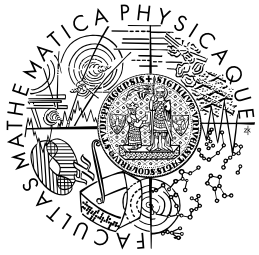


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

MASTER THESIS

Bohdan Ihnatchenko

Multi-Target Machine Translation

Institute of Formal and Applied Linguistics

Supervisor of the master thesis: doc. RNDr. Bojar Ondřej, Ph.D.
Study programme: Computer Science
Study branch: Artificial Intelligence

Prague 2020

I declare that I carried out this master thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

Dedication.

Title: Multi-Target Machine Translation

Author: Bohdan Ihnatchenko

Institute: Institute of Formal and Applied Linguistics

Supervisor: doc. RNDr. Bojar Ondřej, Ph.D., Institute of Formal and Applied Linguistics

Abstract:

In international and highly-multilingual environments, it often happens, that a talk, a document, or any other input, needs to be translated into a massive number of other languages. However, it is not always an option to have a distinct system for each possible language pair due to the fact that training and operating such kind of translation systems is computationally demanding.

Combining multiple target languages into one translation model usually causes a decrease in quality of output for each its translation direction. In this thesis, we experiment with combinations of target languages to see, if a specific grouping of them can lead to better results than just randomly selecting target languages.

We build upon a recent research on training a multilingual Transformer model without any change to its architecture: adding a target language tag to the source sentence.

We trained a large number of bilingual and multilingual Transformer models and evaluated them on multiple test sets from different domains. We found that in most of the cases grouping related target languages into one model caused a better performance compared to models with randomly selected languages. However, we also found that a domain of the test set, as well as domains of data sampled into the training set, usually have a more significant effect on improving or deterioration of multilingual model's translation quality compared to the bilingual one.

Keywords: Neural machine translation, Multi-target MT, linguistic relatedness

Contents

Introduction

With increasing availability of computational resources and enormous amount of publicly available corpora it is now possible to obtain a machine translation (MT) system which produces translations of acceptable quality. But in the use cases similar to conferences, where one speech is translated into multiple target languages, the same amount of models needs to be deployed. Another option is to use multilingual MT system for all needed languages together, which may lead to a decreased quality of translations.

1. TO EDIT: Background

In this chapter, we go through the theory of methods which are used in the work.

1.1 TO EDIT: History of machine translation

The history of machine translation (MT) started with the 'Translation' memorandum by Weaver [1955], where the first goals and ideas were proposed. Only in late 1988 the idea of statistical machine translation was brought back to the research world by IBM's research center Brown et al. [1988].

After years of domination of a statistical approach to MT, first neural machine translation (NMT) systems were introduced: convolutional neural network (CNN) based models [Kalchbrenner and Blunsom, 2013] and sequence-to-sequence models (Sutskever et al., 2014; Cho et al., 2014). Later, recurrent neural network (RNN) based models were improved by introducing an *attention* mechanism [Bahdanau et al., 2014].

Various window lengths in CNN architectures allowed to capture long range relations as well as short range ones; still the range was limited by the maximum window length. In RNN-like architectures, long short-term memory (LSTM) and gated recurrent unit (GRU) cells were used, as their structure allowed to pass the internal state on longer distances due to selective forgetting.

1.2 TO EDIT: Transformer model

Introduced in Vaswani et al. [2017], Transformer model is used as a base for numerous state-of-the-art systems as can be seen for example in WMT18 [Bojar et al., 2018] and WMT19 [Barrault et al., 2019] results.

Transformer model uses the *self attention* mechanism to encode contextual information in each word position. *Position encoding* allows passing the position information without explicit sequential connections as in RNN. The architecture of the *Transformer* model is shown on Figure 1.1. For tasks involving very long sequences authors also proposed *restricted* self-attention, which considers only a neighborhood of size r in the input sequence centered around the respective output position. As was stated by *Transformer*'s authors, there are three main points why self-attention mechanism should be preferred (which are compared with RNN and CNN in Table 1.1):

- total computational complexity per layer;
- the amount of computation that can be parallelized;
- the path length between long-range dependencies in the network.

Layer type	Complexity per layer	Sequential operations	Maximum path length
Self-Attention	$O(n^2 \cdot d)$	$O(1)$	$O(1)$
Recurrent	$O(n \cdot d^2)$	$O(n)$	$O(n)$
Convolutional	$O(k \cdot n \cdot d^2)$	$O(1)$	$O(log_k(n))$
Self-Attention (restricted)	$O(r \cdot n \cdot d)$	$O(1)$	$O(n/r)$

Table 1.1: **Maximum path lengths, per-layer complexity and minimum number of sequential operations for different layer types.** n is the sequence length, d is the representation dimension, k is the kernel size of convolutions and r the size of the neighborhood in restricted self-attention.

1.3 TO EDIT: Preprocessing: BPE

The words that are rare or belong to some other domain, than the domain of the training data the model has seen during the training, are called OOV words. The byte pair encoding (BPE) algorithm [Gage, 1994] of data compression was successfully adopted for approaching this issue by Sennrich et al. [2016]. Its idea is that the less frequent words are segmented into subword units, while the more frequently occuring ones remain to be represented as a single tokens (see Example 1.3.1).

In short, the algorithm is the following:

- Specify the vocabulary size
- Parse the training set and collect all unique characters

Source: You can definitely ride a bicycle or a motorcycle in Zaporizhzhia.
Segmented: You can defi@@ nitely ride a bi@@ cycle or a motor@@ cycle in Z@@ a@@ p@@ o@@ r@@ i@@ z@@ h@@ z@@ h@@ i@@ a@@ .

Example 1.3.1. Suppose, this sentence appears in an English corpus with rare proper names in other languages or even scripts. Then it might look after the segmentation in the following way: the frequent words left the same, some were split by subwords, a rare personal noun is even splitted by characters. The '@@' at the end of a subword is added when it is not the end of a word.

1.4 Translation evaluation

1.4.1 History

In 1966 first machine translation evaluation methods were proposed by the Automatic Language Processing Advisory Committee (ALPAC). The proposed metrics were "intelligibility" and "fidelity" [ALPAC, 1966, p 67]. Trained human raters were needed to measure the metrics.

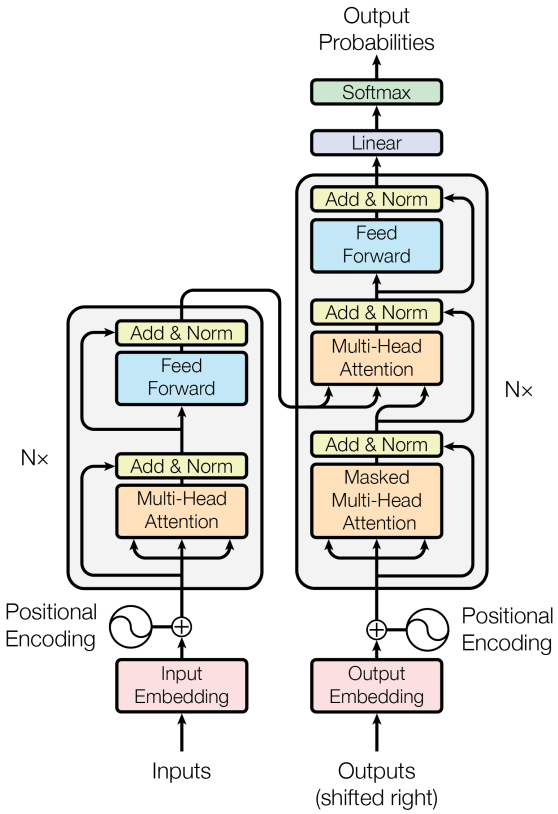


Figure 1.1: **Transformer model architecture.**

Later, after years of using manual evaluation, automatical evaluation metrics were created, such as word error rate (WER) Su et al. [1992], translation edit rate (TER) Snover et al. [2006], etc. Nowadays the most popular metric is bilingual evaluation understudy (BLEU) which is described in the next section.

1.4.2 BLEU - bilingual evaluation understudy

In Papineni et al. [2002], a novel method of automatic machine translation evaluation was introduced – bilingual evaluation understudy (BLEU). Its advantages are the high speed and low cost of evaluation, language independence and high correlation with judgements of highly skilled human raters.

Shortly, BLEU score consists of modified n-gram precision scores corrected by brevity penalty, which ensures the produced translation lenght is close to the reference one. BLEU score is computed for the whole test corpus.

Modified *n*-gram precision score

The main element of the metric is the *precision* measure. It is computed in the following way: the number of candidate translation words (unigrams) that are present in any reference translation is divided by the total number of words in the candidate translation. This approach leads to overrating candidate translation which consists of only one or a couple of words that occur in reference translations, as can be seen in Example 1.4.1.

Intuitively, after a word from the reference translation has occurred, it should not be considered in the calculation anymore. This intuition is formalized as the

modified unigram precision. It is computed in the following way:

1. count the maximum number of occurrences of a word in any reference translation;
2. clip the total count of every candidate word by the maximum reference count;
3. sum the clipped counts;
4. divide this sum by the total (not clipped) number of candidate words.

As a result, the sentence which may receive a high precision score will receive a more realistic evaluation measured by modified precision score, as can be seen in Example 1.4.1.

Candidate: of of of of of of of of of of
Reference: London is the capital of England and of the United Kingdom of Great Britain and Northern Ireland.
Precision: $10/10 = 1.0$
Modified unigram precision: $3/10 = 0.3$

Example 1.4.1. Precision and modified unigram precision. Similarly is computed modified n-gram precision score for any n, but n-gram counts are collected instead.

Sentence length

A produced translation should not be too short or too long. Any excessive length will be automatically penalized by the modified precision: extra words will not get the credit. However, too short outputs could game the metric. This balance is usually achieved by pairing precision with *recall*. However, in BLEU, multiple reference sentences can be used for one source sentence, so recalling all possible translations from every reference is not what is needed. BLEU authors introduced the *brevity penalty* factor for this purpose. In short, it penalizes produced translations that are shorter than the references. To avoid excessive penalization of shorter sentences, the *brevity penalty* is computed on the whole translated set. In the equation below, r is the test corpus' effective reference length and c is the total length of the candidate translation corpus. To compute r the best match lengths for each candidate sentence in the corpus are added.

$$BP = \begin{cases} 1, & \text{if } c > r; \\ e^{1-r/c}, & \text{otherwise} \end{cases} \tag{1.1}$$

Equation

Combining all the above, the metric works in this way (Equation (1.2)):

1. compute the geometric mean of the modified n-gram precisions (p_n), using n-grams up to length N and positive weights (w_n) summing to one.

2. compute *brevity penalty* as in Equation (1.1).

3. multiply results of steps 1. and 2.

The authors proposed to use $N = 4$ and uniform weights $w_i = 1/4$.

The metric value is in the range from 0 to 1. However, popular implementations such as SacreBLEU [Post, 2018] report it in percentage points from 0 to 100.

$$BLEU = BP \cdot \exp\left(\sum_{n=1}^{N} w_n \log p_n\right) \tag{1.2}$$

1.5 Multi-target machine translation

In this section, we have a closer look at the area of MT, which this thesis is dedicated to – multi-target MT. First, we talk about multi-lingual MT in general: multi-way, multi-source and multi-target. Later we describe the specific approach from multi-lingual MT – complete sharing of model parameters, which we rely on in this work.

1.5.1 Multi-lingual machine translation

With constant improvement of neural MT systems performance, researchers started to experiment with incorporating multiple source languages, target languages, or both into one model, and the results are promising:

- having L1→L2 and L2→L3 corpora, of which only the two separate pairs L1-L2 and L2-L3 are parallel, allows to train a model that can produce L1→L3 translation of decent quality [Johnson et al., 2017];

- having a high-resource L1 and low-resource L2 from the same language group helps increase Source→L2 translation quality with pretraining on Source→L1 data [Dabre et al., 2017].

