# A Systematic Analysis of Vocabulary and BPE Settings for Optimal Fine-tuning of NMT

Sharami, J. Pourmostafa Roshan; Shterionov, D.; Spronck, P.

*Published in:*
arXiv

*DOI:*
10.48550/arXiv.2303.00722

*Publication date:*
2023

*Document Version*
Peer reviewed version

# A Systematic Analysis of Vocabulary and BPE Settings for Optimal Fine-tuning of NMT: A Case Study of In-domain Translation

**Javad Pourmostafa Roshan Sharami, Dimitar Shterionov, and Pieter Spronck**
Department of Cognitive Science and Artificial Intelligence, Tilburg University, Tilburg, The Netherlands
{j.pourmostafa,d.shterionov,p.spronck}@tilburguniversity.edu



## Abstract

The effectiveness of Neural Machine Translation (NMT) models largely depends on the vocabulary used at training; small vocabularies can lead to out-of-vocabulary problems – large ones, to memory issues. Subword (SW) tokenization has been successfully employed to mitigate these issues. The choice of vocabulary and SW tokenization has a significant impact on both training and fine-tuning an NMT model. Fine-tuning is a common practice in optimizing an MT model with respect to new data. However, new data potentially introduces new words (or tokens), which, if not taken into consideration, may lead to suboptimal performance. In addition, the distribution of tokens in the new data can differ from the distribution of the original data. As such, the original SW tokenization model could be less suitable for the new data.

Through a systematic empirical evaluation, in this work we compare different strategies for SW tokenization and vocabulary generation with the ultimate goal to uncover an optimal setting for fine-tuning a domain-specific model. Furthermore, we developed several (in-domain) models, the best of which achieves 6 BLEU points improvement over the baseline.

## 1 Introduction

Fine-tuning is a common practice in optimizing an MT model with respect to new data. It can be either in the context of domain adaptation where an existing model is tuned for a certain domain (different from what the model was originally trained for) (Dakwale and Monz, 2017; Wang et al., 2019; Mahdieh et al., 2020), or simply to improve the performance of the model (Wang et al., 2017). Regardless of the fine-tuning objective, the newly introduced data brings in new information. For example, it could be the case that the new data contains new words that have not been seen in previous cycles of training the model; the word segmentation is suboptimal for the new data (Lim et al., 2018; Yeung, 2019; Sato et al., 2020). If not properly addressed, this new information may not have the desired effect on the MT system. For example, if new words are not included in the vocabulary a mismatch in vocabulary causing an out-of-vocabulary (OOV) issues will occur.

In this paper, we present an empirical evaluation of several models trained on different combinations of settings for generating the subwords and vocabularies, aiming to identify a best-case setup for fine-tuning an MT engine. That is, we aim to investigate which fine-tuning conditions (or settings) of a domain-specific model lead to the best performance. Our research focuses on fine-tuning for improving the translation quality of an engine rather than on domain adaptation. With our research we aim to answer the following main research question:

RQ1 Given a machine translation model and a fine-tuning data set, what is the optimal combination of subword generation approach and vocabulary?

In addition, we investigate two secondary research questions:

RQ2 How does fine-tuning, i.e. training an MT system on one data set, followed by training on another, compare to training an MT system with all data at once?

RQ3 What is the time reduction, if any, when fine-tuning, compared to training an MT system with all data at once?

To answer these questions we exploit two data sets ($\sim$0.9M and $\sim$1.1M parallel sentences) to train and one data set ($\sim$179k parallel sentences) to fine-tune several MT systems. This we do so that we can investigate how much we can improve a model trained on sufficient amount of data (approximately $1M$ sentence pairs) and then fine-tuned with extra data, which on its own would not be enough to train a model.

Each fine-tuned alternative, is trained on a different set of options of how the subwords and the vocabulary are created. For the fine-tuning process, we proposed a method to find the best fine-tuning setup based on available data. In our case study, for example, we have access to the data of both models (initial and fine-tuned), however, as already discussed in (Freitag and Al-Onaizan, 2016; Dakwale and Monz, 2017), initial models are mostly deployed in an application; thus data might not be available at the production time. As such, it is paramount to have a guideline based on the available data THAT determines how to best generate sub-words and vocabularies. In this work we use Byte-Pair Encoding (Sennrich et al., 2016a) for subword units.

We also trained models with all available data at once to assess whether fine-tuning has any benefit. We also evaluated MT systems trained on the small in-domain data set first and then fine-tuned on the larger sets. That is in order to test the hypothesis that a system trained on a small, focused data set and then fine-tuned on a larger set is worse than the other way round.[1]

It is noteworthy that the point of this research, however, is to investigate what is the best fine-tuning setup; and not to find the best model. That is, if we start from data set A (regardless of whether it is in-domain, out-of-domain, selected, or other) and then we fine tune on data set B, what should we take under consideration with respect to BPE and vocabulary.

This paper is organized as follows. We first cover the data we used in our experiments. In Section 3, we present our decision points. Our empirical experiments including details on the data, subwords, vocabulary, systems specifications, baselines and results are shown in Section 4. Section 5 presents our analysis with respect to the RQs. We cover the related work and conclude our work in Section 6 and 7, respectively.

## 2 Data

In our research we used two data sets: (i) two corpora of *selected in-domain data* in which sentence pairs have been selected from an out-of-domain corpus, i.e., $\sim$0.9M and $\sim$1.1M selected from the $\sim$31M sentences of collected WMT corpora[2] according to the data selection method presented in (Pourmostafa Roshan Sharami et al., 2022); and (ii) a small ($\sim$179K parallel sentences) *original in-domain data* set.

