Hyper-X: A Unified Hypernetwork for Multi-Task Multilingual Transfer

Ahmet Üstün¹, Arianna Bisazza¹, Gosse Bouma¹, Gertjan van Noord¹, Sebastian Ruder²

¹University of Groningen
²Google Research
a.ustun@rug.nl

Abstract

Massively multilingual models are promising for transfer learning across tasks and languages. However, existing methods are unable to fully leverage training data when it is available in different task-language combinations. To exploit such heterogeneous supervision, we propose Hyper-X, a single hypernetwork that unifies multi-task and multilingual learning with efficient adaptation. This model generates weights for adapter modules conditioned on both tasks and language embeddings. By learning to combine task and language-specific knowledge, our model enables zero-shot transfer for unseen languages and task-language combinations. Our experiments on a diverse set of languages demonstrate that Hyper-X achieves the best or competitive gain when a mixture of multiple resources is available, while being on par with strong baselines in the standard scenario. Hyper-X is also considerably more efficient in terms of parameters and resources compared to methods that train separate adapters. Finally, Hyper-X consistently produces strong results in few-shot scenarios for new languages, showing the versatility of our approach beyond zero-shot transfer.

1 Introduction

Transfer learning across languages and tasks has long been an important focus in NLP (Ruder et al., 2019). Recent advances in massively multilingual transformers (MMTs; Devlin et al., 2019; Conneau et al., 2020) show great success in this area. A benefit of such models is their ability to transfer task-specific information in a high-resource source language to a low-resource target language (Figure 1, 1). Alternatively, such models can leverage knowledge from multiple tasks for potentially stronger generalization (Figure 1, 2).

Over time, many research communities have been developing resources for specific languages of focus (Strassel and Tracey, 2016; Nivre et al., 2018; Wilie et al., 2020). In practice, it is thus common for data to be available for different tasks in a mixture of different languages. For instance, in addition to English data for both POS tagging and Named Entity Recognition (NER), a treebank with POS annotation may be available for Turkish, while NER data may be available for Arabic. This example is illustrated in Figure 1, 3.

In contrast to existing cross-lingual transfer paradigms such as single-task zero-shot transfer (Hu et al., 2020) or few-shot learning (Lauscher et al., 2020a), multi-task learning on such a mixture of datasets (mixed-language multi-task) poses an opportunity to leverage all available data and
to transfer information across both tasks and languages to unseen task–language combinations (Ponti et al., 2021).

Standard fine-tuning strategies, however, are limited in their ability to leverage such heterogeneous task and language data. Specifically, MMTs are prone to suffer from catastrophic forgetting and interference (Wang et al., 2020) when they are fine-tuned on multiple sources. Adapters (Houlsby et al., 2019), a parameter-efficient fine-tuning alternative are commonly used for transfer either across tasks (Mahabadi et al., 2021b) or languages (Üstün et al., 2020) but require training a new adapter for each new language (Pfeiffer et al., 2020b).

In this paper, we propose a unified hypernetwork, Hyper-X that is particularly suited to this setting by leveraging multiple sources of information including different languages and tasks within a single model. The core idea consists of taking language and task embeddings as input, and generating adapter parameters via a hypernetwork for the corresponding task-language combination. By parameterizing each task and language separately, Hyper-X enables adaptation to unseen combinations at test time while exploiting all available data resources.

Additionally, Hyper-X can make seamless use of masked language modelling (MLM) on unlabelled data, which enables it to perform zero-shot adaptation to languages not covered by the MMT during pre-training. MLM also enables Hyper-X to learn a language representation even without available task-specific data.

In sum, our work brings together a number of successful transfer ‘ingredients’ that have been explored in very recent literature (see Table 1), namely multi-task learning, multilingual learning, further pre-training, along a high degree of compute- and time-efficiency.

We evaluate Hyper-X for cross-lingual transfer on two sequence labelling tasks, namely part-of-speech (POS) tagging and named-entity recognition (NER) in 16 languages—7 of which are not covered in pre-training—across the three experimental setups depicted in Figure 1. Our experiments demonstrate that Hyper-X is on par with strong baselines for cross-lingual transfer from English. In the multi-task and mixed-language settings, Hyper-X shows a large improvement compared to the standard baselines and matches the performance of the less efficient adapter-based model due to its ability to leverage heterogeneous sources of supervision. Analysis highlights that Hyper-X is superior in terms of efficiency–performance trade-offs. Finally, we evaluate our model in a few-shot setting, where Hyper-X consistently achieves competitive performance across different languages and tasks, which suggests the usability of our approach in continuous learning scenarios.

