# $m^4Adapter$: Multilingual Multi-Domain Adaptation for Machine Translation with a Meta-Adapter

**Wen Lai**[1], **Alexandra Chronopoulou**[1,2], **Alexander Fraser**[1,2]

[1]Center for Information and Language Processing, LMU Munich, Germany
[2]Munich Center for Machine Learning, Germany
{lavine, achron, fraser}@cis.lmu.de

## Abstract

Multilingual neural machine translation models (MNMT) yield state-of-the-art performance when evaluated on data from a domain and language pair seen at training time. However, when a MNMT model is used to translate under domain shift or to a new language pair, performance drops dramatically. We consider a very challenging scenario: adapting the MNMT model both to a new domain and to a new language pair at the same time. In this paper, we propose $m^4Adapter$ (Multilingual Multi-Domain Adaptation for Machine Translation with a Meta-Adapter), which combines domain and language knowledge using meta-learning with adapters. We present results showing that our approach is a parameter-efficient solution which effectively adapts a model to both a new language pair and a new domain, while outperforming other adapter methods. An ablation study also shows that our approach more effectively transfers domain knowledge across different languages and language information across different domains.[1]

## 1 Introduction

Multilingual neural machine translation (MNMT; Johnson et al., 2017; Aharoni et al., 2019; Fan et al., 2021), uses a single model to handle translation between multiple language pairs. There are two reasons why MNMT is appealing: first, it has been proved to be effective on transferring knowledge from high-resource languages to low-resource languages, especially in zero-shot scenarios (Gu et al., 2019; Zhang et al., 2020); second, it significantly reduces training and inference cost, as it requires training only a single multilingual model, instead of a separate model for each language pair.

Adapting MNMT models to multiple domains is still a challenging task, particularly when domains are distant to the domain of the training corpus. One approach to address this is *fine-tuning* the model on out-of-domain data for NMT (Freitag and Al-Onaizan, 2016; Dakwale and Monz, 2017). Another approach is to use lightweight, learnable units inserted between transformer layers, which are called *adapters* (Bapna and Firat, 2019) for each new domain. Similarly, there is research work on adapting MNMT models to a new language pair using fine-tuning (Neubig and Hu, 2018) and adapters (Bapna and Firat, 2019; Philip et al., 2020; Cooper Stickland et al., 2021b).

Although effective, the above approaches have some limitations: i) Fine-tuning methods require updating the parameters of the whole model for each new domain, which is costly; ii) when fine-tuning on a new domain, catastrophic forgetting (McCloskey and Cohen, 1989) reduces the performance on all other domains, and proves to be a significant issue when data resources are limited. iii) adapter-based approaches require training domain adapters for each domain and language adapters for all languages, which also becomes parameter-inefficient when adapting to a new domain and a new language because the parameters scale linearly with the number of domains and languages.

In recent work, Cooper Stickland et al. (2021a) compose language adapters and domain adapters in MNMT and explore to what extent domain knowledge can be transferred across languages. They find that it is hard to decouple language knowledge from domain knowledge and that adapters often cause the 'off-target' problem (i.e., translating into a wrong target language (Zhang et al., 2020)) when new domains and new language pairs are combined together. They address this problem by using additional in-domain monolingual data to generate synthetic data (i.e., back-translation; Sennrich et al., 2016) and randomly dropping some domain adapter layers (AdapterDrop; Rücklé et al., 2021).

Motivated by Cooper Stickland et al. (2021a),

[1]Our source code is available at https://github.com/lavine-lmu/m4Adapter



*Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 4282–4296
December 7-11, 2022 

we consider a challenging scenario: adapting a MNMT model to multiple new domains and new language directions simultaneously in low-resource settings without using extra monolingual data for back-translation. This scenario could arise when one tries to translate a domain-specific corpus with a commercial translation system. Using our approach, we adapt a model to a new domain and a new language pair using just 500 domain- and language-specific sentences.

To this end, we propose $m^4Adapter$ (Multilingual Multi-Domain Adaptation for Machine Translation with Meta-Adapter), which facilitates the transfer between different domains and languages using meta-learning (Finn et al., 2017) with adapters. Our hypothesis is that we can formulate the task, which is to adapt to new languages and domains, as a multi-task learning problem (and denote it as $D_i$-$L_1$-$L_2$, which stands for translating from a language $L_1$ to a language $L_2$ *in a specific domain* $D_i$). Our approach is two-step: initially, we perform meta-learning with adapters to efficiently learn parameters in a shared representation space across multiple tasks using a small amount of training data (5000 samples); we refer to this as the meta-training step. Then, we fine-tune the trained model to a new domain and language pair simultaneously using an even smaller dataset (500 samples); we refer to this as the meta-adaptation step.

In this work, we make the following contributions: i) We present $m^4Adapter$, a meta-learning approach with adapters that can easily adapt to new domains and languages using a single MNMT model. Experimental results show that $m^4Adapter$ outperforms strong baselines. ii) Through an ablation study, we show that using $m^4Adapter$, domain knowledge can be transferred across languages and language knowledge can also be transferred across domains without using target-language monolingual data for back-translation (unlike the work of Cooper Stickland et al., 2021a). iii) To the best of our knowledge, this paper is the first work to explore meta-learning for MNMT adaptation.

## 2 Related Work

**Domain Adaptation in NMT.** Existing work on domain adaptation for machine translation can be categorized into two types: *data-centric* and *model-centric* approaches (Chu and Wang, 2018). The former focus on maximizing the use of in-domain monolingual, synthetic, and parallel data (Domhan and Hieber, 2017; Park et al., 2017; van der Wees et al., 2017), while the latter design specific training objectives, model architectures or decoding algorithms for domain adaptation (Khayrallah et al., 2017; Gu et al., 2019; Park et al., 2022). In the case of MNMT, adapting to new domains is more challenging because it needs to take into account transfer between languages (Chu and Dabre, 2019; Cooper Stickland et al., 2021a).

**Meta-Learning for NMT.** Meta-learning (Finn et al., 2017), which aims to learn a generally useful model by training on a distribution of tasks, is highly effective for fast adaptation and has recently been shown to be beneficial for many NLP tasks (Lee et al., 2022). Gu et al. (2018) first introduce a model-agnostic meta-learning algorithm (MAML; Finn et al., 2017) for low-resource machine translation. Sharaf et al. (2020), Zhan et al. (2021) and Lai et al. (2022) formulate domain adaptation for NMT as a meta-learning task, and show effective performance on adapting to new domains. Our approach leverages meta-learning to adapt a MNMT model to a *new domain* and to a *new language pair* at the same time.

**Adapters for NMT.** Bapna and Firat (2019) train *language-pair* adapters on top of a pre-trained generic MNMT model, in order to recover lost performance on high-resource language pairs compared to bilingual NMT models. Philip et al. (2020) train adapters for *each language* and show that adding them to a trained model improves the performance of zero-shot translation. Chronopoulou et al. (2022) train adapters for *each language family* and show promising results on multilingual machine translation. Cooper Stickland et al. (2021b) train *language-agnostic* adapters to efficiently fine-tune a pre-trained model for many language pairs. More recently, Cooper Stickland et al. (2021a) stack language adapters and domain adapters on top of an MNMT model and they conclude that it is not possible to transfer domain knowledge across languages, except by employing back-translation which requires significant in-domain resources. In this work, we introduce adapters into the meta-learning algorithm and show that this approach permits transfer between domains and languages.

Our work is mostly related to Cooper Stickland et al. (2021a), however we note several differences: i) we study a more realistic scenario: the corpus

of each domain and language pair is low-resource (i.e., the meta-training corpus in each domain for each language pair is limited to 5000 sentences and the fine-tuning corpus to 500 sentences), which is easier to obtain; ii) our approach can simultaneously adapt to new domains and new language pairs *without using back-translation*. iii) we also show that $m^4Adapter$ can transfer domain information across different languages and language knowledge across different domains through a detailed ablation analysis.

## 3 Method

Our goal is to efficiently adapt an MNMT model to new domains and languages. We propose a novel approach, $m^4Adapter$, which formulates the multilingual multi-domain adaptation task as a multi-task learning problem. To address it, we propose a 2-step approach, which combines *meta-learning* and *meta-adaptation* with adapters. Our approach permits sharing parameters across different tasks. The two steps are explained in Subsections 3.1 and 3.2.

