Monolingual Adapters for Zero-Shot Neural Machine Translation

Jerin Philip[†*] and **Alexandre Bérard**[†] and **Matthias Gallé**[†] and **Laurent Besacier**[†‡]
[†]NAVER LABS Europe [‡]Université Grenoble Alpes
jerin.philip@research.iiit.ac.in
alexandre.berard@naverlabs.com
laurent.besacier@univ-grenoble-alpes.fr
matthias.galle@naverlabs.com

Abstract

We propose a novel adapter layer formalism for adapting multilingual models. They are more parameter-efficient than existing adapter layers while obtaining as good or better performance. The layers are specific to one language (as opposed to bilingual adapters) allowing to compose them and generalize to unseen language-pairs. In this zero-shot setting, they obtain a median improvement of +2.77 BLEU points over a strong 20-language multilingual Transformer baseline trained on TED talks.

1 Introduction

Of the many virtues of multilingual neural machine translation (MNMT), arguably the most attractive is the promise of improving performance in the low resource setting (Johnson et al., 2017; Arivazhagan et al., 2019; Dabre et al., 2020). These models even allow for the extreme of these cases, namely to translate in language pair directions which are unseen at training time (*zero-shot setting* in this paper). Unfortunately, while performance in the low-resource setting indeed increases significantly, their zero-shot performance remains very low (Johnson et al., 2017). In this paper, we propose a neural architecture that allows to translate from any of the source languages towards any of the target languages seen in the training data, regardless of the presence of that specific language direction during training. For that, we build upon the recently proposed *adapter layers* for NMT (Bapna and Firat, 2019), by using *monolingual* (language-specific) adapter layers, instead of *bilingual* (language-pair specific) ones. This design difference improves their compositionality, permitting to combine any encoder adapter with other decoder adapters. Monolingual adapter layers perform as good as bilingual adapter layers in the non–zero-shot setting, are effective in the zero-shot setting and have the additional advantage of requiring fewer parameters.

[*]Work done during an internship at NAVER LABS Europe.

2 Related Work

Zero-shot translation is direct translation in a language pair unseen during training. Aharoni et al. (2019) analyze the zero-shot performance of MNMT models as a function of the number of language pairs. They observe that having more languages results in better zero-shot performance. However, several artifacts arise, as described by Dabre et al. (2020); Zhang et al. (2020); Aharoni et al. (2019); Arivazhagan et al. (2019), like off-target translation and insufficient modeling capacity of the MNMT models. Zhang et al. (2020) use language-aware layer normalization and linear transformation to improve some drawbacks of MNMT; they also rely massively on back-translation to improve zero-shot translation.

Adaptation to a new language pair may be addressed by training a multilingual model then fine-tuning it with parallel data in the language pair of interest (Neubig and Hu, 2018; Variš and Bojar, 2019; Stickland et al., 2020). Escolano et al. (2020) propose plug-and-play encoders and decoders per language, which take advantage of a single representation in each language but at the cost of larger model sizes. In order to add only a few trainable parameters per task, adapter modules – initially introduced for computer vision (Rebuffi et al., 2017, 2018) – were proposed for language modeling by Houlsby et al. (2019). Bapna and Firat (2019) used them for parameter-efficient adaptation in MNMT. The parameters of the original MNMT network (the *parent* model) remain fixed, which permits a high degree of parameter sharing. The final multilingual model (the *adapted* model) is just slightly larger than the original one. (Bapna

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pages 4465–4470,
November 16–20, 2020.