Even if the concept of combining multiple languages into one model and possible outcomes of such combination may seem intuitive, there exist multiple approaches of how exactly this might be performed. As for current time, Dabre et al. [2019] categorizes MNMT (multi-lingual neural machine translation) in the following way (Figure 1.2):

Multi-Way Translation. The goal is constructing a single NMT system for one-to-many, many-to-one or many-to-many translation using parallel corpora for more than one language pair.

Low or Zero-Resource Translation. Large amounts of parallel texts of high quality are available for most of European languages. However, it is not true for most of other languages in the world. Three main directions have been studied for these cases. *Transfer learning*: Transferring translation knowledge from a high-resource language pair to improve the translation of a low-resource language pair. *Pivot translation*: Using a high-resource language (usually English) as a pivot to translate between a language pair. *Zero-shot translation*: Translating between language pairs without parallel corpora.

Multi-Source Translation. Having the source side represented by multiple languages may increase translation quality in general or help to remove ambiguities present in one or another source language (e.g. cases, noun genders, etc.).

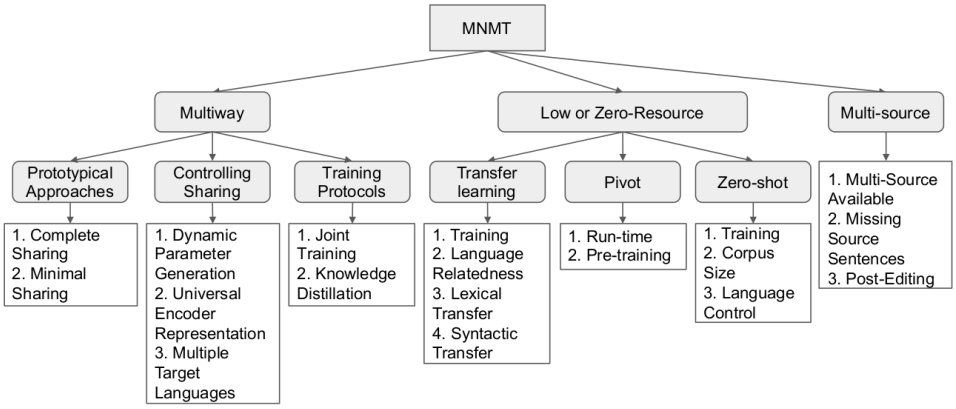


Figure 1.2: **MNMT research categorized.** According to resource scenarios and underlying modeling principles. By Dabre et al. [2019]

1.5.2 Massively multi-lingual machine translation with complete sharing

Johnson et al. [2017] proposed a way to build a multi-lingual machine translation model without any changes to the *Transformer* architecture. The only change was performed on the input data. To make the *Transformer* model process multi-lingual data, they added the desired target language tag to the source sentence. This way they achieved a *complete sharing* of parameters and subword vocabulary among all the source and target languages, i.e. the same encoder with the same parameters is used for every source language, the same decoder is used for every target direction, and the same vocabulary is used for both encoding and decoding of any input and output.

For example, the following En→Cz sentence pair:

Hello world! → Ahoj světe!

is modified to:

<2cs> Hello world! → Ahoj světe!

With the given method, it is possible to produce translations in multiple languages using the same model just by altering the prepended target language tag. It was also demonstrated that this method slightly improves translation quality for low resource languages when compared to monolingual translation model.

In Aharoni et al. [2019], models with up to 103 languages were tested. English centric in-house dataset was used to train En→{Any} and {Any}→En multilingual models. The average number of examples per language pair is 940k: for 13 out of the 102 pairs there were less than one million examples available.

In one of the experiments, they varied the number of languages in the model and measured the model's performance on the specified set of translation directions. They started with a 5-to-5 model with English, Arabic, French, Russian, and Ukrainian selected. Given that the dataset was English-centric, they trained the 5-to-5 model to translate in En→{Ar, Fr, Ru, Uk} and {Ar, Fr, Ru, Uk}→En directions. Therefore, name 5-to-5 refers to the model's ability to accept source sentence in 5 languages and to translate into the same five languages. For 25-to-25 model they added 20 more randomly selected languages to the 5-to-5 setup. In all the cases they trained a large Transformer model with 473.7M parameters. As can be seen in Table 1.2, the quality of translation is significantly worse when a model is trained to translate more languages.

	En-Ar	En-Fr	En-Ru	En-Uk
5-to-5	**12.42**	**37.30**	**24.86**	16.48
25-to-25	11.77	36.79	23.24	**17.17**
50-to-50	11.65	35.83	21.95	15.32
75-to-75	10.69	34.35	20.70	14.59
103-to-103	10.25	34.42	19.90	13.89

Table 1.2: **BLEU scores for translation in one direction (part of Table 7 from [Aharoni et al., 2019]).** We see that the model trained on 5-to-5 English centric dataset (English to any and any to English) scores 12.42 BLEU for English-Arabic test set and the performance decreases as more languages are added to the model. Every language from 5 languages of 5-to-5 data set is included into 25-to-25 set, as well as every language from 25-to-25 data set is included into 50-to-50 and so forth.

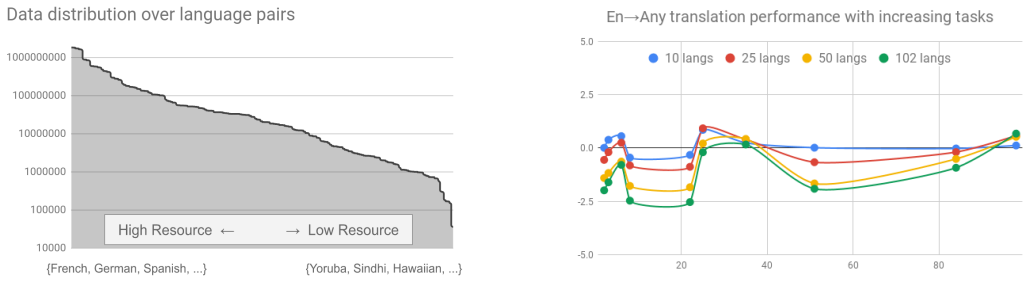


Figure 1.3: **Translation performance for 102 languages from Arivazhagan et al. [2019].** Axis X is shared between left and right plot. On axis X, there are languages sorted by the amount of training data. Left: amount of training data (axis Y) for a language. Right (best viewed in color): Effect of increasing the number of languages on the translation quality. On the axis X the languages are sorted the same way as on the left plot. The points visualized are the 10 languages that are present in all setups from En ↔ 10 to En ↔ 102.

1.6 Conclusion

In this chapter we introduced theoretical and historical background for this work. Firstly, we took a short walk through the history of machine translation. Then we described the most used type of NMT models – self-attention *Transformer* model. After that we went over the history of translation evaluation in general and the most used method of automatic evaluation – BLEU – in particular. In the end, multi-lingual neural machine translation was reviewed with more detailed view into 'complete sharing' scheme.

2. Experiment setup

In this chapter, we describe the data used for experiments, training setup and experiments that were run to answer the questions asked in this thesis.

2.1 XXX TODO: Questions and constraints

Constraints:

Translation quality for multi-lingual system is better or insignificantly worse than for mono-lingual one-to-one tranlsation system.

Maximum possible target languages are combined in one model.

Questions:

How, *on average*, does adding one more randomly selected target language to the multitarget model affect its En→De performance?

How is it different if we add a linguistically similar, not a randomly selected language?

How does adding one more language from the same language family or group *on average* affect translation performance for a selected language pair (e.g. En→De)?

2.2 Experiments

2.2.1 Starting point

The approach described in Section 1.5.2 with combining multiple translation directions into the standart *Transformer* model can be also used to train just multi-target models, i.e. with one source language and multiple target languages. The following papers (Arivazhagan et al. [2019], Aharoni et al. [2019]), which further develop the approach, describe and try many different interesting cases. However, in each setting there is usually only one model of each kind considered. For example, when in Aharoni et al. [2019] compares 5-to-5, 25-to-25, 50-to-50, etc. models, there is only one 5-to-5 model, one 25-to-25, etc.

To conduct our experiments, we use this approach, but with the following differences:

- We fix English as a source language, as we are exploring the multi-target experiments only, .
- For every translation direction and every setting we train multiple models. E.g. for the En→De translation direction and 1-to-5 setting there are couple of En→{De + 4 randomly selected targets} models.
- We use only up to 5 target languages in the model because of:
  - limited resources;
  - our selected datasets (which will be described in the next section) do not contain more than 5-6 languages of the same language group.

2.2.2 Proposed experiments

Bilingual baseline

Bilingual models. The purpose is to have a reference point to be able to reason how does every additional target language affects the model's performance. Siddhant et al. [2019] shows that using target language tags results in the same model efficiency as separately encoding the target. Therefore, we use target tags in this setting too, so that we can use the same training pipeline.

Multi-lingual baselines (RANDOM)

Multilingual models with a random set of target languages. The purpouse is twofold: to show BLEU score decrease with increasing number of target languages and to serve as a baseline for multitarget models with target languages grouped by in non-random way, e.g. by language group or linguistic similarity.

Group by language group (SIMILAR)

Multilingual models with a set of target languages from the same language group. Due to shared parts of vocabulary and linguistic properties we expect to see better results than for multi-lingual baselines. Ideally the results could be comparable with bilingual baselines.

2.3 Dataset(s)

2.3.1 TO EDIT: English to 36 languages

To observe effects of linguistic similarity of target languages, it is important to examine enough possible variations of those. The OPUS dataset (Tiedemann [2012]) is an open and free collection of texts that covers more than 90 languages with data from several domains.[1]

For our experiments the source language is English only.

Given the list of target languages in this dataset (see full list in Appendix A.1.2), we decided to select these two groups of languages for the SIMILAR experiment:

- Germanic group: da, de, is, no, nl, sv.
- Slavic with cyrillic script: bg, mk, ru, uk.

We made use of the sampling and splitting of the data created by the ELITR project.[2] For each of the language pairs and each sub-dataset the data was split to training, validation and testing sets. For each of the two latter sets, 2000 random sentences were selected and the rest of the data remained for the training set. In cases where the sub-dataset contained less than 16000 sentence pairs, no data went to the validation set. Later, for each language pair there were 1000000 sentence pairs sampled from all training sub-sets. XXX FIX: To be more explicit that the sampling is directed towards certain domains. This is

[1] Available at `http://opus.nlpl.eu/`

[2] `https://elitr.eu/wp-content/uploads/2019/07/D11.FINAL_.pdf`

somewhat unclear. Firstly, if available, the sentences were taken from Europarl, then EUbooks, OpenSubtitles, and then all remaining sub-datasets. The same procedure was used to sample XXX TODO: check x000 of validation set sentences per each language pair. The test sets were left separate, so that the result on each domain would be observable.

Later we found an overlap in the source side of different language pairs. Although this would not directly lead to unfair increase of the test score, such sentence pairs were removed from the training sets. This filtering decreased the number of sentence pairs to 0.85-0.95 millions per language pair. XXX TODO: describe the figure with stats XXX TODO: describe the table with groups of sub-datasets XXX TODO: group 4: open folder, save file XXX TODO: group 5: Tanzil - completely different domain, Books of 18th cent.- dated vocabulary Wikipedia - automaticaly aligned sentences.

group	subdataset names	description
1	Europarl/vx, DGT, MultiUN, EUbookshop, JRC-Acquis, ECB, EMEA	Proceedings and documents from Europarl, UN, etc.
2	NewsCommentary, GlobalVoices, WMT-News	News articles and commentaries
3	OpenSubtitles, Tatoeba	Short sentences, human speech, general domain
4	OpenOffice, PHP, KDE4, Gnome	Software documentation or interface elements
5	Tanzil, Books, Wikipedia	Other

Table 2.1: **Groups of subdatasets in OPUS.**

2.3.2 TO EDIT: UN parallel corpus: English to 5 languages

For an additional experiment about adding non-related target languages into the mix, we used 'The UN Parallel Corpus v1.0' [Ziemski et al., 2016]. It consists of more that 11 millions sentence 6-tuples in six UN languages, i.e. every English sentece has a translation into Arabic (ar), Chinese (zh), French (fr), Russian (ru), and Spanish (es).