### 3.1 Meta-Training

The goal of meta-learning is to obtain a model that can easily adapt to new tasks. To this end, we meta-train adapters in order to find a good initialization of our model's parameters using a small training dataset of source tasks $\{\mathcal{T}_1, \ldots, \mathcal{T}_t\}$.

We first select $m$ tasks, as we describe in § 3.1.1. Then, for each of the $m$ sampled tasks, we sample $n$ examples. We explain the task sampling strategy in § 3.1.2. This way, we set up the *m-way-n-shot* task. After setting up the task, we use a meta-learning algorithm, which we describe in § 3.1.3, to meta-learn the parameters of the adapter layers. The architecture of the adapters and their optimization objective are presented in § 3.1.4. Algorithm 1 details the meta-training process of our approach.

#### 3.1.1 Task Definition

Motivated by the work of Tarunesh et al. (2021), where a multilingual multi-task NLP task is regarded as a Task-Language pair (TLP), we address multilingual multi-domain translation as a multi-task learning problem. Specifically, a translation task in a specific textual domain corresponds to a Domain-Language-Pair (**DLP**). For example, an English-Serbian translation task in the 'Ubuntu' domain is denoted as a DLP 'Ubuntu-en-sr'. Given $d$ domains and $l$ languages, we have $d \cdot l \cdot (l-1)$ tasks of this form.[2] We denote the proportion of the dataset size of all DLPs for the $i^{th}$ DLP as $s_i = |\mathcal{D}^i_{train}| / \left( \sum_{a=1}^{n} |\mathcal{D}^a_{train}| \right)$, where $s_i$ will be used in temperature-based sampling (see more details in § 3.1.2). The probability of sampling a batch from the $i^{th}$ DLP during meta-training is denoted as $P_{\mathcal{D}}(i)$. The distribution over all DLPs, is a multinomial (which we denote as $\mathcal{M}$) over $P_{\mathcal{D}}(i)$: $\mathcal{M} \sim P_{\mathcal{D}}(i)$.

#### 3.1.2 Task Sampling

Given $d$ domains and $l$ languages, we sample some DLPs per batch among all $d \cdot l \cdot (l-1)$ tasks. We consider a standard *m-way-n-shot* meta-learning scenario: assuming access to $d \cdot l \cdot (l-1)$ DLPs, a *m-way-n-shot* task is created by first sampling $m$ DLPs ($m \ll l \cdot (l-1)$); then, for each of the $m$ sampled DLPs, $(n+q)$ examples of each DLP are selected; the $n$ examples for each DLP serve as the support set to update the parameter of pre-trained model, while $q$ examples constitute the query set to evaluate the model.

Task sampling is an essential step for meta-learning. Traditional meta-learning methods sample the tasks uniformly (Sharaf et al., 2020), through ordered curriculum (Zhan et al., 2021), or dynamically adjust the sampled dataset according to the model parameters (parameterized sampling strategy, Tarunesh et al., 2021). We do not employ these strategies for the following reasons: i) sampling uniformly is simple but does not consider the distribution of the unbalanced data; ii) Although effective, curriculum-based and parameterized sampling consider features of all $d \cdot l \cdot (l-1)$ DLPs. Because of this, the amount of DLPs is growing exponentially with the number of languages and domains. In contrast, we follow a temperature-based heuristic sampling strategy (Aharoni et al., 2019), which defines the probability of any dataset as a function of its size. Specifically, given $s_i$ as the percentage of the $i^{th}$ DLP in all DLPs, we compute the following probability of the $i^{th}$ DLP to be sampled:

$$P_{\mathcal{D}}(i) = s_i^{1/\tau} / \left( \sum_{a=1}^{n} s_a^{1/\tau} \right)$$

where $\tau$ is a temperature parameter. $\tau = 1$ means that each DLP is sampled in proportion to the size of the corresponding dataset. $\tau \to \infty$ refers to sampling DLPs uniformly.

[2] Given $l$ languages, we focus on complete translation between $l \cdot (l-1)$ language directions.

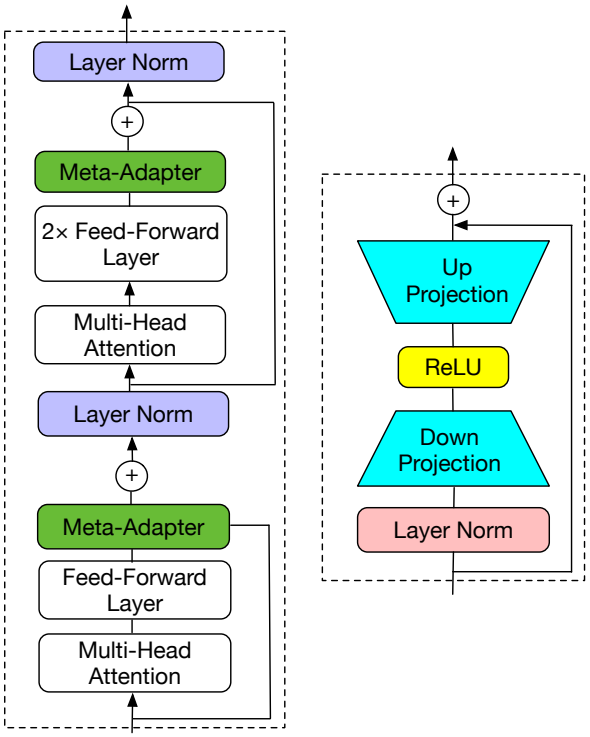



Figure 1: $m^4Adapter$ architecture.

#### 3.1.3 Meta-Learning Algorithm

Given $\theta$ as the parameters of the pre-trained model, $\psi$ as the parameters of the adapters, MAML aims to minimize the following objective:

$$\min_{\psi} \sum_{\mathcal{T}_i \sim \mathcal{M}} \mathcal{L}_i \left( U_i^k(\theta, \psi) \right)$$

where $\mathcal{M}$ is the multinomial distribution over DLPs, $\mathcal{L}_i$ is the loss function and $U_i^k$ is a function which keeps $\theta$ frozen and only returns $\psi$ after $k$ gradient updates calculated on batches sampled from $\mathcal{T}_i$. Note that, to minimize this goal, the traditional MAML algorithm requires computing gradients of the form $\frac{\partial}{\partial \psi} U_i^k(\psi)$, which leads to the costly computation of second-order derivatives. To this end, we follow *Reptile* (Nichol et al., 2018), an alternative first-order meta-learning algorithm that uses a simple update rule:

$$\psi \leftarrow \psi + \beta \frac{1}{|\{\mathcal{T}_i\}|} \sum_{\mathcal{T}_i \sim \mathcal{M}} (\psi_i^{(k)} - \psi)$$

where $\psi_i^{(k)}$ is $U_i^k(\theta, \psi)$ and $\beta$ is a hyper-parameter. Despite its simplicity, it was recently shown that Reptile is at least as effective as MAML in terms of performance (Dou et al., 2019). We therefore employ Reptile for meta-learning in our experiments.

#### 3.1.4 Meta-Adapter

Adapters (Swietojanski and Renals, 2014; Vilar, 2018; Houlsby et al., 2019) are lightweight feed-forward modules. They are described by the following Equation: $W_{\text{up}}\, f\left(W_{\text{down}}\, \text{LN}\left(\mathbf{h}\right)\right) + \mathbf{h}$. An adapter consists of a layer normalization $\text{LN}(\cdot)$ (Ba et al., 2016) of the input $\mathbf{h}$, which is passed to a down-projection $W_{\text{down}} \in R^{z \times d}$, a non-linear activation $f(\cdot)$ (in our case, ReLU) and an up-projection $W_{\text{up}} \in R^{d \times z}$, where $d$ is the bottleneck dimension of the adapter module and the only tunable hyperparameter. The output is combined with a residual connection. Adapters are added between sub-layers of a pre-trained Transformer (Vaswani et al., 2017) model (see the right part of Figure 1), usually after the feed-forward layer.

Using adapters is appealing for multiple reasons: i) we only update the adapter parameters $\psi$ during the whole fine-tuning process, which makes training faster especially for large pre-trained models; ii) they obtain a performance comparable to that of traditional fine-tuning. However, as the adapter parameters $\psi$ are randomly initialized they may not perform well in the few-shot setting. Moreover, adding a new set of adapters for each domain or language pair (Bapna and Firat, 2019; Cooper Stickland et al., 2021a) quickly becomes inefficient when we need to adapt to many new domains and language pairs. To address this problem, we propose training a *Meta-Adapter*, which inserts adapter layers into the meta-learning training process (see the left part of Figure 1). Different from the traditional adapter training process, we only need to train a single meta-adapter to adapt to all new language pairs and domains.

