

Efficient Test Time Adapter Ensembling for Low-resource Language Varieties

Xinyi Wang¹ Yulia Tsvetkov² Sebastian Ruder³ Graham Neubig¹

¹Language Technology Institute, Carnegie Mellon University

²Paul G. Allen School of Computer Science & Engineering, University of Washington

³DeepMind

xinyiwl@cs.cmu.edu, yuliats@cs.washington.edu
ruder@google.com, gneubig@cs.cmu.edu

Abstract

Adapters are light-weight modules that allow parameter-efficient fine-tuning of pre-trained models. Specialized language and task adapters have recently been proposed to facilitate cross-lingual transfer of multilingual pre-trained models (Pfeiffer et al., 2020b). However, this approach requires training a separate language adapter for every language one wishes to support, which can be impractical for languages with limited data. An intuitive solution is to use a related language adapter for the new language variety, but we observe that this solution can lead to sub-optimal performance. In this paper, we aim to improve the robustness of language adapters to uncovered languages without training new adapters. We find that ensembling multiple existing language adapters makes the fine-tuned model significantly more robust to other language varieties not included in these adapters. Building upon this observation, we propose Entropy Minimized Ensemble of Adapters (EMEA), a method that optimizes the ensemble weights of the pre-trained language adapters for each test sentence by minimizing the entropy of its predictions. Experiments on three diverse groups of language varieties show that our method leads to significant improvements on both named entity recognition and part-of-speech tagging across all languages.

1 Introduction

Massively multilingual pre-trained models (Devlin et al., 2019; Huang et al., 2019; Conneau and Lample, 2019; Conneau et al., 2020) combined with cross-lingual transfer now define the state of the art on a variety of NLP tasks (Hu et al., 2020). Within this paradigm, multilingual pre-trained models are fine-tuned on annotated data of a task in a high-resource language, and transferred to other languages. Several recent works propose parameter-efficient fine-tuning methods that insert small *adapter* modules between the layers

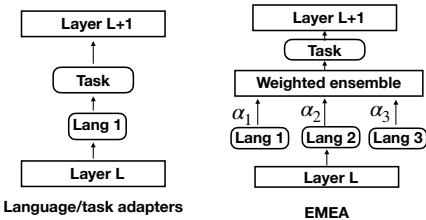


Figure 1: Comparison of the standard cross-lingual adapter and our method of entropy minimized ensembling of adapters (EMEA), which combines multiple language adapters to improve robustness to new language varieties at *test time*.

of pre-trained models (Rebuffi et al., 2017; Houlsby et al., 2019). In this line of work, the pre-trained model is usually frozen while only the adapters are fine-tuned for a downstream task, which is conducive to both improving the model’s learning ability and compactness with respect to storage on disk or in memory. The adapters can be applied to the cross-lingual transfer setting by training separate language and task adapters (Pfeiffer et al., 2020b; Üstün et al., 2020). Specifically, Pfeiffer et al. (2020b) propose to perform zero-shot transfer by first training language-level adapters on monolingual data in different languages and then a task adapter on annotated data in the source language.

One drawback of this framework is that a separate language adapter is required for each target language, which is problematic in cases where the data to train these adapters cannot be easily obtained, such as for languages with diverse regional or demographic variations. In fact, certain language varieties are not included in the standard language identification tools, which makes it challenging to reliably obtain even unlabeled data (Salameh et al., 2018; Caswell et al., 2020; Demszky et al., 2021). To give just one example, the Nordic languages and dialects form a dialect continuum where the total number of language varieties is difficult to estimate, and language varieties constantly emerge in culturally and linguistically diverse areas (Svendsen and Røyneeland, 2008; Røyneeland and Jensen, 2020). Although highly related, these language

varieties have many systematic differences, which need to be addressed by NLP systems that equitably serve all speakers (Kumar et al., 2021). One potential mitigation strategy is directly using an adapter trained on another similar language variety, but we find this sub-optimal in experiments (§ 4).