First, we splitted data into training, validation and test sets. We selected 30000 sentence tuples into the test set, then from the remaining data we took 30000 sentence tuples into the validation set. From the remaining data we removed all sentence tuples that overlapped the test set in the same way as in Section 2.3.1, which gave us 10987284 sentence tuples.

After that, we divided each 6-tuple into five sentence pairs, where English was used as a source and every other language as a target. We also added the target language tag as in Section 2.3.1.

As a result, we received a dataset in the same format as the en-to-36. The main differences are:

- the size: training data for each translation direction contains ten times more sentence pairs;

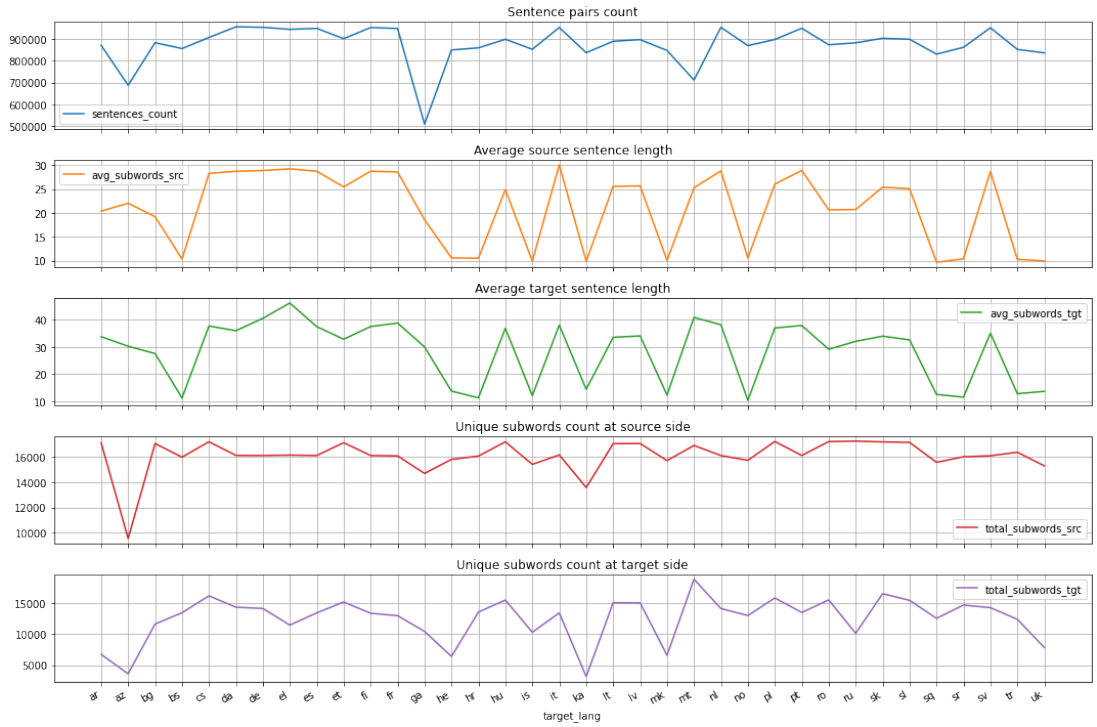


Figure 2.1: **Training data language statistics.** Languages are on the X axis sorted as in Appendix A.1.2. From top to bottom: total number of sentence pairs in training set per language, average number of subwords per sentence on the source side, the same on the target side, total number of unique subwords for this target language on the source side, the same on the target side.

- variety: the training, validation, and test sets consist of data from the same domain;
- only one test set, due to the previous point.

When using the same model as for en-to-36, these differences may allow us to see how the model's translation quality changes if we add more target languages in the mix. In the following text the 'en-to-5' notation refers to this dataset. We specify explicitly, if this dataset is used in a experiment; otherwise the en-to-36 dataset is used.

2.4 Method

In this section we describe how the models are trained, which metrics are collected and how are they analyzed.

2.4.1 XXX TODO: Training tasks

XXX TODO: what is a task, multilingual task, sampling tasks for RANDOM and SIMILAR XXX TODO: introduce model En→{L1, L2, ...} notation

2.4.2 TO EDIT: Data preprocessing and selection

Preprocessing

Both en-to-36 and en-to-5 datasets are preprocessed before any sampling in this way:

- the vocabulary is created using the BPE on both target and source parts of the training set only;
- using the vocabulary, the train set, the validation set, and the test set are segmented and then stored in this preprocessed form.

Note, that due to the fact that every sentence starts from a target tag, these target tags are not splitted by the BPE.

Dataset subsampling

Both training sets from Section 2.3 contain data for many language pairs, but in order to train some specific model, we do not need them all. The same holds for the validation sets.

Therefore, for each specific model En→{L1, L2, ...} we need to subsample the *En→{L1, L2, ...} training set* from *the whole training set* (e.g. en-to-36), the *En→{L1, L2, ...} validation set*.

Training set

To prepare a training set for a specific model, we need to select sentence pairs with relevant source languages from *the whole training set*. This is done similarly for both en-to-5 and en-to-36 datasets. Further, the notation '*En→{L1, L2, ...} training set*' refers to a subsampled in this way training set.

For example, let us take En→{Fr, De} setup, which means that the model to be trained should take a source sentence in English and produce translation either in French or in German. The language of model's output depends on the target tag at the beginning of the input sentence, i.e. <2fr> tag in source sentence leads to French target.

To train such a model, only related sentence pairs are subsampled from the whole training set. In this case, from *the whole training set* we select only those sentence pairs, which source side starts with tags <2fr> or <2de>. Such a subsampled dataset is then used to train this model, and refered to as the *En→{Fr, De} training set*.

Validation set

For any model the validation set is constructed from the big validation set by selecting only relevant sentence pairs in the same way as the training set, i.e. pairs with the target in one of the examined languages. For the example setup from above, En→{De,Fr}, the validation set consists of an equal amount of En→De and En→Fr sentence pairs. E.g. if in the complete validation set there are 1000 sentence pairs for each of possible target languages, then for En→{De,Fr} model the validation set will contain 2000 sentence pairs, and for En→{De,Es,Fr} it will contain 3000 sentence pairs.

Test set

For experiments with en-to-5 dataset (Section 2.3.2) the test sets are created in the same way as validation sets. For en-to-36 dataset (Section 2.3.1) the test set is divided on subsets by the source dataset. It means, that each of source datasets (like OpenSubtitles/v11, Europarl/v7, etc.) there exists a separate test set.

XXX TODO: Why is this needed? In the experiments proposed above the expected number of models to be trained is quite big. First of all, there should be 36 models for *mono-target baseline* for En→36 dataset. For *multi-target random* experiment the number is much bigger. For example, let us consider a case with En→3 models - each model translates from English to 3 target languages. Specifying that each of 36 target languages from En→36 dataset should appear at least in 3 En→3 models, series of random generation of En→3 setups gave the smallest amount of such setups equal to 44. For En→5 case with 5 target languages in each model and with the same restriction of minimum occurance the same procedure gave the minimum amount of needed models equal to 34.

2.4.3 XXX TODO: Model settings

The initial parameter selection is made with respect to Popel and Bojar [2018]. First of all, the hyperparameters of MT model are tuned on couple of language pairs from one dataset. The parameters leading to the same result in shorter time were preferred. Then the selected parameters were used on all experimends with the dataset.

Model settings?

XXX TODO: paste model settings here

Tuning early stopping on early runs

The initial early stopping setting was that after 5 consecutive validation steps without improvement of validation loss value the training process stopped. However, during the training of the first couple of bilingual models the following situation has happened quite often: further improving performance on validation set by couple of tenths of BLEU points took as much time as reaching the pre-optimal state.

In the Figure 2.2, it can be seen that the path from the beginning of training to the optimal point B (26.9 BLEU) took as much time as its further improvement by 0.2 BLEU at point D (27.1 BLEU). However, there were certain models with a bit bigger improvement after a much longer time, e.g. 0.8 BLEU points on Figure 2.3.

This behaviour makes our decision on where to stop particularly complicated for multilingual models, as discussed in Section 2.4.6. After considering also some of preliminary multilingual runs, the 'patience' parameter of early stopping was set to 15. After 15 consequtive validation steps without a metric improvement, the training process is stopped.

XXX FIX: it needs to be presented earlier and *commented including motivation*. The red lines are unnatural at the first sight. In order to visually highlight when an increase in the validation score is observed, we plot the number of "steps stalled", see the red line in Figure 2.2. The higher the diagonal line grows, the longer we have to wait for an improvement. For instance, we see that the path..from beg to B took...

2.4.4 XXX TODO: Training

During the training procedure, once per specified number of updates occurs the checkpointing of the model. The model weights are saved to the disk and number of measurements are logged.

Those measurements are:

- training loss value (mean value for all updates since last checkpoint)
- learning rate value
- training speed (processed words per second)
- training time since last checkpoint
- number of updates happened from the beginning till this checkpoint

Hardware usage should also be recorded if possible:

- GPU usage
- CPU usage
- memory usage
- disk I/O
- network I/O

The hardware metrics are not important for model's evaluation but may help early spot mistakes like underuse of GPU or CPU, lack of RAM, etc. This is why they could possibly be recorded continuously. XXX TODO: link to this point from wandb section

2.4.5 TO EDIT: Validation

The validation set is used to track model's performance during the training on an unseen set of data and to perform early stopping. This model-specific validation set is created as described in Section 2.4.2. These measurements are only used during the training and not for the evaluation.

Once per specified number of steps the validation occurs: validation metrics are recorded, for any metric which value was improved current model weights are saved as best model by this metric. If early stopping condition occurred then the training process is stopped.

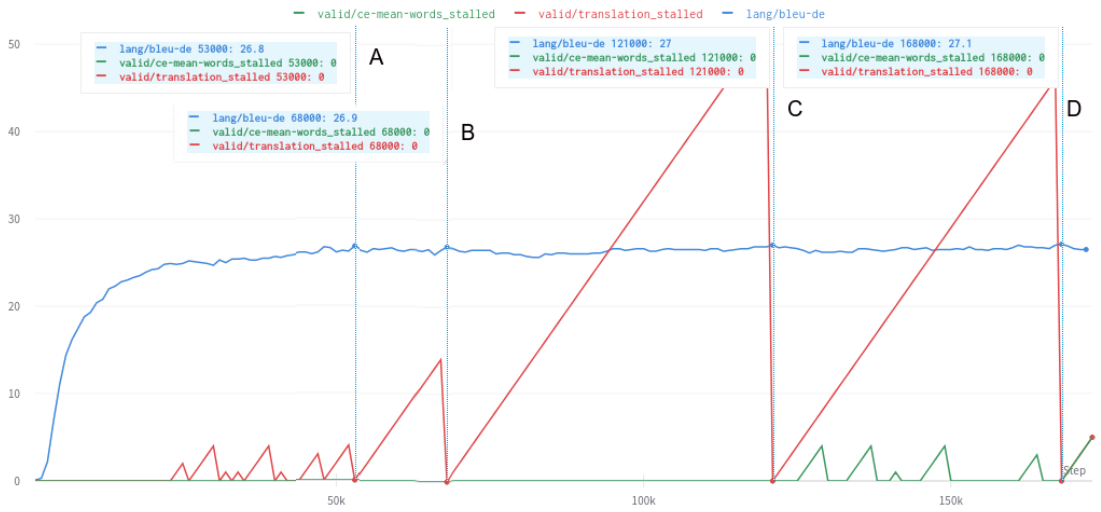


Figure 2.2: **Example change of model's performance on validation set in time.** Preliminary En→De model.
Blue: validation metric (value on the left axis in BLEU)
Red: validation metric (BLEU) stalled. Each consecutive validation step when the metric is not improved this value is incremented by 1. When the metric is improved this value is reset to 0.
Green: loss function value on validation set is stalled. Same logic as for *Red*.
BLEU score values at the points of improvement: A – 26.8, B – 26.9, C – 27.0, D – 27.1.