Let $\theta$ denote the parameters of the pre-trained model and $\psi$ the parameters of the adapter. Given a target task $\mathcal{T}$ in the domain $\mathcal{D}_{\mathcal{T}}$ and a loss function $\mathcal{L}_{\mathcal{T}}(\cdot)$, we train a meta-adapter to minimize the following objective through gradient descent:

$$\min_{\psi} \mathcal{L}_{\mathcal{T}}\left(\theta, \psi; \mathcal{D}_{\mathcal{T}}\right)$$

where the parameters of pre-trained model $\theta$ are frozen and the adapter parameters $\psi$ are randomly initialized, leading to a size of $\psi \ll \theta$. This makes our approach more efficient than meta-learning an entire model (see more details in Section 6.1).

### 3.2 Meta-Adaptation

After the meta-training phase, the parameters of the adapter are fine-tuned to adapt to new tasks (as

**Algorithm 1** $m^4Adapter$ (Multilingual Multi-Domain Adaptation with Meta-Adapter)

**Input:** $\mathcal{D}_{train}$ set of DLPs for meta training; Pre-trained MNMT model $\theta$

**Initialize** $P_D(i)$ based on temperature sampling
**while** not converged **do**
  ▷ *Perform Reptile Updates*
  Sample $m$ DLPs $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_m$ from $\mathcal{M}$
  **for** i = 1,2,. . . ,m **do**
    $\psi_i^{(k)} \leftarrow U_i^k(\theta, \psi)$, denoting $k$ gradient updates from $\psi$ on batches of DLP $\mathcal{T}_i$ while keeping $\theta$ frozen
  **end for**
  $\psi \leftarrow \psi + \frac{\beta}{m} \sum_{i=1}^{m} (\psi_i^{(k)} - \psi)$
**end while**
**return** Meta-Adapter parameter $\psi$

both the domain and language pair of interest are not seen during the meta-training stage) using a small amount of data to simulate a low-resource scenario.

We find that this step is essential to our approach, as it permits adapting the parameters of the meta-learned model to the domain and language pair of interest. This step uses a very small amount of data (500 samples), which we believe could realistically be available for each DLP.

## 4 Experiments

**Datasets.** We split the datasets in two groups: *meta-training* or *training dataset* (used in step 1, § 3.1) and *meta-adapting* or *adapting dataset* (used in step 2, § 3.2). We first meta-learn the adapters on the training dataset (that contains DLPs different to the ones we will evaluate on), then fine-tune to new domains and language pairs on the adapting dataset (a small dataset of the DLPs we will evaluate on). We list the datasets used, each treated as a different domain: *EUbookshop, KDE, OpenSubtitles, QED, TED, Ubuntu, Bible, UN, Tanzil, Infopankki*. The datasets cover the following languages (ISO 639-1 language code[3]): *en, de, fr, mk, sr, et, hr, hu, fi, uk, is, lt, ar, es, ru, zh* and are publicly available on OPUS[4] (Tiedemann, 2012).

**Data Preprocessing.** For each *training dataset*, we strictly limit the corpus of each DLP to a maximum of 5000 sentences to simulate a low-resource setting. For each *adapting dataset*, we use 500 sentences in each DLP to fine-tune the MNMT model, simulating a few-shot setting. For the validation and test set, we select 500 sentences and avoid overlap with the adapting dataset by de-duplication. We filter out sentences longer than 175 tokens and preprocess all data using sentencepiece[5] (Kudo and Richardson, 2018). More details for the data used in this paper can be found in the Appendix A.1.

**Baselines.** We compare $m^4Adapter$ with the following baselines: i) **m2m**: Using the original m2m model (Fan et al., 2021) to generate the translations. ii) **m2m + FT**: Fine-tuning m2m on all DLPs. iii) **m2m + tag**: Fine-tuning m2m with domain tags (Kobus et al., 2017) on all DLPs. iv) **agnostic-adapter**: Mixing the data from all DLPs to train the adapters (Cooper Stickland et al., 2021b), to obtain language and domain-agnostic adapters. v) **stack-adapter**: Training two adapters for each language pair and domain, then stacking both adapters (Cooper Stickland et al., 2021a). Taking 'Ubuntu-en-sr' as an example, this approach first trains a *language pair adapter* for 'en-sr' using all data containing 'en-sr' in all domains (also including the 'Ubuntu' domain) and a *domain adapter* for 'Ubuntu' using all data covering all language pairs in the 'Ubuntu' domain. Then, the two adapters are stacked together. vi) **meta-learning**: Traditional meta-learning methods using the MAML algorithm (Sharaf et al., 2020) on all DLPs.

**Implementation.** We use m2m, released in the HuggingFace repository[6] (Wolf et al., 2020). For adapter training, we use the implementation of the AdapterHub repository[7] (Pfeiffer et al., 2020). We use DeepSpeed[8] (Rasley et al., 2020) to accelerate the pre-training of big models. Note that all baseline systems except *stack-adapter* train a single MNMT model or a single adapter on all DLPs in the *training datasets* and then fine-tune to a specific DLP on a single *adapting dataset*. For *stack-adapter*, the number of language pair adapters and domain adapters to be trained is proportional to the number of language pairs and the number of domains (see more details in Appendix A.2).

**Evaluation.** We measure case-sensitive detokenized BLEU with SacreBLEU[9] (Post, 2018). For

[3] en.wikipedia.org/wiki/List_of_ISO_639-1_codes
[4] https://opus.nlpl.eu
[5] github.com/google/sentencepiece
[6] github.com/huggingface/transformers
[7] github.com/adapter-hub/adapter-transformers
[8] github.com/microsoft/DeepSpeed
[9] github.com/mjpost/sacrebleu

| | BLEU | specific domain | | |
|---|---|---|---|---|
| | | TED | Ubuntu | KDE |
| **m2m** | 18.18 | 16.20 | 20.61 | 22.04 |
| **m2m + FT** | 20.84 | 17.53 | 28.81 | 29.19 |
| **m2m + tag** | 22.70 | 18.70 | **31.86** | 31.53 |
| **agnostic-adapter** | 23.70 | **19.82** | 31.07 | 32.74 |
| **stack-adapter** | 21.06 | 18.34 | 29.17 | 30.26 |
| **meta-learning** | 20.01 | 17.57 | 28.11 | 28.59 |
| $m^4$***Adapter*** | **23.89** | 19.77 | 31.46 | **32.91** |

Table 1: Performance on the *meta-training stage* (DLPs of *training dataset*): average BLEU on DLPs over all domains (left); average BLEU on DLPs per domain (right, under *specific domain*).

Chinese we use the SacreBLEU tokenizer (*tok zh*) and convert all traditional characters generated by the model to simplified characters using HanziConv.[10] We also evaluate our models using chrF (Popović, 2015) due to the recent criticism of BLEU score (Mathur et al., 2020); the results are listed in the Appendix A.3.1.

## 5 Results

Our goal is to evaluate the adaptability of $m^4Adapter$ on a variety of new domains and new language pairs simultaneously. In the *meta-training* stage, we perform meta-learning of the model on 180 DLPs, which contain 6 domains (*EUbookshop, KDE, OpenSubtitles, QED, TED, Ubuntu*) and 30 language pairs (*en, et, mk, sr, hr, hu*), comparing our approach to different baseline systems. In the *meta-adaptation* stage, we fine-tune both our model and the baselines to 3 domains (*UN, Tanzil, Infopankki*) and 30 language pairs (using *ar, en, es, fr, ru, zh*) of the same dataset simultaneously. Table 1 shows the results in the *meta-training* step and Table 2 presents the main results of our model in the *meta-adaptation* step compared to the baselines (results for all DLPs are in Appendix A.3.2).

Motivated by Lai et al. (2022), we compare our approach to multiple baselines in terms of domain robustness. As shown in Table 1, $m^4Adapter$ obtains a performance that is on par or better than *agnostic-adapter*, which is a robust model. Note that $m^4Adapter$ also outperforms *m2m+tag*, which was shown to be the most robust model in Cooper Stickland et al. (2021a). After showing empirically that we obtain a robust model, we verify its adaptability (see Table 2 and § 6.2.1) and language transfer ability (§ 6.2.2) through a series of experiments.