Instead, we propose two methods to combine existing language adapters to *adapt the model to new language varieties at test time* without any training data. First, we find that simply ensembling multiple related language adapters can significantly improve the fine-tuned model, compared with using individual language adapters. Second, we propose Entropy Minimized Ensemble of Adapters (EMEA; Fig. 1), which adapts the ensemble weight of the language adapters for each test instance by minimizing the ensembled model’s prediction uncertainty. Our experiments show that EMEA further improves over vanilla ensembling for three groups of uncovered language varieties on both the named entity recognition and part-of-speech tagging tasks.

2 Adapters for Cross-lingual Transfer

To facilitate our discussion, we briefly summarize the MAD-X framework (Pfeiffer et al., 2020b) for zero-shot cross-lingual transfer and identify its shortcomings. The goal of MAD-X is to fine-tune a multilingual pretrained model \mathcal{M} to m downstream tasks T_1, T_2, \dots, T_m , each of which could be in n languages L_1, L_2, \dots, L_n . To this end, MAD-X relies on language and task adapters, which are lightweight functions inserted in the Transformer layers in \mathcal{M} —usually a feed-forward down-projection followed by an up-projection. Specifically, let h be the output of an intermediate layer in \mathcal{M} , then $\mathcal{L}_j(h)$ is the transformation that projects h into the embedding space for language L_j , and $\mathcal{T}_i(\mathcal{L}_j(h))$ is the transformation that projects $\mathcal{L}_j(h)$ into the embedding space for task T_i .

MAD-X trains the adapters $\mathcal{T}_i(\cdot)$ and $\mathcal{L}_j(\cdot)$ in two steps. First, for each language L_j , its adapter \mathcal{L}_j is inserted into \mathcal{M} to replace the output of each layer h with $\mathcal{L}_j(h)$. The resulting model, which we denote as $\mathcal{L}_j \circ \mathcal{M}$, is trained on unlabeled data in L_j using an unsupervised objective such as masked language modeling (MLM; Devlin et al., 2019). Second, for each task T_i , its adapter \mathcal{T}_i is inserted on top of a *src* language adapter \mathcal{L}_{src} . The resulting model $\mathcal{T}_i \circ \mathcal{L}_{src} \circ \mathcal{M}$ is trained on the downstream task T_i in language L_{src} . After these two steps, $\mathcal{T}_i \circ \mathcal{L}_j \circ \mathcal{M}$ can be used to perform zero-shot cross-

lingual transfer for any task T_i and language L_j .

Shortcomings This approach requires a separate adapter for each language one wishes to support. The online database AdapterHub¹ aims to improve the efficiency and reuse of trained language and task adapters (Pfeiffer et al., 2020a) but currently supports only about 50 languages, and hence most languages are not covered. More importantly, as mentioned in the introduction, certain languages have diverse regional varieties and difficulty of reliably obtaining data for them makes adapter-based approaches especially brittle in these cases. In the following § 3, we propose strategies to improve the robustness of language adapters to uncovered languages without training new adapters.

3 Generalizing Language Adapters to Related Languages

We consider the setting where we have a multilingual pretrained model \mathcal{M} as well as the pretrained task adapters $\mathcal{T}_1, \mathcal{T}_2, \dots, \mathcal{T}_m$ and language adapters $\mathcal{L}_1, \mathcal{L}_2, \dots, \mathcal{L}_n$. We want to use \mathcal{M} and the existing adapters to support a new language L_{new} , which is not in $\{L_1, L_2, \dots, L_n\}$ on a given task T without training a new adapter for \mathcal{L}_{new} .

Related Language Adapters One potential solution is to find the most related language $L_{rel} \in \{L_1, L_2, \dots, L_n\}$ and then use $\mathcal{T} \circ \mathcal{L}_{rel} \circ \mathcal{M}$ to do inference in L_{new} . However, this has two disadvantages. First, the task adapter \mathcal{T} is only trained in the setting of $\mathcal{T} \circ \mathcal{L}_{src} \circ \mathcal{M}$, so it might not generalize well to the test time setting of $\mathcal{T} \circ \mathcal{L}_{rel} \circ \mathcal{M}$ (as shown in § 4.1). Second, while the pretrained model \mathcal{M} may be relatively robust against distribution shifts (Hendrycks et al., 2020), the specialized language adapters might make the model brittle to language variations because they are trained for specific languages. Our experiments in § 4.1 show that this solution indeed leads to poor performance.