Figure 2.3: **Small improvement during long training.** In this case (En→Fi), the difference is a bit more visible: 21.3 at the first point and 21.9 at the best. Colors and scales are the same as at Figure 2.2.

For the validation set, we collect not only the loss function value but also the metric of interest, which is BLEU score. However, this BLEU scores are not used for the model's evaluation but only during the training process. The BLEU of the whole model's validation set is not something we are interested in. For the discussed example we collect validation bleu:fr and bleu:de scores which represent BLEU scores for French and German parts of validation set. E.g., to compute bleu:fr we select only En→Fr sentence pairs from the validation set.

Also, an aggregated value of the bleu:xx scores, i.e. the mean of BLEU scores over all target languages of the current model, is also recorded and may be used for early stopping: ending the training process when the metric is not improved during last N validation steps.

Altogether, the following validation metrics are recorded after the validation step:

- loss function value
- bleu:xx which is BLEU score for each of model's target languages
- aggregated value of all bleu:xx values
- translation time of the model's validation set

2.4.6 TO EDIT: Finishing the training

When should we stop the training? It is not possible to say precisely when did the model acquire its best performance because of stochastic nature of the training algorightm (SGD). Because of that we need to use some method to decide when training process should be stopped.

Number of epochs

The easiest approach is to specify the number of epochs after which the training is stopped. This could be a good solution for the case when all models that will be compared are trained on the same amount of data from the same domain. But in our case, adding one more target language adds a constant amount of sentence pairs to the training set. Roughly, if the number of epochs is specified as a stop condition, a bilingual En→De model will see the German training data x times, when multilingual En→De, Fr, Es will only see the German training data $x/3$ times.

Early stopping

Early stopping is a regularization technique used to avoid possible overfitting of a model on the training data. In general, it works in the following way: after every validation step it checks if the metric value improved during last N validations. The metric to be controlled and number of validation steps N are the parameters of this method (see Figure 2.4).

Another situation is even more probable in the area of NMT with generally large training datasets: model's validation performance is either stalled or slightly

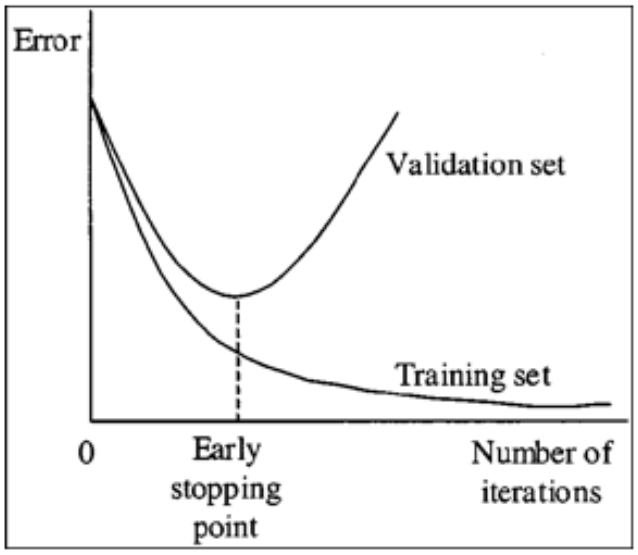


Figure 2.4: **Early stopping to prevent overfitting (Fig. 1 from Gençay and Qi [2001]).** At the 'early stopping' point the model's performance on unseen validation set of data does not improve anymore. Further training leads to poorer performance on unseen data. Stopping the training at this point results in better model's performance on unseen data. Image:

improved (see Figure 2.5). In this case early stopping helps to avoid unnecessary spendings on computational resources.

In our case we could use early stopping to ensure more equal conditions for models with different sizes of training data. A suitable number N could be found experimentaly, but which metric should be used?

Given that the task is to train a model that is as good as possible in **every** of its target directions, the BLEU score of the whole translated validation set for this set of languages does not say anything about the model's performance in each specified translation direction.

Aggregated value of BLEU scores

Therefore, we should use separate BLEU scores which represent model's performance in each of translation directions. The most intuitive and naive way is to compare BLEU scores for each target language.

However, most of frameworks and toolkits can monitor only one metric for the early stopping. Considering that different validation BLEU scores are computed for different parts of the validation set and in which are in different languages, they cannot be directly compared and may have different scale.

For example, a model for the En→{De,Fr} direction is being trained. Before the moment, an En→De model has already been trained and had the best BLEU score of 25 on the German part of the validation set. A En→Fr model has also been trained, and its result on the French part of the validation score is 35. So for the currently training En→{De,Fr}, one percentage point change for the En→De direction is not equal to the same change for the En→Fr direction.

Geometric mean is known to be good for aggregating multiple metrics with

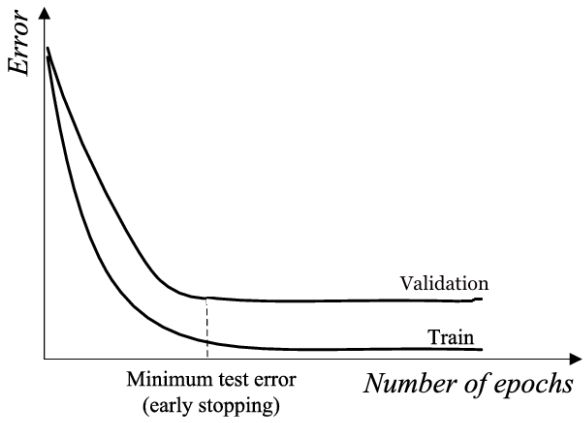


Figure 2.5: **Early stopping as the model is not improving.** Even though the metric value on the training set is still slowly improving, the its value on the unseen validation set is stalled. Further spending of computational resources is unjustified.

different scale (see Equation (2.1)).

$$geometric\_mean = \left(\prod_{i=1}^{n} x_i\right)^{\frac{1}{n}} = \sqrt[n]{x_1 x_2 \cdots x_n} \tag{2.1}$$

2.4.7 TO EDIT: Testing

After the training is finished, the received models should be evaluated on unseen test data. After translating the test set for each of target directions of the model the following record is created:

- model name
- source language
- target languages
- tested target language
- BLEU score for this part of translation
- metric, based on which the best model was saved
- dataset name (for en-to-36)

Let us return to the example setup is En→{De, Fr} and suppose the reported validation metrics are the mean loss function value on test set and 'translation' (geometric mean of all reported BLEU scores, see Section 2.4.5). After the training is finished, there will be two models: best by loss value and best by 'translation'. For each of those two records are created: for En→Fr translation and for En→Es. In total, 4 results are recorded.

If the model was trained and tested on en-to-36 dataset, than 4 times n records are created, where n is number of OPUS subdatasets from which the data was sampled.

2.4.8 Analysis

After required set of models is trained and their test metrics are collected, data should be analysed.

For example, let us take these four models: En→{De, Fr}, En→{De, Az}, En→{De, Bg}, and En→{Bg, Az}. After the training, they provide us with three results for En→De direction 2-target baseline, one value for En→Fr, two values for En→Bg and two for En→Az. These aggregated En→{De, X} results will be later compared with aggregated En→{De, X1, X2} for three target languages, En→{De, X1, X2, X3} for four target languages, where X1, ... Xi are some other targets.

Next, the En→{De, RANDOM} notation refers to a multilingual model that was trained in the RANDOM experiment (randomly selected targets), where one of the targets is German. In the same way, En→{De, GERMANIC} refers to a model from the GERMANIC experiment (targets selected from Germanic languages list).

2.5 Training tools

In the following section we describe the tools that are uset to implement what was shown in Section 2.4.

2.5.1 Toolkits

There exists a number of different tools that can be used for training a NMT model. General purpouse deep learning programming libraries like Tensorflow[3] and PyTorch[4] are most popular for deep learning related research. With their help it is possible to construct any of today's state-of-the-art NMT models; pre-built and pre-trained models are initially present in such frameworks, but it is also possible to describe a model from scratch.

Another option is presented by specialized NMT tool kits. They usually contain efficient and tested implementations of NMT models as well as some of usefull preprocessing tools. For the experiments described in 2.2 there is a need to train significant amount of models with the same architecture and settings but different datasets. Due to that fact, in this work the use of specialized NMT tool kit is more suitable. Let us consider the foolowing list of broadly used tool kits as for year 2020, presented in Koehn [2020]:

- OpenNMT (based on Torch/pyTorch)[5]
- Sockeye (based on MXNet)[6]
- Fairseq (based on pyTorch)[7]
- Marian (stand-alone implementation in C++)[8]

[3] `https://tensorflow.org/`
[4] `https://pytorch.org/`
[5] `https://opennmt.net`
[6] `https://github.com/awslabs/sockeye`
[7] `https://github.com/pytorch/fairseq`
[8] `marian-nmt.github.io`

- Google's Transformer (based on Tensorflow)[9]
- Tensor2Tensor (based on Tensorflow) [10]

We chose *MARIAN-NMT* tool kit[11] as a fast solution with stable and efficient *Transformer* Vaswani et al. [2017] implementation, minimum of third-party dependencies, and ability to train models on multiple GPU units in parallel.

2.5.2 Computational cluster

To be able to train large number of models in a reasonable amount of time we needed to use computational cluster with GPU cards. The computational clusters available at the institution are operating under SGE[12] scheduling software and are equipped with GPU cards with minimum CUDA *compute capability* 6.1.

2.5.3 Training pipeline

Considering data storage quota limitation and high utilization of computational resources by the cluster's users, the following training pipeline was designed:

- Prepare task list
- Iterate over the list working with at most N tasks in parallel
- For each task
  - Subsample the dataset taking only those sentence pairs with target languages specified in the task
  - Run the training procedure for limited amount of time (e.g. for one hour only) starting with previous checkpoint if it already exists
  - Regularly compute metrics on the developement set and report them
  - On the event of evaluation on the developenemt set save the best model for each metric
  - After time is out the training is stopped and subsampled datasets are removed
- If for next selected task the model is already trained then select next task from the list
- If for next selected task the model is currently being trained then decrease the number N of tasks processed in parallel

[9] `https://github.com/tensorflow/models/tree/master/official/transformer`
[10] `https://github.com/tensorflow/tensor2tensor`
[11] Junczys-Dowmunt et al. [2018]
[12] `https://arc.liv.ac.uk/trac/SGE`

2.5.4 Inspecting the training process

As the number of models trained and being trained is growing, monitoring of the training process becomes more and more complicated. If the experiments are also being run on different computational clusters it becomes very possible that a parameter mistakenly set up to different value or a corrupted dataset, or even hardware version may lead to an unexpected difference in results.

