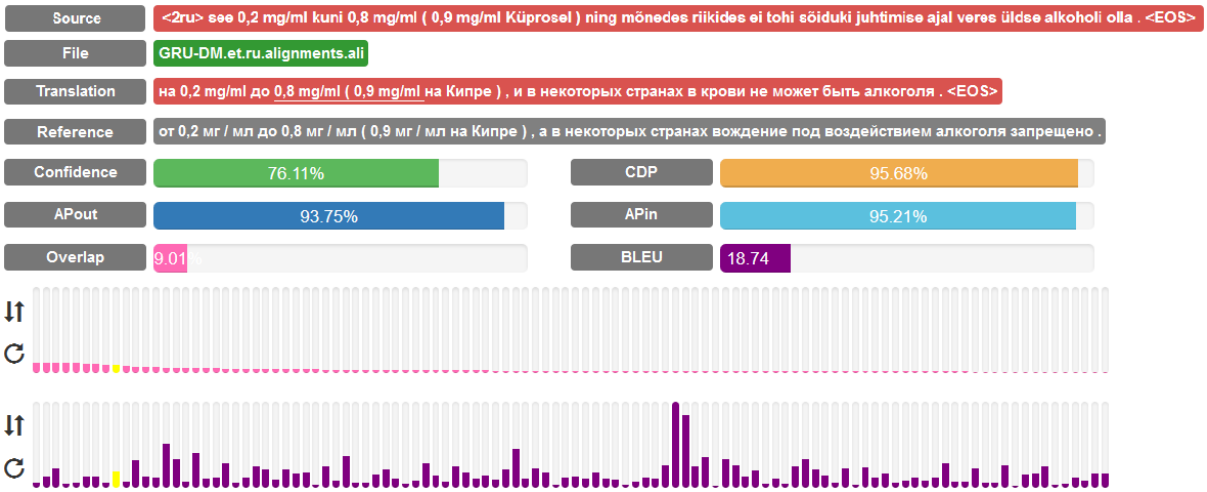
Figure 55: Navigation charts allow to jump to a sentence based on its length in characters (red), confidence (green), coverage deviation penalty (dark yellow), absentmindedness penalty for

input (dark blue) and output (light blue). The currently active sentence is highlighted in bright yellow. All charts are sortable and scrollable for a better user experience

Overlap

The confidence score considers hypotheses translations that are long and have a significant overlap with the source sentence as a worse translation, while tolerating considerable overlap for shorter sentences.

In addition to contributing to the final confidence score, the overlap ratio has been added as an individual score for sorting, navigating and comparing sentences from a dataset as shown in Figure 56.



Source	see 0,2 mg/ml kuni 0,8 mg/ml (0,9 mg/ml Küprosel) ning mõnedes riikides ei tohi sõiduki juhtimise ajal veres üldse alkoholi olla.
Hypothesis	на 0,2 mg/ml до 0,8 mg/ml (0,9 mg/ml на Кипре) , и в некоторых странах в крови не может быть алкоголя.
Match	0,8 mg/ml (0,9 mg/ml

Figure 56: An example translation from Estonian into Russian, showing useful features for debugging translation outcomes - underlining of the longest matching substring between the source and translated sentences; sorting translations by overlap (pink bars) or BLEU score (purple bars); reference translation (grey background).

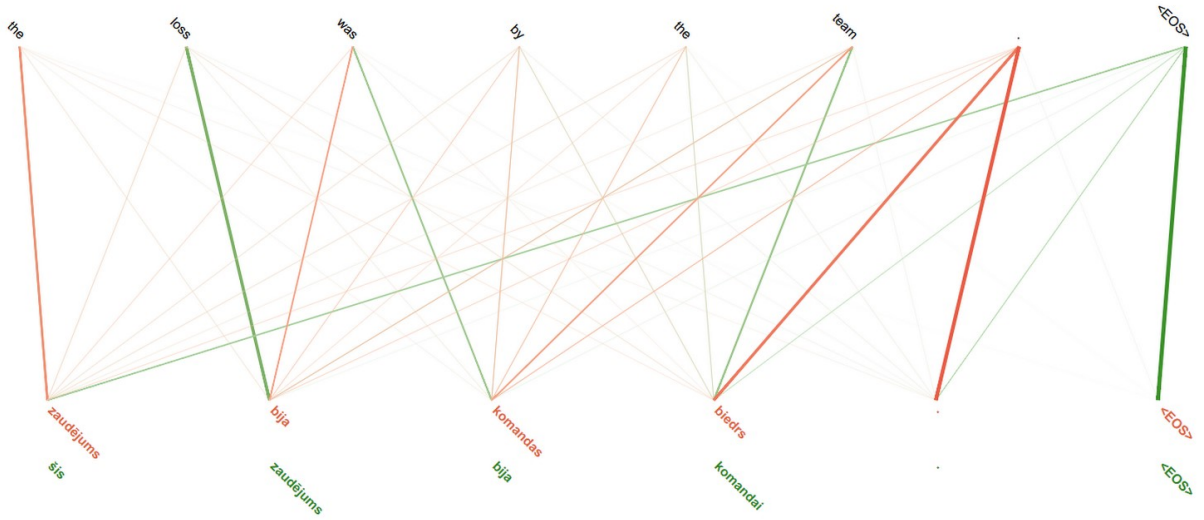
The system also underlines the longest matching substring between the source and translation in cases where the overlap is high enough (over 10%). An example is shown in Figure 56, where the overlap ratio is 20.19%.

References and BLEU

We believe that simply displaying the reference next to the hypothesis is helpful more often than not. Having provided references also allows to calculate BLEU scores for the translations, providing yet another dimension for sorting (Figure 56). Unlike overlap, the BLEU scores do not influence the overall confidence scores.

Comparing Translations

A major feature of the tool is the option to directly compare two translations of the same source sentence. To perform the comparison, all source sentences for both input datasets must match, but the target sentences may differ in output token order as well as count. Comparisons may be performed between translations obtained from any two of the five currently supported NMT frameworks (Nematus, Neural Monkey, OpenNMT (Klein et al., 2017), Marian and Sockeye (Hieber et al., 2017)) or even an arbitrary input file, as long as it's formatted according to the specification provided in the instruction file[49].



Source	the loss was by the team.
Hypothesis 1	zaudējums bija komandas biedrs.
Hypothesis 2	šis zaudējums bija komandai.
Reference	zaudē komanda.

Figure 57: A direct comparison of attention alignments for translating the same sentence with two different NMT systems.

Figure 57 shows an example comparison of a sentence translated by two different NMT systems. On the top row is the source text and the bottom rows represent output from each individual NMT system color-coded to match the colours of the alignment lines. The second hypothesis (in green) exhibits stronger and more reliable output alignments to the content words while the first shows strong alignments coming from the stop sign. In this example neither hypothesis matches the reference, but since it is only two words long for a source sentence of triple the length, it can hint to an oversimplified translation by the translator (assuming English was the original) and does not mean that both hypotheses are completely wrong. In fact, the second hypothesis is a fairly decent representation of the source sentence.

[49] Using other input formats - https://github.com/M4t1ss/SoftAlignments\#how-to-get-alignment-files-from-nmt-systems

Command line visualization

The command line visualization is available in three different formats: (1) using twenty-five different shades of grey as shown in Figure 58 (right); (2) using five gradually shaded Unicode block elements as shown in Figure 58 (left); and (3) using nine gradually filled Unicode block elements. Each sentence is output via a graphical matrix, where rows represent the source input tokens or subword units and columns representing the target side. The corresponding tokens are printed out on the bottom (target) or far right side (source) of the matrix. Unlike the authors of *alitextview.pl*, we chose to represent the source tokens on the right, so that the graphical matrix starts at the beginning of the line for each sentence. After each sentence, one empty line is printed.

One obvious use case for the command line visualization is to directly compare alignments of NMT attention with the ones produced by SMT. This type of visualization is also the fastest, therefore it can be used to quickly check alignments for a specific sentence. Fixed-width Unicode fonts can be used in almost all text editors, so redirecting output to a text file to share with others is also a valid application. However, to view the *colour* version from a text file, it needs to be interpreted as xterm color sequences, e.g. using "*less -R*" in a Linux terminal.

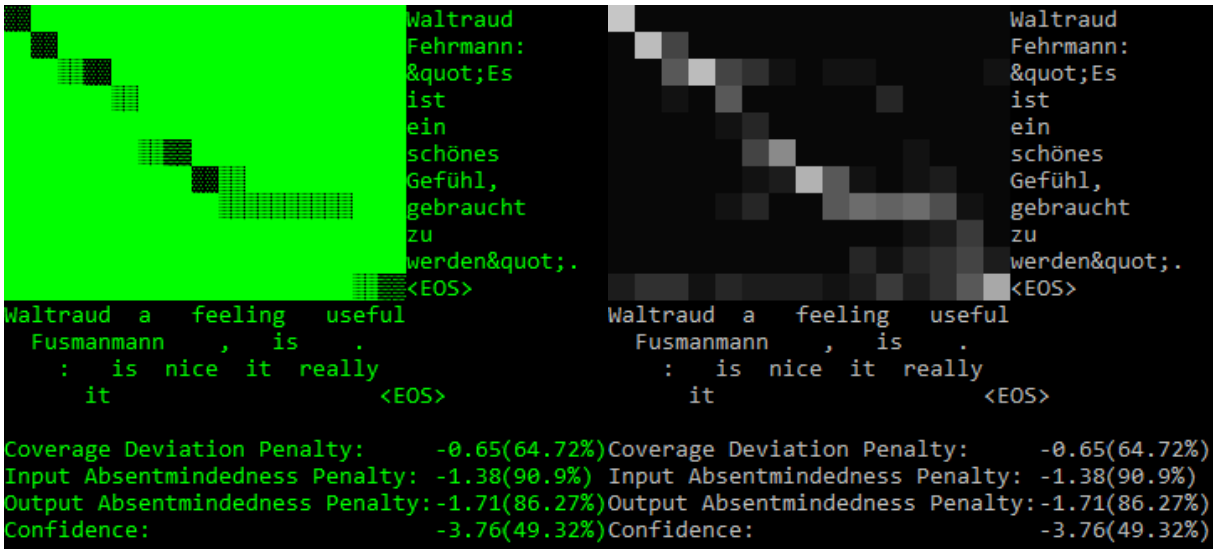


Figure 58: Visualization in the command line, using five differently shaded block elements (left), and twenty-five different tones of grey (right).