**Selected In-domain Data set** The selected in-domain data used for training the initial models (i.e., before doing fine-tuning), was introduced in (Pourmostafa Roshan Sharami et al., 2022). We used, in particular, $Top5$ and $Top6$ because they led to the best translation performance in their work. The selected in-domain data is the result of ranking out-of-domain sentences according to their similarity with an in-domain data set. The language pair is English-French. Furthermore, as indicated by their work, the volume of the data is sufficient to train MT systems with high translation quality.

**Original In-domain Data set** The original in-domain data we experimented with is the International Workshop on Spoken Language Translation (IWSLT) 2014 corpus (Cettolo et al., 2014). It is a collection of TED talks. To evaluate our models during training and find the models' performance we used one development set (dev2010) and two test sets (test2010 and test2011), respectively. IWSLT 2014 and WMT are commonly used in the context of Domain Adaptation (DA) as an in-domain data set (Axelrod et al., 2011; Luong and Manning, 2015; Chen et al., 2016; Wang et al., 2017; Pourmostafa Roshan Sharami et al., 2022), which facilitates for better replicability.

[1] This hypothesis has been proven in other domains, e.g. in robots control systems (Spronck et al., 2008).

[2] `http://statmt.org/wmt15/translation-task.html`

Table 1 shows statistics of the data we used in our experiments.

| Type of data | Name | Sentences |
|---|---|---|
| Selected in-domain | Top5 | 895k |
| | Top6 | ∼1M |
| Original in-domain (IWSLT 2014) | TED training | 179K |
| | TED dev2010 | 887 |
| | TED test2010 | 1664 |
| | TED test2011 | 818 |

**Table 1:** Summary of in-domain data sets (in-domain and out-of-domain), plus out-of-domain data sets.

## 3 Decision points

As noted in Section 1, with this work we aim to identify the most effective way of fine-tuning an MT system with respect to *subwords* and *vocabulary*. Consider a model $M$ trained on a data set $D$ which is representative for a certain domain $d$ and a fine-tuning data set $E$. The following decision points need to be made:

**Available data:** Choose which fine-tuning data set or a combination of data sets from $E$ should be used. As shown in previous work, using all available data is not always beneficial as it does not always contribute to the overall performance (especially when it comes to specific domains) while introducing computational overhead (Wang et al., 2019; Soto et al., 2020; Pourmostafa Roshan Sharami et al., 2022).

**Subwords:** Choose a model to construct subword units. Use either (i) the BPE model learned on the set $D$, and thus used in the training of model $M$ ($D_{BPE}$), (ii) learn a new BPE model on the selected fine-tuning data set ($E_{BPE}$) or (iii) learn a new BPE on the concatenation of $D$ and $E$ ($(D+E)_{BPE}$). This is mainly because there might be new and unique words in the fine-tuning data that did not appear in the set $D$, for which the original BPE model would be suboptimal. That can be the case if we tune an existing model toward a different domain other than $d$. However, comparing (ii) and (iii), training data from the original model may not be available and as such only (ii) could be a viable option.

**Vocabulary:** Choose the vocabulary, that is, either (i) use the vocabulary of the original model $M$ ($|D|$), (ii) extend it with tokens from the fine-tuning set $E$ ($|D+E|$) or (iii) create a new vocabulary from $E$ ($|E|$). This is important because a relevant vocabulary set is the pillar of the MT performance. Thus, finding such a set mitigates the impact of OOV and rare words.

These two factors – subwords and vocabulary – need to be jointly considered as each of them has a significant impact on the MT performance. To this end, we face an optimization problem along two dimensions. As the different options at each dimension are independent of the rest, [3] the solutions can be enumerated as combinations over these options. This gives us 9 combinations: apply the BPE model of the original MT system $D_{BPE}$ on the data for fine-tuning $E$ and train three systems with the different vocabulary options $|D|$, $|D+E|$ and $|E|$; train a new BPE model on the fine-tuning data, $E_{BPE}$ and apply it on the data, generating three different vocabularies ($|D|$, $|D+E|$ and $|E|$).

Following these decision points, given a fine-tuning data set we can consider three BPE models. With these models we (i) *create the vocabulary sources*; and (ii) *create the training sets for fine-tuning*. Typically these two processes are tied to each other, i.e. once the BPE model is learned and applied on the training data, the vocabulary is the set of subword units that appear in the (processed) data. However, this is not a hard constraint. That is, we can use a vocabulary that is derived from data processed with a different BPE model than the one of the training or fine-tuning data. For instance, data set $E$ can be processed with BPE $E_{BPE}$ but the vocabulary used for training can still be based on $D$ derived from applying $D_{BPE}$.

## 4 Experiments

To find the best possible setting for fine-tuning an in-domain model, we followed the decision points in Section 3, conducted experiments with the English-French data presented in Section 2 and compared our fine-tuned models with each other and to different baselines using BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and chrF (Popović, 2015).

Figure 1 illustrates our experimental approach.

According to the given data and our task, i.e., fine-tuning a model trained on $Top5$ or $Top6$ ($D =$

[3] It is noteworthy that we are aware that vocabulary sets in every combination are dependent on BPE models, and here we only defined "combinations" as independent compared to their other peers.