Adapter Ensembling As a first solution to this problem, we propose an extremely simple strategy of averaging the transformed outputs of multiple language adapters. Specifically, we use both the source language adapter \mathcal{L}_{src} and adapters from related languages with similar linguistic properties to the new language. Let \mathcal{R} be the set of the source and related language adapters. To do inference on a task T for the new language L_{new} , we transform

¹<https://adapterhub.ml/>

the output h of each layer in \mathcal{M} with the language adapters as $\mathcal{L}_{\text{avg}}(h) = \frac{1}{R} \sum_{i=1}^R \mathcal{L}_i(h)$.

Entropy Minimized Ensemble of Adapters

While ensembling is a simple and effective strategy to combine multiple potentially beneficial language adapters, the equal weighing of all language adapters could be sub-optimal for L_{new} ; different language varieties, or even sentences, could benefit from a different weighting of the pretrained language adapters. To further improve adapter ensembling, we generalize $\mathcal{L}_{\text{avg}}(h)$ into a learnable weighted average:

$$\mathcal{L}_{\text{wavg}}(h) = \sum_{i=1}^R \alpha_i \mathcal{L}_i(h)$$

where $\alpha_1, \alpha_2, \dots, \alpha_R$ are learnable weights satisfying $\alpha_i \geq 0$ and $\sum_{i=1}^R \alpha_i = 1$. Next, we propose Entropy Minimized Ensemble of Adapters (EMEA) method, which learns the adapter weightings *for each sentence* without additional training.

The intuition behind our method is that a good adapter weight α for a test input x should make the model more confident in its prediction for x , that is, it should lead to lower model entropy over the input (Shannon, 1948; Wang et al., 2021). Specifically for structured prediction tasks, we want to classify each word x_w in a test input x with W words into one of the possible C classes. We consider the entropy: $H(x; \alpha) = -\sum_{w=1}^W \sum_{c=1}^C P(c|x_w; \alpha) \log P(c|x_w; \alpha)$, where $P(c|x_w; \alpha)$ is the prediction of the model $\mathcal{T} \circ L_{\text{wavg}}(h) \circ \mathcal{M}$. Since $P(c|x_w; \alpha)$ is a function of the ensemble weights α , we can calculate the gradient of α as $g_i = \nabla_{\alpha_i} H(x; \alpha)$.

To minimize the entropy loss, we can simply do gradient descent steps on each α_i using the corresponding gradient g_i by $\alpha_i = \alpha_i - \gamma g_i$, where γ is the learning rate. We can then use the updated α to calculate the final prediction for x . In § 4, we find that a single step of gradient update already leads to better performance than simple ensembling. We can additionally perform multiple steps of gradient descent to obtain a better α at the cost of lower inference speed. Alg. 1 shows the pseudo code of our method².

4 Experiments

Data We focus on zero-shot cross-lingual transfer with English as the source language. We

Algorithm 1: Training with EMEA

```

Input :Uniform weights  $\alpha^0$ , weighted adapter
       output;  $L_{\text{wavg}}(h, \alpha^0)$ ; test data  $x$ ; number of
       update steps  $T$ 
Output:Prediction  $\hat{y}$ 
1 for  $t$  in  $0, 1, \dots, T-1$  do
    $\triangleright$  Calculate entropy
2    $H(x, \alpha) \leftarrow \text{Entropy}(\mathcal{T} \circ L_{\text{wavg}}(h, \alpha^t) \circ \mathcal{M})$ 
    $\triangleright$  Calculate gradient
3    $g^t = \nabla_{\alpha} H(x; \alpha^t)$ 
    $\triangleright$  Update weighting
4    $\alpha^{t+1} \leftarrow \text{Update}(\alpha^t, g^t)$ 
5 end
    $\triangleright$  Calculate final prediction
6    $\hat{y} \leftarrow \text{Predict}(\mathcal{T} \circ L_{\text{wavg}}(h, \alpha^T) \circ \mathcal{M})$ 

```

Related	Additional	Test
hi	en,ar	mr, bn, ta, bho
is	en,de	fo, no, da
ru	en	be, uk, bg

Table 1: Test language groups and their corresponding language adapters. Adapters from languages in the first two columns are applied to the test languages in the third column.

conduct experiments on named entity recognition (NER) and part-of-speech tagging (POS). We use the WikiAnn dataset (Pan et al., 2017) for NER and Universal Treebank 2.0 for POS tagging (Nivre et al., 2018).