5.2.4 System description

The visualization tool is developed in *Python* and *PHP*. It is published in a GitHub repository[50] and open-sourced with the MIT License.

Both visualizations can be run directly from the command line. The web version is capable of launching on a local machine without the requirement for a dedicated web server.

[50] NMT Attention Alignment Visualizations - https://github.com/M4t1ss/SoftAlignments

Scoring attention

This section provides details about how the previously mentioned confidence scores are calculated and outlines what is needed to make good use of each option.

The basis of our scoring methods was influenced by Wu et al. (2016), who defined a coverage penalty for punishing translations that do not pay enough attention to input tokens (14). To complement that, we introduce a set of our own metrics:

- **Coverage deviation penalty** (*CDP*) penalizes attention deficiency and excessive attention per input token.
- **Absentmindedness penalties** (AP_{out}, AP_{in}) penalize output tokens that pay attention to too many input tokens, or input tokens that produce too many output tokens.
- **Confidence** is the sum of the three metrics – *CDP*, AP_{out} and AP_{in}.
- **Overlap penalty** (OP) penalizes translations that copy large fractions from source sentences

Coverage deviation penalty

Unlike *CP*, *CDP* penalizes not just attention deficiency but also excessive attention per input token. The aim is to penalize the sum of attentions per input token for going too far from 1.0, so that tokens with the total attention of 1.0 get a score of 0.0 on the logarithmic scale, while tokens with less attention (like 0.13) or more attention (like 3.7) get lower values. We thus define the coverage deviation penalty (*15*). The metric is on a logarithmic scale, and it is normalised by the length *J* of the input sentence in order to avoid assigning higher scores to shorter sentences.

Absentmindedness penalties

To target scattered attention per output token, we introduce an output absentmindedness penalty (*16*). It evaluates the dispersion via the entropy of the predicted attention distribution, resulting in values from 1.0 for the lowest entropy to 0.0 for the highest. The values are again on the log-scale and normalised by the source sentence length *L* (*i* is the output token index, *j* - the input token index, α – attention probability).

The absentmindedness penalty can also be applied to the input tokens after normalising the distribution of attention per input token (19).

$$AP_{in} = -\frac{1}{L}\sum_j \sum_i \alpha_{ij} \cdot log\alpha_{ij} \quad (19)$$

Overlap penalty

A stronger penalty (20) is allocated to longer sentences that copy large amounts from the source while shorter ones get more tolerance (e.g., the three-word English sentence “Thanks Barack Obama.” can be perfectly translated into “Paldies Barack Obama.” although $\frac{2}{3}$ of words in the translation are the same in the source).

$$OP = (0.8 + (L_t * 0.01)) * (3 - ((1 - S) * 5)) * (0.7 + S) * tan(S) \quad (20)$$

In all of the metrics L is the length of the source sentence; L_t - length of the target sentence; S - similarity between the source sentence and the translation on the scale of 0 - 1; α_{ji} - the attention weight between source token *i* and translation token *j*.

The final confidence score sums up all three above mentioned metrics (21).

$$confidence = \begin{cases} CDP + AP_{out} + AP_{in}, \ if \ similarity < 0.3 \\ CDP + AP_{out} + AP_{in} - OP, \ otherwise \end{cases} \quad (21)$$

For visualization purposes, each of the scores needed to be set on the same scale of 0-100%. To achieve that, we applied (22),

$$\text{percentage} = e^{-C(X^2)} \quad (22)$$

where *X* is the score to convert and *C* is a constant of either 1 for the *CDP* or 0.05 for the other scores (AP_{out}, AP_{in}, *confidence*).

System architecture

The code can be divided into two logical parts: (1) processing input data and generating output data and (2) displaying and navigating the generated output data in a web browser. The former part is written in Python and handles all input data, generates output data, displays the command line visualization or launches a temporary web server for the web browser visualization. Each time a web visualization is launched, a new folder is created within */web/data* where all necessary output data files are stored, a temporary *PHP* web server is launched on *127.0.0.1:47155*, and the address is opened as a new tab in the default web browser. After stopping the script all data remains in the */web/data* and can be accessed later as well.

The latter part is responsible for everything that is shown in the browser. It mainly consists of *PHP*, *HTML* and *JavaScript* code that facilitates quick navigation between sentences even in larger data files, as well as navigation charts and sorting, visualization export to image files and a responsive user interface. If necessary, this part can be used as a stand-alone website for displaying and interacting with pre-generated results.

Requirements and usage

The requirements are as follows:

- *Python* (2 or 3) and *NumPy*
- *PHP* 5.4 or newer (for web visualization)
- Nematus, OpenNMT, Sockeye or Neural Monkey (for training NMT systems)
- Nematus, AmuNMT (Junczys, 2016), OpenNMT, Sockeye or Neural Monkey (for translating and extracting attention data)
    - Or any NMT framework that can output an attention matrix for each translation (may require format conversion)

To use the tool, first translate a set of sentences using a supported NMT framework with the option of saving alignments[51] switched on. The sources combined with the resulting translations + attention matrices can then be used as input for the *process_alignments.py* script. Depending on the selected output type, alignments will either be displayed in the command line or a new tab will be opened in the default web browser. Example input files from each supported NMT framework are provided along with commands to run them.

5.2.5 Finding faulty translations

This section summarises several tips and tricks that may come in handy when using the tool to look for faulty translations of various kinds. Here we also list common causes associated with the problems. Some peculiarities to pay attention to may include:

- Short sentences with a low confidence, CDP, AP_{in} or AP_{out}

  All of the metrics do not necessarily need to be low, but translations that exhibit at least one of them to be under 30% are often worth looking into.
- Long sentences with a high overlap

  As stated before, for short, several words long sentences, it may be completely normal to have an overlap of 50% or more, but if it occurs in sentences that are 10 or more words long, it may indicate that the system has only partially translated the source or not translated anything at all. When completely untranslated sentences are found, it is worth checking the training data for any source-target sentence pairs that are equal. Removing them from the training data should help.
- Sentences with a low BLEU score, but normal or even high confidence, CDP, AP_{in} and AP_{out}

  The BLEU metric has its flaws and one of them is comparing each hypothesis to only one reference, while it is often possible to translate the same sentence in several different ways. In cases when the only low-scoring metric output by the tool is the BLEU score, it is often that the translation is perfectly good, but just different from the reference. Such sentences are often useful examples to show that lower BLEU scores of neural MT systems do not necessarily represent lower quality translations and are cheaper to find than performing full manual human evaluations.

A separate recommendation specifically for comparing two translations is to look at the attention alignment lines and try to find ones with source tokens having strong alignments to different hypothesis tokens, while maintaining relatively similar confidence scores. Such translations are often synonyms.

5.2.6 Conclusions

In this section, we described our tool for visualizing attention alignments generated by neural machine translation systems and for estimating confidence of the translation. The tool aims to help researchers better understand how their systems perform by enabling to quickly locate better and worse translations in a bigger test set.

[51] How to get alignment files from NMT systems - https://github.com/M4t1ss/SoftAlignments

Compared to other similar tools, ours relies on the confidence scores and does not require reference translations to facilitate this easier navigation. This allows to integrate it, for example, in an NMT system with a web interface, providing users with an explanation for the result of a specific translation. However, if reference translations are provided, several additional features become available.

One limitation of the tool is the inability to make full use of attention alignments from NMT systems with a very high amount of attention matrices in the neural network. For example, convolutional neural network MT systems (Gehring et al., 2017) tend to be trained with 15 or more layers with an attention matrix in each of them. Self-attentional transformer network NMT systems (Vaswani et al., 2017) may be trained with 6 layers each having 8 attention heads – resulting in 48 attention matrices. Even when all attentions are summed up, the result looks like every source token is connected to every hypothesis token as can be seen in Figure 59.

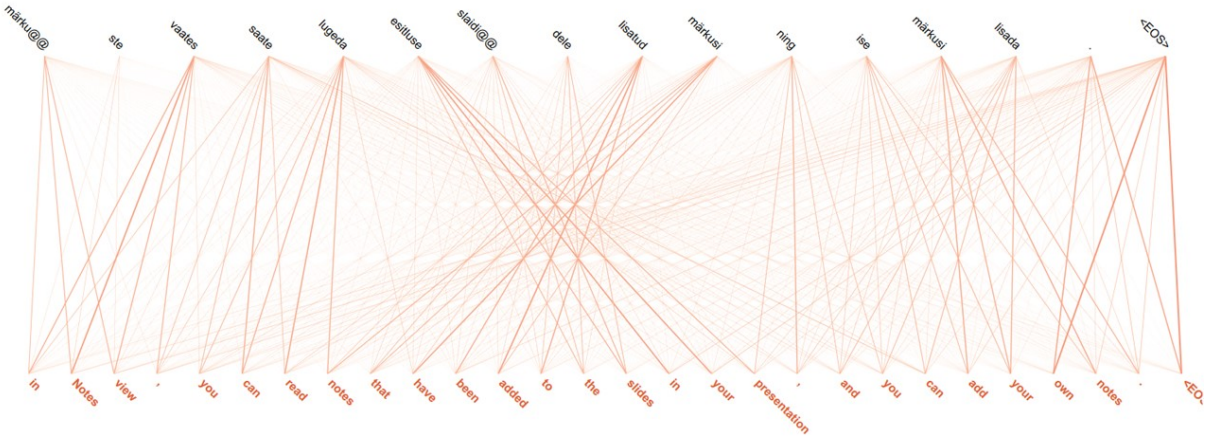


Figure 59: An example of attention alignments from a 15-layer encoder and 15-layer decoder convolutional neural machine translation system trained with FairSeq[52].

5.3 CLEANING CORPORA TO IMPROVE NEURAL MACHINE TRANSLATION PERFORMANCE

Large parallel corpora that are automatically obtained from the web, documents or elsewhere often exhibit many corrupted parts that are bound to negatively affect the quality of the systems and models that learn from these corpora. This section describes frequent problems found in data and such data affects neural machine translation systems, as well as how to identify and deal with them. The solutions are summarised in a set of scripts that remove problematic sentences from input corpora. The section is based on the papers of Rikters (2018b) and Pinnis et al. (2018). The author's contribution to this work is 100%.