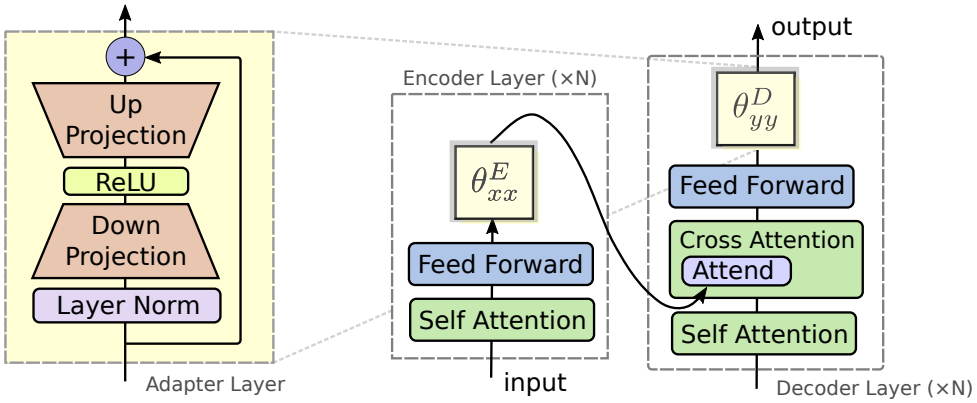

Figure 1: Proposed modification of the adapter layers from Bapna and Firat (2019). We use languages as the tasks in the encoder and in the decoder. *xx* and *yy* denote source and target languages respectively.

and Firat, 2019) show that adapters mitigate one major problem of MNMT models: performance drop in high-resource languages.

The motivation of that work was not zero-shot, and it is not obvious how to use them in such a scenario as the adapter layers are *language-pair* specific. While in Section 5 we propose a way of using those adapters through pivoting adapter layers, the main contribution of this paper is monolingual adapters which allow combining any encoder adapter with other decoder adapters.

adapt	#tasks	+ params/task	zero-shot
FT	$\mathcal{O}(n^2)$	$\mathcal{O}(K)$	✗
biling.	$\mathcal{O}(n^2)$	$\mathcal{O}(k)$	✓(pivot)
mono.	$\mathcal{O}(n)$	$\mathcal{O}(k)$	✓

Table 1: Number of parameters required for different adaptation techniques. FT denotes fine-tuning. K is the total number of model parameters, k is the number of parameters per set of adapter layers (with $k \ll K$), and n is the number of languages.

3 Monolingual adapters

Adapter modules (Rebuffi et al., 2017; Houlsby et al., 2019) were formulated for NMT by Bapna and Firat (2019): lightweight adapter layers are transplanted between the layers of a pre-trained network and fine-tuned on the adaptation corpus. As shown in Figure 1 (left), an adapter layer is a down projection to a bottleneck dimension followed by an up projection to the initial dimension. The bottleneck allows to limit the number of parameters of the adapter module. The residual connection coupled with a near-identity initialization enables a pass-through and allows keeping at least the performance of the parent model. In their initial formulation, Bapna and Firat (2019) proposed adapters for each *language pair* (bilingual adapters), while we propose monolingual adapters.

We illustrate the mechanism in Figure 1: our monolingual-adapter layers are inserted into each of the transformer encoder and decoder layers. When translating from language *xx* to language *yy*, we only activate the encoder adapter layers for *xx*, denoted by θ^E_{xx}; and the decoder adapter layers for *yy*, denoted by θ^D_{yy}.

Our formulation is different from Bapna and Firat (2019), who propose adapter layers for each language direction ($\theta_{xx\rightarrow yy}$). In a multiparallel setting (*i.e.*, where parallel data is available for all language pairs), this requires training $n(n-1)$ sets of layers, where n is the number of languages. Our monolingual (language-specific) adapters only require $2n$ layers. Table 1 summarizes the amount of parameters needed for adaptation with regular fine-tuning (FT), bilingual adapters (Bapna and Firat, 2019) and our proposed monolingual adapters. In our setting of 20 languages, fine-tuning would multiply the number of parameters by 380 (20×19). As the bottleneck dimension determines the increase of parameters, we experiment with both 64 (used in past work) and 1024, which matches the total number of parameters for bilingual adapters (see Table 2).

4 Experimental Setup

4.1 Datasets

We use the TED talks (Qi et al., 2018) in all our experiments, and all the numbers are BLEU scores

type	FT	mono.	biling.	mono.
bottleneck	–	64	64	1024
increase	×380	×1.47	×4.53	×3.73

Table 2: Increase in parameters over all tasks (380 language directions). FT denotes fine-tuning. The number of parameters of the parent model in this work is 68M. These parameter counts include the embeddings.

over the test set.[1] The TED talks dataset is multiparallel, *i.e.*, each English sentence has translations in multiple languages. Here, we restrict to the top 20 languages,[2] resulting in training corpora ranging between 108k and 214k parallel sentences. We use the dataset as a full multiparallel corpus (data aligned in all directions) and simulate an English-centric setting by using only parallel corpora with English as one of the languages.