As shown in Table 2, $m^4Adapter$ performs well when adapting to the *meta-adaptation* domains and language pairs at the same time. We observe that no baseline system outperforms the original m2m model. This implies that these models are unable to transfer language or domain knowledge from the MNMT model. One possible explanation is that these models already exhibit over-fitting and catastrophic forgetting when trained on *meta-training* domains and language pairs in such limited resource scenarios.

Because of the unpredictability of the baseline systems' performance, it is difficult to draw reliable conclusions. For example, in the UN domain, *meta-learning* is on par with the original m2m model. However, performance on Tanzil and Infopankki is much worse than the one of the original m2m model. The *agnostic-adapter* also performs comparably with the original m2m model in the same domains, which shows that it is a robust model. Still, it obtains much worse performance on UN. In contrast, $m^4Adapter$ has a more stable performance when adapting to new domains and language pairs.

In addition, $m^4Adapter$ has the ability to improve the performance of some DLPs on which baseline models obtain extremely low BLEU scores, especially in some distant domains. For example, in Tanzil-ar-ru, the traditional meta-learning method only gets 1.70 BLEU score, while $m^4Adapter$ gets 4.33.

## 6 Analysis

In this section, we conduct additional experiments to better understand the strengths of $m^4Adapter$. We first investigate the benefits in terms of speed in $m^4Adapter$ training and adapting (Section 6.1), then investigate the cross-lingual domain transfer and cross-domain language transfer through an ablation study (Section 6.2).

### 6.1 Efficiency of $m^4$*Adapter*

We compare the efficiency of baselines to traditional fine-tuning and list their number of trainable parameters and training/adapting time in Table 3.

$m^4Adapter$ only updates the adapter parameters while freezing the MNMT model's parameters (just like *agnostic-adapter*). Therefore, it has fewer trainable parameters compared to fine-tuning (0.75% of the parameters of the entire model).

[10] github.com/berniey/hanziconv

| | DLP (meta-adaptation domain) | | | specific DLP | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | UN | Tanzil | Infopankki | UN-ar-en | Tanzil-ar-en | Infopankki-ar-en | UN-ar-ru | Tanzil-ar-ru | Infopankki-ar-ru |
| **m2m** | 32.28 | 8.72 | 17.40 | 38.94 | 6.44 | 22.57 | 22.96 | 3.64 | 15.05 |
| **m2m + FT** | 29.93 | 8.26 | 15.88 | 35.11 | 6.85 | 21.33 | 19.10 | 3.05 | 14.19 |
| **m2m + tag** | 29.88 | 8.06 | 15.93 | 34.39 | 6.63 | 20.12 | 19.37 | 2.65 | 13.68 |
| **agnostic-adapter** | 30.56 | 8.42 | 17.36 | 36.13 | 6.12 | 23.08 | 20.64 | 3.63 | 14.96 |
| **stack-adapter** | 29.64 | 8.14 | 17.19 | 35.31 | 5.83 | 22.14 | 19.17 | 2.34 | 13.85 |
| **meta-learning** | 32.21 | 7.02 | 16.73 | 37.13 | 5.50 | 18.91 | 22.68 | 1.70 | 15.23 |
| $m^4Adapter$ | **33.53** | **9.87** | **18.43** | **39.05** | **8.56** | **23.21** | **25.22** | **4.33** | **17.48** |
| **Δ** | +1.25 | +1.15 | +1.03 | +0.11 | +2.12 | +0.64 | +2.26 | +0.69 | +2.43 |

Table 2: Main results on the *meta-adaptation stage*: average BLEU scores on all DLPs with different adaptation domain (left) and BLEU scores on some examples of specific DLP (right). Δ denotes improvement over m2m.

| Method | #Param. | $\text{Time}_T$ | $\text{Time}_A$ |
|---|---|---|---|
| **m2m** | 418M (100%) | - | - |
| **m2m + FT** | 418M (100%) | 100% | 100% |
| **m2m + tag** | 418M (100%) | 100% | 100% |
| **agnostic-adapter** | 3.17M (0.75%) | 42% | 150% |
| **stack-adapter** | $k\cdot$ 3.17M ($k\cdot$ 0.75%) | $k\cdot$ 42% | 200% |
| **meta-learning** | 418M (100%) | 75% | 500% |
| $m^4Adapter$ | 3.17M (0.75%) | 34% | 300% |

Table 3: Number of trainable parameters and Training/Adapting time relative to fine-tuning. $k$ denotes the number of DLPs during the training process.

Furthermore, the parameters of $m^4Adapter$ are significantly fewer than those of *stack-adapter*, which are $k$ times larger than those of standard adapter-based approaches. This happens because domain adapters and language pair adapters must be trained in each DLP when training the *stack-adapter* model. Adapter-based approaches train 34%-42% faster than fine-tuning due to parameter efficiency. The adaptation time of $m^4Adapter$, on the other hand, is often longer since it requires updating the high-level gradient. Our approach requires more time than traditional adapter methods but is faster compared with updating the entire model using traditonal meta-learning. For example, the adaptation time for *m2m+FT* is $40s$, while for $m^4Adapter$ it is $120s$, which is still a lot faster than standard *meta-learning* ($200s$).

### 6.2 Ablation Study

We conduct a number of experiments with extensive analysis to validate the domain transfer ability of the $m^4Adapter$ across different language pairs (§ 6.2.1), as well as the language transfer ability across multiple domains (§ 6.2.2).

#### 6.2.1 Domain Transfer via Languages

To investigate the capacity of our models to transfer domain knowledge across different languages, we define domain transfer via languages, i.e., the ability to transfer domains while keeping the languages unchanged. We first fine-tune the MNMT model in some of the *meta-training* domains under the specified language pair, and then we adapt these trained models to new *meta-adaptation* domains of the same language pair. To be more specific, we first choose 6 languages (*en, et, mk, sr, hr, hu*), forming 30 language pairs. Then, we choose six of these seven domains (*EUbookshop, KDE, OpenSubtitles, QED, TED, Ubuntu, Bible*) across all selected 30 language pairs as the *meta-training* dataset (180 DLPs) to fine-tune the MNMT model, and another one domain as the adapting domain across all selected language pairs (30 DLPs) to evaluate the adaptability of the fine-tuned MNMT model to the new domain. Table 4 provides the results for domain transfer across languages.

From Table 4, we observe that almost all baseline systems and $m^4Adapter$ outperform the original m2m model (except for the *EUbookshop* domain), indicating that the model encodes language knowledge and can transfer this knowledge to new *meta-adaptation* domains. Our approach is comparable to the performance of *agnostic-adapter*, which performs the best among all baseline systems.

We also discover that domain transfer through languages is desirable in some distant domains. For example, the original m2m model only got BLEU scores of 2.01 and 19.01 in the *Bible* and *OpenSubtitles* domain (*hr-sr* language pair). However, domain transfer through $m^4Adapter$ resulted in a considerable performance boost and achieved a BLEU score of 13.69 and 54.30.

We notice that none of the baselines outperforms the original m2m model in the *EUbookshop* domain, which means that the language knowledge learned from the baseline model does not transfer to this particular domain. Our approach, on the other hand, has a strong domain transfer ability. We investigated the reason, which was caused

| | meta-adaptation domain | | | | | | | specific DLP (hr-sr) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | EUbookshop | KDE | OpenSubtitles | QED | TED | Ubuntu | Bible | EUbookshop | KDE | OpenSubtitles | QED | TED | Ubuntu | Bible |
| **m2m** | 17.77 | 22.05 | 14.13 | 18.34 | 16.20 | 20.62 | 9.80 | 11.43 | 25.37 | 19.01 | 12.25 | 8.14 | 22.33 | 2.01 |
| **m2m + FT** | 12.73 | 24.56 | 16.22 | 20.46 | 18.74 | 31.32 | 11.30 | 9.79 | 21.05 | 53.34 | 23.87 | 20.81 | **34.08** | 12.57 |
| **m2m + tag** | 13.03 | 25.34 | 16.12 | 17.75 | 17.04 | 26.29 | 11.49 | 10.13 | 29.64 | 49.54 | 19.78 | 20.43 | 34.15 | 13.25 |
| **agnostic-adapter** | 16.24 | **25.85** | 17.90 | 21.71 | **20.08** | **31.53** | 11.75 | 9.05 | **30.64** | 54.04 | 22.79 | 21.19 | 28.83 | 10.59 |
| **stack-adapter** | 13.25 | 24.19 | 17.21 | 19.56 | 18.37 | 28.27 | 10.38 | 10.55 | 24.50 | 42.94 | 22.02 | 20.95 | 25.41 | 10.14 |
| **meta-learning** | 13.61 | 24.91 | 16.22 | 17.70 | 16.40 | 24.93 | 11.84 | 7.90 | 27.85 | 52.50 | 20.41 | 19.00 | 31.24 | 10.42 |
| $m^4Adapter$ | **18.99** | 25.22 | **17.94** | **21.71** | 19.86 | 31.37 | **12.12** | **12.05** | 30.49 | **54.30** | **23.92** | **21.32** | 33.71 | **13.69** |
| **Δ** | +2.75 | -0.63 | +0.04 | +0.00 | -0.22 | -0.16 | +0.37 | +3.00 | -0.15 | +0.26 | +1.13 | +0.13 | +4.88 | +3.1 |