5.3.1 Introduction

MT systems - both, SMT and NMT - rely on large amounts of parallel data for training the models. It is often the case that larger amounts of corpora lead to higher quality models,

[52] Facebook AI Research Sequence-to-Sequence Toolkit https://github.com/facebookresearch/fairseq

therefore a common practice is automatic extraction of such corpora from web resources, digitised books and other sources. Such data is prone to be noisy and include all kinds of problematic sentences alongside the high-quality ones. Data quality plays an important role in training of statistical and, especially, neural network-based models like NMT, which is quick to memorise bad examples. In the case of training SMT and NMT systems, often the only pre-processing is done using scripts from the Moses Toolkit (Koehn et al., 2007), which is only capable of removing sentences that are longer or shorter than a specified amount or the source-target length ratio is too high.

In this section, we explore the types of low-quality sentences commonly found in parallel corpora. We also compare the benefits of using additional filters to remove these sentences before training MT systems in contrast to using only the Moses scripts. We introduce a set of corpora cleaning tools[53] that remove sentences that have some of the most common problems found in large corpora. It is published in GitHub with the MIT open-source license.

5.3.2 Related work

Zipporah (Xu and Koehn, 2017) is a trainable tool for selecting a high-quality subset of data from a huge amount of noisy data. The authors report that it can improve MT quality by up to 2.1 BLEU, but in order to use it, the tool requires a known high-quality dataset for training.

Wolk (2015) proposes a method that uses online MT engines to translate source sentences from a parallel corpus and compare them with the given target sentences. It is very expensive to use on real-world parallel corpora, containing tens of millions of parallel sentences. The author reports results on using the method on rather small corpora of only several million words.

Khadivi and Ney (2005) introduce a parallel corpora filtering method based on word alignment models. Similar to Zipporah, this method also relies on training using a high-quality corpus.

5.3.3 Problems in corpora

This section outlines some often-occurring problems in parallel corpora. The specific examples were obtained from the English-Estonian part of the ParaCrawl[54] corpus.

One of the most common defects in parallel corpora is a high mismatch between the non-alphabetic characters between source and target sentences (Figure 60). Also, often there are sentences that are completely or mostly composed of characters outside the scope of the language in question (Figure 61).

In parallel corpora, we may occasionally see the same sentence of one language aligned to multiple different ones of the other language (Figure 62), but this is not always a bad indication, since they may just be paraphrases of the same concept (Figure 63). It is also wise to check if sentences in specific languages actually consist of text in that language (Figure 64)

[53] Corpora Cleaning Tools: https://github.com/M4t1ss/parallel-corpora-tools
[54] Large-Scale Parallel Web Crawl: http://statmt.org/paracrawl

as there may be citations and other parts of foreign language texts, especially in news domain corpora.

Finally, a little less common observation for automatically gathered corpora, but somewhat more often in automatically generated (translated) parallel corpora is the repeating of tokens (Figure 65). Sentences like this may not always be incorrect, but they introduce ambiguity when used to train MT systems.

English	Estonian
Add to my wishlist	Hommikul (200 + 200 = 400 kcal)
Dec 2009	ÊßÇÌí 2009

Figure 60: An example of a high mismatch in non-alphabetical character counts between source and target.

Ô²Õ¡Ö Õ¥Ö Õ Õ¥Õ¦,	ლÇáÝÇÑÓ ÇáäÈíáლ,Nader-87	அறவுரையாளராக ".
MÃ©szÃ¡ros IstvÃ¡n	الفلسفة	走った森,

Figure 61: Examples of sentences with over 50% non-alphabetical symbols.

English	Estonian
I voted in favour.	kirjalikult. – (IT) Hääletasin poolt.
I voted in favour.	Ma andsin oma poolthääle.

Figure 62: An example of an English sentence aligned to multiple different Estonian sentences.

English	Estonian
That is the wrong way to go.	See ei ole õge.
This is simply wrong.	See ei ole õge.

Figure 63: Multiple English paraphrased sentences aligned to one Estonian sentence.

English	Estonian
Zaghachi See okwu	3 Comments
Täna mängitud: 25 910	Täna mängitud: 25 929

Figure 64: Examples of sentences with a different identified language than the one specified.

English	Estonian
1 If , , and are the roots of , compute .	1 Juhul kui , Ja on juured , Arvutama .
we have that and or or or .	meil on, et ja või või või .
NXT Spray - NAPURA	NXT SPRAY NXT SPRAY

Figure 65: An example repeating tokens (underlined).

5.3.4 Corpora filters

The filters described in this section are mainly intended for parallel corpora consisting of two files with identical line-counts where each line of one file is related to the same line of the other file. Several of the filters are applicable to monolingual data as well and can be used to clean data for unsupervised MT training, back-translation, and other use-cases.

Unique parallel sentence filter – removes duplicate source-target sentence pairs.

Equal source-target filter - removes sentences that are identical in the source side and the target side of the corpus.

Multiple sources - one target and **multiple targets - one source** filters – removes repeating sentence pairs where the same source sentence is aligned to multiple different target sentences and multiple source sentences aligned to the same target sentence.

Non-alphabetical filters – remove sentences that contain over 50% non-alphabetical symbols on either the source side or the target and sentence pairs that have significantly more (at least 1:3) non-alphabetical symbols in the source side than in the target side (or vice versa).

Repeating token filter – especially useful for filtering back-translated parallel corpora that are created by translating a clean monolingual corpus into another language using NMT. NMT output may sometimes exhibit repeated words in the generated translation, which indicates that the system had problems translating a part of the sentence and it used the repetitions to fill the gap. In such cases the source-target sentence pair is likely to not be a good parallel sentence, therefore the repeating token filter removes them.

Correct language filter – uses language identification software (Lui and Baldwin, 2012) to estimate the language of each sentence and removes any sentence that has a different identified language from the one specified.

Moses Scripts and Subword NMT – calls Moses scripts for tokenising, cleaning, truecasing, and Subword NMT (Sennrich et al., 2016c) for splitting into subword units. This process prepares the corpus up to the point where it can be passed on to the NMT system for training.

5.3.5 Experiments and results

Corpora cleaning

Table 37: Detailed results on filtering English-Estonian/Finnish/Latvian larger common parallel corpora from WMT shared tasks.

	Paracrawl		Rapid			Europarl		
	En-Et	**En-Fi**	**En-Et**	**En-Fi**	**En-Lv**	**En-Et**	**En-Fi**	**En-Lv**
Corpus size	1298103	624058	226978	583223	306588	652944	1926114	638789
Unique	26	37	23	161463	80894	23218	52686	19652
	0.00%	0.01%	0.01%	27.68%	26.39%	3.56%	2.74%	3.08%
src == tgt	242816	41611	428	3488	2929	490	528	707
	18.71%	6.67%	0.19%	0.60%	0.96%	0.08%	0.03%	0.11%
* sources 1 target	267235	17239	1108	1513	990	1176	6631	979
	20.59%	2.76%	0.49%	0.26%	0.32%	0.18%	0.34%	0.15%
* targets 1 source	69225	9532	752	1016	329	462	3536	435
	5.33%	1.53%	0.33%	0.17%	0.11%	0.07%	0.18%	0.07%
> 50% non-alpha	200338	12919	1226	5647	1699	66	285	72
	15.43%	2.07%	0.54%	0.97%	0.55%	0.01%	0.01%	0.01%
Non-alpha mismatch	23777	12737	6674	13311	6361	7211	24847	4012
	1.83%	2.04%	2.94%	2.28%	2.07%	1.10%	1.29%	0.63%
Repeating tokens	11210	1397	175	396	171	727	2594	703
	0.86%	0.22%	0.08%	0.07%	0.06%	0.11%	0.13%	0.11%
Language mismatch	283152	36233	14762	24854	8739	8924	10932	3301
	21.81%	5.81%	6.50%	4.26%	2.85%	1.37%	0.57%	0.52%
Total removed	1097779	131705	25148	211688	102112	42274	102039	29861
	85%	**21%**	**11%**	**36%**	**33%**	6%	5%	5%

We used the toolkit to clean parallel corpora provided in the WMT17[55] and WMT18[56] news MT shared tasks for English ↔ Estonian/Finnish/Latvian. Detailed results of the cleaning process for three of the largest corpora - ParaCrawl, Rapid corpus of EU press releases (Rapid) and European Parliament Proceedings Parallel Corpus (Europarl) - are shown in Table 37.

The results show that ParaCrawl is the most problematic corpus, especially the Estonian part, where 85% had to be removed. The most frequent problems are 1) specified and identified language mismatch; 2) identical sentences appearing on source and target sides; 3) multiple source sentences aligned to the same target sentence; 4) an overwhelming amount of non-alphabetical characters; and 5) multiple target sentences aligned to the same source sentence. All examples of bad sentences in Section 5.3.3 were selected from the removed parts of the English-Estonian ParaCrawl corpus.

[55] Second Conference on Machine Translation - http://statmt.org/wmt17
[56] Third Conference on Machine Translation - http://statmt.org/wmt18

The Rapid corpus had an overall higher quality with only about 25% of parallel sentences removed. For the three languages it exhibited three main defects - 1) duplicate parallel sentences; 2) specified and identified language mismatch; and 3) mismatch in amounts of non-alphabetical symbols between source and target sentences.

Table 38: Detailed results on filtering English-Finnish/Latvian smaller parallel corpora from WMT shared tasks.

	Wiki	**DCEP**	**Leta**	**Books**
	En-Fi	**En-Lv**		
Corpus size	153728	3542280	15671	9577
Unique	0	2277397	454	434
	0.00%	64.29%	2.90%	4.53%
src == tgt	42438	339861	2	4
	27.61%	9.59%	0.01%	0.04%
* sources 1 target	161	12474	2	35
	0.10%	0.35%	0.01%	0.37%
* targets 1 source	339	9450	15	12
	0.22%	0.27%	0.10%	0.13%
> 50% non-alpha	488	31842	0	13
	0.32%	0.90%	0.00%	0.14%
Non-alpha mismatch	4616	38838	946	20
	3.00%	1.10%	6.04%	0.21%
Repeating tokens	38	1242	47	8
	0.02%	0.04%	0.30%	0.08%
Language mismatch	74507	48910	59	1074
	48.47%	1.38%	0.38%	11.21%
Total removed	122587	2760014	1525	1600
	80%	**78%**	**10%**	**17%**

Europarl was by far the cleanest corpus, having only 5-6% of sentences removed by the cleaning toolkit. For all languages, most removed sentences were due to the same two defects as in the Rapid corpus.

We combined and shuffled all three English-Estonian corpora, resulting in 1 012 824 (46.50% of total) sentence parallel corpus for training NMT systems described in the next section. The total amount of English-Finnish parallel sentences was 2 719 104 (82.72% of total) after adding a cleaned version of the Wiki Headlines corpus, and English-Latvian - 1 617 793 (35.85% of total) parallel sentences after adding cleaned versions of LETA translated news, Digital Corpus of European Parliament (DCEP), and Online Books corpora (cleaning details in Table 38). We used the development datasets provided by the WMT shared tasks.