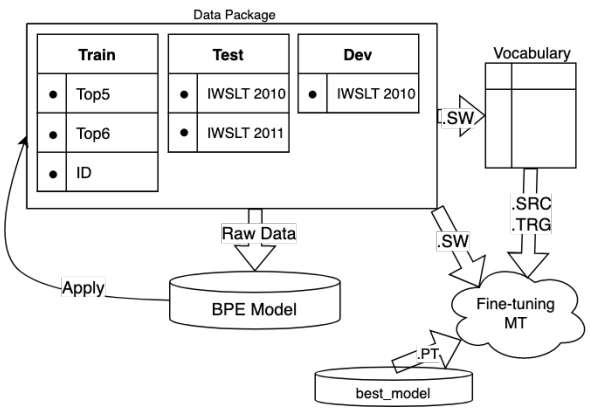


**Figure 1:** An overview of fine-tuning an MT. The data package shows the data sets we used for experiments. These can be used for training the initial models or fine-tuning the trained models.

$Top5$ or $D = Top6$) with original in-domain data ($E$ =in-domain data) we have the 54 options, 27 for each trained model. There are 3 options for the data source of the vocabulary ($Top5$ or $Top6$, $ID$ or the combination thereof) and 3 options for building the BPE which then impact the segmentation of the data used to build the vocabulary but also the segmentation of the fine-tuning data.[4] That is, there are 3 options to build a BPE model to be used for segmenting the data from which the vocabulary will be created and 3 options to build a BPE model to be used to segment the fine-tuning data.

It is noteworthy that the number of merge operations for BPE is 50K, and a separate BPE model was created for each source and target.

Figure 2 gives an overview of the different options.

Based on these 54 options we define three types of experiments. First, experiments in which the BPE model is built on either the original data $D = Top5/6$ or on the fine-tuning data $E = ID$ (but not on their combination) and the vocabulary is generated from the same data. With this type of experiments we investigate (hypothetical) scenarios where either there is no access to the original data set ($D$) nor vocabulary ($|D|$), and as such only the fine-tuning dataset ($E$) can be used, or these are available and we can exploit them directly without spending time or resources on processing the fine-tuning data to extract $|E|$. Second, experiments in which the vocabulary is built on both $D$ and $E$ ($Top5/6 + ID$). This would be considered a very favourable scenario, where both $D$ and $E$ are available and can be exploited jointly. Under such assumption, we also build baseline models on $D+E$ (see Section 4.2). Third, experiments in which the BPE used to segment the data on which the vocabulary is built is different from the BPE used to segment the fine-tuning data. These experiments would cover (hypothetical) scenarios in which the vocabulary is given, but it does not correspond to the exact way the subwords of the fine-tuning data have been generated. With this, third, set of experiments, we want to see whether it is possible to reach sufficient quality under limiting conditions.

To reduce the amount of computational time and resources we implemented 11 of the 27 sets of experiments. For the second type of experiments we excluded those in which different BPE models are used for the vocabulary and for the fine-tuning data. That is because given that both $D$ and $E$ are mixing BPE models would be unnecessary and impractical. Following the same reasoning, experiments from type 1 and 3 where two BPE models – one learned from $D+E$ and another learned either from $D$ or from $E$ – were excluded. These leaves us with the 11 experiments enumerated in Table 3.

### 4.1 NMT System Description

We used the OpenNMT-py[5] framework (Klein et al., 2017) for training as well as fine-tuning our NMT models. We fine-tuned transformer models (Vaswani et al., 2017) for a maximum of 200K steps; intermediate models were saved and validated every 1000 steps until reached convergence. We set an early stopping condition such that the fine-tuning process was stopped after 10 validations steps with no improvement. Since the models we fine-tuned were proposed by (Pourmostafa Roshan Sharami et al., 2022), we kept the NMT setup consistent and used the same hyperparameters. To run all NMT systems effectively and aligned with our research questions, we also set other hyperparameters as suggested by the OpenNMT-py community to simulate Google's default setup (Vaswani et al., 2017).

For fine-tuning, we distributed the training over three NVIDIA Tesla V100 GPUs. We encoded all data as a sequence of subwords units using the Byte Pair Encoding (BPE) algorithm (Sennrich et al., 2016a). The number of BPE merge operations

[4]As noted at the end of Section 3 we are not strictly constrained against mixing different word segmentations for the training data and for the vocabulary in a fine-tuning test case.

[5]`https://opennmt.net/OpenNMT-py/`

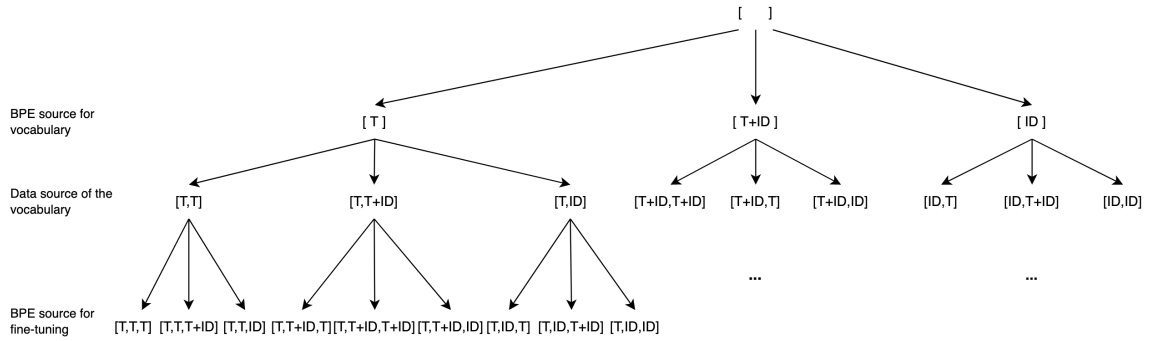


**Figure 2:** Tree's leaves show different combinations that we experimented with. For sake of simplicity in our report, we assume $Top5$ and $Top6$ as one single training data. That is, we experimented with both $Top5$ and $Top6$, however, we did not expand the tree for each of them. "$T$" also abbreviated from "$Top$".

is 50K, and a separate BPE model was created for each source and target.

