Cross-Lingual Transfer with Language-Specific Subnetworks for Low-Resource Dependency Parsing

Rochelle Choenni
University of Amsterdam
The Institute for Logic, Language and Computation (ILLC)
r.m.v.k.choenni@uva.nl

Dan Garrette
Google Research
dhgarrette@google.com

Ekaterina Shutova
University of Amsterdam
The Institute for Logic, Language and Computation (ILLC)
e.shutova@uva.nl

Large multilingual language models typically share their parameters across all languages, which enables cross-lingual task transfer, but learning can also be hindered when training updates from different languages are in conflict. In this article, we propose novel methods for using language-specific subnetworks, which control cross-lingual parameter sharing, to reduce conflicts and increase positive transfer during fine-tuning. We introduce dynamic subnetworks, which are jointly updated with the model, and we combine our methods with meta-learning, an established, but complementary, technique for improving cross-lingual transfer. Finally, we provide extensive analyses of how each of our methods affects the models.

1. Introduction

Large multilingual language models, such as mBERT (Devlin et al. 2019), are pretrained on data covering many languages, but share their parameters across all languages. This modeling approach has several powerful advantages, such as allowing similar languages to exert positive influence on each other, and enabling cross-lingual task transfer (i.e., fine-tuning on some source language(s), then using the model on different target languages) (Pires, Schlinger, and Garrette 2019). These advantages are

Action Editor: Kevin Duh. Submission received: 2 December 2022; revised version received: 13 April 2023; accepted for publication: 16 April 2023.

https://doi.org/10.1162/coli_a_00482

particularly enticing in low-resource scenarios since without sufficient training data in the target language, the model's effectiveness hinges on its ability to derive benefit from other languages' data. In practice, however, even state-of-the-art multilingual models tend to perform poorly on low-resource languages (Lauscher et al. 2020; Üstün et al. 2020), due in part to **negative interference** effects—parameter updates that help the model on one language, but harm its ability to handle another—which undercut the benefits of multilingual modeling (Arivazhagan et al. 2019; Wang, Lipton, and Tsvetkov 2020; Ansell et al. 2021).

In this article, we propose novel methods for using language-specific subnetworks, which control cross-lingual parameter sharing, to reduce conflicts and increase positive transfer during fine-tuning, with the goal of improving the performance of multilingual language models on low-resource languages. While recent work applies various subnetwork based approaches to their models statically (Lu et al. 2022; Yang et al. 2022; Nooralahzadeh and Sennrich 2022), we propose a new method that allows the model to dynamically update the subnetworks during fine-tuning. This allows for sharing between language pairs to a different extent at the different learning stages of the models. We accomplish this by using pruning techniques (Frankle and Carbin 2018) to select an optimal subset of parameters from the full model for further language-specific fine-tuning. Inspired by studies that show that attention-heads in BERT-based models have specialized functions (Voita et al. 2019; Htut et al. 2019), we focus on learning subnetworks at the attention-head level. We learn separate—but potentially overlapping—head masks for each language by fine-tuning the model on the language, and then pruning out the least important heads.

Given our focus on low-resource languages, we also combine our methods with meta-learning, a data-efficient technique to learn tasks from a few samples (Finn, Abbeel, and Levine 2017). Motivated by Wang, Lipton, and Tsvetkov (2020), who find that meta-learning can reduce negative interference in the multilingual set-up, we test how much our subnetwork methods can further benefit performance in this learning framework, as well as compare the subnetwork based approach to a meta-learning baseline. Our results show that a combination of meta-learning and dynamic subnetworks is particularly powerful. To the best of our knowledge, we are the first to adapt subnetwork sharing to the meta-learning framework.

We extensively test the effectiveness of our methods on the task of dependency parsing. We use data from Universal Dependencies (UD) (Nivre et al. 2016) comprising 82 datasets covering 70 distinct languages, from 43 language families; 58 of the languages can be considered truly low-resource. Our experiments show, quantitatively, that our language-specific subnetworks, when used during fine-tuning, act as an effective sharing mechanism: permitting positive influence from similar languages, while shielding each language's parameters from negative interference that would otherwise have been introduced by more distant languages. Moreover, we show substantial improvements in cross-lingual transfer to new languages at test time. Importantly, we are able to achieve this while relying on data from just 8 treebanks before few-shot fine-tuning at test time.

Finally, we perform extensive analyses of our models to better understand how different choices affect generalization properties. We analyze model behavior with respect to several factors: typological relatedness of fine-tuning and test languages, data-scarcity during pretraining, robustness to domain transfer, and their ability to predict rare and unseen labels. We find interesting differences in model behavior that can provide useful guidance on which method to choose based on the properties of the target language.