Machine translation

To observe the actual benefit of filtering data for NMT, we trained NMT models using filtered and non-filtered data in both translation directions for the three language pairs. We used Sockeye to train transformer architecture models with 6 encoder and decoder layers, 8

transformer attention heads per layer, word embeddings and hidden layers of size 512, dropout of 0.2, shared subword unit vocabulary of 50 000 tokens, maximum sentence length of 128 symbols, and a batch size of 3072 words. All models were trained until they reached convergence on development data.

Table 39: Translation quality results (BLEU scores) for all translation directions on development data. The best results are marked in bold. The second row shows how much of the initial parallel corpora remained after filtering for each language pair.

	En-Et	**Et-En**	**En-Fi**	**Fi-En**	**En-Lv**	**Lv-En**
Unfiltered	15.45	21.55	**20.07**	**25.25**	21.29	24.12
Corpus after filtering	46.50%		82.72%		35.85%	
Filtered	**15.8**	**21.62**	19.64	25.04	**22.89**	**24.37**
Difference	**+0.35**	**+0.07**	-0.43	-0.21	**+1.60**	**+0.25**

The final NMT system results in Table 39 show that corpora filtering improves NMT quality for Estonian and Latvian systems, but not Finnish. The lack of improvement for Finnish is mainly due to the Europarl being the largest (about $\frac{3}{5}$ of total) and at the same time the cleanest corpus for this language pair. The biggest corpora for Estonian and Latvian - ParaCrawl (about $\frac{3}{5}$ of total) and DCEP (about $\frac{4}{5}$ of total) respectively were also the most problematic ones with 85% and 78% sentences removed respectively.

Figure 66 shows training progression of all 12 NMT systems. Filtered systems are depicted with solid lines, unfiltered ones - with dotted lines, Estonian systems are in light/dark blue colours, Finnish - orange/yellow, and Latvian are in light/dark red colours. The figure shows that the filtered Estonian and Latvian systems are much quicker to learn than the unfiltered ones, but eventually, they converge close to the unfiltered systems. As for the Finnish systems - there is no significant difference between filtered and unfiltered, as at times one is higher than the other or vice versa.

It is generally visible that in both translation directions the filtered systems achieve higher BLEU scores and reach higher quality quicker. For both English ↔ Estonian systems, the unfiltered version catches up to the filtered one later on in the training, but never quite reaches or surpasses it.

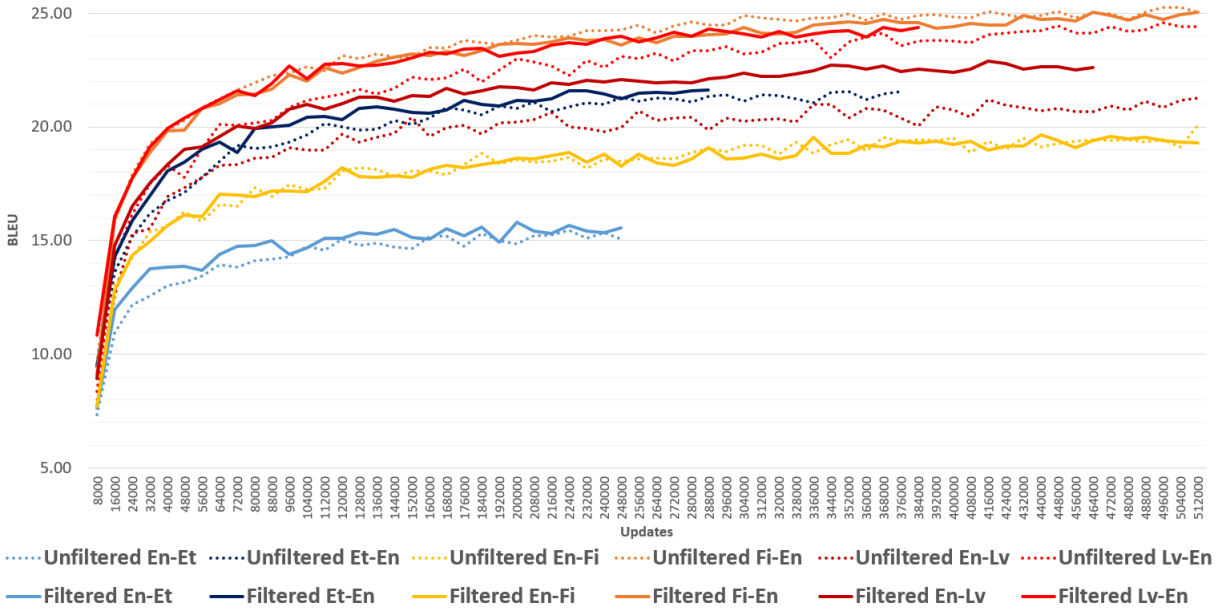


Figure 66: Training progress of English ↔ Estonian/Finnish/Latvian NMT systems.

News translation shared task

To test the full potential of the described methods, the highest-scoring English ↔ Estonian and English ↔ Finnish models were further developed and submitted to the WMT 18 shared task: machine translation of news. The submitted systems were named *tilde-c-nmt-2bt* and *tilde-c-nmt-1bt* respectively. All systems ranked in the top 3-7 by automatic evaluation (BLEU score) out of 17-23 submissions in the constrained track (using only resources that were provided in the shared task).

System and data overview

The English → Estonian and Estonian → English NMT systems (*tilde-c-nmt-2bt*) (Pinnis et al., 2018) are averaged from multiple best NMT models. The models were trained using two sets of back-translated data in a 1-to-1 proportion - one back-translated using a system trained on parallel-only data and the other using an NMT system trained on parallel + the first set of back-translated data. The English → Finnish and Finnish → English NMT systems (tilde-c-nmt-1bt) were trained identically to the Estonian systems, but back-translation was performed only once.

The data processing workflow from Section 5.3.4 was used to clean and prepare data provided in the shared task. The filters were applied to the given parallel sentences, monolingual news sentences before performing back-translation, and both sets of synthetic parallel sentences that resulted from back-translating the monolingual news.

NMT systems and results

To get the highest-quality translation results, we use a multi-pass hybrid approach for training NMT systems. With each trained NMT system, we supplement the parallel training data with an additional set of back-translated for the next system (see Figure 67) resulting in multiple passes of training data during training. The final translations are produced using only

the final NMT system (i.e., NMT3), unlike the multi-pass approach mentioned in Section 2.4.2, in which each input sentence is passed through multiple MT systems.

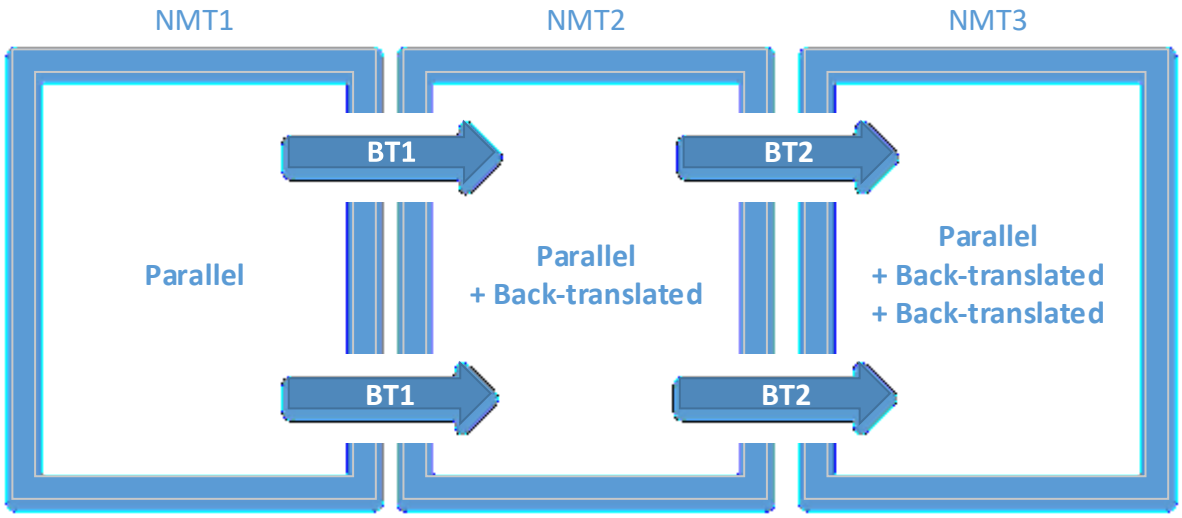


Figure 67: Multi-pass NMT training via double back-translation.

First, we trained baseline models using only filtered parallel datasets (Parallel-only in Figure 68). Then, we back-translated the first batches of monolingual news data and trained intermediate NMT systems (Parallel + First Back-translated). Finally, we used the intermediate NMT systems to back-translate the second batches of monolingual news data and trained final NMT systems (Parallel + Second Back-translated). The final step was performed only for English ↔ Estonian systems. Training progress in Figure 68 shows that the English → Estonian system benefits from the additional data, but the system in the other direction - not so much.