4.2 Training

Architecture We use the Transformer architecture (Vaswani et al., 2017), implemented in `fairseq` (Ott et al., 2019), which we modify to include monolingual and bilingual adapters. We train a joint BPE model (Sennrich et al., 2016) on all languages, with inline casing (Berard et al., 2019) and 64k merge operations (resulting in a 70k vocabulary size). The Transformer architecture used in this work[3] has 4 attention heads, 6 encoder layers, 6 decoder layers, an embedding size of 512 and a feed-forward dimension of 1024.

MNMT Training We train a standard MNMT model following similar settings as Johnson et al. (2017). A single many-to-many model is trained on all the data English-centric data, using a source-side control token to indicate the target language. This model, which we call "parent", serves as an initialization for our adapter-enabled models. We use Adam (Kingma and Ba, 2015) with an inverse square root schedule, with 4000 warmup updates and a maximum learning rate of 0.0005. We set the maximum batch size per GPU to 4000 tokens, and train on 4 GPUs with mixed-precision (Ott et al., 2018). We apply dropout with a rate of 0.3, and label smoothing with a rate of 0.1. Like Arivazhagan et al. (2019), we mitigate the training size imbalance between language pairs by following a temperature-based sampling strategy with $T = 5$. To ensure all languages are represented adequately in the vocabulary, we use the same temperature-based sampling strategy for training the BPE model. This MNMT model is trained for 120 epochs over all the English-centric training data (38 language pairs). As shown in Table 3, it is a strong MNMT baseline.

Adapter Variations With monolingual adapters enabled, we optimize the adapter parameters for an additional 60 epochs with the same English-centric data. This setting lets us study the zero-shot capabilities of monolingual adapters. We also consider an "adaptation" setting where the monolingual adapters see data in all language pairs (380). In this setting, we only optimize adapter parameters for 10 epochs due to the increase of training time owing to more data. We use a bottleneck dimension of 64 for bilingual adapters, and try two values for the monolingual adapters: 64 and 1024. Table 2 shows how many extra parameters are added in each setting.

To train the adapters, we use the same settings as the parent MNMT model but reset the learning rate schedule and freeze all model parameters except the new adapter parameters. We train the 380 sets of bilingual adapters sequentially, as they are independent from each other. However, the monolingual adapters are trained all at once. To do so, we aggregate the training data for all language directions, using the same temperature-based sampling strategy as the parent model. For ease of implementation, we build homogeneous batches (*i.e.*, only containing sentences for one language direction) and only activate corresponding adapters. An epoch consists in a pass over the training data in all language directions (≈ 160k line pairs × 380 lang dirs ≈ 62M examples in the adaptation case, and ≈ 7.1M examples in the zero-shot case).

5 Results and Discussion

We evaluate the effectiveness of monolingual adapters in two settings: adaptation, where multiparallel training data is available for the adapters; and in the zero-shot setting where translation is done on unseen language pairs.

[1] Obtained by running `multi-bleu.perl`, or SacreBLEU with the `--tok none` option, as the TED talks dataset is pre-tokenized.

[2] `en, ar, he, ru, ko, it, ja, zh-cn, es, fr, pt-br, nl, tr, ro, pl, bg, vi, de, fa, hu`