To address these and other issues that may occur during the training process we use Weights&Biases[13] experiment tracking tool. Its main features that are useful in this prospective are following:

- Metric visualization
  - Training and validation loss curves (Figure 2.6 left subplot)
  - Scatter plots (Figures 2.7 and 2.6 middle subplot)
- Artifact storage
  - Model checkpoints storage
    - stores 'heavy' model files which cannot be stored in *git*
    - along with *git* it makes possible to move training to the different computational cluster system
  - Sample translations of validation set
    - helps to observe improvements of translation quality in time
    - lets verify that model is actually produces meaningfull translation
- Customizable reports
- Hardware utilization

[13] Biewald [2020]

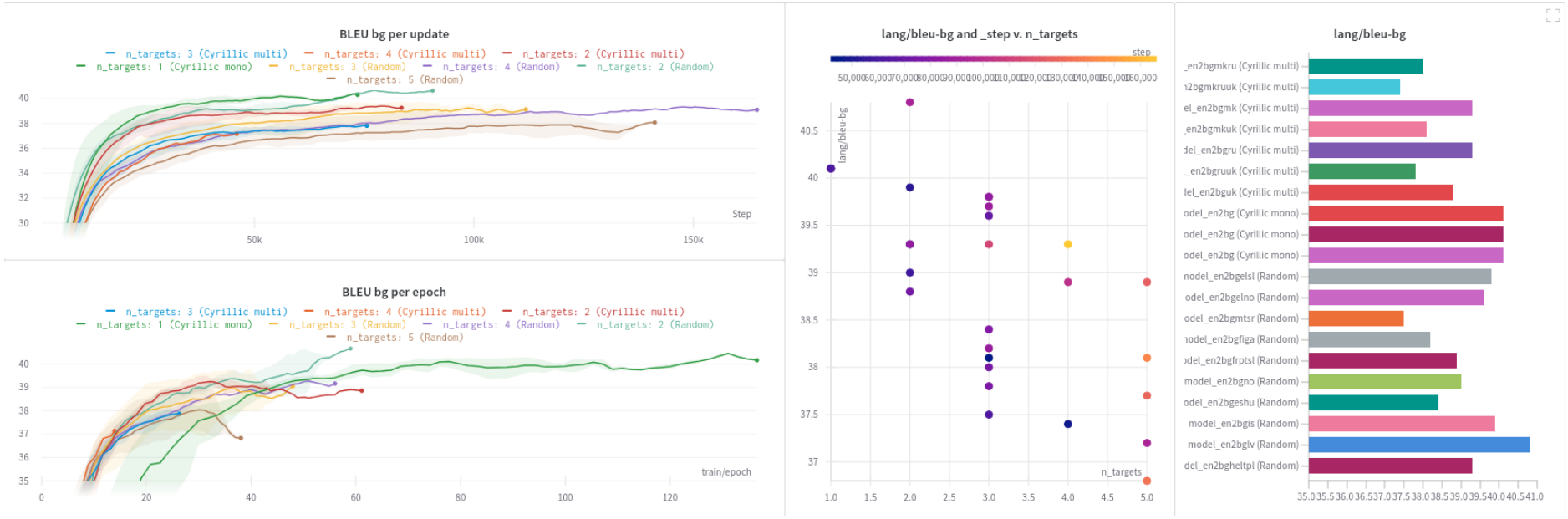


Figure 2.6: **Training progress dashboard for one translation direction.** Here is a part of interactive report for 'Slavic languages with Cyrillic script vs. random' experiment. In this specific case models' performance on Bulgarian part of validation set is compared. Note: the visualized BLEU scores are only used during the training and are not used for evaluation.
Left: BLEU score for En→Bg translation direction is monitored with training step on X axis (top) and training epoch (bottom). Each curve represents mean value (line) and its min/max value (range) at the point of time of multiple models' results. Models are grouped by the number of target languages and experiment subgroup (bilingual En→Bg, multilingual En→{Slavic} and En→{Random}).
Middle: Number of targets (axis X) vs. BLEU on En→Bg validation set (axis Y) vs. update steps (colos with scale at the top).
Right: Individual models' En→Bg validation BLEU scores.

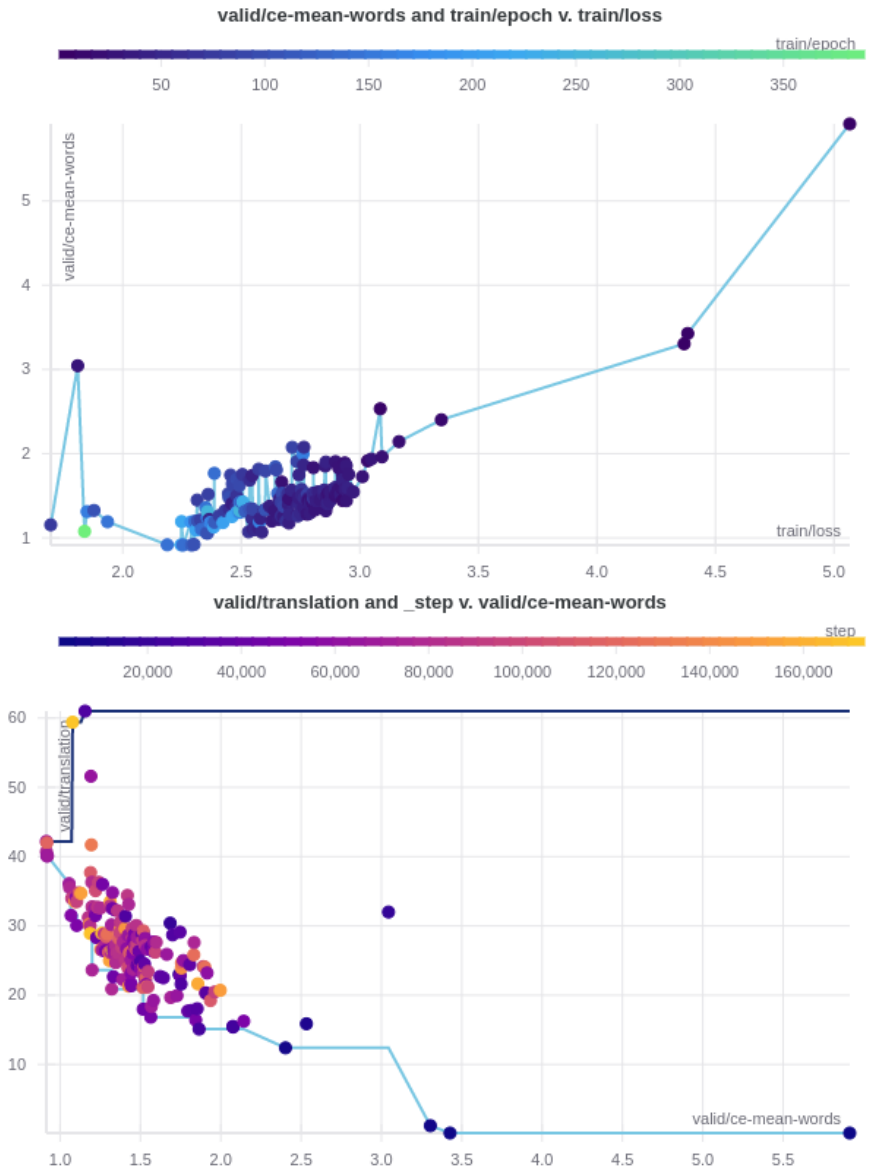


Figure 2.7: **Overall convergence dashboard.** In these two interactive graphs, each point represents one model. Models that are currently training are visualized here together with completely converged models and those which training process is currently on hold.

Top: the X axis represents the training loss value, the Y axis represents the value for the same loss function calculated on the validation set. The color of each point represents current training epoch for the model. Normally for any model the point XXX FIX: moves from top right part of this graph to the bottom left part, representing both training and validation loss being gradually decreased during the training procedure. The point that moves to the middle left part of the graph may signalize about either overfitting of the model on training set, or difference in data distribution in training and validation set, or else some mistake in training settings. This is useful for finding which training runs need attention and perhaps debugging.

Right: in this plot loss value on the validation set (axis X) is compared with geometric mean of BLEU scores for each of target languages. For any model during the training, its point usually XXX FIX: move from bottom right corner into the cluster of other points. The model the point of which 'arrives' to any other location than the cluster may need special attention.

3. Bilingual and multi-lingual baselines

In this chapter, we describe the baseline experiments. Bilingual baselines are needed to specify the starting point: how good can a model perform on a specific translation direction for each test set.

After bilingual results are collected and inspected, it is time for multi-lingual baselines. For this purpose, we trained models with randomly selected sets of target languages. This way, we can see how much adding more target languages to the model changes its performance on the same specific translation direction.

Most of the experiments are done on the en-to-36 dataset with a couple of additional experiments on the en-to-5 dataset.

3.1 TO EDIT: Bilingual baseline

We trained bilingual models on the en-to-36 dataset and received a number of values for each translation direction, each value reflecting the performance on a particular domain of the corpus. Test results for relevant target directions (i.e. languages from 'Germanic' and 'Slavic with Cyrillic script' from Section 2.2.2) are shown in Table 3.1. For example, for En→De direction, we trained a bilingual model. After that, we evaluated the model on the test set and received BLEU scores for each sub-dataset, as shown in Figure 3.1. Later, when an En→{De, others} model is trained and evaluated on the same test set, its En→De performance on each sub-dataset will be compared with these values.

In Figure 3.1 we can see that BLEU scores for datasets in group 1 (Table 2.1), i.e. Europarl/v7, MultiUN/v1, DGT/v4 and JRC-Acquis/v3.0 are the highest; the lowest it is for sub-datasets from groups 4 and 5, such as PHP/v1, KDE/v4, Tanzil/v1, GNOME/v1, and Books/v1.

BLEU score values for different test sets cannot be compared directly. However, too big or too small value can give us some insight about the data.

Let us look closer at some example translations from different sub-datasets of the En→De test set. In Example 3.1.1, we can see a translation produced by our bilingual En→De model compared with the reference translation and two unnamed online translation systems. The sentence pair is from the Europarl/v7 sub-dataset of En→De test set. As was stated in Section 2.3.1, Europarl/v7 was a prioritized source of data to be sampled to the training set. Even though our translation has different wording comparing to the reference one, the sense is preserved. Interestingly, at the same time, our translation is much closer to the ones produced by online MT systems.

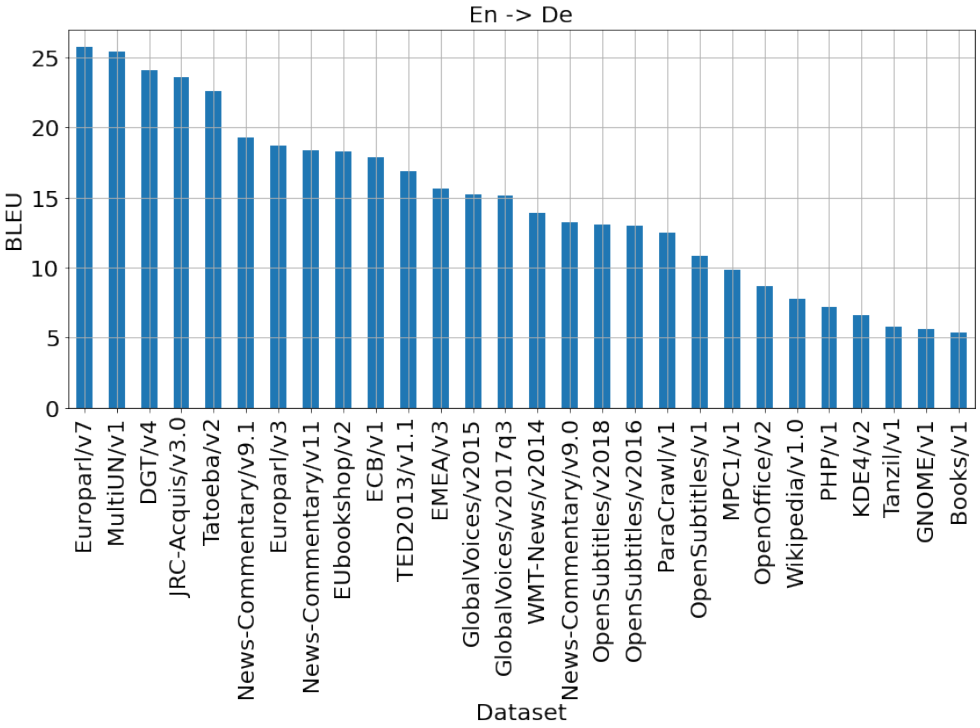


Figure 3.1: **En→De bilingual results.** Datasets on the X axis are sorted by declining BLEU score. XXX FIX: Color each column by dataset's group or add prefixes

Source (En): <2de> Finally, I fully support the compromise agreement reached by our committee on Article 5 (4).
Reference translation (De): Ich unterstütze ohne jede Einschränkung die von unserem Ausschuss zu Artikel 5 Absatz 4 erzielte Kompromissvereinbarung.
Our bilingual En→De: Schließlich unterstütze ich die von unserem Ausschuss erzielte Kompromiss zu Artikel 5 Absatz 4 voll und ganz.
OMT-G: Schließlich unterstütze ich voll und ganz die Kompromissvereinbarung, die unser Ausschuss zu Artikel 5 Absatz 4 getroffen hat.
OMT-D: Schließlich unterstütze ich voll und ganz die von unserem Ausschuss erzielte Kompromissvereinbarung zu Artikel 5 Absatz 4.