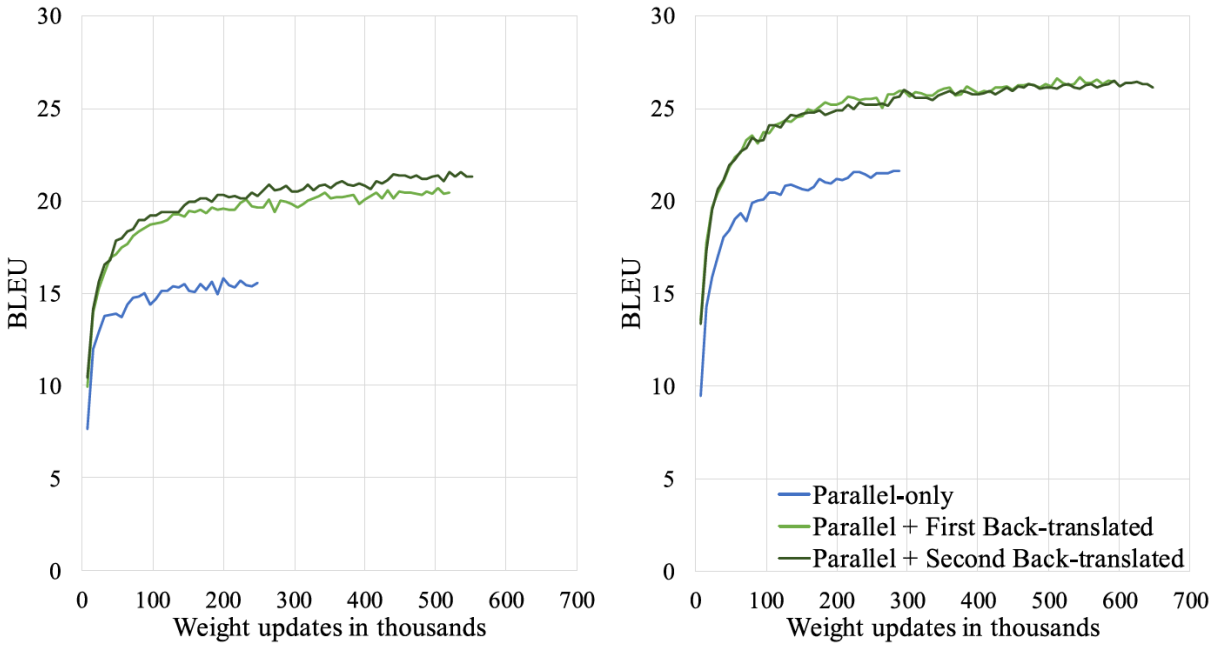


Figure 68: NMT system training progress (BLEU scores on the validation set) for English → Estonian (left) and Estonian → English (right).

For the final translations, we used a post-processing script (Rikters et al., 2017a) to replace consecutive repeating n-grams and repeating n-grams that have a preposition between them (i.e., victim of the victim) with a single n-gram.

The automatic evaluation results of the NMT systems, which were trained on all training datasets, using the SacreBLEU evaluation tool (Post, 2018) are given in Table 40. The results show that the multi-pass hybrid approach for back-translating additional monolingual data turned out to be the most competitive, reaching 3rd place according to automatic evaluation. Table 41 shows the manual evaluation results of the two final submissions to the shared task. The manual evaluation results show that there was no statistically significant difference between the first three Et → En systems and first seven En → Et systems, meaning that both *tilde-c-nmt-2bt* systems were tied for 1st place.

Table 40: Automatic evaluation results of the submitted systems (*tilde-c-nmt-2bt* and *tilde-c-nmt-1bt*) at the WMT18 shared news translation task, only considering constrained systems.

System	**BLEU**	
	Score	**Rank**
Estonian → English	28.0	7 of 23
English → Estonian	23.6	3 of 18
Finnish → English	23.0	5 of 17
English → Finnish	16.9	5 of 18

Table 41: Automatic (BLEU) and human ranking of the submitted systems (*tilde-c-nmt-2bt*) at the WMT18 shared news translation task, only considering primary constrained systems. Human rankings are shown by clusters according to Wilcoxon signed-rank test at p-level p<=0.05,

System	Rank		
	BLEU	Human	
		Cluster	Ave %
Estonian → English	7 of 23	1-7 of 9	3 of 9
English → Estonian	4 of 9	1-3 of 9	3 of 9

5.3.6 Conclusions

This section introduced several types of problematic sentences that can be found in large text corpora and a set of filters that help to remove them in order to train higher quality neural machine translation models using the remaining clean part of the corpora. Results show that in cases where the majority of given parallel corpora are very noisy and there is a small fraction of high-quality corpora, cleaning boosts NMT performance. This is especially evident for translation into morphologically rich languages like Estonian and Latvian.

In this section, we mainly focused on cleaning parallel corpora, but the toolkit is also capable of cleaning monolingual corpora separately. In the MT system training workflow, cleaning monolingual data is useful before performing back-translation of an in-domain corpus, so that only filtered sentences get translated.

6. CONCLUSIONS

The research conducted in this thesis analyses a variety of methods for combining multiple machine translation systems. The research is mostly dedicated to combining statistical and neural machine translation methods in theoretical and practical implementations; it also includes a theoretical overview of system combinations of rule-based and other less popular machine translation paradigms. A majority of this research is focused on translation from and into Latvian, several additional experiments are performed with other morphologically rich languages, such as Czech, Estonian, Finnish, German and Russian.

The author of this thesis carried out the majority of research and development for the described systems and continues to advance their further evolution. MT translations were evaluated using automatic metrics. For most of them, medium or small-scale manual evaluation was also performed. The four main results of the thesis are:

- a method for hybrid MT combination using chunking and neural language models;
- a method for hybrid NMT combination using neural network attention alignments;
- a method for multi-pass incremental training for NMT;
- graphical tools for overviewing and debugging the processes.

The work conducted in this thesis is a substantial contribution to the field of machine translation on a national and international level:

1) the author's initial idea of employing an LM to score translations and choose the best has proven to be useful even after the paradigm shift from SMT to NMT;
2) among noteworthy contributions of this work are also several highest-quality MT systems (Estonian ↔ Russian and Estonian ↔ English) along with details and required tools for reproducibility;
3) the tool for NMT output comprehension using attention alignments not only clearly displays the relation between the source text and the translation, but also is the first and only tool that allows the user quickly locate worse example translations to better understand shortcomings of the MT system in question.

The method for hybrid MT combination via chunking and neural language models has proven to outperform individual similar-quality systems in machine translation of texts with very long sentences. The method demonstrated good performance when working with SMT output, while for NMT output and shorter sentences the chunking method had little to poor influence. Nevertheless, even without chunking part, it is still often very useful to rescore NMT output or choose the best translation using a neural LM.

The hybrid combination method for NMT via neural network attention alignments complies with the emerging technology of neural network MT. It helps distinguish low quality resulting translations from high-quality ones without any references and use them in a hybrid combination setup. Aside from using the method for combining MT output, it has been employed in several MT quality estimation research papers (Ive et al., 2018; Yankovskaya et al., 2018).

The hybrid method of multi-pass incremental training for NMT allowed to be between the top-3 best systems in the annual news translation competition when translating into a morphologically-rich and low-resourced language – Estonian. Since the difference in human evaluation between the top-3 systems was not statistically significant (while it was statistically significant when compared to all other systems), both systems can be considered as the current state-of-the-art for Estonian ↔ English MT. The method has also proven to be competitive for systems translating into Finnish, Latvian and other complex languages and it is anticipated that it will be widely used in this year's WMT shared task for news translation.

The developed graphical tools help to inspect how translations are composed from component systems, and overview results of generated translations to locate better or worse results quickly. Aside from being useful for researchers to help them understand how systems produced specific output, these tools can also help people using public online MT systems, by outlining correlation between source and translation words. The NMT visualization and debugging tool is used to teach students in Charles University, the University of Tartu and in the University of Zurich. It is also currently the most cited publication of the author and has received the most *stars* (27) and *forks* (12) on GitHub, indicating that it is appreciated by the research community.

Since in most cases, when evaluating the three main hybrid methods - chunking; attention-based; and multi-pass, the author observed improvements in automatic and human evaluation of the translations, the hypothesis proposed in the thesis, i.e., it is possible to achieve higher quality translations for the Baltic languages by combining output from multiple different MT systems than produced by each component system individually, can be considered as proven.

BIBLIOGRAPHY

AUTHORS' PUBLICATIONS

Pinnis, M., Rikters, M., Krišlauks, R. (2018, October). Tilde's Machine Translation Systems for WMT2018. In the proceedings of *The 3rd Conference on Machine Translation*.

Rikters, M. (2018b, September) Impact of Corpora Quality on Neural Machine Translation. In *The 8th Conference on Human Language Technologies - the Baltic Perspective (Baltic HLT 2016)*

Rikters, M., Pinnis, M., Rozis, R., Krišlauks, R. (2018b, September) Advancing Estonian Machine Translation. In *The 8th Conference on Human Language Technologies - the Baltic Perspective (Baltic HLT 2016)*

Rikters, M. (2018a, July). Debugging Neural Machine Translations. In *The 13th International Baltic Conference on Databases and Information Systems*

Rikters, M., Pinnis, M., Krišlauks, R. (2018a, May). Training and Adapting Multilingual NMT for Less-resourced and Morphologically Rich Languages. In Proceedings of *The 11th International Conference on Language Resources and Evaluation (LREC 2018)*. Paris, France: European Language Resources Association (ELRA).

Rikters, M., Fishel, M. (2017, September). Confidence Through Attention. In the proceedings of *The 16th Machine Translation Summit*.

Rikters, M., Bojar, O. (2017b, September). Paying Attention to Multi-word Expressions in Neural Machine Translation. In the proceedings of *The 16th Machine Translation Summit*.

Rikters, M., Amrhein, C., Del, M., Fishel, M. (2017, September). C-3MA: Tartu-Riga-Zurich Translation Systems for WMT17. In the proceedings of *The 2nd Conference on Machine Translation*.

Rikters, M., Fishel, M., Bojar, O. (2017a, August). Visualizing Neural Machine Translation Attention and Confidence. In *The Prague Bulletin for Mathematical Linguistics* issue 109.

Rikters, M. (2016d, December). Neural Network Language Models for Candidate Scoring in Hybrid Multi-System Machine Translation. In *CoLing 2016, 6th Workshop on Hybrid Approaches to Translation (HyTra 6)*.

Rikters, M. (2016c, October). Searching for the Best Translation Combination Across All Possible Variants. In *The 7th Conference on Human Language Technologies - the Baltic Perspective (Baltic HLT 2016)* (pp. 92-96).

Rikters, M. (2016b, September). Interactive multi-system machine translation with neural language models. In *Frontiers in Artificial Intelligence and Applications*.

Rikters, M. (2016a, July). K-Translate-Interactive Multi-System Machine Translation. In *The 12th International Baltic Conference on Databases and Information Systems* (pp. 304-318). Springer International Publishing.

Rikters, M., Skadiņa, I. (2016b, May). Syntax-based multi-system machine translation. In N. C. C. Chair) et al. (Eds.), In Proceedings of *The 10th International Conference on Language Resources and Evaluation (LREC 2016)*. Paris, France: European Language Resources Association (ELRA).

Rikters, M., Skadiņa, I. (2016a, April) Combining machine translated sentence chunks from multiple MT systems. In *The 17th International Conference on Intelligent Text Processing and Computational Linguistics (CICLing 2016).*

Rikters, M. (2015, July). Multi-system machine translation using online APIs for English-Latvian. In *ACL-IJCNLP 2015, 4th Workshop on Hybrid Approaches to Translation (HyTra 4)*.