2 Background

2.1 Adapters

Adapters (Rebuffi et al., 2018) are light-weight bottleneck layers inserted into a MMT to fine-tune the model for a new task (Houlsby et al., 2019), language (Pfeiffer et al., 2020b) or domain (Bapna and Firat, 2019). The pre-trained weights of the transformer remain fixed and only adapter parameters are updated. This setup prevents catastrophic forgetting (McCloskey and Cohen, 1989) by encapsulating specialized knowledge.

Formally, an adapter module $A_i$ at layer $i$ consists of a down-projection $D_i \in \mathbb{R}^{h \times b}$ of the input.
While \( \mathcal{H} \) can be any differentiable function, it is commonly parameterized as a simple linear transform \((W_h)\) that generates a flat vector with the dimension of \(d_h\), which corresponds to the total number of model parameters. \(W_h\) is shared across all input sources, enabling maximum sharing.

3 Hyper-X

We propose, Hyper-X, an efficient adaptation of a MMT by exploiting multiple sources of information for transfer to an unseen language or task-language pairs. Specifically, Hyper-X learns to combine task and language-specific knowledge in the form of embeddings using a hypernetwork. Conditioned on the task and language embeddings, the hypernetwork generates composite adapter layers for the corresponding task-language combination (e.g. NER in Turkish), thereby enabling transfer to arbitrary task-language pairs at test time. Figure 2 provides an overview of our model.

By jointly learning from task and language information, Hyper-X overcomes some of the limitations of prior work: Unlike adapter-based approaches (Pfeiffer et al., 2020b; Üstün et al., 2020) that transfer cross-lingual information only to the task of the task adapter, our model is capable of leveraging supervision—and positive transfer—from both multiple tasks and languages. Moreover, unlike Ponti et al. (2021) who require annotated data in one of the target tasks for each language, Hyper-X is able to perform zero-shot transfer even when there is no annotated data from any of the target tasks, by using MLM as an auxiliary task for each language.

3.1 A Hypernetwork for Task-Language Adapters

We use a standard hypernetwork as the parameter generator function. However, instead of generating the full model parameters, our hypernetwork generates the parameters for each adapter layer. Concretely, the hypernetwork \( \mathcal{H} \) generates adapter parameters where each adapter layer \( A_i \) consists of down and up-projection matrices \((D_i, U_i)\):

\[
D_i, U_i \equiv \mathcal{H}(s^{(h)})
\]

Decoupling Tasks and Languages In Hyper-X, we condition the parameter generation on the input task and language. Therefore, given a combination of task \( t \in \{t_1, ..., t_m\} \) and language \( l \in \{l_1, ..., l_n\} \), the source embedding contains

Figure 2: Overview of Hyper-X. The hypernetwork (1) takes the concatenation of task, language and layer embeddings as input and generates a flat parameter vector. Before the final transformation, the source projector network projects the combination of these embeddings to a smaller dimension. The parameter vector is then reshaped and cast to weights of the adapter (2), which are inserted into a transformer layer (3).
knowledge from both sources: \( s^{(h)} \approx (t, l) \). We parameterize each task and language via separate embeddings, which enables adaptation to any task-language combination. Task and language embeddings (\( s^{(t)}, s^{(l)} \)) are low-dimensional vectors that are learned together with the parameters of the hypernetwork. During training, for each mini-batch we update these embeddings according to the task and language that the mini-batch is sampled from.

**MLM as Auxiliary Task** Hyper-X learns separate tasks and languages embeddings—as long as the task and language have been seen during training. As annotated data in many under-represented languages is limited, we employ MLM as an auxiliary task during training to enable computing embeddings for every language. Moreover, MLM enables a better zero-shot performance for languages that are not included in MMT pre-training (see § 6.2 for a detailed analysis of the impact of MLM).

**Sharing Across Layers** In addition to the task and language embeddings, we learn a layer embedding \( s^{(i)} \) (Mahabadi et al., 2021b; Ansell et al., 2021) corresponding to the transformer layer index \( i \) where the respective adapter module is plugged in. Since Hyper-X generates an adapter for each Transformer layer, learning independent layer embeddings allows for information sharing across those layers. Moreover, as layer embeddings allow the use of a single hypernetwork for all Transformer layers, they reduce the trainable parameters, i.e., size of the hypernetwork, by a factor corresponding to the number of layers of the main model.