[3] `transformer_iwslt_de_en` in `fairseq`

		xx→en					*en→yy*					*xx→yy*
		ar	*de*	*he*	*it*	*avg*$_{19}$	*ar*	*de*	*he*	*it*	*avg*$_{19}$	*avg*$_{342}$
(1)	Aharoni et al. (2019)	27.84	30.50	34.37	33.64	-	12.95	23.31	23.66	30.33	-	-
	Tan et al. (2019)	31.07	34.63	36.81	38.06	-	-	-	-	-	-	-
	Bilingual baselines	***32.99***	***37.36***	***39.00***	***39.73***	***32.42***	***17.22***	*29.94*	***27.47***	*35.42*	*24.37*	*14.96*
(2)	Aharoni et al. (2019)	28.32	32.97	33.18	35.14	-	14.25	27.95	24.16	33.26	-	-
	Parent	30.68	36.53	36.00	38.77	31.66	15.40	28.60	24.53	34.02	23.26	9.73
	Parent adaptation	29.63	35.83	35.10	37.91	30.85	14.45	27.49	22.87	32.43	22.25	14.82
	Mono-1024 zero-shot	30.87	35.87	35.87	38.14	31.21	*16.71*	***30.81***	*26.78*	***36.05***	***24.85***	12.94
	Mono-1024 adaptation	*32.66*	*37.03*	*37.76*	*38.81*	*32.29*	16.24	29.76	25.77	35.02	24.07	***15.83***

Table 3: BLEU scores of our models on the TED test sets compared to the literature. (1) Bilingual models. (2) Many-to-many MNMT models. The best model for each case is highlighted in italics and the best overall is in bold. Note that "(2) Aharoni et al. (2019)" is a 58-language model. "Parent" is our MNMT model trained on English-centric data. "Parent adaptation" is the same model fine-tuned for 10 epochs on the full multiparallel corpus (similar setting as "Mono-1024 adaptation", but without adapters). "Bilingual baselines" are models trained on one language direction only, with the same architecture as "Parent". Pivot-translating through English with "Parent" gives an average zero-shot performance (*xx→yy*) of 14.39 BLEU.

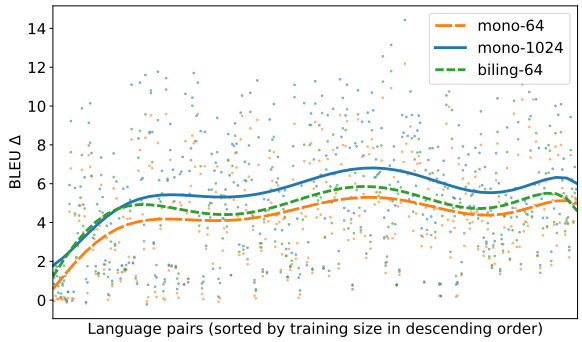

Figure 2: Comparison of bilingual adapters and monolingual adapters on the TED test sets in the adaptation setting, over 380 language pairs (sorted by available data). The y-axis shows the absolute BLEU differences with the parent model (trained on English-centric data only). The trendlines are obtained by interpolating a polynomial of degree 7 over the individual points.

5.1 Adaptation

In this setting, the adapter layers are trained on multiparallel data in 380 language pairs. Figure 2 shows the absolute difference in BLEU with the parent model, trained on English-centric data only, on each language pair. We compare bilingual adapters of dimension 64 with monolingual adapters of dimension 1024 or 64. As can be seen from the trendlines, while *mono-64* performs slightly (but consistently) worse than *biling-64*, *mono-1024* (which has a lower parameter budget than *biling-64*) obtains even better results, ranging from an absolute difference of -0.22 to +14.43, with a median of +5.59.

Because multilingual models are known to degrade performance on high-resource language directions, we study specifically translation to and from English. For *en→yy*, *mono-1024* (median +1.65) consistently outperforms *biling-64* (+1.24) and *mono-64* (+0.48) over the 19 language pairs. For *xx→en* however, *biling-64* adapters are slightly superior to both *mono-64* and *mono-1024* (+1.08 vs +0.09 and +0.50 respectively).

5.2 Zero-shot

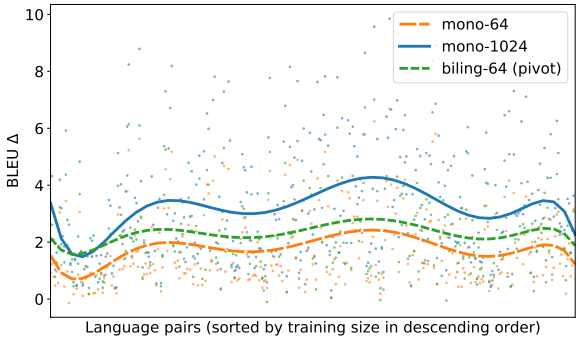

Figure 3: Comparison of bilingual adapters and monolingual adapters on the TED test sets in the zero-shot setting. The parent model and adapter-enabled models have only seen *xx→en* and *en→yy* data, and are tested on the remaining 342 pairs. The y-axis shows the absolute BLEU differences with the parent model.