Model We use the mBERT (Devlin et al., 2019) model, which shows good performance for low-resource languages on the structured prediction tasks (Pfeiffer et al., 2020b; Hu et al., 2020). We use the English annotated data to train the task adapter. Each experiment is run with 3 different random seeds and we report the average performance. More details can be found in Appendix A.

Languages Due to the scarcity of datasets for dialects, we focus on three groups of closely related languages to simulate the setup of language varieties. Each group has a language with a pre-trained adapter available on the AdapterHub (Pfeiffer et al., 2020a), and we test on the languages without adapters. The language with adapter and the target languages for each group are: 1. Hindi (hi): Marathi (mr), Bengali (bn), Tamil (ta), Bhojpuri (bho); 2. Icelandic (is): Faroese (fo), Norwegian (no), Danish (da); 3. Russian (ru): Bulgarian (bg), Ukrainian (uk), Belorussian (be). For our methods, we additionally use the adapter for English (the *src* language), and optionally for another highly related language if there is one available on the AdapterHub. The adapters used are listed in Tab. 1.

²Code can be found at <https://github.com/cindyxinyiwang/emea>

Task	Method	mr	bn	ta	avg.	fo	no	da	avg.	be	uk	bg	avg.	avg.
NER	En	48.0	54.4	29.6	44.0	57.5	73.3	80.5	70.4	67.1	67.6	71.1	68.6	61.0
	Related	51.7	47.0	30.8	43.1	54.3	72.7	79.3	68.7	66.2	65.8	69.8	67.3	59.7
	CL	48.1	55.2	28.9	44.1	57.5	73.6	80.6	70.6	67.0	67.8	71.0	68.6	61.1
	Fusion	49.8	58.3	33.7	47.2	56.0	69.3	77.8	67.7	70.1	69.1	72.3	70.5	61.8
POS	Ensemble	55.5	55.3	35.8	48.8	57.4	74.0	80.8	70.7	70.5	72.2	74.2	72.3	63.9
	EMEA-s1	57.2	61.2	37.4	51.9	59.2	74.3	81.3	71.6	71.5	72.9	74.9	73.1	65.5
	EMEA-s10	57.5	63.2	38.3	53.0	61.6	74.9	82.0	72.8	72.9	72.9	75.1	73.6	66.5
	Method	mr	bho	ta	avg.	fo	no	da	avg.	be	uk	bg	avg.	avg.
POS	En	62.6	39.5	53.4	51.8	71.6	84.6	87.6	81.1	85.3	81.4	84.6	83.7	72.2
	Related	53.2	46.9	47.0	49.0	72.8	82.4	86.9	80.7	84.0	79.5	82.9	82.1	70.6
	CL	62.6	39.6	53.6	51.9	71.7	84.2	87.7	81.2	85.6	81.5	84.7	83.9	72.3
	Fusion	59.8	42.3	53.5	51.8	72.9	81.3	86.0	80.0	85.8	80.0	83.3	83.0	71.6
POS	Ensemble	62.2	45.5	53.7	53.8	73.9	83.6	87.9	81.8	85.9	81.6	84.6	84.0	73.2
	EMEA-s1	62.1	45.1	54.3	53.8	74.0	83.5	87.8	81.7	86.2	81.4	84.6	84.0	73.2
	EMEA-s10	62.5	44.9	55.6	54.3	73.8	83.7	88.0	81.8	86.0	81.6	84.9	84.2	73.5

Table 2: F1 of the baselines and our methods for each language group. EMEA-s1 updates the adapter weights with a single gradient step while EMEA-s10 updates for 10 steps.

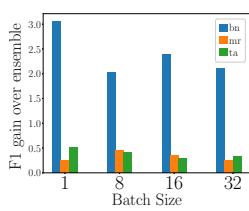


Figure 2: Improvements over ensemble with different batch size.

