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

Wen Lai, Alexandra Chronopoulou, Alexander Fraser

Center for Information and Language Processing, LMU Munich, Germany

7th December, 2022

Background

- We consider **two problems** of Multilingual Multi-Domain Neural Machine Translation (MNMT) adaptation:
  - **Domain Adaptation**: adapt the MNMT model to a new domain.
  - **Language Adaptation**: adapt the MNMT model to a new language pair.

Background

- We consider **two problems** of Multilingual Multi-Domain Neural Machine Translation (MNMT) adaptation:
  - **Domain Adaptation**: adapt the MNMT model to a new domain.
  - **Language Adaptation**: adapt the MNMT model to a new language pair.
- Common Practice
  - *fine-tuning* the model on new domain / language pair data for NMT (Freitag and Al-Onaizan, 2016; Dakwale and Monz, 2017).
  - use lightweight, learnable units inserted between transformer layers, which are called *adapters* (Bapna and Firat, 2019).

Challenges

- Fine-tuning methods require updating the parameters of the whole model for each new domain, which is costly;

Challenges

- Fine-tuning methods require updating the parameters of the whole model for each new domain, which is costly;
- When fine-tuning on a new domain, catastrophic forgetting reduces the performance on all other domains, and proves to be a significant issue when data resources are limited;

Challenges

- Fine-tuning methods require updating the parameters of the whole model for each new domain, which is costly;
- When fine-tuning on a new domain, catastrophic forgetting reduces the performance on all other domains, and proves to be a significant issue when data resources are limited;
- 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;

Challenges

- Fine-tuning methods require updating the parameters of the whole model for each new domain, which is costly;
- When fine-tuning on a new domain, catastrophic forgetting reduces the performance on all other domains, and proves to be a significant issue when data resources are limited;
- 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;
- Current methods consider the two problems separately.

Research Question

- Can we improve both two adaptation problems at the same time?

Research Question

- Can we improve both two adaptation problems at the same time?
- Can we transfer the language knowledge across domains and domain knowledge across languages?

Overview

- 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.

Overview

- 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.
- we propose a 2-step approach:
  - **Meta-Training**: 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);
  - **Meta-Adaptation**: we fine-tune the trained model to a new domain and language pair simultaneously using an even smaller dataset (500 samples).

Task Definition

- 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'.

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.
- We follow a temperature-based heuristic sampling strategy Aharoni et al., 2019[1], which defines the probability of any dataset as a function of its size.

[1] Massively Multilingual Neural Machine Translation (Aharoni et al., NAACL 2019)

Meta-Learning Algorithm

- We follow *Reptile* (Nichol et al., 2018[2]), an alternative first-order meta-learning algorithm that uses a simple update rule:

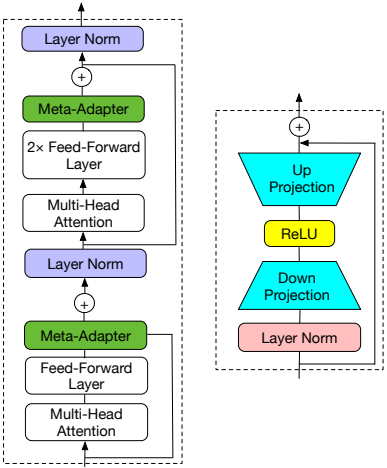
$$\psi \leftarrow \psi + \beta \frac{1}{|\{\mathcal{T}_i\}|} \sum_{\mathcal{T}_i \sim \mathcal{M}} (\psi_i^{(k)} - \psi)$$

- Where $\psi_i^{(k)}$ is $U_i^k(\theta, \psi)$ and β is a hyper-parameter.

[2] On first-order meta-learning algorithms (Nichol et al., arXiv preprint arXiv:1803.02999 (2018))

Meta-Adapter

- We propose training a *Meta-Adapter*, which inserts adapter layers into the meta-learning training process.
- 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.



Meta-Adaptation

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

Meta-Adaptation

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

Datasets

- We split the datasets in two groups: *meta-training* and *meta-adapting*.
- 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 et al., 2012).

[3] https://en.wikipedia.org/wiki/List_of_ISO_639-1_codes
[4] https://opus.nlpl.eu

Baselines

- **m2m**: Using the original m2m model (Fan et al., 2021) to generate the translations;
- **m2m + FT**: Fine-tuning m2m on all DLPs;
- **m2m + tag**: Fine-tuning m2m with domain tags (Kobus et al., 2017) on all DLPs;
- **agnostic-adapter**: Mixing the data from all DLPs to train the adapters (Cooper Stickland et al., 2021b), to obtain language and domain-agnostic adapters;
- **stack-adapter**: Training two adapters for each language pair and domain, then stacking both adapters (Cooper Stickland et al., 2021a);
- **meta-learning**: Traditional meta-learning methods using the MAML algorithm (Sharaf et al., 2020) on all DLPs.