REFERENCES

Ahsan, A., Kolachina, P. (2010). Coupling Statistical Machine Translation with Rule-based Transfer and Generation, *AMTA-The Ninth Conference of the Association for Machine Translation in the Americas*. Denver, Colorado.

Akiba, Y., Taro W., Eiichiro S. (2002). Using language and translation models to select the best among outputs from multiple MT systems. In *Proceedings of the 19th international conference on Computational linguistics-Volume 1*, pp. 1-7. Association for Computational Linguistics.

Baccouche, M., Mamalet, F., Wolf, C., Garcia, C., Baskurt, A. (2011). Sequential Deep Learning for Human Action Recognition. In *Human Behavior Understanding* (pp. 29–39). https://doi.org/10.1007/978-3-642-25446-8

Bach, N., Huang, F., Al-Onaizan, Y. (2011). Goodness: A method for measuring machine translation confidence. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 211–219, Portland, Oregon, USA. Association for Computational Linguistics.

Bahdanau, D., Cho, K., Bengio, Y. (2015). Neural Machine Translation by Jointly Learning to Align and Translate. *Proceedings of the International Conference on Learning Representations (ICLR)*.

Baldwin, T., Kim, S. N. (2010). Multiword expressions. In *Handbook of Natural Language Processing, Second Edition*, pages 267–292. Chapman and Hall/CRC.

Banerjee, S., Lavie, A. (2005). METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization* (Vol. 29, pp. 65-72).

Barrault, L. (2010). MANY: Open source machine translation system combination. *The Prague Bulletin of Mathematical Linguistics 93*: 147-155.

Bengio, Y., Ducharme, R., Vincent, P., Janvin, C. (2003). A Neural Probabilistic Language Model. *The Journal of Machine Learning Research*, *3*(Feb), 1137–1155.

Bertoldi, N., Haddow, B., Fouet, J.-B. (2010). Improved Minimum Error Rate Training in Moses. *Prague Bulletin of Mathematical Linguistics*, *91*(February), 7–16. https://doi.org/10.2478/v10108-009-0011-9

Brown, P. F., Cocke, J., Della Pietra, S., Della Pietra, V. J., Jelinek, F., Mercer, R. L., Roossin, P. S. (1988). A Statistical Approach to French/English Translation. In *RIAO* (pp. 810-829).

Brown, P., Pietra, V. D., Pietra, S. D., Mercer, R. (1993). The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* 19.263–311.

Bojar, O., Federmann, C., Fishel, M., Graham, Y., Haddow, B., Huck, M., Koehn, P., Monz, C. (2018). Findings of the 2018 Conference on Machine Translation (WMT18). *WMT-2018*, *2*, 131–198.

Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Huck, M., Hokamp, C., Koehn, P., Logacheva, V., Monz, C., Negri, M., Post, M., Scarton, C., Specia, L., Turchi, M. (2015). *Findings of the 2015 workshop on statistical machine translation. In Proceedings of the Tenth Workshop on Statistical Machine Translation (pp. 1-46).* Association for Computational Linguistics Lisbon, Portugal.

Bojar, O., Chatterjee, R., Federmann, C., Haddow, B., Huck, M., Hokamp, C., Koehn, P., Logacheva, V., Monz, C., Negri, M., Post, M., Scarton, C., Specia, L., Turchi, M. (2016). Findings of the 2016 conference on machine translation. *In Proceedings of the First Conference on Machine Translation (pp. 131–198).* Association for Computational Linguistics Berlin, Germany.

Bojar, O., Helcl, J., Kocmi, T., Libovický, J., Musil, T. (2017a). Results of the WMT17 Neural MT Training Task. In *Proceedings of the Second Conference on Machine Translation (WMT17)*, Copenhagen, Denmark.

Bojar, O., Chatterjee, R., Federmann, C., Graham, Y., Haddow, B., Huang, S., Huck, M., Koehn, P., Liu, Q., Logacheva, V., Monz, C., Negri, M., Post, M., Rubino, R., Specia, L., Turchi, M (2017b). Findings of the 2017 Conference on Machine Translation (WMT17). In *Proceedings of the Second Conference on Machine Translation (WMT17)*, Copenhagen, Denmark.

Bouamor, D., Semmar, N., Zweigenbaum, P. (2012). Identifying bilingual multi-word expressions for statistical machine translation. In *LREC 2012*, pages 674–679.

Callison-Burch, C., Flournoy, R.S. (2001). A program for automatically selecting the best output from multiple machine translation engines. In *Proceedings of the Machine Translation Summit VIII*, vol. 318.

Castano, M. A., Casacuberta, F., Vidal, E. (1997). Machine translation using neural networks and finite-state models. *Theoretical and Methodological Issues in Machine Translation (TMI)*, 160-167.

Chen, T., Li, M., Li, Y., Lin, M., Wang, N., Wang, M., Xiao, T., Xu, B., Zhang, C., Zhang, Z. (2015). MXNet: A Flexible and Efficient Machine Learning Library for Heterogeneous Distributed Systems. *Neural Information Processing Systems, Workshop on Machine Learning Systems*, 1–6.

Chen, W., Matusov, E., Khadivi, S., Peter, J.-T. (2016). Guided alignment training for topic-aware neural machine translation. In *AMTA 2016*, Vol., page 121.

Chen, Y., Eisele, A. (2012). MultiUN v2: UN Documents with Multilingual Alignments. In proceedings of *LREC 2012*, 2500–2504.

Chen, Y., Eisele, A., Federmann, C., Hasler, E., Jellinghaus, M., Theison, S. (2007). Multi-engine machine translation with an open-source decoder for statistical machine translation. In *Proceedings of the Second Workshop on Statistical Machine Translation* (pp. 193-196). Association for Computational Linguistics.

Cho, K., van Merrienboer, B., Gulcehre, C., Bahdanau, D., Bougares, F., Schwenk, H., Bengio, Y. (2014a). Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 1724–1734.

Cho, K., van Merriënboer, B., Bahdanau, D., Bengio, Y. (2014b). On the Properties of Neural Machine Translation: Encoder–Decoder Approaches. *Syntax, Semantics and Structure in Statistical Translation*, 103.

Cohn, T., Hoang, C. D. V., Vymolova, E., Yao, K., Dyer, C., Haffari, G. (2016). Incorporating structural alignment biases into an attentional neural translation model. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 876–885, San Diego, California. Association for Computational Linguistics.

Crego, J., Kim, J., Klein, G., Rebollo, A., Yang, K., Senellart, J., Akhanov, E., Brunelle, P., Coquard, A., Deng, Y., Enoue, S. (2016). SYSTRAN's Pure Neural Machine Translation Systems. *arXiv preprint arXiv:1610.05540*.

Dandapat, S., Forcada, M. L., Groves, D., Penkale, S., Tinsley, J., Way, A. (2010). OpenMaTrEx: a free/open-source marker-driven example-based machine translation system. In *International Conference on Natural Language Processing* (pp. 121-126). Springer Berlin Heidelberg.

Dauphin, Y. N., Fan, A., Auli, M., Grangier, D. (2016). Language Modeling with Gated Convolutional Networks. *arXiv preprint arXiv:1612.08083*.

Deksne, D., Skadiņš, R., Skadiņa, I. (2008). Dictionary of multiword expressions for translation into highly inflected languages. In *LREC 2008*.

Doddington, G. (2002). Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*. Morgan Kaufmann Publishers Inc.

Dostert, L. E. (1954). The Georgetown-IBM experiment, in *Machine Translation of Languages* (W. N. Locke and A. D. Booth, eds.), Chapter 8, especially pp. 129ff.

Dyer, C., Chahuneau, V., Smith, N. A. (2013). A simple, fast, and effective reparameterization of ibm model 2. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 644-648).

Dyer, C., Weese, J., Setiawan, H., Lopez, A., Ture, F., Eidelman, V., Ganitkevitch, J., Blunsom, P., Resnik, P. (2010). cdec: a decoder, alignment, and learning framework for finite-state and context-free translation models. *Proceedings of the ACL 2010 System Demonstrations*. Association for Computational Linguistics. Retrieved from https://dl.acm.org/citation.cfm?id=1858935

Eisele, A., Federmann, C., Uszkoreit, H., Saint-Amand, H., Kay, M., Jellinghaus, M., Hunsicker, S., Herrmann, T. Chen, Y. (2008). Hybrid architectures for multi-engine machine translation. *Proceedings of Translating and the Computer*, *30*.

Eisele, A., Federmann, C., Saint-Amand, H., Jellinghaus, M., Herrmann, T., Chen, Y. (2008). Using Moses to integrate multiple rule-based machine translation engines into a hybrid system. In *Proceedings of the Third Workshop on Statistical Machine Translation* (pp. 179-182). Association for Computational Linguistics.

Eisele, A. (2005). First steps towards multi-engine machine translation. *Proceedings of the ACL Workshop on Building and Using Parallel Texts*. Association for Computational Linguistics.

Felice, M., Specia, L. (2012). Linguistic features for quality estimation. In *Proceedings of the Seventh Workshop on Statistical Machine Translation*, pages 96–103, Montreal, Canada.

Feng, Y., Liu, Y., Mi, H., Liu, Q., Lü, Y. (2009). Lattice-based system combination for statistical machine translation. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3* (pp. 1105-1113). Association for Computational Linguistics.

Firat, O., Cho, K., Bengio, Y. (2016). Multi-Way, Multilingual Neural Machine Translation with a Shared Attention Mechanism. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies* (pp. 866–875). Stroudsburg, PA, USA: Association for Computational Linguistics. https://doi.org/10.18653/v1/N16-1101

Forcada, M. L., Ginestí-Rosell, M., Nordfalk, J., O'Regan, J., Ortiz-Rojas, S., Pérez-Ortiz, J. A., Sánchez-Martínez, F., Ramírez-Sánchez, G., Tyers, F. M. (2011). Apertium: a free/open-source platform for rule-based machine translation. *Machine translation, 25*(2), 127-144.

Forcada, M. L., Ñeco, R. P. (1997). Recursive hetero-associative memories for translation. In *International Work-Conference on Artificial Neural Networks* (pp. 453-462). Springer Berlin Heidelberg.

Freitag, M., Al-Onaizan, Y. (2016). Fast Domain Adaptation for Neural Machine Translation. Retrieved from http://arxiv.org/abs/1612.06897

Freitag, M., Peter, J., Peitz, S., Feng, M., Ney, H. (2015). Local System Voting Feature for Machine Translation System Combination. In *EMNLP 2015 Tenth Workshop on Statistical Machine Translation (WMT 2015)*, pages 467-476, Lisbon, Portugal.