### 4.2 Baseline Models

We compared our fine-tuned models not only to each other but also to three different categories of baselines:

(1) We only trained NMT systems on the original or in-domain data (with no further training/fine-tuning) to set up the following baselines. B1 -– an NMT model trained on the original in-domain data; B2 and B3 -– NMT models trained on the selected in-domain data Top5 and Top6 respectively. As stated before, we did not fine-tune any models to establish B1, B2, and B3, however, we use them as baselines to measure fine-tuning improvement over the IWSLT data set.

(2) We considered several state-of-the-art fine-tuned models. These are compared with our models to show the impact of our fine-tuning process. These are B4 (Luong and Manning, 2015), B5 (Axelrod et al., 2011), B6 (Chen et al., 2016), B7 (Wang et al., 2017). It is worth mentioning that we defined these baselines to evaluate the impact of the fine-tuning procedure itself before comparing the proposed combinations with each other.

(3) We mixed original ($Top5$ or $Top6$) and in-domain ($ID$) data sets and then trained a model. That is, we did not fine-tune any models in this category. This helps to show and measure the difference between merge and fine-tuning operations.

The baseline results are shown in Table 2.

| # | Test set 2010 | | | Test set 2011 | | |
|---|---|---|---|---|---|---|
| | BLEU↑ | TER↓ | CHRF2↑ | BLEU↑ | TER↓ | CHRF2↑ |
| B1 | 31.9 | 56.6 | 57.0 | 38.3 | 49.7 | 61.0 |
| B2 | 30.9 | 59.1 | 57.0 | 36.7 | 51.5 | 62.0 |
| B3 | 31.3 | 58.3 | 58.0 | 36.5 | 50.9 | 62.0 |
| B4 | 32.2 | N/A | N/A | 35.0 | N/A | N/A |
| B5 | 32.2 | 58.3 | N/A | 35.5 | N/A | N/A |
| B6 | 30.3 | 58.3 | N/A | 33.8 | N/A | N/A |
| B7 | 32.8 | 58.3 | N/A | 36.5 | N/A | N/A |
| B8 | 31.8 | 57.3 | 57.3 | 37.9 | 50.2 | 62.2 |
| B9 | 32.2 | 56.8 | 57.6 | 38.8 | 48.7 | 62.7 |

**Table 2:** Results of the baseline models. B1, B2, and B3 represent the NMT models trained on the original ID data, $Top5$, and $Top6$, respectively. B4, ..., B7 represent models fine-tuned in previous studies. B8 and B9 represent the models trained on the mixture of (ID, $Top5$) and (ID, $Top6$), respectively.

### 4.3 Results and Analysis

The performance of our fine-tuned MT systems is evaluated with respect to two test sets using case insensitive BLEU (Papineni et al., 2002), TER (Snover et al., 2006) and chrF2 (Popović, 2015) metrics, as implemented withing the sacreBLEU toolkit (Post, 2018). Our results are reported in Table 3). We also analyzed the results of different combinations with respect to statistical differences (see Section 5.1).

According to Table 3, the best-to-worst ranking of fine-tuning combinations for both $Top5$ and $Top6$ according to BLEU, TER and chrF2 are $C3, C1, C9, C11, C2, C10, C4, C8, C6, C7, C5$. In our experiments, $C3$ achieved the highest BLEU score among all combinations; except in two cases where other combinations achieved the same scores as follows: (1) $C1$ on test set 2010 for translation of $Top5$ and $Top6$; and (2) $C9$ on test set 2010 and 2011 for translation of $Top5$.

$C3$, $C1$, and $C9$ also achieved the highest chrF2 and lowest TER scores, suggesting the best setting for fine-tuning our in-domain model could be

a combination of (i) a BPE model created from the initial models' training data (i.e., the one used for training models – $top5$ or $Top6$); and (ii) a vocabulary set created from the fine-tuning data (i.e., $ID$); or the initial's models training data or the combination thereof. It is noteworthy that fine-tuning a model using this setting is not always feasible as we may not have access to the data used to train the original model. But, in case of availability, it is suggested to follow the $C3$'s setting.

The next promising fine-tuning setting is $C11$ which suggests combining the initial models' training data with the fine-tuning data. However, if they both are available, we prefer to follow the $C3, C1$, or $C9$ as these may have a better impact on the translation quality.

The fifth, sixth, and seventh suggested fine-tuning settings according to the evaluation metrics are $C2$, $C10$, and $C4$, respectively. These combinations indicated a model fine-tuned with (i) a BPE model created from the fine-tuning data (i.e., $ID$), and (ii) a vocabulary set created from either the fine-tuning data (i.e., $ID$) or the initial's models training data or the combination thereof could be employed to have an effective fine-tuning process. However, fine-tuning a model with $C10$ and $C4$ is only feasible if one has access to the data used to train models. That is, if the only available data is the one used to fine-tune a model, then it is recommended to follow $C2$.