2. Background and Related Work

2.1 Pruning and Sparse Networks

Frankle and Carbin (2018) were the first to show that neural network pruning (Han et al. 2015; Li et al. 2016) can be used to find a subnetwork that matches the test accuracy of the full network. Later studies confirmed that such subnetworks also exist within (multilingual) BERT (Prasanna, Rogers, and Rumshisky 2020; Budhraja et al. 2021; Li et al. 2022), and that they can even be transferred across different NLP tasks (Chen et al. 2020). While these studies are typically motivated by a desire to find a smaller, faster version of the model (Jiao et al. 2020; Lan et al. 2019; Sanh et al. 2019; Held and Yang 2022; Zhang et al. 2021), we use pruning to find multiple simultaneous subnetworks (one for each fine-tuning language) within the overall multilingual model, which we use during both fine-tuning and inference to guide cross-lingual sharing.

2.2 Selective Parameter Sharing

Naseem, Barzilay, and Globerson (2012) used categorizations from linguistic typology to explicitly share subsets of parameters across separate languages' dependency parsing models. Large multilingual models have, however, been shown to induce implicit typological properties automatically, and different design decisions (e.g., training strategy) can influence the language relationships they encode (Chi, Hewitt, and Manning 2020; Choenni and Shutova 2022). Rather than attempting to force the model to follow an externally defined typology, we instead take a data-driven approach, using pruning methods to automatically identify the subnetwork of parameters most relevant to each language, and letting subnetwork overlap naturally dictate parameter sharing.

A related line of research aims to control selective sharing by injecting language-specific parameters (Üstün et al. 2020; Wang, Lipton, and Tsvetkov 2020; Le et al. 2021; Ansell et al. 2021; Pfeiffer et al. 2020), which is often realized by inserting adapter modules into the network (Houlsby et al. 2019). Our approach, in contrast, uses subnetwork masking of the existing model parameters to control language interaction.

Lastly, Wang, Lipton, and Tsvetkov (2020) separate language-specific and language-universal parameters within *bilingual* models, and then meta-train the language-specific parameters only. However, given that we work in a multilingual as opposed to a bilingual setting, most parameters are shared by at least a few languages, and are thus somewhere between purely language-specific and fully universal. Our approach, instead, allows for parameters to be shared among any specific subset of languages.

Analyzing and Training Shared Subnetworks. The idea of sharing through sparse subnetworks was first proposed for multi-task learning (Sun et al. 2020), and was recently studied in the multilingual setting: Foroutan et al. (2022) show that both language-neutral and language-specific subnetworks exist in multilingual models, and Nooralahzadeh and Sennrich (2022) show that training *task-specific* subnetworks can help in cross-lingual transfer as well.

Moreover, Lin et al. (2021) train multilingual models using *language-pair-specific* subnetworks for neural machine translation (NMT), and Hendy et al. (2022) build on their work, but use *domain-specific* subnetworks instead. In both studies, subnetworks are found using magnitude pruning and kept static during training. In addition, while Lin et al. (2021) show that their method can perform well in a zero-shot setting, their strategy for merging masks for new language-pairs relies on the availability of translation data between English and both the source and target language. This makes their approach unsuitable in low-resource scenarios where such resources are not available. In addition, they show that their methods work for unseen language-pairs, but the individual languages are not unseen during training on NMT.

Furthermore, Ansell et al. (2021) learn real-valued (composable) masks instead of binary ones. Thus, instead of fully enabling or disabling parameters, they essentially apply new weights to them, making the workings of these masks more similar to that of adapter modules (Pfeiffer et al. 2020).

Finally, in concurrent work, Lu et al. (2022) show that using language-specific subnetworks at the pretraining stage can mitigate negative interference for speech recognition, and Xu et al. (2022) apply subnetworks during the backward pass only. We instead apply subnetworks during fine-tuning and few-shot fine-tuning at test time, allowing us to both make use of existing pretrained models and apply our models to truly low-resource languages. Moreover, we go beyond existing work by experimenting with *structured* subnetworks, by allowing subnetworks to dynamically change during fine-tuning, and by extensively analyzing the effects and benefits of our methods.

2.3 Meta-learning

Meta-learning is motivated by the idea that a model can "learn to learn" many tasks from only a few samples. This has been adapted to the multilingual setting by optimizing a model to be able to quickly adapt to new languages: By using meta-learning to fine-tune a multilingual model on a small set of (higher-resource) languages, the model can then be adapted to a new language using only a few examples (Nooralahzadeh et al. 2020). In this work, we use the Model-Agnostic Meta-Learning algorithm (MAML) (Finn, Abbeel, and Levine 2017), which has already proven useful for cross-lingual transfer of NLP tasks (Nooralahzadeh et al. 2020; Wu et al. 2020; Gu et al. 2020), including being applied to dependency parsing by Langedijk et al. (2022), whose approach we follow for our own experiments.