Table 4: Domain transfer via languages: average BLEU scores on all DLPs in each *meta-adaptation* domain (left) and BLEU scores on a random selection of one specific DLP in *hr-sr* (right). Δ denotes improvement over *agnostic-adapter*.

| | meta-adaptation language pair | | | | specific DLP (de-en) | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | de-en | en-fr | fi-uk | is-lt | EUbookshop | KDE | OpenSubtitles | QED | TED | Ubuntu |
| **m2m** | 24.52 | 29.20 | 12.34 | 12.55 | 19.59 | 26.48 | 15.89 | 26.34 | 28.14 | 30.65 |
| **m2m + FT** | 23.29 | 24.44 | 11.29 | 9.59 | 16.04 | 23.17 | 13.34 | 21.39 | 26.20 | 39.59 |
| **m2m + tag** | 22.52 | 24.97 | 11.71 | 11.22 | 15.86 | 23.67 | 11.72 | 20.64 | 25.97 | 37.25 |
| **agnostic-adapter** | 28.33 | **30.93** | **15.42** | **14.38** | 20.16 | **28.72** | 17.97 | **27.66** | **33.63** | 41.89 |
| **stack-adapter** | 23.37 | 24.96 | 11.51 | 11.09 | 16.14 | 22.51 | 13.84 | 22.29 | 27.67 | 36.73 |
| **meta-learning** | 25.08 | 28.26 | 13.40 | 12.83 | 17.88 | 21.20 | 16.32 | 24.96 | 30.32 | 39.81 |
| $m^4Adapter$ | **28.37** | 30.80 | 15.24 | 14.05 | **20.20** | 28.19 | **18.06** | 27.18 | 33.32 | **43.24** |
| Δ | +0.04 | -0.13 | -0.18 | -0.33 | +0.04 | -0.53 | +0.09 | -0.48 | -0.31 | +1.35 |

Table 5: Language transfer via domains: average BLEU scores on all DLPs in each meta-adaptation language pair (left) and BLEU scores on one specific DLP in *de-en* (right). Δ denotes the improvement over the *agnostic-adapter*.

by a significant overfitting issue while adapting to the *EUbookshop* domain. The previous fine-tuning strategy converged too early, resulting in significant overfitting of the model to the *meta-training* dataset, which performed exceedingly badly in adapting to the new domain (see the loss decline curve in Appendix A.5 for more details). This phenomenon is also consistent with our previous findings (§ 5) that our approach is more stable than the baseline systems in adapting to new domains.

### 6.2.2 Language Transfer via Domains

To study the ability of our model to transfer language knowledge across different domains, we define language transfer via domains, i.e., the ability to transfer languages while keeping the domains unchanged. To this end, we first fine-tune the MNMT model in some *meta-training* DLPs, and then we adapt these trained models to *meta-adaptation* language pairs of the same domains. To achieve this, we first select 180 DLPs as the *meta-training* dataset to train the model, which contains 6 domains (*EUbookshop, KDE, OpenSubtitles, QED, TED, Ubuntu*) and 30 language pairs (*en, et, mk, sr, hr, hu*); then adapt these trained model to 4 of the *meta-adaptation* language pairs (*de-en, en-fr, fi-uk, is-lt*). The findings of language transfer across domains are shown in Table 5.

According to Table 5, the performance of traditional fine-tuning approaches (*m2m+FT*, *m2m+tag*) are poorer than the original m2m model, which means that these methods do not transfer the learned domain knowledge to the new *meta-adaptation* language pair. This meets our expectation since m2m is trained on a big dataset and learns a great quantity of linguistic information, which limits its capacity to transfer language information in small datasets. This explanation can be demonstrated by the results of the *meta-learning* approach. As shown in Table 5, *meta-learning* yields slightly higher BLEU scores compared to the original m2m model, which arguably supports the conclusion that the original m2m model already has strong linguistic information. These small improvements from *meta-learning* can be attributed to leveraging the limited data available.

In contrast, adapter-based methods (*agnostic-adapter* and $m^4Adapter$) permit cross-lingual transfer across domains. $m^4Adapter$ shows a performance that is on par or better than the *agnostic-adapter*, the most competitive model in all baseline systems. The results of the *stack-adapter* show that it cannot perform language transfer across domains through naively stacking domain adapters and language adapters. This is consistent with the conclusions of Cooper Stickland et al. (2021a).

Similarly, $m^4Adapter$ has demonstrated significant language transfer ability in distant domains. In

*ubuntu-de-en*, for example, $m^4Adapter$ achieves a BLEU score of 43.24, which is significantly higher than the original m2m model's BLEU of 30.65.

## 7 Conclusion

We present $m^4Adapter$, a novel multilingual multi-domain NMT adaptation framework which combines meta-learning and parameter-efficient fine-tuning with adapters. $m^4Adapter$ is effective on adapting to new languages and domains simultaneously in low-resource settings. We find that $m^4Adapter$ also transfers language knowledge across domains and transfers domain information across languages. In addition, $m^4Adapter$ is efficient in training and adaptation, which is practical for online adaptation (Etchegoyhen et al., 2021) to complex scenarios (new languages and new domains) in the real world.

## 8 Limitations

This work has two main limitations. i) We have only evaluated the proposed method on limited and balanced bilingual training data to simulate the low-resource scenario. However, some domains in our setting are in fact highly imbalanced. ii) We only evaluated $m^4Adapter$ on machine translation, perhaps it would be plausible to expand our method to other NLP tasks, such as text generation or language modeling. Since our framework leverages a multilingual pretrained model and only trains adapters, we believe it could easily be applied to other tasks besides MT.

## Acknowledgement

This work was supported by funding to Wen Lai's PhD research from LMU-CSC (China Scholarship Council) Scholarship Program. This work has received funding from the European Research Council under the European Union's Horizon 2020 research and innovation program (grant agreement #640550). This work was also supported by the DFG (grant FR 2829/4-1).

## References

Roee Aharoni and Yoav Goldberg. 2020. Unsupervised domain clusters in pretrained language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7747–7763, Online. Association for Computational Linguistics.

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884, Minneapolis, Minnesota. Association for Computational Linguistics.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1538–1548, Hong Kong, China. Association for Computational Linguistics.

Yanda Chen, Ruiqi Zhong, Sheng Zha, George Karypis, and He He. 2022. Meta-learning via language model in-context tuning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 719–730, Dublin, Ireland. Association for Computational Linguistics.

Alexandra Chronopoulou, Dario Stojanovski, and Alexander Fraser. 2022. Language-family adapters for multilingual neural machine translation. *arXiv preprint arXiv:2209.15236*.

Chenhui Chu and Raj Dabre. 2019. Multilingual multi-domain adaptation approaches for neural machine translation. *arXiv preprint arXiv:1906.07978*.

Chenhui Chu and Rui Wang. 2018. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

Asa Cooper Stickland, Alexandre Berard, and Vassilina Nikoulina. 2021a. 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. Association for Computational Linguistics.

Asa Cooper Stickland, Xian Li, and Marjan Ghazvininejad. 2021b. Recipes for adapting pre-trained monolingual and multilingual models to machine translation. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3440–3453, Online. Association for Computational Linguistics.

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

Tobias Domhan and Felix Hieber. 2017. Using target-side monolingual data for neural machine translation through multi-task learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1500–1505, Copenhagen, Denmark. Association for Computational Linguistics.

Zi-Yi Dou, Keyi Yu, and Antonios Anastasopoulos. 2019. Investigating meta-learning algorithms for low-resource natural language understanding tasks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1192–1197, Hong Kong, China. Association for Computational Linguistics.