The other suggested fine-tuning settings according to their evaluation scores in descending order are: $C8, C6, C7, C5$. It is worth mentioning that these setups used both the fine-tuning data (i.e., $ID$) and the initial models' training data, however, did not perform well in terms of translation quality. That shows the importance of BPE models and vocabulary for fine-tuning.

After analyzing our results and extracting a comprehensive guideline of data, BPE, and vocabulary for fine-tuning models, we compared combinations with three categories of baselines to show the effectiveness of the fine-tuning itself regardless of the different fine-tuning combinations. According to the results summarised in Table 2 and Table 3, all fine-tuned models outperformed the baselines. For example, C3 [6] BLEU scores were increased by roughly 13% (31.9 to 36.1), 16.8% (30.9 to 36.1), 10% (32.8 to 36.1), and 13.5% (31.8 to 36.1) compared to baselines $B1, B2, B7$, and $B8$ respectively. [7] These figures indicate that the fine-tuning process was a better option than training a model at once. These results raise an interesting question about how data should be fed to the neural network at training to achieve optimal performance (both in terms of translation quality as well as training time).

[6] Top5←ID and evaluated with test set 2010

## 5 Discussions

In this section, first we discuss the pairwise statistical significance of the evaluation scores between the fine-tuned NMT models. Second, we investigate the training time with fine-tuning compared to the training time of baselines. Third, we show to what extent the choice of an initial model for fine-tuning affects the performance of translation.

### 5.1 Statistical Significance Test

We computed pairwise statistical significance of the results shown in Table 3 in terms of BLEU scores by using bootstrap resampling and 95% confidence interval for both test sets (2010 and 2011) based on 1000 iterations, and samples of 300 sentences. According to the test output, most fine-tuned models have a statistically significant difference except those systems pairs listed in Table 4.

In addition to the analysis presented in Section 4.3, this shows two main points: (1) if there is no access to both initial and fine-tuned models' data we can achieve quite similar performance only from the initial data. For example, the initial model –$Top5$– fine-tuned as per $C1$ and $C3$ on the 2010 test set have no differences in terms of BLEU score. (2) having one single data set versus two different ones for creating BPE models and vocabulary may not always have a significant impact on the the model performance. For example, while $C2$ was trained with one single data set (ID), $C4$ employed both ID and Top5 and still achieved the same performance (on test set 2010).

### 5.2 Training Time

In Table 5 we present the running time (RT) for training the baselines ($B1, B2, B3, B8$ and $B9$) and for fine-tuning for the best model ($C3$) for both $Top5$ and $Top6$.[8] Fine-tuning time is about

[7] We chose B7 as the representative of the second baseline category because it achieved the highest BLEU scores among the fine-tuned baselines.

[8] Fine-tuning for all models and SW and vocabulary combinations took approximately 1h and 30 minutes. As such we limit our discussion to $C3$.

| # | X | Y | Z | Models | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | | Top5 ← ID | | | | | | Top6 ← ID | | | | | |
| | | | | Test set 2010 | | | Test set 2011 | | | Test set 2010 | | | Test set 2011 | | |
| | | | | BLEU↑ | TER↓ | chrF2↑ | BLEU↑ | TER↓ | chrF2↑ | BLEU↑ | TER↓ | chrF2↑ | BLEU↑ | TER↓ | chrF2 |
| C1 | Top5/6 | Top5/6 | Top5/6 | 36.1 | 52.4 | 60.3 | 43.7 | 44.1 | 66.1 | 36.4 | 52.3 | 60.4 | 44.1 | 43.7 | 66.4 |
| C2 | ID | ID | ID | 35.9 | 52.5 | 60 | 42.8 | 44.9 | 65.2 | 36.1 | 52.5 | 60.0 | 44.0 | 43.7 | 65.8 |
| C3 | Top5/6 | ID | Top5/6 | 36.1 | 52.4 | 60.4 | 44.0 | 43.6 | 66.6 | 36.4 | 52.2 | 60.4 | 44.4 | 43.5 | 66.4 |
| C4 | ID | Top5/6 | ID | 35.5 | 52.8 | 59.7 | 43.3 | 44.5 | 65.6 | 35.5 | 53.0 | 59.6 | 43.6 | 44.1 | 65.7 |
| C5 | Top5/6 | Top5/6 | ID | 33.4 | 53.8 | 58.6 | 40.7 | 45.4 | 64.2 | 33.6 | 53.3 | 58.7 | 40.9 | 45.5 | 64.2 |
| C6 | ID | Top5/6 | Top5/6 | 35.4 | 53.0 | 59.7 | 43.5 | 44.6 | 65.5 | 35.5 | 53.6 | 59.9 | 43.3 | 44.4 | 65.7 |
| C7 | Top5/6 | ID | ID | 33.4 | 53.3 | 58.6 | 40.9 | 45.1 | 64.2 | 33.2 | 53.4 | 58.2 | 40.2 | 45.4 | 64.0 |
| C8 | ID | ID | Top5/6 | 35.4 | 52.8 | 59.6 | 42.8 | 44.6 | 65.1 | 35.1 | 53.1 | 59.6 | 43.3 | 44.4 | 65.7 |
| C9 | Top5/6 | Top5/6+ID | Top5/6 | 36.1 | 52.4 | 60.1 | 44.0 | 43.6 | 66.4 | 35.9 | 52.6 | 60.4 | 44.3 | 43.5 | 66.6 |
| C10 | ID | Top5/6+ID | ID | 35.6 | 52.7 | 59.9 | 43.0 | 44.9 | 65.2 | 36.1 | 52.3 | 60.2 | 44.0 | 43.6 | 66.0 |
| C11 | Top5/6+ID | Top5/6+ID | Top5/6+ID | 36.0 | 52.5 | 60.1 | 44.1 | 43.6 | 65.9 | 36.4 | 52.3 | 60.3 | 44.0 | 43.8 | 66.4 |