Main Results

- Motivated by Lai et al., 2022[a], we compare our approach to multiple baselines in terms of domain robustness.
- *m*^{4}*Adapter* obtains a performance that is on par or better than *agnostic-adapter*, which is a robust model.

[a] Improving Both Domain Robustness and Domain Adaptability in Machine Translation (Lai et al., COLING 2022)

	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**

Main Results

	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*^{4}*Adapter	**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

- *m*^{4}*Adapter* 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.

Efficiency

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

Method	#Param.	Time_T	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*^{4} *Adapter	3.17M (0.75%)	34%	300%

Efficiency

- We compare the efficiency of baselines to traditional fine-tuning and list their number of trainable parameters and training/adapting time in the table.
- *m*^{4}*Adapter* only updates the adapter parameters while freezing the MNMT model's parameters. therefore, it has fewer trainable parameters compared to fine-tuning.

Method	#Param.	Time$_T$	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*^{4}*Adapter	3.17M (0.75%)	34%	300%

Efficiency

- We compare the efficiency of baselines to traditional fine-tuning and list their number of trainable parameters and training/adapting time in the table.
- *m^4Adapter* only updates the adapter parameters while freezing the MNMT model's parameters. therefore, it has fewer trainable parameters compared to fine-tuning.
- Our approach requires more time than traditional adapter methods but is faster compared with updating the entire model using traditonal meta-learning.

Method	#Param.	Time_T	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%

Domain Transfer via Languages

- We define domain transfer via languages, i.e., the ability to transfer domains while keeping the languages unchanged.

	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*^{4}*Adapter	**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

Domain Transfer via Languages

- We define domain transfer via languages, i.e., the ability to transfer domains while keeping the languages unchanged.

	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*^{4}*Adapter	**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

- We observe that almost all baseline systems and *m*^{4}*Adapter* outperform the original m2m model, indicating that the model encodes language knowledge and can transfer this knowledge to new *meta-adaptation* domains.

Domain Transfer via Languages

- We define domain transfer via languages, i.e., the ability to transfer domains while keeping the languages unchanged.

	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*^{4}*Adapter	**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

- We observe that almost all baseline systems and *m*^{4}*Adapter* outperform the original m2m model, 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.

Domain Transfer via Languages

- We define domain transfer via languages, i.e., the ability to transfer domains while keeping the languages unchanged.

	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*^{4}*Adapter	**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

- We observe that almost all baseline systems and *m*^{4}*Adapter* outperform the original m2m model, 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.

Language Transfer via Domains

- We define language transfer via domains, i.e., the ability to transfer languages while keeping the domains unchanged.

	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*^{4}*Adapter	**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

Language Transfer via Domains

- We define language transfer via domains, i.e., the ability to transfer languages while keeping the domains unchanged.

	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*^{4}*Adapter	**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

- The performance of traditional fine-tuning approaches 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.

Language Transfer via Domains

- We define language transfer via domains, i.e., the ability to transfer languages while keeping the domains unchanged.

	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*^{4}*Adapter	**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

- The performance of traditional fine-tuning approaches 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.
- In contrast, *m*^{4}*Adapter* shows a performance that is on par or better than the *agnostic-adapter*, the most competitive model in all baseline systems.

Conclusion

- We present *m*^{4}*Adapter*, a novel multilingual multi-domain NMT adaptation framework which combines meta-learning and parameter-efficient fine-tuning with adapters.
- *m*^{4}*Adapter* is effective on adapting to new languages and domains simultaneously in low-resource settings.
- We show that *m*^{4}*Adapter* transfers domain knowledge across different languages and language information across different domains.
- In addition, *m*^{4}*Adapter* is efficient in training and adaptation, which is practical for online adaptation Etchegoyhen et al., 2021[5] to complex scenarios (new languages and new domains) in the real world.

[5] Online Learning over Time in Adaptive Neural Machine Translation (Etchegoyhen et al., RANLP 2021)

Thank You!

Email: lavine@cis.lmu.de
Homepage: `https://lavine-lmu.github.io`
Address: Oettingenstraße 67, 80538 Munich, Germany

Paper Code Blog