Galley, M., Graehl, J., Knight, K., Marcu, D., DeNeefe, S., Wang, W., Thayer, I. (2006). Scalable inference and training of context-rich syntactic translation models. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics* (pp. 961-968). Association for Computational Linguistics.

Gamon, M., Aue, A., Smets, M. (2005). Sentence-level MT evaluation without reference translations: Beyond language modeling. In *Proceedings of EAMT*, pp. 103-111.

Gao, J. (2011). *A Quirk Review of Translation Models*.

Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y. N. (2017). Convolutional Sequence to Sequence Learning. *Proceedings of the 34th International Conference on Machine Learning*, in PMLR 70:1243-1252

Gers, F. A., Schmidhuber, J., Cummins, F. (2000). Learning to Forget: Continual Prediction with LSTM. *Neural Computation 12*, no. 10: 2451-2471.

Girardi, C., Bentivogli, L., Farajian, M. A., Federico, M. (2014). MTEQuAl: a Toolkit for Human Assessment of Machine Translation Output. In *COLING 2014*, pages 120–123, Dublin, Ireland, Dublin City University and ACL.

Ha, T.-L., Niehues, J., Waibel, A. (2016). Toward Multilingual Neural Machine Translation with Universal Encoder and Decoder. In *Proceedings of the 13th In- ternational Workshop on Spoken Language Translation (IWSLT 2016)* (p. 16).

Hajič, J., Hric, J., Kuboň, V. (2000). Machine translation of very close languages. In P*roceedings of the sixth conference on Applied natural language processing* (pp. 7-12). Association for Computational Linguistics.

He, X., Toutanova, K. (2009). Joint optimization for machine translation system combination. *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 3-Volume 3*. Association for Computational Linguistics.

Heafield, K. (2011). KenLM: Faster and smaller language model queries. In *Proceedings of the Sixth Workshop on Statistical Machine Translation*, pp. 187-197. Association for Computational Linguistics.

Heafield, K., Lavie, A. (2010). Combining Machine Translation Output with Open Source: The Carnegie Mellon Multi-Engine Machine Translation Scheme. *The Prague Bulletin of Mathematical Linguistics 93*: 27-36.

Helcl, J, Libovický, J. (2017). Neural Monkey: An Open-source Tool for Sequence Learning. *The Prague Bulletin of Mathematical Linguistics*, (107):5–17. ISSN 0032-6585. doi: 10.1515/pralin-2017-0001.

Hieber, F., Domhan, T., Denkowski, M., Vilar, D., Sokolov, A., Clifton, A., Post, M. (2017). Sockeye: A Toolkit for Neural Machine Translation. *Arxiv.Org*.

Hochreiter, S., Schmidhuber, J. (1997). Long short-term memory. In *Neural computation 9*, no. 8: 1735-1780.

Huang, L., Knight, K., Joshi, A. (2006). Statistical syntax-directed translation with extended domain of locality. In *Proceedings of AMTA* (pp. 66-73).

Hutchins, J. (2005). *The history of machine translation in a nutshell*.

Hutchins, J. (2012). Milestones in the history of machine translation. *Science*.

Jayaraman, S., Lavie, A. (2005). Multi-engine machine translation guided by explicit word matching. *Proceedings of the ACL 2005 on Interactive poster and demonstration sessions*. Association for Computational Linguistics.

Jean, S., Cho, K., Memisevic, R., Bengio, Y. (2014). On Using Very Large Target Vocabulary for Neural Machine Translation. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1–10.

Johnson, M., Schuster, M., Le, Q. V., Krikun, M., Wu, Y., Chen, Z., Thorat, N., Viégas, F., Wattenberg, M., Corrado, G., Hughes, M. (2016). Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *arXiv preprint arXiv:1611.04558*.

Johnson, R., King, M., Des Tombe, L. (1985). Eurotra: A multilingual system under development. *Computational Linguistics*, 11(2-3), 155-169.

Jurafsky, D., Martin, J. H. (2014). *Speech and language processing* (Vol. 3). London: Pearson.

Junczys-Dowmunt, M., Dwojak, T., Hoang, H. Is Neural Machine Translation Ready for Deployment? A Case Study on 30 Translation Directions. In *Proceedings of the 9th International Workshop on Spoken Language Translation (IWSLT)*, Seattle, WA, 2016.

Kalchbrenner, N., Blunsom, P. (2013). Recurrent Continuous Translation Models. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 1700–1709.

Kendall, M. (1938). A new measure of rank correlation. *Biometrika*, 30(1-2):81–89.

Khadivi, S., Ney, H. (2005). Automatic Filtering of Bilingual Corpora for Statistical Machine Translation. In Natural Language Processing and Information Systems, 10th International Conference on Applications of Natural Language to Information Systems (Vol. 3513, pp. 263–274). Springer, Berlin, Heidelberg. https://doi.org/10.1007/11428817_24

Kim, Y., Jernite, Y., Sontag, D., Rush, A. M. (2016). Character-Aware Neural Language Models. *Thirtieth AAAI Conference on Artificial Intelligence*.

Klakow, D., Peters, J. (2002). Testing the correlation of word error rate and perplexity. In *Speech Communication, 38(1)*, pp.19-28.

Klein, G., Kim, Y., Deng, Y., Crego, J., Senellart, J., Rush, A. M. (2017). OpenNMT: Open-source Toolkit for Neural Machine Translation. *Proceedings of ACL 2017, System Demonstrations*, 67–72. https://doi.org/10.18653/v1/P17-4012

Klejch, O., Avramidis, E., Burchardt, A., Popel, M. (2015). MT-ComparEval: Graphical evaluation interface for Machine Translation development. In *The Prague Bulletin of Mathematical Linguistics 104*.1: 63-74.

Koehn, P., Hoang, H., Birch, A., Callison-Burch, C., Federico, M., Bertoldi, N., Cowan, B., Shen, W., Moran, C., Zens, R., Dyer, C. (2007). Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions* (pp. 177-180). Association for Computational Linguistics.

Koehn, P., Och, F. J., Marcu, D. (2003). Statistical phrase-based translation. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1* (pp. 48-54). Association for Computational Linguistics.

Krause, B., Lu, L., Murray, I., Renals, S. (2017). Multiplicative LSTM for sequence modelling. In *5th International Conference on Learning Representations* (p. 9). Toulon, France. Retrieved from https://openreview.net/forum?id=SJCS5rXFl

Krizhevsky, A., Sutskever, I., Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. Advances In Neural Information Processing Systems, 1–9.evsky, A., Sutskever, I., & Hinton, G. E. (2012). ImageNet Classification with Deep Convolutional Neural Networks. *Advances In Neural Information Processing Systems*, 1–9.

Li, Z., Callison-Burch, C., Dyer, C., Ganitkevitch, J., Khudanpur, S., Schwartz, L., Thornton, W.N., Weese, J., Zaidan, O. F. (2009). Joshua: An open source toolkit for parsing-based machine translation. In *Proceedings of the Fourth Workshop on Statistical Machine Translation* (pp. 135-139). Association for Computational Linguistics.

Lui, M., Baldwin, T. (2012). langid.py: An off-the-shelf language identification tool. In *Proceedings of the ACL 2012 System Demonstrations* (pp. 25–30). Association for Computational Linguistics. Retrieved from https://dl.acm.org/citation.cfm?id=2390475

Majchráková, D., Dušek, O., Hajič, J., Karčová, A., Garabík, R. (2012). Semi-automatic detection of multiword expressions in the Slovak dependency treebank.

Mäenpää, P., Ranta, A. (1999). The type theory and type checker of GF. In *Colloquium on Principles, Logics, and Implementations of High-Level Programming Languages*. Workshop on Logical Frameworks and Meta-languages, Paris.

Madnani, N., (2011). iBLEU: Interactively debugging and scoring statistical machine translation systems. In *The Fifth IEEE International Conference on Semantic Computing (ICSC)* (pp. 213-214). IEEE.

Mellebeek, B., Owczarzak, K., Van Genabith, J., Way, A. (2006) Multi-engine machine translation by recursive sentence decomposition. In: *AMTA 2006 - 7th Conference of the Association for Machine Translation of the Americas*, 8-12 August 2006, Cambridge, Massachusetts, USA.

Mikolov, T., Deoras, A., Kombrink, S., Burget, L., Černocký, J. (2011). Empirical evaluation and combination of advanced language modeling techniques. In *Twelfth Annual Conference of the International Speech Communication Association*.

Neff, M. S., McCord, M. C. (1990). *Acquiring lexical data from machine-readable dictionary resources for machine translation*. IBM Thomas J. Watson Research Division.

Nagao, M. (1984). A framework of a mechanical translation between Japanese and English by analogy principle. *Artificial and human intelligence*, 351-354.

Nyberg, E. H., Mitamura, T. (1992). The KANT system: Fast, accurate, high-quality translation in practical domains. In *Proceedings of the 14th conference on Computational linguistics - Volume 3* (pp. 1069-1073). Association for Computational Linguistics.

Och, F. J., Ney, H. (2000). A Comparison of Alignment Models for Statistical Machine Translation. In *Proceedings of the 17th conference on Computational linguistics*, pages 1086–1090. Association for Computational Linguistics. ISBN 1-555-55555-1.

Och, F. J., Ney, H. (2003). A systematic comparison of various statistical alignment models. *Computational linguistics*, *29*(1), 19-51.

Och, F. J. (2003). Minimum error rate training in statistical machine translation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics* - Volume 1 (pp. 160-167). Association for Computational Linguistics.

Paikens, P., Rituma, L., Pretkalniņa, L. (2013). Morphological analysis with limited resources: Latvian example. In *Proceedings of the 19th Nordic Conference of Computational Linguistics (NODALIDA 2013)*; May 22-24; 2013; Oslo University; Norway. NEALT Proceedings Series 16, number 085, pages 267–277. Linkoping University Electronic Press.

Pal, S., Srivastava, A., Dandapat, S., Van Genabith, J., Liu, Q., Way, A. (2014). USAAR-DCU Hybrid Machine Translation System for ICON 2014. In *Proceedings of the 11th International Conference on Natural Language Processing, Goa, India*.

Papineni, K., Roukos, S., Ward, T., Zhu, W.-J. (2001). BLEU. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics - ACL '02* (p. 311). Morristown, NJ, USA: Association for Computational Linguistics.