**Combining Multiple Sources** To combine language, task and layer embeddings, we use a simple source projector network \( P_s \) as part of our hypernetwork. This module consisting of two feed-forward layers with a ReLU activation takes the concatenation of the three embeddings and learns a combined embedding \( s^{(p)} \in \mathbb{R}^{d_p} \) with a potentially smaller dimension:

\[
\begin{align*}
    s^{(h)} &= s^{(t)} \oplus s^{(l)} \oplus s^{(i)} \quad (4) \\
    s^{(p)} &= P_s(s^{(h)}) \quad (5)
\end{align*}
\]

where \( s^{(h)} \in \mathbb{R}^{d_h} \) refers to the concatenated embedding before the \( P_s \), with \( d_s = d_t + d_l + d_i \). This component enables learning how to combine source embeddings while also reducing the total number of trainable parameters.

### 4 Experiments

**Dataset and Languages** We conduct experiments on two downstream tasks: part-of-speech (POS) tagging and named entity recognition (NER). For POS tagging, we use the Universal Dependencies (UD) 2.7 dataset (Zeman et al., 2020) and for NER, we use WikiANN (Pan et al., 2017) with the train, dev and test splits from Rahimi et al. (2019). In addition to these two tasks, we also use masked language modelling (MLM) on Wikipedia articles as an auxiliary task. We limit the number of sentences from Wikipedia to 100K for each language, in order to control the impact of dataset size and to reduce the training time.

For the language selection, we consider: (i) typological diversity based on language family, script and morphosyntactic attributes; (ii) a combination of high-resource and low-resource languages based on available data in downstream task; (iii) presence in the pre-training data of mBERT; and (iv) presence of a language in the two task-specific datasets.\(^2\) We provide the details of the language and dataset selection in Appendix A.

**Experimental Setup** We evaluate Hyper-X for zero-shot transfer in three different settings: (1) **English single-task**, where we train the models only on English data for each downstream task separately. (2) **English multi-task**, where the models are trained on English POS and NER data at the same time. (3) **Mixed-language multi-task**, where we train the models in a multi-task setup, but instead of using only English data for both POS and NER, we use a mixture of task-language combinations. In order to measure zero-shot performance in this setup, following Ponti et al. (2021) we create two different partitions from all possible language-task combinations in such a way that a task-language pair is always unseen for one of the partitions (e.g. NER-Turkish and POS-Arabic in Figure 1). Details of partitions and our partitioning strategy are given in Appendix A.

#### 4.1 Baselines and Model Variants

**mBERT** (Devlin et al., 2019) is a MMT that is pre-trained for 104 languages. We use mBERT by fine-tuning all the model parameters on the \(^2\) (i) and (ii) are necessary for a realistic setting and to evaluate full-scale cross-lingual capabilities; (iii) allows us to measure if models are able to extend the limits of the MMT; (iv) enables us to assess supervision from a mixture of task and language combinations.
available sources. As this standard approach enables cross-lingual transfer from both a single source or a set of language-task combinations, we compare it to Hyper-X in all three settings. Moreover, we use mBERT as the base model for both Hyper-X and the other baselines.

MAD-X (Pfeiffer et al., 2020b) is an adapter-based modular framework for cross-lingual transfer learning based on MMTs. It combines a task-specific adapter with language-specific adapters that are independently trained for each language using MLM. We train MAD-X language adapters on the same Wikipedia data that is used for Hyper-X, for all languages with a default architecture.3 Finally, for the mixed-language setup, as the original MAD-X does not allow standard multi-task training, we train the task adapters by using multiple source languages but for NER and POS separately. We call this model MAD-X MS.

Parameter Space Factorization (Ponti et al., 2021) is a Bayesian framework that learns a parameter generator from multiple tasks and languages for the softmax layer on top of a MMT. However, if a language lacks annotated training data, this model cannot learn the required latent variable for the corresponding language. Therefore, we evaluate this baseline only for the mixed-language multi-task setting using the same partitions as Hyper-X. We use the original implementation with default hyper-parameters and low-rank factorization.

Model Variants We evaluated two variants of Hyper-X in order to see the impact of Hypernetwork size: Hyper-X Base model fine-tunes 76m parameters ($d_s = 192$), compatible with MAD-X in terms of total number of trainable parameters, and Hyper-X Small updates only 13m parameters ($d_s = 32$). Table 3 shows the parameter counts together with the corresponding runtime.

4.2 Training Details

For all the experiments, we used a batch size of 32 and a maximum sequence length of 256. We trained Hyper-X for 100,000 updates steps by using a linearly decreasing learning rate of $1e^{-4}$ with 4000 warm-up steps. We evaluated checkpoints every 5000 steps, and used the best checkpoint w.r.t. the average validation score for testing. As for baselines, we trained mBERT and MAD-X tasks adapters for 20 epochs by using learning rate of $1e^{-5}$ and $1e^{-4}$ respectively with the same scheduler and warm-up steps. Since MAD-X requires prerequisite language adapters, we trained language adapters for 100,000 steps for each language separately.