Example 3.1.1. Bilingual En→De model's output of test set sentence translation (from Europarl/v7 sub-dataset) compared with the reference one and online translation systems OMT-G and OMT-D. Here and further for the online translation system, the target tag is omitted, and the target language is selected directly in the system. For our system, the following sentence is firstly preprocessed (see Section 2.4.2).

The next prioritized sources for sampling training data were Eurobookshop and OpenSubtitles. The first dataset has domain and vocabulary similar to the Europarl dataset. OpenSubtitles dataset has data of a different domain: transcribed human speech from films and series; it has much shorter sentences and the speech of different register.

In Example 3.1.2, we can see an issue of another kind that might happen: a short sentence might not have all the needed information. Here English 'you' in the reference translation translates as 'ihr' (2. person, plural), and in our translation as 'Sie' (3. person, plural) which refers to a polite form of 'you'. One of the online MT systems translates it as 'du' (2. person, singular). The difference in exact translation of 'you' affects the translation of 'know', because in German the verb has different conjugation for each person and case, comparing to English, where s/es are only added to the verb for 3. person, singular.

Source (En): <2de> Do you know it?
Reference translation (De): Kennt ihr das?
Our bilingual En→De: Wissen Sie das?
OMT-G: Weißt du es?
OMT-D: Kennen Sie es?

Example 3.1.2. Example sentence from OpenSubtitles/v2018 sub-dataset of the En→De test set.

The main reason for introducing these detailed and domain-specific baselines is that we want to make comparisons of BLEU scores as reliable as possible. Specifically, since different languages are differently covered by the text domains, a single BLEU over a mixed test set would likely hide important observations.

3.2 XXX TODO: Multilingual baseline

Next after we have trained bilingual models and collected the results, we trained multilingual baseline models – the models with randomly selected sets of targets. We generated RANDOM task in a way we described in Section 2.4.1.

As we have seen in Section 1.5.2, models with more languages in the mix usually perform slightly or significantly worse than bilingual ones.

However, there might be different unexpected effects due to slight domain-wise differences in corpora content for different target languages.

XXX TODO: ? When the size of the model is fixed, adding more translation directions usually causes worsening of its performance.

XXX FIX: refer to the table, explain, something like Across all considered configurations (e.g. English to 5 target languages), there are too many specific settings how to conduct the experiment. Already the choice of the particular languages offers too many setups and we cannot afford to run them all. We thus randomly sample and report the average BLEU and standard deviation over all the particular runs we performed. The actual number of runs is given in the column 'count'.

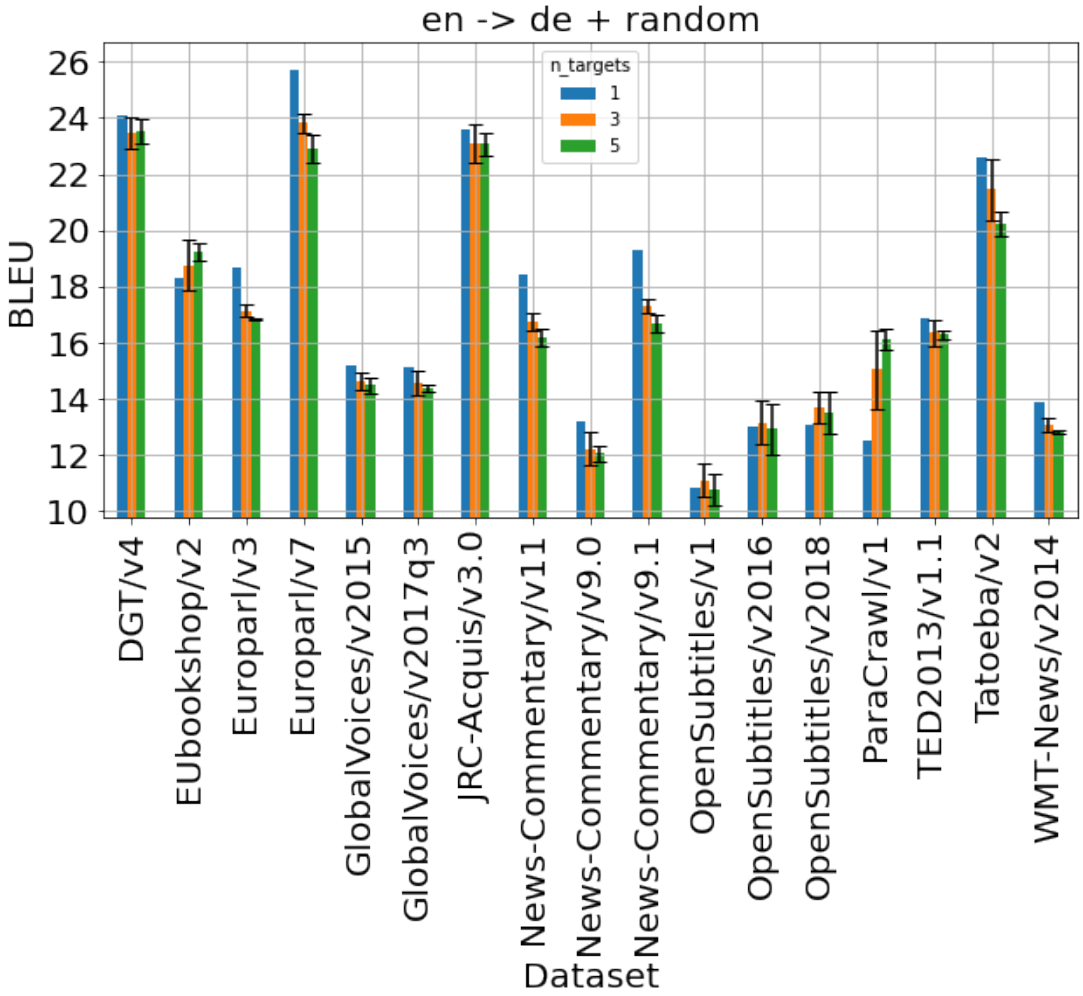


Figure 3.2: **En→De multilingual baseline results (RANDOM).** BLEU scores for En→De of multigarget models with randomly selected target languages and German as one of the targets. Datasets with BLEU lower than 10 are removed from this figure.

target dataset	bg	da	de	is	mk	nl	no	ru	sv	uk
Books/v1	—	—	5.4	—	—	4.8	—	8.3	—	—
DGT/v4	33.1	27.4	24.1	—	—	25.9	—	—	28.9	—
ECB/v1	—	20.9	17.9	—	—	21.2	—	—	—	—
EMEA/v3	15.1	16.4	15.6	—	—	15.8	—	—	17.6	—
EUbookshop/v2	38.2	24.1	18.3	—	—	18.9	—	—	24.7	—
Europarl/v3	—	24.6	18.7	—	—	23.4	—	—	23.6	—
Europarl/v7	41.4	32.5	25.7	—	—	26.0	—	—	33.3	—
GNOME/v1	—	—	5.6	2.4	—	8.9	—	—	—	—
GlobalVoices/v2015	—	—	15.2	—	10.6	18.6	—	13.2	—	—
GlobalVoices/v2017q3	—	—	15.1	—	10.7	19.1	—	14.4	—	—
JRC-Acquis/v3.0	30.8	27.3	23.6	—	—	25.7	—	—	29.1	—
KDE4/v2	6.9	8.5	6.6	4.2	4.8	8.1	—	4.1	8.4	1.3
MPC1/v1	—	—	9.8	—	—	—	—	—	—	—
MultiUN/v1	—	—	25.4	—	—	—	—	14.6	—	—
News-Commentary/v11	—	—	18.4	—	—	19.2	—	23.9	—	—
News-Commentary/v9.0	—	—	13.2	—	—	—	—	18.2	—	—
News-Commentary/v9.1	—	—	19.3	—	—	—	—	22.1	—	—
OpenOffice/v2	—	—	8.7	—	—	—	—	—	8.6	—
OpenSubtitles/v1	19.3	17.1	10.8	—	—	12.5	23.1	16.2	13.4	—
OpenSubtitles/v2016	23.2	14.8	13.0	24.3	24.3	13.7	27.0	19.5	14.8	11.2
OpenSubtitles/v2018	23.7	15.6	13.1	23.1	24.6	13.4	29.6	19.2	15.3	12.2
PHP/v1	—	—	7.2	—	—	12.6	—	3.3	8.9	—
ParaCrawl/v1	—	—	12.5	—	—	17.9	—	11.1	—	—
SETIMES/v1	23.2	—	—	—	6.4	—	—	—	—	—
SETIMES/v2	27.5	—	—	—	10.4	—	—	—	—	—
TED2013/v1.1	—	—	16.9	—	—	19.1	—	14.7	—	—
Tanzil/v1	5.7	—	5.8	—	—	4.6	6.1	2.4	4.4	—
Tatoeba/v2	—	—	22.6	—	—	28.9	—	27.7	—	13.3
UN/v20090831	—	—	—	—	—	—	—	9.9	—	—
Ubuntu/v14.10	—	—	—	—	—	8.6	—	—	—	—
WMT-News/v2014	—	—	13.9	—	—	—	—	—	—	—
WikiSource/v1	—	—	—	—	—	—	—	—	5.3	—
Wikipedia/v1.0	11.7	—	7.8	—	—	9.6	—	10.6	—	—

Table 3.1: BLEU scores for bilingual models

n_targets	mean	std	count
1	41.40	—	1
2	40.60	0.20	3
3	39.39	0.62	8
4	39.40	0.71	2
5	38.45	0.52	6

(a) En→Bg for *Europarl/v7* dataset.

n_targets	mean	std	count
1	19.50	–	1
2	18.88	0.39	4
3	17.45	0.52	4
4	17.80	0.42	2

(b) En→Ru for *OpenSubtitles/v2016* dataset.

Table 3.2: **BLEU score change with adding target languages.** (a) First row: for mono-lingual En→Bg model test BLEU score is 41.40. Second row: for 3 (column *count*) En→Any models with two target languages (column *n_targets*) one of which is Bulgarian the mean BLEU score is 40.60 with standard deviation 0.20. (b): same way as (a)

3.3 XXX TODO: Additional experiments with richer dataset

Additionaly to the previus results, we trained a couple of 1-to-1, 1-to-2 and 1-to-3 models on the bigger dataset from Section 2.3.2 – en-to-5. This dataset has the two main differences:

- it has much more sentence pairs per target language;
- it is created from a parallel corpus.

The first point refers to the fact, that for each target language it contains 10 millions of sentence pairs; 5 thousand of which were used for a validation set, another 5 thousand were used for a test set, and all remaining is a test set.