Monolingual adapters can be naturally used for zero-shot translation, where a new language pair is provided at inference time. For this, we simply use the encoder adapters of the source language and the decoder adapters of the target language. To evaluate zero-shot translation, we use the adapter-enabled models trained on English-centric data and translate the test sets in the 342 language pairs not involving English (19 × 18).

Absolute improvements in BLEU scores of the

adapter-enabled models over the MNMT parent model are shown in Figure 3. A median improvement of +1.26 is observed in the *mono-64* setting, while the *mono-1024* setting brings a median improvement of +2.77. The smallest difference (over the parent model) observed in each case was -0.14 and +0.30 respectively, indicating near-systematic improvement by using monolingual adapter layers. These results demonstrate the compositionality property of our monolingual adapters.

Because of the English-centric nature of TEDx, we also apply bilingual adapters to the zero-shot setting. We do this by composing the encoder and decoder adapter layers through a pivot language. That is, to translate *xx→yy*, we choose the bilingual adapter corresponding to *xx→en* in the encoder and *en→yy* in the decoder. As can be seen in Figure 3, this slightly outperforms *mono-64* but not *mono-1024*.

6 Ablation Study

We investigate the individual contribution of the encoder and decoder adapter layers at inference time. We compare the full model using *mono-1024* adapters against the two options of activating (1) only encoder adapters (2) only decoder adapters.

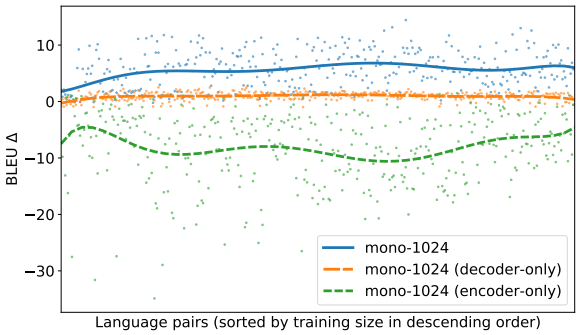

Figure 4: Ablation study for the adaptation setting, where we compare the full model using *mono-1024* adapter against its degraded versions with (1) only encoder adapters (2) only decoder adapters.

The interpolated curves for all language pairs are in Figure 4 for the adaptation setting and in Figure 5 for the zero-shot setting. In the adaptation setting, enabling only the decoder layers brings a median improvement of +1.03 over the parent model, while enabling only the encoder gives -7.00 BLEU (versus +5.59 when both encoder and decoders are enabled). In the zero-shot setting, the contribution of the encoder is larger (+1.69) than the decoder

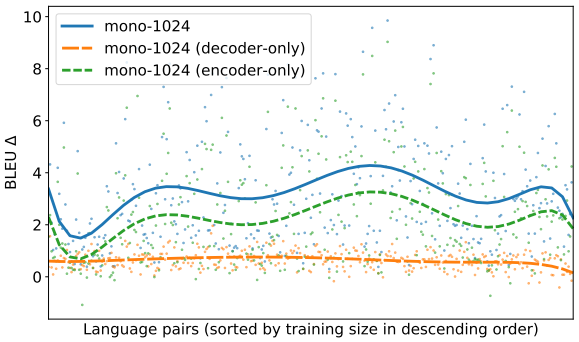

Figure 5: Ablation study for the zero-shot setting, where we compare the full model using *mono-1024* adapter against its degraded versions with (1) only encoder adapters (2) only decoder adapters.

(+0.63), compared to +2.77 when both encoder and decoder adapters are enabled.