In terms of model size, we use a bottleneck dimension of 256 to learn adapters for Hyper-X. Similarly, we train language and adapters with dimension of 256 and 48 for MAD-X to create a comparable baseline. In Hyper-X, as input to the hypernetwork, dimensions for task, language and layer embeddings are all set to 64 (total 192). During training, we create homogeneous mini-batches for each task-language combination to learn the corresponding embeddings together with the hypernetwork. Moreover, following Mahabadi et al. (2021b), we also update the original layer-norm parameters. During multi-task training, we use temperature-based sampling with $T = 5$ to balance each task-language pair during training (See Appendix § B.1 for details).

5 Zero-shot Transfer Results

Table 2 shows the aggregate zero-shot results in NER and POS tagging respectively. In addition to the average scores across all 15 zero-shot languages, we show the average of the 8 ‘seen’ and 7 ‘unseen’ languages separately with respect to language coverage of mBERT. We present results for English single-task, English multi-task and Mixed-language multi-task settings.

Overall, Hyper-X Base performs on par with the strongest baseline when transferring from English. In the presence of additional sources, such as a mixture of task-language pairs, Hyper-X outperforms both mBERT and parameter space factorization (PSF). In comparison to MAD-X, Hyper-X generally performs better on seen languages. We relate this to the unified hypernetwork enabling maximum sharing between languages and higher utilization of the pre-trained capacity in contrast to the isolated adapters. On unseen languages, Hyper-X is outperformed by MAD-X in most cases. However, we emphasize that MAD-X requires training separate language adapters for each new language,
Table 2: Zero-shot cross-lingual transfer results averaged over 3 runs on Named-Entity Recognition (NER; F1) and Part-of-Speech Tagging (POS; Accuracy) for mBERT, MAD-X (Pfeiffer et al., 2020b), parameter space factorization (PSF; Ponti et al., 2021) and Hyper-X. We highlight the best results per-setting in bold. We also report the total number of parameters and fine-tuning time for all models. Note that Hyper-X corresponds to a single model trained for each partition while MAD-X consists of \( N \) independently trained adapters for each task and language. MAD-X MS refers to an adapted version of the original model trained on multiple source languages but each task separately.

### English Single-Task
When English is used as the only source language for each task separately, Hyper-X (Base) performs on par with MAD-X for NER (52.7 vs 52.8 F1) but falls behind for POS tagging (63.5 vs 65.4 Acc.) on average. Both models significantly outperform mBERT. Looking at the individual language results, Hyper-X performs slightly better on ‘seen’ languages compared to MAD-X in NER and POS tagging respectively. For ‘unseen’ languages, both MAD-X and Hyper-X benefit from MLM, which results in large improvements with respect to mBERT. Between the two models, MAD-X achieves a higher average score in both NER and POS tagging.

### English Multi-Task
In a multi-task setting where only English data is available, fine-tuning mBERT for both target tasks at the same time gives mixed results compared to single-task training—in line with previous findings noting catastrophic forgetting and interference in MMTs (Wang et al., 2020). Hyper-X Base, on the other hand, shows a small but consistent improvement on the majority of languages, with 0.2 (F1) and 0.1 (Acc.) average increase in NER and POS tagging respectively. This confirms that Hyper-X is able to mitigate interference while allowing for sharing between tasks when enough capacity is provided.\(^4\)

### Mixed-Language Multi-Task
In this setting, a mixture of language data is provided for NER and POS via two separate training partitions while keeping each task-language pair unseen in one of these partitions. All the models including mBERT achieve better zero-shot scores compared to the previous settings. Among the baselines, parameter space factorization (PSF) gives a larger improvement compared to mBERT on both tasks, indicating the importance of task- and language-specific parametrization for adapting a MMT. Hyper-X Base produces the largest performance gain among the models that trains only a single model: it achieves 9.0 (F1) and 4.3 (Acc.) average increase for NER and POS. Although both PSF and Hyper-X enable adaptation conditioned on a mixture of task and language combinations, we relate the difference between PSF and Hyper-X to the contrast in parameter generation. PSF only generates parameters of the softmax layer and is thus unable to adapt deeper layers of the model. Hyper-X, on the other hand, generates adapter layer parameters inserted throughout the model, which provide a higher degree of adaptation flexibility. Hyper-X outperforms PSF particularly on unseen languages as it benefits from MLM as an auxiliary task.