The second point means, that for every sentence pair for one of the languages there is a sentence pair with the same source side for every other target language.

model targets	Es	Fr	Ru
Es, Fr, Ru	56.33	45.03	40.35
Es, Fr	57.94	46.84	—
Fr, Ru	—	45.66	41.95
Es, Ru	57.31	—	42.16
Es	59.94	—	—
Fr	—	48.64	—
Ru	—	—	44.20

Table 3.3: **Results of an additional experiment on the larget corpus.** Targets in the first column refer to a model's targets, given that English is always the source language. E.g. the first row shows results for a En→{Es, Fr, Ru} model. Further columns show BLEU scores for specific target direction, written in the column's header. For example, the En→{Es, Fr, Ru} model's BLEU score for En→Fr part of the test set is 45.03. The En→{Es, Fr} model has a En→Es score of 57.94 and a En→Fr score of 46.84, but does not have any score for En→Ru part of test set.

4. Group by language groups

XXX TODO: Intro here 1 to 2, 3, 4, 5, etc. models on en-to-36 dataset (0.9 mil. sentences per target language) compared with random runs

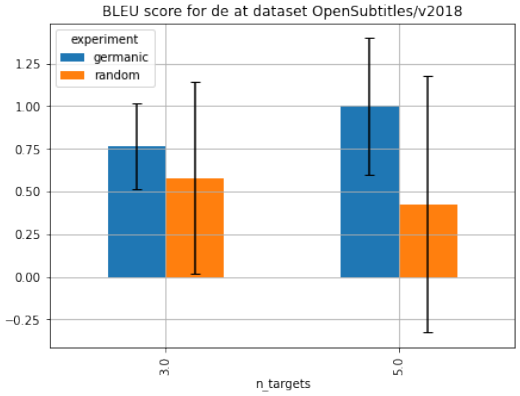
4.1 Germanic group

Here Germanic group consists of German, Dutch, Swedish, Danish, Norwegian and Islandic. Models En→{Germanic} are compared to En→{non-Germanic}, where non-Germanic consists of any langauge except from the Germanic group. In Figure 4.1 and Figure 4.2, some selected results are visualized along with vocabulary changes. Results for OpenSubtitles/v2018 mean the BLEU score on test set part sampled from OpenSubtitles/v2018. In both figures, the subfigure (a) shows the result on spontaneous or pseudo-spontaneous speech transcripts, i.e. subtitles, while sub-figure (b) shows the result for prepared speeches or documents from Europarl or UN meetings.

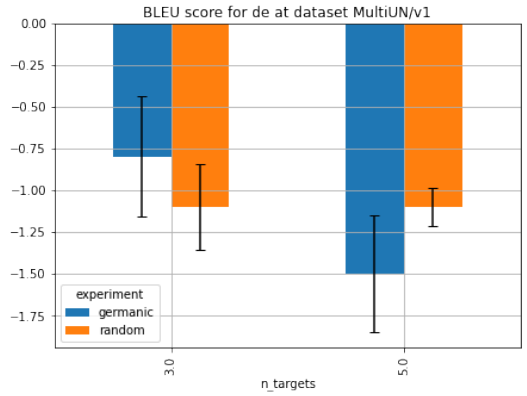
In this cases observations are twofold:

- For test sets with lower bilingual BLEU score, adding more target languages to the model improves the score; adding related target languages improves it even more.
- Adding more target languages improves translation result on test sets from spontaneous speech domain but worsens them for prepared speech or documents.

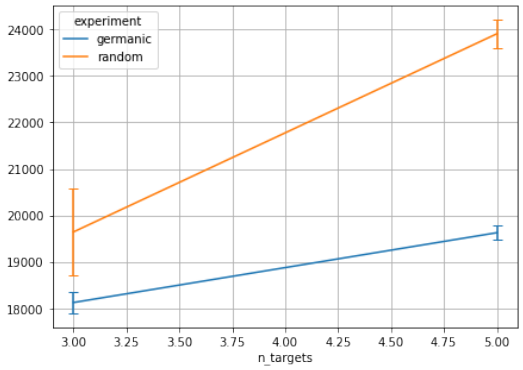
XXX FIX: Describe observation (c): When comparing the vocabulary size of models where languages related to the target German are added with models where random languages are added, the vocabulary clearly grows faster with random languages. This behaviour is expected and confirms that related languages contain similar patterns of subwords.



(a) OpenSubtitles/v2018, the baseline is the bilingual score: 13.1 BLEU

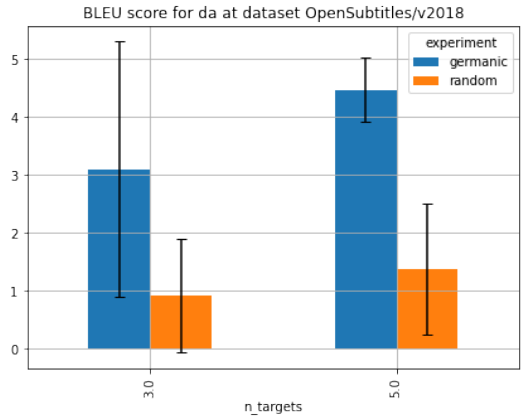


(b) MultiUn, bilingual score: 25.4 BLEU

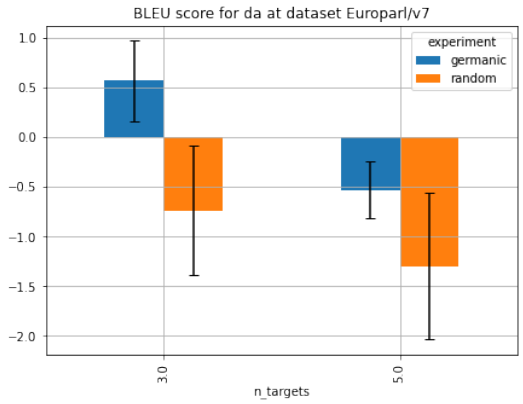


(c) Subword dictionary size used for target side

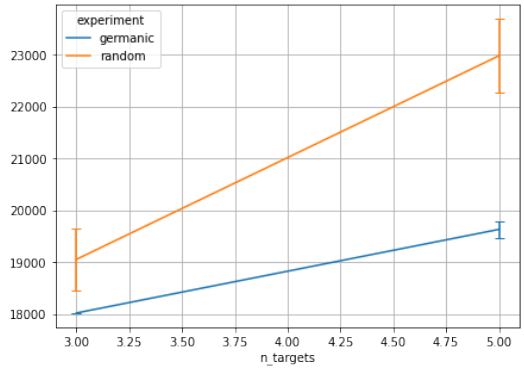
Figure 4.1: **En→De BLEU score difference: Random vs. Germanic.** On X axis - number of target languages. On Y axis - difference score comparing with bilingual baseline BLEU. Black vertical lines show standard deviation across the runs we sampled. (a) Adding random target languages as well as related ones slightly improves German translation score on speech transcript. (b) Adding neither random target languages nor related ones helps with prepared speeches transcripts and documents in German. (c) Adding a related target language into the mix introduces fewer new unique subwords.



(a) OpenSubtitles/v2018, bilingual score: 15.6 BLEU



(b) Europarl/v3, bilingual score: 32.5 BLEU



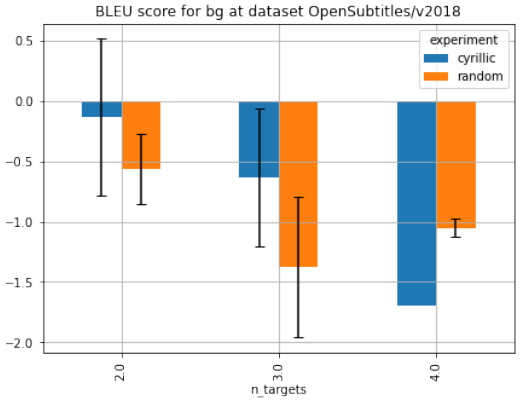
(c) Subword dictionary size used for target side

Figure 4.2: **En→Da BLEU score difference: Random vs. Germanic.** The axis are same as above. (a) For OpenSubtitles test set which which consists of human speech transcripts, adding similar target language to the mix significantly imporves the result. (b) For Europarl/v7 which consists of prepared speeches transcripts and documents, adding more germanic languages to the mix did not worsen Danish translation quality (unlike the case with German). (c) Adding random target language to the mix adds more subwords to the target subword dictionary

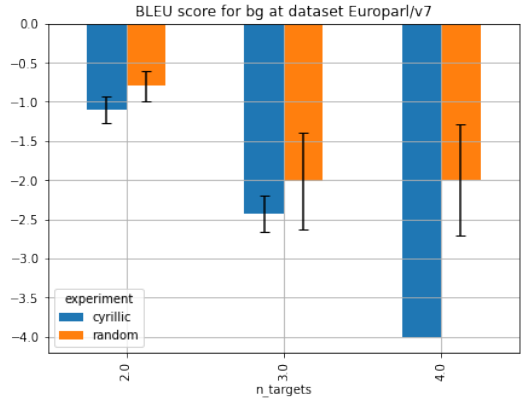
4.2 Slavic with Cyrillic script

Here *Slavic with Cyrillic script* group consists of Bulgarian, Macedonian, Russian and Ukrainian. Models En→{Cyrillic} are compared to En→{non-Cyrillic}, where non-Cyrillic consists of any langauge except from those from the group above. In Figures 4.3 and 4.4, some selected results are visualized along with vocabulary changes. Test sets for subfigures (a) and (b) were selected the same way as in Section 4.1.

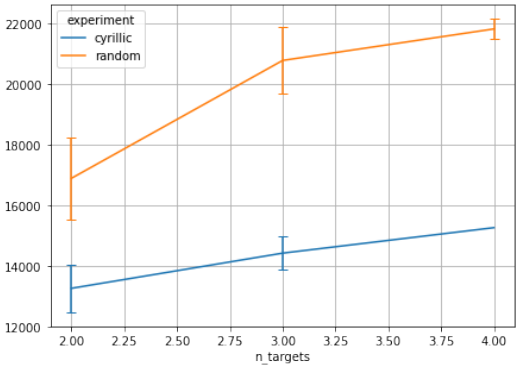
From the two opposite observations of 4.1 in this case the second one is observed: low results of bilingual baselines are getting slightly better or remain the same, good results are getting slightly or significantly worse as more languages are added to the mix.



(a) OpenSubtitles/v2018, bilingual score: 23.7 BLEU



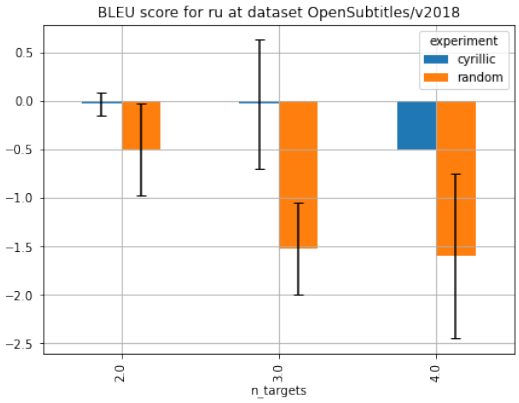
(b) Europarl/v7, bilingual score: 41.4 BLEU



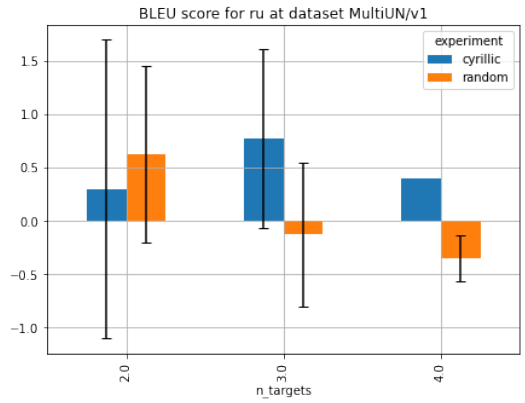
(c) Subword dictionary size used for target side

Figure 4.3: **En→Bg BLEU score difference: Random vs. Slavic with Cyrillic script.** The axes are same as for Figures 4.2 and 4.1. There is not any data for Cyrillic and 5 targets as there are only 4 such languages in the en-to-36 dataset. Both (a) and (b) show a significant decrease in translation quality as more languages are added to the setup. In (c) it is clearly visible how adding a random language with non-cyrillic script increases target subword vocabulary size.