Thierry Etchegoyhen, David Ponce, Harritxu Gete, and Victor Ruiz. 2021. Online learning over time in adaptive neural machine translation. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP 2021)*, pages 411–420, Held Online. INCOMA Ltd.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *Journal of Machine Learning Research*, 22(107):1–48.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. 2017. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR.

Markus Freitag and Yaser Al-Onaizan. 2016. Fast domain adaptation for neural machine translation. *arXiv preprint arXiv:1612.06897*.

Jiatao Gu, Yong Wang, Yun Chen, Victor O. K. Li, and Kyunghyun Cho. 2018. Meta-learning for low-resource neural machine translation. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3622–3631, Brussels, Belgium. Association for Computational Linguistics.

Jiatao Gu, Yong Wang, Kyunghyun Cho, and Victor O.K. Li. 2019. Improved zero-shot neural machine translation via ignoring spurious correlations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1258–1268, Florence, Italy. Association for Computational Linguistics.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

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. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Huda Khayrallah, Gaurav Kumar, Kevin Duh, Matt Post, and Philipp Koehn. 2017. Neural lattice search for domain adaptation in machine translation. In *Proceedings of the Eighth International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 20–25, Taipei, Taiwan. Asian Federation of Natural Language Processing.

Catherine Kobus, Josep Crego, and Jean Senellart. 2017. Domain control for neural machine translation. In *Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017*, pages 372–378, Varna, Bulgaria. INCOMA Ltd.

Taku Kudo and John Richardson. 2018. SentencePiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71, Brussels, Belgium. Association for Computational Linguistics.

Wen Lai, Jindřich Libovický, and Alexander Fraser. 2022. Improving both domain robustness and domain adaptability in machine translation. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 5191–5204, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

Hung-yi Lee, Shang-Wen Li, and Ngoc Thang Vu. 2022. Meta learning for natural language processing: A survey. *arXiv preprint arXiv:2205.01500*.

Ilya Loshchilov and Frank Hutter. 2018. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Nitika Mathur, Timothy Baldwin, and Trevor Cohn. 2020. Tangled up in BLEU: Reevaluating the evaluation of automatic machine translation evaluation metrics. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4984–4997, Online. Association for Computational Linguistics.

Michael McCloskey and Neal J Cohen. 1989. Catastrophic interference in connectionist networks: The sequential learning problem. In *Psychology of learning and motivation*, volume 24, pages 109–165. Elsevier.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages

875–880, Brussels, Belgium. Association for Computational Linguistics.

Alex Nichol, Joshua Achiam, and John Schulman. 2018. On first-order meta-learning algorithms. *arXiv preprint arXiv:1803.02999*.

Cheonbok Park, Hantae Kim, Ioan Calapodescu, Hyun Chang Cho, and Vassilina Nikoulina. 2022. DaLC: Domain adaptation learning curve prediction for neural machine translation. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1789–1807, Dublin, Ireland. Association for Computational Linguistics.

Jaehong Park, Jongyoon Song, and Sungroh Yoon. 2017. Building a neural machine translation system using only synthetic parallel data. *arXiv preprint arXiv:1704.00253*.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Jerin Philip, Alexandre Berard, Matthias Gallé, and Laurent Besacier. 2020. Monolingual adapters for zero-shot neural machine translation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4465–4470, Online. Association for Computational Linguistics.

Maja Popović. 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. Association for Computational Linguistics.

Matt Post. 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. Association for Computational Linguistics.

Jeff Rasley, Samyam Rajbhandari, Olatunji Ruwase, and Yuxiong He. 2020. Deepspeed: System optimizations enable training deep learning models with over 100 billion parameters. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3505–3506.

Andreas Rücklé, Gregor Geigle, Max Glockner, Tilman Beck, Jonas Pfeiffer, Nils Reimers, and Iryna Gurevych. 2021. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Amr Sharaf, Hany Hassan, and Hal Daumé III. 2020. Meta-learning for few-shot NMT adaptation. In *Proceedings of the Fourth Workshop on Neural Generation and Translation*, pages 43–53, Online. Association for Computational Linguistics.

Pawel Swietojanski and Steve Renals. 2014. Learning hidden unit contributions for unsupervised speaker adaptation of neural network acoustic models. In *2014 IEEE Spoken Language Technology Workshop (SLT)*, pages 171–176. IEEE.

Ishan Tarunesh, Sushil Khyalia, Vishwajeet Kumar, Ganesh Ramakrishnan, and Preethi Jyothi. 2021. Meta-learning for effective multi-task and multilingual modelling. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3600–3612, Online. Association for Computational Linguistics.

Jörg Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pages 2214–2218, Istanbul, Turkey. European Language Resources Association (ELRA).

Marlies van der Wees, Arianna Bisazza, and Christof Monz. 2017. Dynamic data selection for neural machine translation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1410, Copenhagen, Denmark. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

David Vilar. 2018. Learning hidden unit contribution for adapting neural machine translation models. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 500–505, New Orleans, Louisiana. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing.

In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Runzhe Zhan, Xuebo Liu, Derek F Wong, and Lidia S Chao. 2021. Meta-curriculum learning for domain adaptation in neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 35, pages 14310–14318.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639, Online. Association for Computational Linguistics.

# A Appendix

## A.1 Datasets

All datasets used in our experiments are publicly available on OPUS. Despite the fact that OPUS contains corpora from various domains and languages, some recent works (Aharoni and Goldberg, 2020; Lai et al., 2022) have raised concerns about using OPUS corpora as they can be noisy. We therefore performed the following cleaning and filtering preprocess on the original OPUS corpus: i) remove sentences that contain more than 50% punctuation; ii) to ensure that the training set did not contain any corpora from the validation or test sets, all corpora were de-duplicated; iii) sentences longer than 175 tokens were removed; iv) we used a language detection tool[11] (*langid*) to filter out sentences with mixed languages.

As described in Section 4, during the training phase, although most of the DLPs were limited to a maximum of 5000 sentences, there was still a fraction of DLPs with a corpus of less than 5000 samples which we list it in Table 6.

## A.2 Model Configuration

Our $m^4Adapter$ model is trained in the following way: it first samples $m$ tasks based on temperature $\tau$, then makes $k$ gradient updates for each task $\mathcal{T}_i$. Finally, it updates the parameters of $\psi$. In our set of experiments, we use the AdamW (Loshchilov and Hutter, 2018) optimizer, which is shared across all DLPs. We fix the initial learning rate to $5e-5$ with a dropout probability $0.1$. In our experiments, we consider values of $m \in \{4, 8, 16\}$, $k \in \{1, 2, 3, 4, 5\}$, $\alpha \in \{0.1, 0.5, 1.0\}$ and $\tau \in \{1, 2, 5, \infty\}$ and choose the best setting ($m$ = 8, $k$ = 3, $\beta$ = 1.0, $\tau$ = 1) based on the average BLEU scores over all DLPs. Each $m^4Adapter$ model is trained for 3 epochs and adapts to each DLP for 1 epoch to simulate a fast adaptation scenario.

[11] https://fasttext.cc/docs/en/language-identification.html

| DLP | #Num. | DLP | #Num. |
|---|---|---|---|
| EUbookshop-hu-sr | 59 | Ubuntu-hu-sr | 140 |
| EUbookshop-hu-mk | 976 | Ubuntu-hr-sr | 438 |
| EUbookshop-en-sr | 1104 | Ubuntu-hr-hu | 479 |
| EUbookshop-et-sr | 1141 | Ubuntu-et-sr | 912 |
| EUbookshop-hr-sr | 1280 | Ubuntu-en-sr | 1519 |
| EUbookshop-mk-sr | 1320 | Ubuntu-et-mk | 1545 |
| EUbookshop-hr-hu | 1328 | Ubuntu-hr-mk | 1880 |
| EUbookshop-en-mk | 1836 | Ubuntu-mk-sr | 2091 |
| EUbookshop-et-mk | 2000 | Ubuntu-hu-mk | 2118 |
| EUbookshop-hr-mk | 2003 | Ubuntu-et-hu | 2147 |
| EUbookshop-et-hr | 2861 | Ubuntu-et-hr | 2542 |
| EUbookshop-en-hr | 4668 | Ubuntu-en-mk | 2644 |
| - | - | Ubuntu-en-et | 4998 |
| - | - | Ubuntu-en-hu | 4999 |

Table 6: Data statistics (number of sentences) of DLPs that contain less than 5000 sentences.

| | UN | Tanzil | Infopankki |
|---|---|---|---|
| $\tau = 1$ | **33.53** | **9.87** | 18.43 |
| $\tau = 2$ | 33.52 | 9.81 | **18.46** |
| $\tau = 5$ | 33.33 | 9.77 | 18.19 |
| $\tau = \infty$ | 33.44 | 9.80 | 18.44 |

Table 7: Different temperature settings.