---

\(^4\)MAD-X learns independent adapters for each target task, which does not allow for positive cross-task transfer.
Table 3: Compute efficiency with respect to number of fine-tuned parameters and training time for mBERT, PSF, MAD-X and Hyper-X. Training time includes both NER and POS-tagging. For MAD-X, the total number of parameters and training time is calculated for 16 (l) languages and 2 (t) tasks.

<table>
<thead>
<tr>
<th>Model</th>
<th>#Params.</th>
<th>Training Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>mBERT</td>
<td>177m</td>
<td>2h</td>
</tr>
<tr>
<td>PSF</td>
<td>185m</td>
<td>4h</td>
</tr>
<tr>
<td>MAD-X</td>
<td>76m</td>
<td>116h</td>
</tr>
<tr>
<td>Language Adapters</td>
<td>4.7m x l</td>
<td>7h x l</td>
</tr>
<tr>
<td>Task Adapters</td>
<td>0.9m x t</td>
<td>2h x t</td>
</tr>
<tr>
<td>Hyper-X Small</td>
<td>13m</td>
<td>16h</td>
</tr>
<tr>
<td>Hyper-X Base</td>
<td>76m</td>
<td>18h</td>
</tr>
</tbody>
</table>

Finally, Hyper-X tends to perform slightly better on seen languages compared to the adapted multi-source version of MAD-X. However, MAD-X outperforms Hyper-X on unseen languages by 1.2 (F1) and 2.8 (Acc.) for NER and POS respectively. Besides the expected benefits of independently trained language adapters in MAD-X, we relate this to the limited cross-task supervision for unseen languages in Hyper-X for this setting. Especially, when the target task is POS, most of the unseen languages have only 100 sentences available in NER dataset, which leaves only a little margin for improvements.

6 Analysis

6.1 Parameter and Time Efficiency

Table 3 shows the fine-tuned parameter counts and the training time required for the baselines and Hyper-X models. Unlike mBERT, PSF and Hyper-X, MAD-X consists of 16 and 2 independently trained language and task adapters respectively. In terms of parameter efficiency, MAD-X and Hyper-X Base models correspond to 43% of mBERT’s parameters. However, in terms of training time, Hyper-X Base is trained only once for about 18 hours, as opposed to MAD-X’s considerably high total training time (116 hours in total). Thus, considering the competitive zero-shot performances across different languages and settings, Hyper-X Base provides a better efficiency-performance trade-off. Furthermore, in the case of adding more languages, MAD-X’s parameter count and training time increase linearly with the number of new languages, while Hyper-X’s computational cost remains the same.

As Hyper-X model variants, we evaluated two different sizes of the source embedding (d_s; 32→192). Although Hyper-X Small is much more parameter-efficient (7.2% of mBERT’s parameters) and takes slightly less time to train (16h), its zero-shot performance is significantly lower than the base model, especially for unseen languages. Nevertheless, Hyper-X Small remains a valid alternative for particularly ‘seen’ languages.

6.2 Impact of Auxiliary MLM Training

Figure 3 demonstrates the impact of auxiliary MLM training in Hyper-X Base for the mixed-language multi-task setting. As this setting provides training instances for each task and language, we evaluated the impact of MLM by removing the corresponding Wikipedia data first for ‘seen’ languages, then for ‘all’ languages. As shown in the figure, although the availability of MLM data slightly increases seen language performance, it mainly boosts the scores in unseen languages: +6.2 F1 and +10.5 Acc. for NER and POS respectively. Furthermore, when MLM data is removed for only seen languages, Hyper-X can mostly recover performance on seen languages, confirming the dominant effect of MLM on unseen languages.

6.3 Impact of Source Languages

In the mixed-language multi-task setting, we deliberately avoid grouping languages from same families to different partitions, in order to restrict the transfer from the same-language family instances, and to observe the effect of cross-task supervision. However, we also evaluate the impact of source languages in this setup, to measure the degree of potential positive transfer. To this end, we switched the partitions of kk, mt, yue, so that all of them will likely benefit from a high-resource language
from the same family for the same target task. Figure 4 and 5 shows the aggregated results in both Hyper-X Base and mBERT. Firstly, both models benefit from positive transfer. Secondly, although the relative increase in mBERT is slightly higher, Hyper-X still outperforms mBERT with a large margin, showing the robustness of our model with regard to different partitions.

6.4 Few-shot Transfer

Fine-tuning an MMT with a few target instances has been shown to increase zero-shot performances (Lauscher et al., 2020b). Therefore, we evaluate Hyper-X for few-shot transfer on 5 languages—3 of which are high-resource and covered by mBERT and 2 are low-resource and unseen. To this end, we further fine-tune Hyper-X and the corresponding baselines that are trained initially in the English multi-task by using 5, 10, 20, and 50 training instances for each language separately on NER and POS-tagging (see details in Appendix §D).