**Table 3:** Evaluation scores of the fine-tuned NMT systems. X represents the source of the BPE model used to create vocabulary; Y represents the source of the vocabulary set and Z represents the source of the BPE model used to create the data for fine-tuning. With $TopN \leftarrow ID$ we denote that a model trained on $TopN$ is fine-tuned on $ID$ data.

| | Test Set 2010 | Test Set 2011 |
|---|---|---|
| **Top5** | (C1, C3)<br>(C6, C8)<br>(C2, C4)<br>(C5, C7) | (C1, C6)<br>(C4, C5) |
| **Top6** | (C1, C3)<br>(C6, C8) | (C1, C6)<br>(C4, C5) |

**Table 4:** Results of systems pairs that are not statistically significant (for $p < 0.05$). (CX, CY) means models that fined tuned with the setup suggested in combinations CX and CY have no statistically significant difference based on 300 samples.

| # | Complete RT D:H:M | Step | Best model RT D:H:M | Step |
|---|---|---|---|---|
| B1 | 00:03:53 | 18,000 | 00:00:50 | 5,000 |
| B2 | 00:10:33 | 35,000 | 00:05:50 | 20,000 |
| B3 | 00:08:20 | 35,000 | 00:04:26 | 19,000 |
| B8 | 00:05:47 | 23,000 | 00:03:16 | 13,000 |
| B9 | 00:06:17 | 25,000 | 00:04:45 | 15,000 |
| C3-Top5 | 00:01:33 | 12,000 | 00:00:17 | 2,000 |
| C3-Top6 | 00:01:32 | 12,000 | 00:00:16 | 2,000 |

**Table 5:** Running time (RT) for training and fine-tuning. The first baselines B1, B2 and B3 are trained on original in-domain data, $Top5$, and Top6 accordingly; baselines B8 and B9 are combinations of ID and Top5 or ID and Top6. $C3 - Top5$ and $C3 - Top6$ indicate the fine-tuning with the best combination ($C3$) of models trained on $Top5$ and $Top6$ accordingly.

1 hour and 30 minutes compared to the training time of an MT system with all data at once ($B8$ or $B9$) which is about 5 or 6 hours. That shows that the time for fine-tuning is only a fraction (27%) of the time for training models on all data at once. On the one hand, compared to the sum of training and fine-tuning times, that is B2 or B3 followed by C3 which amount at 12 hours and 6 minutes and 9 hours and 52 minutes accordingly, training a model "from scratch" is preferable. On the other hand, training "from scratch" does not achieve the same performance as with fine-tuning (as already stated this in Section 4.3).

### 5.3 Reverse Fine-tuning

We also assess whether the initial model for fine-tuning in our case study ($Top5/6$) is effective or we may need to reverse the order in which data is presented for training. That is training MT systems on the original in-domain data followed by fine-tuning them on the selected in-domain data. It is noteworthy that we conducted this experiment to monitor the performance as well as the sensitivity of the fine-tuning process toward the initial model.

According to Table 6, all fine-tuned models outperformed the reverse fine-tuned models. That means, starting with the selected in-domain models ($Top5/6$) is more efficient than starting with the original in-domain. This also reveals the fact that trained on large data (such as $Top5/6$) prevents overfitting on the small in-domain data (such as $ID$). This transformation to the new parameters can cause a drop in models performance for the test instances from the initial data (Li and Hoiem, 2016; Dakwale and Monz, 2017). However, this is

not entirely true in our case study as we worked on one domain for the entire research. That is, both initial and fine-tuning data are from one specific domain, and only the first one is relatively large. In the future we plan to expand on more domains and assess the impact of different data quantities and domain-specificity.

| # | Test set 2010 | | | Test set 2011 | | |
|---|---|---|---|---|---|---|
| | BLEU↑ | TER↓ | CHRF2↑ | BLEU↑ | TER↓ | CHRF2↑ |
| $ID \leftarrow Top5$ | 30.8 | 59.4 | 56.8 | 36.3 | 52.3 | 61.1 |
| $ID \leftarrow Top6$ | 31.5 | 58.2 | 57.3 | 37.8 | 50.3 | 61.9 |

**Table 6:** The results of reverse fine-tuning. With $ID \leftarrow TopN$ we denote that a model trained on $I$ is fine-tuned on $TopN$ data.

## 6 Related Work

There is a significant amount of research on the topic of fine-tuning MT. Most prior studies have investigated adapting models to a different domain. That is, they first employed a large out-of-domain data and then fine-tuned it on small in-domain data. Among others, (Luong et al., 2015) did the first successful work, where they trained a model on English-German general-domain data and then fine-tuned it on a new domain data (conversational) in the same languages. They claimed an increase of 3.8 BLEU points compared to the original model (25.6 to 29.4) without further training.