## A.3 Additional Results

### A.3.1 chrF Evaluation

In addition to BLEU, we also use chrF (Popović, 2015) as an evaluation metric. Tables 9, 10 and 11 show the results. $m^4Adapter$ is more effective than all baseline systems in terms of chrF, which is consistent with the BLEU scores (that were presented in Tables 2, 4 and 5).

### A.3.2 Results on all DLPs

Figure 2 reports the results for all DLPs, which is consistent with the results in Tables 2 and 9.

## A.4 Analysis

To better understand our proposed method, we investigate the effect of different parameter settings on the results (as described in Section 3.1.2). We also analyse the poor results on *EUbookshop* domain as described in Section 6.2.1.

| shots | avg BLEU |
|---|---|
| 2-shots | 23.80 |
| 4-shots | 23.88 |
| 8-shots | **23.89** |
| 16-shots | 23.85 |
| 32-shots | 23.88 |

Table 8: Different amounts of shots.

### A.4.1 Effect of temperature sampling

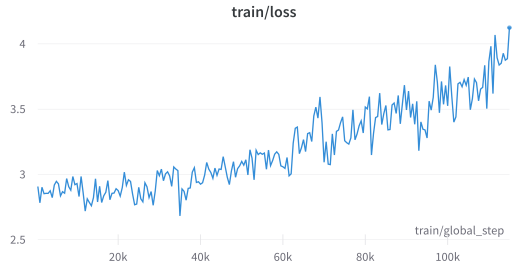
Although the *meta-training* data of all DLPs is limited to a maximum of 5000 sentences, there are still some DLPs with less than 5000 sentences, so we use temperature sampling for each setting for $\tau = 1, 2, 5$ and $\infty$. We first sample the task-based temperature and show the results in Table 7. We notice that the performance of the various temperature settings is very similar. These results meet our expectation as the data we used was limited to a maximum of 5000 sentences in most DLPs, with the exception of some DLPs in the *EUbookshop* and *Ubuntu* domains (see Appendix A.1), which means data is sampled uniformly in different temperature settings.

### A.4.2 Effect of different shots

We also test the performance on different numbers of shots ($n = 2, 4, 6, 8, 16$) and show the results in Table 8. Interestingly, we observe that $m^4Adapter$ is not sensitive to different numbers of shots, unlike other NLP (Chen et al., 2022) and Computer Vision tasks (Finn et al., 2017) which use the meta-learning approach. We argue that this is because the meta-adapter is randomly initialized at each batch, resulting in a gap between training and inference. Narrowing this gap is an important future research direction.

## A.5 Analysis on EUbookshop domain

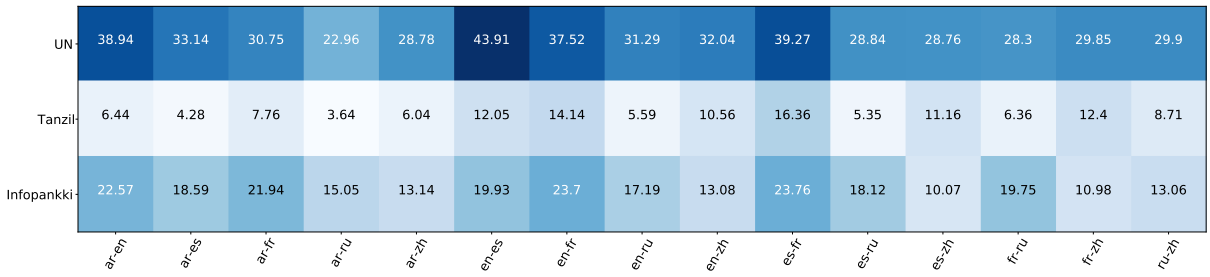
As described in Section 6.2.1, we observed that all baseline systems overfit when trained on data from the *EUbookshop* domain. For example, in the case of the *m2m + FT* baseline, the training loss converges and stops improving at a very early stage. After that, the model overfits the validation set (Figure 2). On the contrary, the training loss of the $m^4Adapter$ does not show signs of overfitting. This is probably due to the much smaller number of parameters that our proposed model trains.



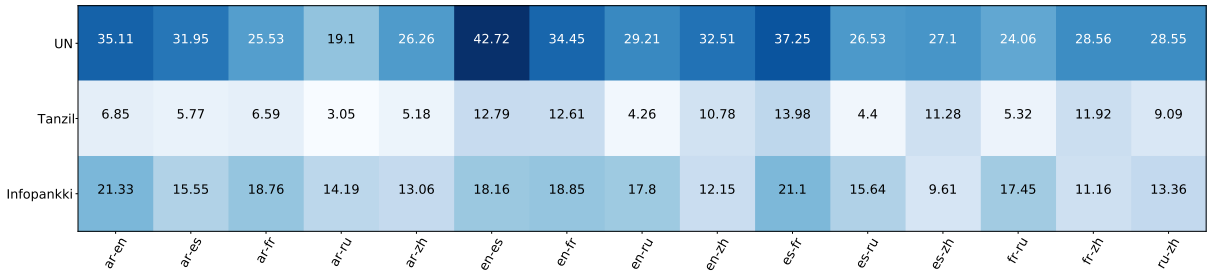

(a) m2m + FT


(b) $m^4Adapter$

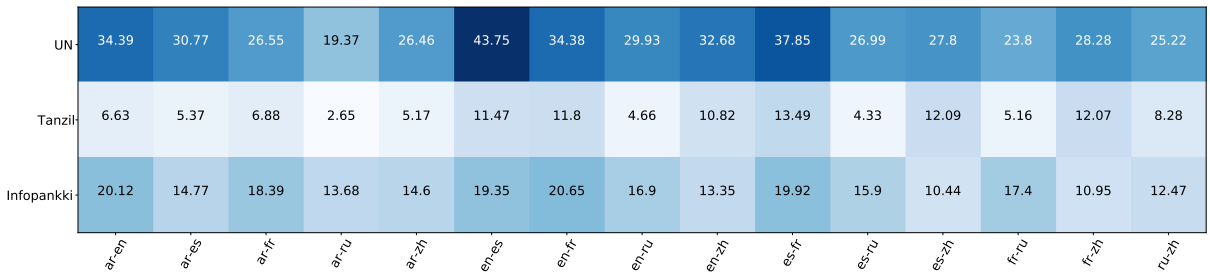
Figure 2: Training loss of *m2m + FT* and $m^4Adapter$ in EUbookshop domain.