Pascanu, R., Mikolov, T., Bengio, Y. (2013). On the difficulty of training recurrent neural networks. *ICML* (3), 28:1310–1318.

Pecina, P. (2008). Reference data for Czech collocation extraction. In *Proc. of the LREC Workshop Towards a Shared Task for MWEs (MWE 2008)*, pages 11–14.

Peter, J. T., Alkhouli, T., Ney, H., Huck, M., Braune, F., Fraser, A., Tamchyna, A., Bojar, O., Haddow, B., Sennrich, R., Blain, F. (2016). The QT21/HimL Combined Machine Translation System. In *Proceedings of the First Conference on Machine Translation* (pp. 344-355).

Phillips, A. B., Brown, R. D. (2009). Cunei machine translation platform: System description. In *3rd Workshop on Example-Based Machine Translation*.

Pierce, J. R., Carroll, J. B., Hamp, E. P., Hays, D. G., Hockett, C. F., Oettinger, A. G., Perlis, A. (1966). Computers in translation and linguistics (ALPAC report). report 1416. *National Academy of Sciences/National Research Council*.

Pinnis, M. (2013). Context independent term mapper for European languages. In *RANLP*, pages 562–570.

Pinnis, M. (2016). Towards Hybrid Neural Machine Translation for English-Latvian. In *Proceedings of the 7th Conference Human Language Technologies - the Baltic Perspective* (pp. 84-91).

Pinnis, M., Krišlauks, R., Deksne, D., Miks, T. (2017). Neural machine translation for morphologically rich languages with improved sub-word units and synthetic data. In *International Conference on Text, Speech, and Dialogue*. Springer

Post, M. (2018). A Call for Clarity in Reporting BLEU Scores. Retrieved from http://arxiv.org/abs/1804.08771

Randolph, J. J. (2005). Free-Marginal Multirater Kappa (multirater K [free]): An Alternative to Fleiss' Fixed-Marginal Multirater Kappa. *Presented at the Joensuu Learning and Instruction Symposium*, vol. 2005.

Ranta, A., Angelov, K., Hallgren, T. (2010). Tools for multilingual grammar-based translation on the web. In *Proceedings of the ACL 2010 System Demonstrations* (pp. 66-71). Association for Computational Linguistics.

Ramisch, C. (2012). A generic framework for multiword expressions treatment: from acquisition to applications. In *Proceedings of ACL 2012 Student Research Workshop*, pages 61–66. Association for Computational Linguistics.

Richardson, S., Dolan, W., Menezes, A., Pinkham, J. (2001). Achieving commercial-quality translation with example-based methods. In P*roceedings of MT Summit VIII* (pp. 293-298). Santiago De Compostela, Spain.

Rozis, R., Skadinš, R. (2017). Tilde MODEL - Multilingual Open Data for EU Languages. In *Proceedings of the 21st Nordic Conference on Computational Linguistics, NoDaLiDa* (pp. 263–265).

Schwenk, H. (2007). Continuous space language models. *Computer Speech & Language*, 21(3), 492-518.

Sennrich, R., Haddow, B., Birch, A. (2016a). Edinburgh Neural Machine Translation Systems for WMT 16. *Proceedings of the First Conference on Machine Translation (WMT-16)*, 2, 371–376. https://doi.org/10.18653/v1/W16-2323

Sennrich, R., Firat, O., Cho, K., Birch, A., Haddow, B., Hitschler, J., Junczys-Dowmunt, M., Läubli, S., Barone, A.V.M, Mokry, J. (2017). Nematus: a Toolkit for Neural Machine Translation. *EACL 2017*, page 65, 2017.

Sennrich, R., Haddow, B., Birch, A. (2016c). Neural Machine Translation of Rare Words with Subword Units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Sennrich, R., Haddow, B., Birch, A. (2016b). Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics* (Volume 1: Long Papers), pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Skadiņa, I., Aker, A., Giouli, V., Tufis, D., Gaizauskas, R., Mieriņa, M., Mastropavlos, N. (2010). A Collection of Comparable Corpora for Under-resourced Languages. In *Proceedings of the Fourth International Conference Baltic HLT 2010* (pp. 161-168).

Skadiņa I., Levāne-Petrova, K., Rābante, G. (2012). Linguistically Motivated Evaluation of English-Latvian Statistical Machine Translation. In *Human Language Technologies – The Baltic Perspective - Proceedings of the Fifth International Conference Baltic HLT 2012*, IOS Press, Frontiers in Artificial Intelligence and Applications, Vol. 247, pp. 221-229.

Skadiņa, I. (2016). Multi-word expressions in English-Latvian. In *Human Language Technologies – The Baltic Perspective: Proceedings of the Seventh International Conference Baltic HLT 2016*, volume 289, page 97. IOS Press.

Skadiņš, R., Goba, K., Šics, V. (2010) Improving SMT for Baltic Languages with Factored Models. *Proceedings of the Fourth International Conference (Baltic HLT 2010)*, Frontiers in Artificial Intelligence and Applications, Vol. 2192., 125-132.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., Makhoul, J. (2006). A study of translation edit rate with targeted human annotation. In *Proceedings of association for machine translation in the Americas* (Vol. 200, No. 6).

Son, L. H., Allauzen, A., Yvon, F. (2012) Continuous Space Translation Models with Neural Networks. *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, pp. 39–48.

Steinberger, R., Pouliquen, B., Widiger, A., Ignat, C., Erjavec, T., Tufis, D., Varga, D. (2006). The JRC-Acquis: A multilingual aligned parallel corpus with 20+ languages. In *Proceedings of LREC 2006*.

Steinberger, R., Eisele, A., Klocek, S., Pilos, S., Schlüter, P. (2012). DGT-TM: A freely available translation memory in 22 languages. In *Proceedings of LREC 2012*.

Sutskever, I., Vinyals, O., Le, Q. V. (2014). Sequence to sequence learning with neural networks. In *Advances in neural information processing systems* (pp. 3104-3112).

Šmite, D. (2013). Methods for literature reviews, Research design in brief, *lecture slides*, University of Latvia.

Tang, Y., Meng, F., Lu, Z., Li, H., Yu, P. L. H. (2016). Neural machine translation with external phrase memory. *CoRR*, abs/1606.01792.

Theano Development Team. (2016). Theano: A Python framework for fast computation of mathematical expressions. *ArXiv E-Prints*, 19. https://doi.org/1605.02688

Thurmair, G. (2009). Comparing different architectures of hybrid machine translation systems. In P*roceedings of the 12th Machine Translation Summit*, August 26-30 2009, Ottawa, Canada; pp.340-347.

Tiedemann, J. (2009). News from OPUS — A Collection of Multilingual Parallel Corpora with Tools and Interfaces. *Recent Advances in Natural Language Processing*, *V*, 237–248.

Toma, P. (1977). Systran as a multilingual machine translation system. In *Proceedings of the Third European Congress on Information Systems and Networks, Overcoming the language barrier* (pp. 569-581).

Tu, Z., Lu, Z., Liu, Y., Liu, X., Li, H. (2016). Coverage-based Neural Machine Translation. *arXiv preprint* arXiv:1601.04811.

Vasiļjevs, A., Skadiņš, R., Tiedemann, J. (2012). LetsMT!: A Cloud-Based Platform for Do-It-Yourself Machine Translation. In *Proceedings of the ACL 2012 System Demonstrations*, 43–48, Jeju Island, Korea: Association for Computational Linguistics.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Polosukhin, I. (2017). Attention is All you Need. In I. G. Garnett, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, & R. Garnett (Eds.), *Advances in Neural Information Processing Systems 30* (pp. 5998–6008). Curran Associates, Inc. https://doi.org/10.1017/S0140525X16001837

Vaswani, A., Zhao, Y., Fossum, V., Chiang, D. (2013). Decoding with Large-Scale Neural Language Models Improves Translation. *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP 2013)*, (October), 1387–1392.

Vauquois, B. (1968). A survey of formal grammars and algorithms for recognition and transformation in mechanical translation. In *Ifip congress (2)* (Vol. 68, pp. 1114-1122).

Vilar, D., Stein, D., Huck, M., Ney, H. (2010). Jane: Open source hierarchical translation, extended with reordering and lexicon models. In *Proceedings of the Joint Fifth Workshop on Statistical Machine Translation and MetricsMATR* (pp. 262-270). Association for Computational Linguistics.

Wolk, K. (2015). Noisy-parallel and comparable corpora filtering methodology for the extraction of bi-lingual equivalent data at sentence level. *Computer Science @BULLET Computer Science*, *16*(162), 169–184. https://doi.org/10.7494/csci.2015.16.2.169

Wu, Y., Schuster, M., Chen, Z., Le, Q.V., Norouzi, M., Macherey, W., Krikun, M., Cao, Y., Gao, Q., Macherey, K., Klingner, J., Shah, A., Johnson, M., Liu, X., Kaiser, L., Gouws, S., Kato, Y., Kudo, T., Kazawa, H., Stevens, K., Kurian, G., Patil, N., Wang, W., Young, C., Smith, J., Riesa, J., Rudnick, A., Vinyals, O., Corrado, G., Hughes, M., Dean, J. (2016). Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.08144. URL http://arxiv.org/abs/1609.08144.

Xuan, H. W., W. Li, Tang, G. Y. (2012). An Advanced Review of Hybrid Machine Translation (HMT). In *Procedia Engineering 29*: 3017-3022.

Xu, H., Koehn, P. (2017). Zipporah: a Fast and Scalable Data Cleaning System for Noisy Web-Crawled Parallel Corpora. *Emnlp*, 2935–2940. Retrieved from http://www.aclweb.org/anthology/D17-1319%0Ahttp://aclweb.org/anthology/D17-1318

Zeiler, M.D., 2012. ADADELTA: an adaptive learning rate method. *arXiv preprint* arXiv:1212.5701.

Zeman, D., Fishel, M., Berka, J., Bojar, O. (2011). Addicter: What Is Wrong with My Translations? *The Prague Bulletin of Mathematical Linguistics*, 96:79–88.

Zhang, H., Gildea, D. (2008). Efficient Multi-Pass Decoding for Synchronous Context Free Grammars. In *ACL* (pp. 209-217).

Zhao, Y., He, X. (2009). Using n-gram based features for machine translation system combination. Proceedings of Human Language Technologies: *The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. Association for Computational Linguistics.

Zhou, L., Hu, W., Zhang, J., Zong, C. (2017). Neural System Combination for Machine Translation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)* (pp. 378–384). Stroudsburg, PA, USA: Association for Computational Linguistics.