Another method to improve the translation performance on the new domain without degrading the performance on the generic domain test set was proposed in (Freitag and Al-Onaizan, 2016). To this end, they ensembled the fine-tuned model with the already trained baseline model; and evaluated their method by IWSLT 2015 evaluation campaign (Cettolo et al., 2015). The authors reported a gain of 7.2 BLEU points on the in-domain test set while still retaining the performance on the out-of-domain test set.

Following that, (Freitag and Al-Onaizan, 2016) demonstrated that while an ensemble approach for fine-tuning seems a good option, the performance of the fine-tuned models still drops for the generic domain task, especially when it comes to domain-specific contexts (e.g., medical and legal domain). This is mainly because the in-domain data set at topic or genre level (van der Wees et al., 2015), comprises new vocabulary and linguistic features that are different from the generic data (Koehn and Knowles, 2017). To fix this problem, they proposed a fine-tuning method based on knowledge distillation (Hinton et al., 2015).

There is also other research carried out to solve the vocabulary mismatches in the context of fine-tuning. For instance, (Sato et al., 2020) proposed a method to adapt the embedding layers of the initial model to the target domain. They performed this by projecting the general word embedding obtained from target-domain monolingual data onto source-domain embedding. The authors reported 3.86 and 3.28 BLEU points gain in English→Japanese and German→English translation, respectively.

As segmenting words and generating vocabulary hugely impact the MT performance (Ataman and Federico, 2018) extensively evaluated the problem of segmenting words at a subword level and compared two word segmentation methods: Byte-Pair Encoding (BPE) (Sennrich et al., 2016b) and the Linguistically-Motivated Vocabulary Reduction (LMVR) (Ataman et al., 2017) for NMT. They compared these approaches in five morphologically-rich languages and reported that LMVR achieved better performance in the tested languages.

## 7 Conclusion and Future Work

In this paper, we conducted a systematic analysis based on a commonly used domain-specific data set (IWSLT 2014) to find the optimal combination of subword generation approach and vocabulary for fine-tuning NMT models. In addition to comparing the performance of 22 models (11 options for two training sets), we investigated how fine-tuning impacts training time and MT performance, compared to training an MT model with all data at once. Through our empirical evaluation, we have created a state-of-the-art model for in-domain translation that can be employed in different contexts, among others, multilingual domain adaptation (Cooper Stickland et al., 2021).

Considering the available data, we present two decision points (i.e., subwords and vocabulary) that need to be made prior to fine-tuning, along with a third one about crossing the BPE source for the training/fine-tuning data and for the vocabulary. Our experiments outline a roadmap with three possible options for fine-tuning in-domain models as follows: (1) In case both the initial model's data ($D$) and fine-tuning data ($E$) are available, it

might be effective to train BPE models and create vocabulary using $D$ and $E$, respectively. (ii) Otherwise, it might be viable to create BPE and vocabulary based on $D$; and (iii) if the last two options were not possible, it suggests creating the decision points all based on $E$. It is worth noting that all fine-tuning strategies the initial MT models improved the baselines, with the maximum gained of 6 BLEU points (30.9 to 36.1).

In our future work, we intend to improve the generalization of our pre-trained (in-domain) models by further training on an out-of-domain corpus, so it possibly enables these models to translate generic inputs as well as their specialized context without forgetting what they already learned. Another research direction would be to investigate the proposed decision points on other language pairs and domains.

The data and fine-tuned models are available at: `https://github.com/JoyeBright/FT-IWSLT2014-BPEVocab`.

## References


[Ataman and Federico2018] Ataman, Duygu and Marcello Federico. 2018. An evaluation of two vocabulary reduction methods for neural machine translation. In *Proceedings of the 13th Conference of the Association for Machine Translation in the Americas (Volume 1: Research Track)*, pages 97–110, Boston, MA, March. Association for Machine Translation in the Americas.

[Ataman et al.2017] Ataman, Duygu, Matteo Negri, Marco Turchi, and Marcello Federico. 2017. Linguistically motivated vocabulary reduction for neural machine translation from turkish to english. *The Prague Bulletin of Mathematical Linguistics*, 108, 06.

[Axelrod et al.2011] Axelrod, Amittai, Xiaodong He, and Jianfeng Gao. 2011. Domain adaptation via pseudo in-domain data selection. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 355–362, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

[Cettolo et al.2014] Cettolo, Mauro, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, and Marcello Federico. 2014. Report on the 11th IWSLT evaluation campaign. In *Proceedings of the 11th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 2–17, Lake Tahoe, California, December 4-5.

[Cettolo et al.2015] Cettolo, Mauro, Jan Niehues, Sebastian Stüker, Luisa Bentivogli, Roldano Cattoni, and Marcello Federico. 2015. The IWSLT 2015 evaluation campaign. In *Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 2–14, Da Nang, Vietnam, December 3-4.

[Chen et al.2016] Chen, Boxing, Roland Kuhn, George Foster, Colin Cherry, and Fei Huang. 2016. Bilingual methods for adaptive training data selection for machine translation. In *Conferences of the Association for Machine Translation in the Americas: MT Researchers' Track*, pages 93–106, Austin, TX, USA, October 28 - November 1. The Association for Machine Translation in the Americas.

[Cooper Stickland et al.2021] Cooper Stickland, Asa, Alexandre Berard, and Vassilina Nikoulina. 2021. Multilingual domain adaptation for NMT: Decoupling language and domain information with adapters. In *Proceedings of the Sixth Conference on Machine Translation*, pages 578–598, Online, November. Association for Computational Linguistics.