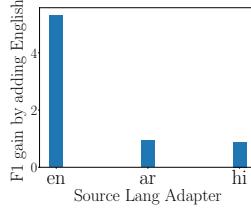


Figure 3: Improvements by adding en adapter for different src language adapters.

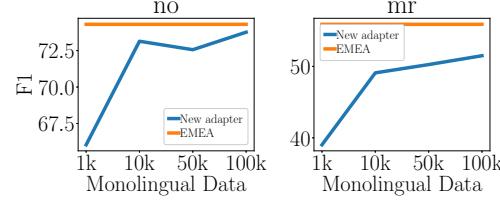


Figure 4: Comparison to training adapter on different amount of monolingual data.

Baselines We compare with several baselines: 1) En: the English adapter; 2) Related: the best performing related language adapter; 3) Continual learning (CL): we use the English language adapter and update its parameters using the entropy loss for each test input; 4) Fusion: learn another set of key, value and query parameters in each layer that uses the layer output as a query to mix together the output of each adapter (Pfeiffer et al., 2021). Since we do not use labeled data in the new language, we train the fusion parameters on English labeled data.

4.1 Results

The results can be found in Tab. 2. For most languages using the English adapter is better than the best individual related language adapter. This confirms our hypothesis that specialized language adapters are not robust to language variations. CL leads to slight improvements for some languages but is generally comparable to En. Fusion improves over En for the NER task but it requires training and storing extra parameters. Its performance is also not consistent across languages and tasks, likely because it is only trained on English labeled data.

Using multiple language adapters brings significant gains Ensembling leads to significant gains for the non-Latin language group. It also brings im-

provements or is comparable to the best baseline on other languages. EMEA delivers further improvements across almost all languages, demonstrating the effectiveness of adapting language adapter weights to each test sentence. With only a single gradient update step on the ensemble weights, EMEA-s1 already leads to significant improvements over ensembling for NER. EMEA-s10 brings additional improvements on both tasks because it learns more optimal ensembling weights with 10 gradient update steps (we list the inference cost for each method in Appendix B). We hypothesize that the proposed methods improve non-Latin languages more because these are low-performing languages that the model is more uncertain about.

Effect of test batch size In Fig. 2 we plot the result of using different test batch sizes with EMEA on the NER task. A smaller batch size leads to more fine-grained test time adaptation with a higher computational cost. Fig. 2 shows that a smaller batch size indeed leads to better performance while using a larger batch size still outperforms the baseline.

Significance of source language adapter We investigate whether the benefit of adding the *src* language adapter comes from the discrepancy between training and testing of the task adapter. We train

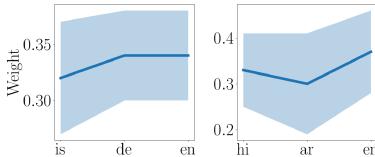


Figure 5: Mean and standard deviation of the weight for each adapter for the *is* (left) and *hi* (right) language groups.

different task adapters with language adapters other than English (*en*), and compare the improvement of adding the *en* adapter for the ensemble. Fig. 3 shows that the *en* adapter provides the largest benefit when it is used to train the task adapter, which verifies that using different language adapters with the task adapter between training and testing leads to sub-optimal cross-lingual transfer performance.

Comparison to training new adapters In order to better understand how much data is required to train new language adapters that are competitive with EMEA, we trained new adapters using a small amount of monolingual data in the target language. We focus on two languages, *mr* and *no*, on the NER task, and show the results in Fig. 4. Note that this setting puts EMEA at a disadvantage because EMEA does not require any training. It takes about 100k monolingual data for *no* to reach comparable performance with our method, while *mr* still lags behind EMEA. As large amounts of monolingual data are difficult to obtain for many language varieties and under-represented languages, EMEA can serve as a useful baseline for applying NLP models to such low-resource settings.

Analysis of weights We plot the mean and standard deviation of ensembling weights from EMEA in Fig. 5. The *en* adapter gets the highest weight for both language groups, in line with the results in Tab. 2 showing *en* as the best individual adapter. For the *hi* language group, the *ar* adapter tends to have the least benefit, probably because it has a different script from the languages we test on.