We have seen that in the adaptation case, using the encoder adapter layers alone leads to a severe drop in performance. This might indicate that – at least during the adaptation – important information is captured in the encoder's adapter layer (in line with previous reports by Kudugunta et al., 2019) or that the decoder adaptation grows dependent on the encoder adapters, to the point where dropping the latter degrades the system. However, further analysis would be needed to confirm either of these hypotheses.

7 Conclusion

This work investigated adapter modules and their compositionality for MNMT, in particular in the zero-shot setting. We introduced *monolingual adapters* and compared them to *bilingual adapters*, which we also applied to zero-shot translation. Our adaptation experiments show the potential of the proposed monolingual adapters, which outperform bilingual adapters while having fewer parameters. In a zero-shot setting, we naturally compose our monolingual adapters and obtain a median improvement of +2.77 BLEU points over a strong MNMT model. Future work will investigate the compositional capability of these adapters, and combine domain and monolingual adapters for NMT.

More generally, this work adds to the growing evidence of the flexibility of adapter layers (Pfeiffer et al., 2020a), and their potential for lightweight fine-tuning, including in zero-shot scenarios (Pfeiffer et al., 2020b) and in a variety of tasks (Üstün et al., 2020).

Acknowledgements. We thank Marc Dymetman and Vassilina Nikoulina for fruitful discussions and comments on this paper.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. In *Proceedings of NAACL-HLT*.

Naveen Arivazhagan, Ankur Bapna, Orhan Firat, Dmitry Lepikhin, Melvin Johnson, Maxim Krikun, Mia Xu Chen, Yuan Cao, George Foster, Colin Cherry, et al. 2019. Massively multilingual neural machine translation in the wild: Findings and challenges. *arXiv preprint arXiv:1907.05019*.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *Proceedings of EMNLP-IJCNLP*.

Alexandre Berard, Ioan Calapodescu, and Claude Roux. 2019. Naver Labs Europe's systems for the WMT19 machine translation robustness task. In *Proceedings of WMT: Shared Task Papers*.

Raj Dabre, Chenhui Chu, and Anoop Kunchukuttan. 2020. A comprehensive survey of multilingual neural machine translation. *arXiv preprint arXiv:2001.01115*.

Carlos Escolano, Marta R Costa-jussà, José AR Fonollosa, and Mikel Artetxe. 2020. Multilingual machine translation: Closing the gap between shared and language-specific encoder-decoders. *arXiv preprint arXiv:2004.06575*.

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 *Proceedings of ICML*.

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. *TACL*.

Diederik P Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *Proceedings of ICLR*.

Sneha Kudugunta, Ankur Bapna, Isaac Caswell, and Orhan Firat. 2019. Investigating multilingual NMT representations at scale. In *Proceedings of EMNLP-IJCNLP*.

Graham Neubig and Junjie Hu. 2018. Rapid adaptation of neural machine translation to new languages. In *Proceedings of EMNLP*.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. In *Proceedings of NAACL-HLT: Demonstrations*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling Neural Machine Translation. In *Proceedings of WMT: Research Papers*.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. Adapterhub: A framework for adapting transformers. *arXiv preprint arXiv:2007.07779*.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. *arXiv preprint arXiv:2005.00052*.

Ye Qi, Devendra Sachan, Matthieu Felix, Sarguna Padmanabhan, and Graham Neubig. 2018. When and why are pre-trained word embeddings useful for neural machine translation? In *Proceedings of NAACL-HLT*.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. In *Proceedings of NeurIPS*.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2018. Efficient parametrization of multi-domain deep neural networks. In *Proceedings of CVPR*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of ACL*.

Asa Cooper Stickland, Xian Li, and Marjan Ghazvininejad. 2020. Recipes for adapting pre-trained monolingual and multilingual models to machine translation.

Xu Tan, Yi Ren, Di He, Tao Qin, Zhou Zhao, and Tie-Yan Liu. 2019. Multilingual neural machine translation with knowledge distillation. In *Proceedings of ICLR*.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly universal dependency parsing. *arXiv preprint arXiv:2004.14327*.

Dušan Variš and Ondřej Bojar. 2019. Unsupervised pretraining for neural machine translation using elastic weight consolidation. In *Proceedings of ACL: Student Research Workshop*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Proceedings of NeurIPS*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of ACL*.