Figure 6 presents the average results comparing mBERT to MAD-X. Similar to the zero-shot results, on seen languages, Hyper-X constantly provides better adaptation than both baselines for NER and POS. On unseen languages, MAD-X gives the best result on average. This is because MAD-X starts with better initial representations for Maltese and Uyghur. When more samples are provided, Hyper-X reduces the initial gap. Overall, Hyper-X consistently achieves the best or competitive performance on the majority of the experiments, except ‘unseen’ languages for POS tagging, showing the effectiveness of our approach beyond the standard zero-shot transfer. Taken together with the parameter and training efficiency, these results show that Hyper-X can be easily extended to new languages without incurring large computing costs.

7 Related Work

Adapters As a parameter-efficient alternative to standard fine-tuning, adapters have been used for quick training (Rücklé et al., 2021), multi-task learning (Stickland and Murray, 2019) and knowledge composition (Pfeiffer et al., 2021a; Wang et al., 2021; Poth et al., 2021). Moreover, Mahabadi et al. (2021a) and He et al. (2022a) extended adapters for better performance with fewer parameters. In the context of multilingual transfer, adapters enable allocation of additional language-specific capacity, thereby mitigating the ‘curse of multilinguality’ (Üstün et al., 2020). Such language adapters (Pfeiffer et al., 2020b; Ansell et al., 2021) achieve high zero-shot results when combined with task adapters and enable generalization to languages unseen during pre-training via MLM-based adaptation (Pfeiffer et al., 2021b). Philip et al. (2020) and Üstün et al. (2021b) also used monolingual adapters for zero-shot and unsupervised NMT.

Hypernetworks in NLP Tay et al. (2021) propose a multi-task model that uses a hypernetwork to condition on input to learn task-specific reparametrizations. Similarly, Mahabadi et al. (2021b) generate task-specific adapters via a hypernetwork. Recently, He et al. (2022b) use a hypernetwork to generate prompts. For multilingual learning, where the input sources correspond to language embeddings, Üstün et al. (2020) and Ansell et al. (2021) learn these embeddings from the typological feature vectors of languages, enabling generalization to unseen languages based on a hypernetwork. In a similar spirit to our work, parameter space factorization (PSF; Ponti et al., 2021), learns task and language-specific embeddings from seen task-language combinations. However, unlike our model, these embeddings are used...
for task/language-specific parametrization in the softmax layer.

8 Conclusion

We have proposed Hyper-X, a novel approach for multi-task multilingual transfer learning, based on a unified hypernetwork that leverages heterogeneous sources of information, such as multiple tasks and languages. By learning to generate composite adapters for each task-language combinations that modify the parameters of a pre-trained multilingual transformer, Hyper-X allows for maximum information sharing and enables zero-shot prediction for arbitrary task-language pairs at test time. Through a number of experiments, we demonstrate that Hyper-X is competitive with the state-of-the-art when transferring from a source language. When a mixture of tasks and languages is available, Hyper-X outperforms several strong baselines on many languages, while being more parameter and time efficient. Finally, we show that for few-shot transfer, Hyper-X is a strong option with a less computing cost than baselines for the initial task adaptation.

9 Limitations

Firstly, although our experiments show the potential of Hyper-X to benefit from multiple tasks for zero-shot transfer, so far we evaluated our model on a limited set of tasks: NER and POS-tagging, which may limit the generalizability of our model to other tasks.

Secondly, for the few-shot transfer, we limit our experiments to languages that we learn via MLM and to existing tasks. Our work does not include languages without MLM data as well as completely new tasks. Learning the task and language embeddings separately, however, creates a possibility to interpolate existing embeddings for new languages or new tasks, which especially may work for the few-shot learning. We leave exploration of these two limitations to future work.

Acknowledgements

We would like to thank Noah Constant, Asa Cooper Stickland and the anonymous reviewers for their helpful feedback on a previous version of this paper. We also would like to thank the Center for Information Technology of the University of Groningen for providing access to the Peregrine HPC cluster.

References


the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), pages 4171–4186.


A Language Selection

Table 4 shows that the details for languages such as language code, UD treebank id and language family. For POS tagging, we use the Universal Dependencies (UD) 2.7 dataset (Zeman et al., 2020) and for NER, we use Wikiann (Pan et al., 2017) with the train, dev and test splits from Rahimi et al. (2019). To partition languages for the mixed-language multi-task setting, we group languages from the same families into the same partitions to avoid a strong supervision from the same language family when evaluating zero-shot predictions for unseen task-language combinations. When there is no available training data in the target treebank, we use the test split for the mixed-language multi-task setting.