5 Related Work

Our work is related to parameter efficient fine-tuning of pretrained models (Bapna et al., 2019; Pfeiffer et al., 2020b; Li and Liang, 2021; Guo et al., 2021). Specifically, (Üstün et al., 2020; Karimi Mahabadi et al., 2021) make adapters more generalizable by learning a parameter generator, while our work aims to utilize existing pretrained adapters without further training. Pfeiffer et al. (2021) propose to learn extra parameters using la-

beled data to combine pretrained multitask adapters whereas our method does not require any training or labeled data. While we focus on language adapters in this work, our method is also applicable to ensembling domain or task adapters. Finally, our method is inspired by the test time adaptation framework proposed for image classification (Sun et al., 2020; Wang et al., 2021; Kedia and Chinthakindi, 2021). Instead of adapting a single model, we focus on efficient utilization of many pre-trained language adapters to improve the model’s robustness to language variations.

6 Discussion and Conclusion

Language and dialect cannot be simply categorized into monolithic entities. Thus a truly intelligent NLP system should be able to recognize and adapt to personalized language varieties after it is trained and deployed. However, the standard system evaluation is built on the assumption that an NLP model is fixed once it is trained. In this paper, we focus on a specific case of this general problem—we find that specialized language adapters might not be robust to unseen language variations, and that utilization of multiple existing pretrained language adapters alleviates this issue. We hope our findings can inspire future work on models that are robust and adaptive to language variations.

We identify two limitations of this paper, which we leave to future work. First, there are limited datasets and benchmarks that evaluate NLP models’ ability to generalize to unseen dialect variations. Therefore, we only test our method on NER and POS tagging tasks because they have the best language coverage. It is an important future direction to construct high-quality datasets that consider language and dialect variations. Second, our method has slower inference speed due to test time computation. Future work can aim to reduce the cost by algorithmic or hardware innovations.

Acknowledgement

This material is based on work supported by the National Science Foundation under grants 2040926 and 2125201. XW is supported by the Apple PhD fellowship. The authors would like to thank Laura Rimell, Sachin Kumar and Hieu Pham for helpful discussions on the drafts of the paper.

References

Ankur Bapna, Naveen Arivazhagan, and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. In *EMNLP*.

Isaac Caswell, Theresa Breiner, Daan van Esch, and Ankur Bapna. 2020. Language ID in the wild: Unexpected challenges on the path to a thousand-language web text corpus. In *COLING*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised cross-lingual representation learning at scale. In *ACL*.

Alexis Conneau and Guillaume Lample. 2019. Crosslingual language model pretraining. In *NeurIPS*.

Dorottya Demszky, Devyani Sharma, Jonathan H. Clark, Vinodkumar Prabhakaran, and Jacob Eisenstein. 2021. Learning to recognize dialect features. In *NAACL*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Demi Guo, Alexander M. Rush, and Yoon Kim. 2021. Parameter-efficient transfer learning with diff pruning. In *ICLR*.

Dan Hendrycks, Xiaoyuan Liu, Eric Wallace, Adam Dziedzic, Rishabh Krishnan, and Dawn Song. 2020. Pretrained transformers improve out-of-distribution robustness. In *ACL*.

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 *ICML*.

Junjie Hu, Sebastian Ruder, Aditya Siddhant, Graham Neubig, Orhan Firat, and Melvin Johnson. 2020. Xtreme: A massively multilingual multi-task benchmark for evaluating cross-lingual generalization. In *ICML*.

Haoyang Huang, Yaobo Liang, Nan Duan, Ming Gong, Linjun Shou, Dixin Jiang, and Ming Zhou. 2019. Unicoder: A universal language encoder by pre-training with multiple cross-lingual tasks. In *EMNLP*.

Rabeeh Karimi Mahabadi, Sebastian Ruder, Mostafa Dehghani, and James Henderson. 2021. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *ACL*, Online.

Akhil Kedia and Sai Chetan Chinthakindi. 2021. Keep learning: Self-supervised meta-learning for learning from inference. In *EACL*.