[Dakwale and Monz2017] Dakwale, Praveen and Christof Monz. 2017. Fine-tuning for neural machine translation with limited degradation across in- and out-of-domain data. In *Proceedings of the 16th Machine Translation Summit (MT-Summit 2017)*, pages 156–169.

[Freitag and Al-Onaizan2016] Freitag, Markus and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation.

[Hinton et al.2015] Hinton, Geoffrey, Oriol Vinyals, and Jeff Dean. 2015. Distilling the knowledge in a neural network.

[Klein et al.2017] Klein, Guillaume, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72, Vancouver, Canada, July. Association for Computational Linguistics.

[Koehn and Knowles2017] Koehn, Philipp and Rebecca Knowles. 2017. Six challenges for neural machine translation. In *Proceedings of the First Workshop on Neural Machine Translation*, pages 28–39, Vancouver, August. Association for Computational Linguistics.

[Li and Hoiem2016] Li, Zhizhong and Derek Hoiem. 2016. Learning without forgetting. In Leibe, Bastian, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision - 14th European Conference, ECCV 2016, Proceedings*, Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), pages 614–629, Germany. Springer. Funding Information: This work is supported in part by NSF Awards 14-46765, 10-53768 and ONR MURIN000014-16-1-2007. Publisher Copyright: © Springer International Publishing AG 2016.; 14th

European Conference on Computer Vision, ECCV 2016 ; Conference date: 11-10-2016 Through 14-10-2016.

[Lim et al.2018] Lim, Robert, Kenneth Heafield, Hieu Hoang, Mark Briers, and Allen Malony. 2018. Exploring hyper-parameter optimization for neural machine translation on gpu architectures.

[Luong and Manning2015] Luong, Minh-Thang and Christopher Manning. 2015. Stanford neural machine translation systems for spoken language domains. In *Proceedings of the 12th International Workshop on Spoken Language Translation: Evaluation Campaign*, pages 76–79, Da Nang, Vietnam, December 3-4.

[Luong et al.2015] Luong, Thang, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, September. Association for Computational Linguistics.

[Mahdieh et al.2020] Mahdieh, Mahdis, Mia Xu Chen, Yuan Cao, and Orhan Firat. 2020. Rapid domain adaptation for machine translation with monolingual data.

[Papineni et al.2002] Papineni, Kishore, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A Method for Automatic Evaluation of Machine Translation. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics (ACL 2002)*, pages 311–318, Philadelphia, Pennsylvania, USA, July. Association for Computational Linguistics.

[Popović2015] Popović, Maja. 2015. chrF: character n-gram F-score for automatic MT evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395, Lisbon, Portugal, September. Association for Computational Linguistics.

[Post2018] Post, Matt. 2018. 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. Association for Computational Linguistics.

[Pourmostafa Roshan Sharami et al.2022] Pourmostafa Roshan Sharami, Javad, Dimitar Sterionov, and Pieter Spronck. 2022. Selecting parallel in-domain sentences for neural machine translation using monolingual texts. *Computational Linguistics in the Netherlands Journal*, 11:213–230, Feb.

[Sato et al.2020] Sato, Shoetsu, Jin Sakuma, Naoki Yoshinaga, Masashi Toyoda, and Masaru Kitsuregawa. 2020. Vocabulary adaptation for domain adaptation in neural machine translation. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4269–4279, Online, November. Association for Computational Linguistics.

[Sennrich et al.2016a] Sennrich, Rico, Barry Haddow, and Alexandra Birch. 2016a. Controlling politeness in neural machine translation via side constraints. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 35–40, San Diego, California, June. Association for Computational Linguistics.

[Sennrich et al.2016b] Sennrich, Rico, Barry Haddow, and Alexandra Birch. 2016b. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany, August. Association for Computational Linguistics.

[Snover et al.2006] Snover, Matthew, Bonnie Dorr, Rich Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas: Technical Papers*, pages 223–231, Cambridge, Massachusetts, USA, August 8-12. Association for Machine Translation in the Americas.

[Soto et al.2020] Soto, Xabier, Dimitar Shterionov, Alberto Poncelas, and Andy Way. 2020. Selecting backtranslated data from multiple sources for improved neural machine translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3898–3908, Online, July. Association for Computational Linguistics.

[Spronck et al.2008] Spronck, Pieter, Ida Sprinkhuizen-Kuyper, and Eric Postma. 2008. Deca: The doping-driven evolutionary control algorithm. *Applied Artificial Intelligence*, 22(3):169–197, mar.

[van der Wees et al.2015] van der Wees, Marlies, Arianna Bisazza, Wouter Weerkamp, and Christof Monz. 2015. What's in a domain? analyzing genre and topic differences in statistical machine translation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 560–566, Beijing, China, July. Association for Computational Linguistics.

[Vaswani et al.2017] Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Guyon, I., U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

[Wang et al.2017] Wang, Rui, Andrew Finch, Masao Utiyama, and Eiichiro Sumita. 2017. Sentence embedding for neural machine translation domain adaptation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 560–566, Vancouver, Canada, July. Association for Computational Linguistics.

[Wang et al.2019] Wang, Xu, Chunyang Chen, and Zhenchang Xing. 2019. Domain-specific machine translation with recurrent neural network for software localization. *Empirical Software Engineering*, 24(6):3514–3545, December.

[Yeung2019] Yeung, Chin Man. 2019. Effects of inserting domain vocabulary and fine-tuning bert for german legal language, November.