B Experimental Details

B.1 Impact of Sampling

Hyper-X is a single model that is trained at once for multiple languages and task simultaneously. However, as the amount of total MLM training data is considerably larger than NER and POS-tagging data, we experimented with two different sampling methods: size proportional sampling and temperature-based sampling \((t = 5)\). For the temperature-based sampling, we independently sample a batch for each task-language combination. Figure 7 shows the impact of different sampling methods on the zero-shot performance for ‘seen’, ‘unseen’ language groups together with average over all languages. As seen, temperature-based sampling, greatly increase performance for all language groups on both NER and POS-tagging. This suggest that when MLM data does not restricted by sampling, it highly influences the learning objective which results a catastrophic forgetting on the target tasks.

B.2 Implementation and Computing Infrastructure

All the experiments are conducted using Tesla V100 GPUs. We did not use parallel training on multiple GPUs, so each experiment was conducted on a single GPU. Parameters that are fine-tuned for each model and total runtime are reported in the section (§ 6.1). We implemented Hyper-X by using Transformers library (Wolf et al., 2020) and the code will be released upon publication. We used adapterhub (Pfeiffer et al., 2020a) for MAD-X, and the original repository for parameter space factorization (Ponti et al., 2021). Hyper-parameters that are used in experiments are given in the section 4. We did not conduct a hyper-parameter search due to the computational limitations, and used the reference values in most cases: only the dimension for language adapters in MAD-X is changed to match with the same parameter count of Hyper-X. Finally for mBERT, we did a preliminary experiments with learning rate of 1e-4 and 1e-5, and pick the latter one as it produced better performance.

C Detailed Results

The results that are averaged over 3 runs for each language are given in Table 6

D Few Shot Experiments

For the few-shot transfer experiments, we fine-tune each model for 50 epochs with the same hyper-parameters. We disable the learning rate decay as only a few training instances are provided to the models. Note that, in these experiments, we always start with the models that are already trained in the zero-shot setting and perform fine-tuning for each language and task separately. For the selection of training samples, we randomly sample instances regardless of the labels, as the initial models are already trained for these tasks on English data.

Table 5 show that few-shot results for NER and POS-tagging respectively.
<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
<th>UD Treebank</th>
<th>Family</th>
<th>Train</th>
<th>NER</th>
<th>Test</th>
<th>POS</th>
<th>Train</th>
<th>POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>English</td>
<td>en</td>
<td>EWT</td>
<td>IE, Germanic</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>12543</td>
<td>2002</td>
<td>2077</td>
</tr>
<tr>
<td>Arabic</td>
<td>ar</td>
<td>PADT</td>
<td>Afro-Asiatic, Semitic</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>6075</td>
<td>990</td>
<td>888</td>
</tr>
<tr>
<td>Breton</td>
<td>br</td>
<td>KEB</td>
<td>IE, Celtic</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>3997</td>
<td>500</td>
</tr>
<tr>
<td>Icelandic</td>
<td>is</td>
<td>PUD</td>
<td>IE, Germanic</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>-</td>
<td>-</td>
<td>1000</td>
</tr>
<tr>
<td>Kazakh</td>
<td>kk</td>
<td>KTB</td>
<td>Turkic, Northwestern</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>32</td>
<td>-</td>
<td>1047</td>
</tr>
<tr>
<td>Tamil</td>
<td>ta</td>
<td>TTB</td>
<td>Dravidian, Southern</td>
<td>15000</td>
<td>10000</td>
<td>10000</td>
<td>400</td>
<td>80</td>
<td>120</td>
</tr>
<tr>
<td>Turkish</td>
<td>tr</td>
<td>IMST</td>
<td>Turkic, Southwestern</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>3664</td>
<td>988</td>
<td>983</td>
</tr>
<tr>
<td>Yoruba</td>
<td>yo</td>
<td>YTB</td>
<td>Niger-Congo, Defoid</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>318</td>
</tr>
<tr>
<td>Faroese</td>
<td>fo</td>
<td>OFT</td>
<td>IE, Germanic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>1208</td>
</tr>
<tr>
<td>Guarani</td>
<td>gn</td>
<td>Thomas</td>
<td>Tupian, Tupi-Guarani</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>98</td>
</tr>
<tr>
<td>Upper Sorbian</td>
<td>yo</td>
<td>UFAL</td>
<td>IE, Slavic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>23</td>
</tr>
<tr>
<td>Maltese</td>
<td>mt</td>
<td>MUDT</td>
<td>Afro-Asiatic, Semitic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>1123</td>
<td>433</td>
<td>518</td>
</tr>
<tr>
<td>Sanskrit</td>
<td>sa</td>
<td>UFAL</td>
<td>Indic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>230</td>
</tr>
<tr>
<td>Uyghur</td>
<td>ug</td>
<td>UDT</td>
<td>Turkic, Southeastern</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>1656</td>
<td>900</td>
<td>900</td>
</tr>
<tr>
<td>Cantonese</td>
<td>yue</td>
<td>HK</td>
<td>Sino-Tibetan</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>-</td>
<td>-</td>
<td>1004</td>
</tr>
</tbody>
</table>