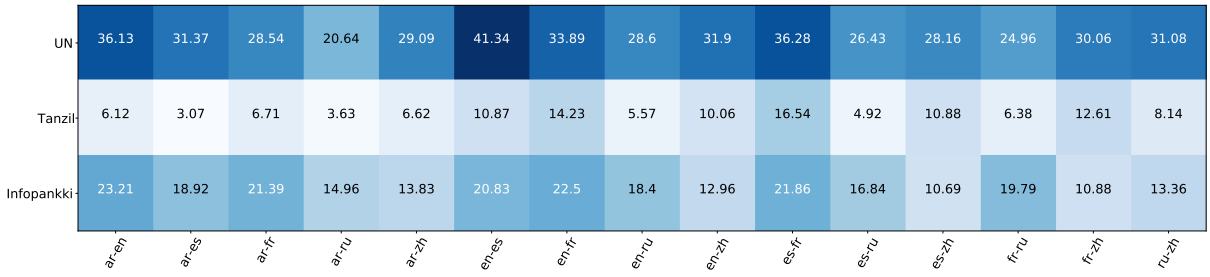

(a) m2m



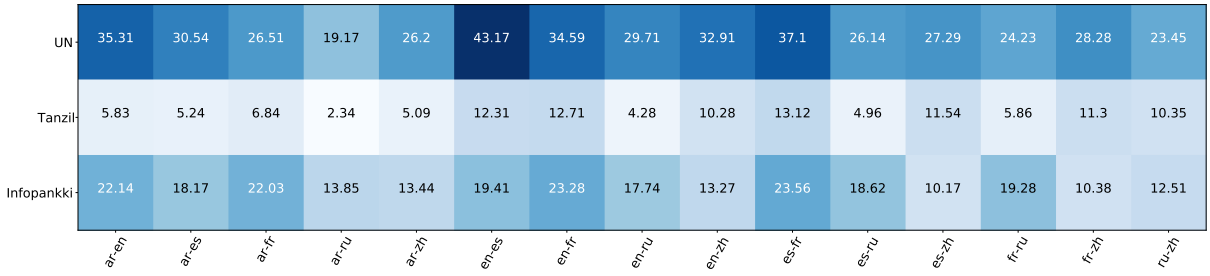

(b) m2m + FT



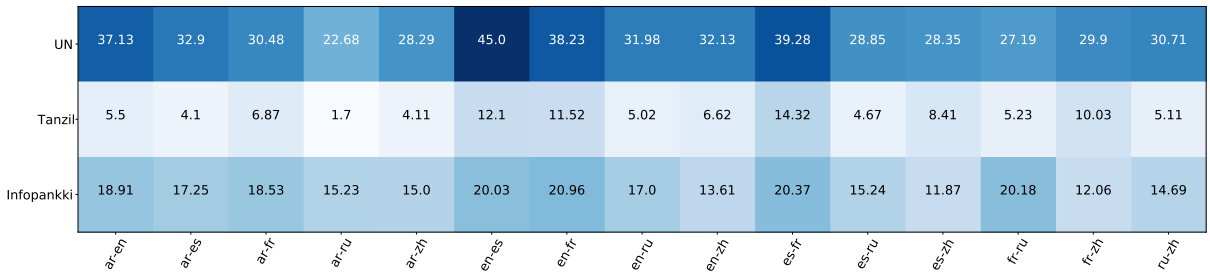

(c) m2m + tag



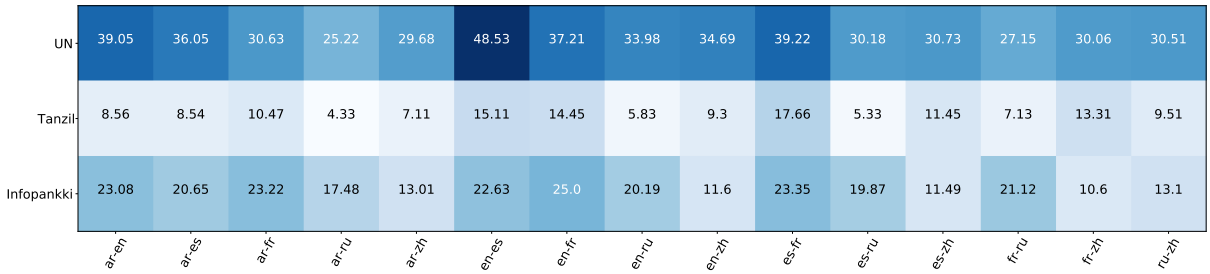

(d) agnostic-adapter




(e) stack-adapter




(f) Meta-Learning




(g) $m^4Adapter$

Figure 2: Main result: BLEU scores in all DLPs

| | **DLP (*meta-adaptation* domain)** | | | **specific DLP** | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | UN | Tanzil | Infopankki | UN-ar-en | Tanzil-ar-en | Infopankki-ar-en | UN-ar-ru | Tanzil-ar-ru | Infopankki-ar-ru |
| **m2m** | 0.480 | 0.227 | 0.377 | 0.602 | 0.280 | 0.479 | 0.484 | 0.191 | 0.450 |
| **m2m + FT** | 0.473 | 0.203 | 0.348 | 0.592 | 0.249 | 0.466 | 0.473 | 0.154 | 0.401 |
| **m2m + tag** | 0.473 | 0.203 | 0.344 | 0.590 | 0.255 | 0.448 | 0.474 | 0.152 | 0.400 |
| **agnostic-adapter** | 0.475 | 0.228 | 0.370 | 0.615 | 0.242 | 0.488 | 0.486 | 0.217 | 0.431 |
| **stack-adapter** | 0.472 | 0.207 | 0.368 | 0.593 | 0.243 | 0.476 | 0.473 | 0.151 | 0.405 |
| **meta-learning** | 0.487 | 0.203 | 0.349 | 0.612 | 0.278 | 0.454 | 0.483 | 0.165 | 0.428 |
| $m^4Adapter$ | **0.525** | **0.230** | **0.384** | **0.649** | **0.299** | **0.491** | **0.536** | **0.228** | **0.521** |

Table 9: Main results on the *meta-adaptation stage*: average chrF scores on all DLPs with different adaptation domain (left) and chrF scores on some examples of specific DLP (right).

| | ***meta-adaptation* domain** | | | | | | | specific DLP (hr-sr) | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | EUbookshop | KDE | OpenSubtitles | QED | TED | Ubuntu | Bible | EUbookshop | KDE | OpenSubtitles | QED | TED | Ubuntu | Bible |
| **m2m** | 0.446 | 0.417 | 0.339 | 0.420 | 0.408 | 0.476 | 0.129 | 0.361 | 0.432 | 0.284 | 0.204 | 0.146 | 0.495 | 0.025 |
| **m2m + FT** | 0.378 | 0.444 | 0.358 | 0.444 | 0.445 | 0.567 | 0.144 | 0.353 | 0.473 | 0.677 | 0.423 | 0.429 | 0.563 | 0.138 |
| **m2m + tag** | 0.388 | 0.445 | 0.359 | 0.414 | 0.428 | 0.520 | 0.135 | 0.359 | 0.502 | 0.671 | 0.360 | 0.428 | 0.581 | 0.118 |
| **agnostic-adapter** | 0.419 | **0.460** | 0.385 | 0.456 | **0.461** | **0.568** | 0.144 | 0.279 | **0.507** | 0.613 | 0.388 | 0.415 | 0.554 | 0.127 |
| **stack-adapter** | 0.382 | 0.436 | 0.390 | 0.438 | 0.441 | 0.546 | 0.134 | 0.358 | 0.427 | 0.562 | 0.381 | 0.427 | 0.526 | 0.124 |
| **meta-learning** | 0.387 | 0.440 | 0.360 | 0.412 | 0.422 | 0.509 | 0.142 | 0.237 | 0.502 | 0.676 | 0.353 | 0.404 | 0.546 | 0.139 |
| $m^4Adapter$ | **0.497** | 0.452 | **0.386** | **0.456** | 0.457 | 0.565 | **0.148** | **0.369** | 0.504 | **0.679** | **0.427** | **0.431** | **0.578** | **0.143** |

Table 10: Domain transfer via languages: average chrF scores on all DLPs in each *meta-adaptation* domain (left) and chrF scores on random select one specific DLP in *hr-sr* (right).

| | ***meta-adaptation* language pair** | | | | **specific DLP(de-en)** | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | de-en | en-fr | fi-uk | is-lt | EUbookshop | KDE | OpenSubtitles | QED | TED | Ubuntu |
| **m2m** | 0.116 | 0.130 | 0.327 | 0.320 | 0.171 | 0.104 | 0.093 | 0.107 | 0.132 | 0.095 |
| **m2m + FT** | 0.112 | 0.094 | 0.253 | 0.243 | 0.164 | 0.091 | 0.089 | 0.105 | 0.134 | 0.090 |
| **m2m + tag** | 0.094 | 0.096 | 0.258 | 0.261 | 0.140 | 0.067 | 0.082 | 0.088 | 0.116 | 0.077 |
| **agnostic-adapter** | 0.116 | 0.127 | **0.343** | 0.331 | 0.168 | 0.102 | 0.093 | 0.108 | 0.134 | 0.092 |
| **stack-adapter** | 0.113 | 0.096 | 0.256 | 0.258 | 0.164 | 0.087 | 0.088 | 0.105 | 0.130 | 0.075 |
| **meta-learning** | 0.115 | 0.125 | 0.317 | 0.309 | 0.170 | 0.101 | 0.092 | 0.108 | 0.133 | 0.092 |
| $m^4Adapter$ | **0.117** | **0.131** | 0.342 | **0.333** | **0.174** | **0.107** | **0.095** | **0.108** | **0.134** | **0.097** |

Table 11: Language transfer via domains: average chrF scores on all DLPs in each *meta-adaptation* language pair (left) and chrF scores on one specific DLP in *de-en* (right).