Sachin Kumar, Antonios Anastasopoulos, Shuly Wintner, and Yulia Tsvetkov. 2021. Machine translation into low-resource language varieties. In *ACL*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. In *Arxiv*.

Joakim Nivre, Mitchell Abrams, Željko Agić, Lars Ahrenberg, Lene Antonsen, Maria Jesus Aranzabe, Gashaw Arutie, Masayuki Asahara, Luma Ateyah, Mohammed Attia, et al. 2018. Universal dependencies 2.2.

Xiaoman Pan, Boliang Zhang, Jonathan May, Joel Nothman, Kevin Knight, and Heng Ji. 2017. Cross-lingual name tagging and linking for 282 languages. In *ACL*, pages 1946–1958.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. 2021. Adapterfusion: Non-destructive task composition for transfer learning. In *EACL*, Online.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020a. Adapterhub: A framework for adapting transformers. In *EMNLP System Demonstration*, Online.

Jonas Pfeiffer, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020b. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *EMNLP*, Online.

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

Unn Røyneland and Bård Uri Jensen. 2020. Dialect acquisition and migration in norway – questions of authenticity, belonging and legitimacy. In *Journal of Multilingual and Multicultural Development*.

Mohammad Salameh, Houda Bouamor, and Nizar Habash. 2018. Fine-grained Arabic dialect identification. In *COLING*.

C.E. Shannon. 1948. A mathematical theory of communication. In *Bell system technical journal*.

Yu Sun, Xiaolong Wang, Liu Zhuang, John Miller, Moritz Hardt, and Alexei A. Efros. 2020. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*.

Bente Ailin Svendsen and Unn Røyneland. 2008. Multietnolectal facts and functions in oslo, norway. In *Internatioanl Journal of Bilingualism*.

Ahmet Üstün, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly Universal Dependency parsing. In *EMNLP*.

Dequan Wang, Evan Shelhamer, Shaoteng Liu, Bruno Olshausen, and Trevor Darrell. 2021. Tent: Fully test-time adaptation by entropy minimization. In *ICLR*.

Method	Example/Second
Single Adapter	250
CL	77
Fusion	200
Ensemble	200
EMEA-s1	62
EMEA-s10	9

Table 3: Decoding speed for different methods used in the paper.

A Implementation Details

We preprocess the data using scripts in XTREME (Hu et al., 2020). We use the best performing adapter configuration in Pfeiffer et al. (2020b). For NER, we train the task adapter for 100 epochs using learning rate of 1e-4. For POS tagging, we train the task adapter for 50 epochs with the learning rate of 1e-4. For EMEA, we search over the learning rate γ of 0.1, 1, 10 on the English validation set and pick $\gamma = 10$ for all experiments.

For Fusion, we use learning rate of 5e-5 which is recommended by (Pfeiffer et al., 2021). We search over the best learning rate for CL on the performance of English labeled data. We use the learning rate of 2e-5 and do 1 step of gradient update for each batch.

For our experiment on training new adapters, we find that training from scratch on no and mr is not competitive when using very small amount of data. Therefore, we continue training from their related language adapters.

B Decoding Speed

We list the inference time for various methods in the paper in Tab. 3. EMEA leads to better performance at a cost of lower inference speed. We leave it to future work to explore strategies that speed up the test time optimization.

C Examples of outputs

We compare the outputs of EMEA with the best baseline on the POS tagging task for Norwegian (no). Although both methods struggle with verb and adjective predictions, EMEA is often better at predicting the correct adjectives compared to the baseline.

src	Lendið, er, kargt, og, oyði, .
tgt	NOUN, VERB, ADJ, CCONJ, ADJ, PUNCT
Base	NOUN, AUX , ADJ, CCONJ, NOUN , PUNCT
EMEA	NOUN, AUX , ADJ, CCONJ, ADJ , PUNCT
src	Útvinnningin, er, í, tveimur, umfórum, .
tgt	NOUN, VERB, ADP, NUM, NOUN, PUNCT
Base	NOUN, AUX , ADP, ADJ , NOUN, PUNCT
EMEA	NOUN, VERB , ADP, NUM , NOUN, PUNCT

Table 4: Example outputs on POS tagging.