Table 4: Languages that are used in the experiments, together with corresponding language code, UD treebank and language families. We used WikiANN (Pan et al., 2017; Rahimi et al., 2019) and UD version 2.7 (Zeman et al., 2020) for NER and POS-tagging respectively.

<table>
<thead>
<tr>
<th>Language</th>
<th>Code</th>
<th>UD Treebank</th>
<th>Family</th>
<th>Train</th>
<th>NER</th>
<th>Test</th>
<th>POS</th>
<th>Train</th>
<th>POS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arabic</td>
<td>ar</td>
<td>PADT</td>
<td>Afro-Asiatic, Semitic</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>6075</td>
<td>990</td>
<td>888</td>
</tr>
<tr>
<td>Breton</td>
<td>br</td>
<td>KEB</td>
<td>IE, Celtic</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>3997</td>
<td>500</td>
</tr>
<tr>
<td>Icelandic</td>
<td>is</td>
<td>PUD</td>
<td>IE, Germanic</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>-</td>
<td>-</td>
<td>1000</td>
</tr>
<tr>
<td>Kazakh</td>
<td>kk</td>
<td>KTB</td>
<td>Turkic, Northwestern</td>
<td>1000</td>
<td>1000</td>
<td>1000</td>
<td>32</td>
<td>-</td>
<td>1047</td>
</tr>
<tr>
<td>Tamil</td>
<td>ta</td>
<td>TTB</td>
<td>Dravidian, Southern</td>
<td>15000</td>
<td>10000</td>
<td>10000</td>
<td>400</td>
<td>80</td>
<td>120</td>
</tr>
<tr>
<td>Turkish</td>
<td>tr</td>
<td>IMST</td>
<td>Turkic, Southwestern</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>3664</td>
<td>988</td>
<td>983</td>
</tr>
<tr>
<td>Yoruba</td>
<td>yo</td>
<td>YTB</td>
<td>Niger-Congo, Defoid</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>318</td>
</tr>
<tr>
<td>Faroese</td>
<td>fo</td>
<td>OFT</td>
<td>IE, Germanic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>1208</td>
</tr>
<tr>
<td>Guarani</td>
<td>gn</td>
<td>Thomas</td>
<td>Tupian, Tupi-Guarani</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>98</td>
</tr>
<tr>
<td>Upper Sorbian</td>
<td>yo</td>
<td>UFAL</td>
<td>IE, Slavic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>23</td>
</tr>
<tr>
<td>Maltese</td>
<td>mt</td>
<td>MUDT</td>
<td>Afro-Asiatic, Semitic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>1123</td>
<td>433</td>
<td>518</td>
</tr>
<tr>
<td>Sanskrit</td>
<td>sa</td>
<td>UFAL</td>
<td>Indic</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>-</td>
<td>-</td>
<td>230</td>
</tr>
<tr>
<td>Uyghur</td>
<td>ug</td>
<td>UDT</td>
<td>Turkic, Southeastern</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>1656</td>
<td>900</td>
<td>900</td>
</tr>
<tr>
<td>Cantonese</td>
<td>yue</td>
<td>HK</td>
<td>Sino-Tibetan</td>
<td>20000</td>
<td>10000</td>
<td>10000</td>
<td>-</td>
<td>-</td>
<td>1004</td>
</tr>
</tbody>
</table>

Table 5: Per language results for few-shot experiments, where models are further fine-tuned with a few training instances (0, 5, 10, 20, 50) from NER and POS datasets. For the language selection, ar, tr, zh are covered by mBERT and mt, ug are unseen.
Table 6: Zero-shot cross-lingual transfer results averaged over 3 runs for Named-Entity Recognition (NER; F1) and Part-of-Speech Tagging (POS; Accuracy) for mBERT (mB), MAD-X (MX) and parameter space factorization (PSF) models, together with Hyper-X Small (HX.32) and Base (HX.192). Superscripts denote the partitioning that is used for mixed-language multi-task setting.