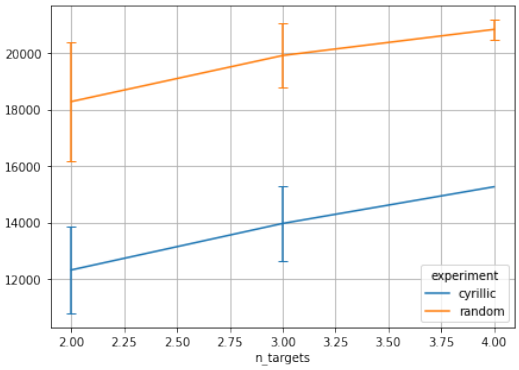
XXX TODO: can describe some more hidden tables here



(a) OpenSubtitles/v2018, bilingual score: 19.2 BLEU



(b) MultiUN, bilingual score: 14.6 BLEU



(c) Subword dictionary size used for target side

Figure 4.4: **En→Ru BLEU score difference: Random vs. Slavic with Cyrillic script.** The axes are same as for Figures 4.2, 4.1 and 4.3. There is only one value for Cyrillic with four targets as there are only four languages in the selected group. Both (a) and (b) show a significant decrease in translation quality. In (c), it is clearly visible how adding a random language with non-Cyrillic script increases target subword vocabulary size.

5. Discussion

5.1 Results

More languages in the mix: - share word ordering patterns - share vocabulary

More training data illustrates the properties of distribution better (some words are rare, some are often used)

5.2 Further work

XXX TODO: Conclusion

On the technical side:

- Created a set up for running a very large number of experiments on a moderate GPU cluster - Proposed and re-assessed a stopping criterion for the experiments. - Found and utilized a tool for visual inspection of a massive number of experiment results (how many results in total?) ...maybe something else, too

On the experimental side, carried out and interpreted the results for many English-to-X multilingual models:

- Setups were run on a domain-diverse training set for up to 36 target languages, and also sometimes complemented on a multi-parallel set for up to 5 target languages. - Confirmed that generally, the more target languages in the model, the worse the performance. - Identified interesting exceptions: (you need to check the following) - Targetting the spoken domain, the quality does not decrease. - Targetting the same script (Cyrillic), the quality decreases less or does not decrease. - maybe something else

Bibliography

Roee Aharoni, Melvin Johnson, and Orhan Firat. Massively Multilingual Neural Machine Translation. In *Proceedings of the 2019 Conference of the North {A}merican Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota, 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1388. URL https://www.aclweb.org/anthology/N19-1388.

ALPAC. *Language and Machines*. National Academies Press, Washington, D.C., jan 1966. ISBN 978-0-309-57056-5. doi: 10.17226/9547. URL http://www.nap.edu/catalog/9547.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, Wolfgang Macherey, Zhifeng Chen, and Yonghui Wu. Massively Multilingual Neural Machine Translation in the Wild: Findings and Challenges. jul 2019. URL http://arxiv.org/abs/1907.05019.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. sep 2014. URL http://arxiv.org/abs/1409.0473.

Loïc Barrault, Ondřej Bojar, Marta R Costa-jussà, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Philipp Koehn, Shervin Malmasi, Christof Monz, Mathias Müller, Santanu Pal, Matt Post, and Marcos Zampieri. Findings of the 2019 Conference on Machine Translation (WMT19). In *Proceedings of the Fourth Conference on Machine Translation (Volume 2: Shared Task Papers, Day 1)*, pages 1–61, Stroudsburg, PA, USA, 2019. Association for Computational Linguistics. doi: 10.18653/v1/W19-5301. URL http://www.aclweb.org/anthology/W19-5301https://www.aclweb.org/anthology/W19-5301.

Lukas Biewald. Experiment tracking with weights and biases, 2020. URL https://www.wandb.com/. Software available from wandb.com.

Ondřej Bojar, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Philipp Koehn, and Christof Monz. Findings of the 2018 conference on machine translation (WMT18). In *Proceedings of the Third Conference on Machine Translation: Shared Task Papers*, pages 272–303, Belgium, Brussels, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6401. URL https://www.aclweb.org/anthology/W18-6401.

P. Brown, J. Cocke, S. Della Pietra, V. Della Pietra, F. Jelinek, R. Mercer, and P. Roossin. A statistical approach to language translation. In *Proceedings of the 12th conference on Computational linguistics* -, volume 1, pages 71–76, Morristown, NJ, USA, 1988. Association for Computational Linguistics. ISBN 963 8431 56 3. doi: 10.3115/991635.991651. URL http://portal.acm.org/citation.cfm?doid=991635.991651.

Kyunghyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of SSST-8, Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, pages 103–111, Stroudsburg, PA, USA, 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-4012. URL http://aclweb.org/anthology/W14-4012.

Raj Dabre, Tetsuji Nakagawa, and Hideto Kazawa. An empirical study of language relatedness for transfer learning in neural machine translation. Technical report, November 2017. URL https://www.aclweb.org/anthology/Y17-1038.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. A Brief Survey of Multilingual Neural Machine Translation. may 2019. URL http://arxiv.org/abs/1905.05395.

Philip Gage. A New Algorithm for Data Compression. *C Users J.*, 12(2):23–38, 1994. doi: 10.5555/177910.177914.

Ramazan Gençay and Min Qi. Pricing and hedging derivative securities with neural networks: Bayesian regularization, early stopping, and bagging. *Neural Networks, IEEE Transactions on*, 12:726 – 734, 08 2001. doi: 10.1109/72.935086.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, Macduff Hughes, and Jeffrey Dean. Google's Multilingual Neural Machine Translation System: Enabling Zero-Shot Translation. *Transactions of the Association for Computational Linguistics*, 5:339–351, 2017. doi: 10.1162/tacl_a_00065. URL https://www.aclweb.org/anthology/Q17-1024.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F T Martins, and Alexandra Birch. Marian: Fast Neural Machine Translation in {C++}. In *Proceedings of ACL 2018, System Demonstrations*, Melbourne, Australia, 2018. URL https://arxiv.org/abs/1804.00344.

Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, Washington, USA, oct 2013. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/D13-1176.

Philipp Koehn. *Neural Machine Translation*. Cambridge University Press, 2020.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei jing Zhu. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*, pages 311–318. Association for Computational Linguistics, 2002.

Martin Popel and Ondřej Bojar. Training tips for the transformer model. *The Prague Bulletin of Mathematical Linguistics*, 110, 03 2018. doi: 10.2478/pralin-2018-0002.

Matt Post. A call for clarity in reporting BLEU scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6319. URL https://www.aclweb.org/anthology/W18-6319.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Stroudsburg, PA, USA, 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1162. URL http://aclweb.org/anthology/P16-1162.

Aditya Siddhant, Melvin Johnson, Henry Tsai, Naveen Arivazhagan, Jason Riesa, Ankur Bapna, Orhan Firat, and Karthik Raman. Evaluating the Cross-Lingual Effectiveness of Massively Multilingual Neural Machine Translation. sep 2019. URL http://arxiv.org/abs/1909.00437.

Matthew Snover, Bonnie J. Dorr, Richard H. Schwartz, and Linnea Micciulla. A study of translation edit rate with targeted human annotation. 2006.

Keh-Yih Su, Ming-Wen Wu, and Jing-Shin Chang. A new quantitative quality measure for machine translation systems. In *Proceedings of the 14th Conference on Computational Linguistics - Volume 2*, COLING '92, page 433–439, USA, 1992. Association for Computational Linguistics. doi: 10.3115/992133.992137. URL https://doi.org/10.3115/992133.992137.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to Sequence Learning with Neural Networks. sep 2014. URL http://arxiv.org/abs/1409.3215.

Jörg Tiedemann. Parallel data, tools and interfaces in opus. In Nicoletta Calzolari (Conference Chair), Khalid Choukri, Thierry Declerck, Mehmet Ugur Dogan, Bente Maegaard, Joseph Mariani, Jan Odijk, and Stelios Piperidis, editors, *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, Istanbul, Turkey, may 2012. European Language Resources Association (ELRA). ISBN 978-2-9517408-7-7.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All you Need. In I Guyon, U V Luxburg, S Bengio, H Wallach, R Fergus, S Vishwanathan, and R Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc., 2017. URL http://papers.nips.cc/paper/7181-attention-is-all-you-need.pdf.

Warren Weaver. *Machine Translation of Languages*. MIT Press, Cambridge, Massachusetts, 1955. ISBN 0-8371-8434-7.

Michał Ziemski, Marcin Junczys-Dowmunt, and Bruno Pouliquen. The united nations parallel corpus v1.0. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 3530–3534, Portorož, Slovenia, May 2016. European Language Resources Association (ELRA). URL `https://www.aclweb.org/anthology/L16-1561`.

List of Figures

List of Tables

Glossary

baseline In machine learning this term refers to a simple or naïve initial solution, which efficiency it then taken as a reference point and later improved. . 12, 42

early stopping Regularization technique to avoid model overfit. Usualy consists of stopping the training process when the value of some selected metric on the validation set is not improved for last number of validation steps. . 16, 19–21, 42

en-to-36 The dataset with source sentences in English and target sentences in 36 languages, described in Section 2.3.1 . 13–16, 21, 27, 42

en-to-5 The dataset created from UN parallel corpus, with source sentences in English and target sentences in one of following 5 languages: Spanish, French, Russian, Arabic and Chinese; described in Section 2.3.2 . 15, 16, 27, 33, 42

epoch Refers to one pass of full training dataset to the learning algorightm . 19, 42

gradient descent An algorithm of iterative optimization of differentiable objective function (loss). 42

latex Is a mark up language specially suited for scientific documents. 42

loss Loss function, also often called 'objective function' and 'error function'. It is optimized during the training process. In our experiments it is mean word cross-entropy score.. 16, 42

overfitting Occurs when the model's performance on unseen validation set stops improving while on the training set it still improves. . 19, 26, 42

self attention A mechanism in sequential models, which alters every element's representation with respect to the other relevant elements in the sequence. . 3, 42

stochastic gradient descent Method for iterative optimizing an objective function. Differs from gradient descent by approximating the gradient on mini-batch of data.. 42

Abbreviations

ALPAC Automatic Language Processing Advisory Committee. 4, 42

ARPA Advanced Research Projects Agency. 42

BLEU bilingual evaluation understudy. 4–6, 9, 10, 12, 16, 17, 19–21, 25–27, 29, 32, 34–38, 42

BPE byte pair encoding. 4, 15, 42

CNN convolutional neural network. 3, 42

GRU gated recurrent unit. 3, 42

LSTM long short-term memory. 3, 42

MT machine translation. 2, 3, 7, 27, 29, 42

NMT neural machine translation. 3, 7, 9, 42

OOV out-of-vocabulary. 3, 42

RNN recurrent neural network. 3, 42

SGD stochastic gradient descent. 19, 42

SMT statistical machine translation. 42

WMT Workshop on Statistical Machine Translation. 3, 42

A. Attachments

A.1 Language lists

The source language is always the same:
en - English
In the following sections there are lists of *target* languages.

A.1.1 Languages from en-to-5

ar - Arabic
fr - French
es - Spanish
ru - Russian
zh - Chinese

A.1.2 Languages from en-to-36

XXX TODO: list with names
XXX TODO: two language groups used in the experiments