MAML iteratively selects a batch of training tasks $\mathcal{T}$, also known as **episodes**. For each task $t \in \mathcal{T}$, we sample a training dataset $\mathcal{D}_t = (\mathcal{D}_t^{trn} \cup \mathcal{D}_t^{tst})$ that consists of a **support set** used for adaptation, and a *query set* used for evaluation. MAML casts the meta-training step as a bilevel optimization problem. Within each episode, the parameters θ of a model f_θ are fine-tuned on the support set of each task t yielding f_{ϕ_t}, that is, the model adapts to a new task. The model f_{ϕ_t} is then evaluated on the query set of task t, for all of the tasks in the batch. This adaptation step is referred to as the **inner loop** of MAML. In the **outer loop**, the original model f_θ is then updated using the gradients of the query set of each $t \in \mathcal{T}$ with respect to the original model parameters θ. MAML strives to learn a good initialization of f_θ, which allows for quick adaptation to new tasks. This set-up is mimicked at test time where we again select a support set from the test task for few-shot adaptation, prior to evaluating the model on the remainder of the task data.

2.4 Dependency Parsing

In dependency parsing, a model must predict, given an input sentence, a **dependency tree**: A directed graph of binary, asymmetrical arcs between words. Each arc is labeled with a dependency relation type that holds between the two words, commonly referred to as the **head** and its **dependent**.

The UD project has brought forth a dependency formalism that allows for consistent morphosyntactic annotation across typologically diverse languages (Nivre et al. 2016). While UD parsing has received much attention in the NLP community, performance on low-resource languages remains far below that of high-resource languages (Zeman et al. 2018). State-of-the-art multilingual parsers generally exploit a pretrained multilingual language model with a deep biaffine parser (Dozat and Manning 2016) on top. The model is then fine-tuned on data (typically) from high-resource languages. This fine-tuning stage has been performed on English data only (Wu et al. 2020), or multiple languages (Tran and Bisazza 2019).

UDify (Kondratyuk and Straka 2019) takes this a step further and is fine-tuned on all available training sets together (covering 75 languages). Moreover, they use a multi-task training objective that combines parsing with predicting part-of-speech tags, morphological features, and lemmas.

On the modeling side, previous studies have attempted to exploit knowledge from the field of Linguistic Typology to further improve upon this training paradigm. For instance, UDapter (Üstün et al. 2020) is trained on 13 languages using the same set-up as UDify, but freezes mBERT's parameters and trains language-specific adapter modules. It induces typological guidance by taking language embeddings predicted from typological features as input. In a related study, Choudhary (2021) tries to induce typological knowledge into UDify by using typology prediction as an auxiliary task instead.

Other studies have taken a data-centric approach instead. van der Goot et al. (2021) propose MACHAMP, a toolkit for multi-task learning of a variety of NLP tasks, including dependency parsing. While using a similar architecture to existing literature, they show that they can further improve performance by resampling datasets according to a multinomial distribution on the batch level to prevent larger datasets from overwhelming the model. In addition, Glavaš and Vulić (2021), propose hierachical source selection, a model-agnostic method for finding the optimal subset of UD treebanks for cross-lingual transfer to a specific target language.

3. Data

We use data from Universal Dependencies v2.9[1] and test on 82 datasets covering 70 unique and highly typologically diverse languages belonging to 19 language families from 43 subfamilies. We consider 54 of these languages to be extremely low-resource as there are fewer than 31 training samples available. For the other 28 languages, 50% have approximately 150–2K training samples and the other 50% have 2K–15K samples available. In total, our test data contains 233 possible arc labels. We use 8 high-resource languages for fine-tuning, based on the selection used by Langedijk et al. (2022) and Tran and Bisazza (2019): English, Arabic, Czech, Estonian,[2] Hindi, Italian, Norwegian, and Russian.

1 `https://universaldependencies.org/`.

2 Note that we swapped out Korean with Estonian as we were unable to learn a high-quality subnetwork for Korean. The choice of Estonian is mainly motivated by the high-resource data requirement in combination with the fact that the subfamily, that is, Uralic, was not represented by our fine-tuning languages yet.

4. Methodology

In §4.1–4.2 we describe the model that will be used throughout our experiments and the training strategy. In §4.3 we then explain how we define and select subnetworks, and how we apply them to our models. In §4.4 we explain how our approach is adapted to the meta-learning setting, and in §4.5–4.6 we describe our test set-up and baselines.

4.1 Model

Our implementation is derived from UDify (Kondratyuk and Straka 2019), but uses only the parsing task rather than its full multi-task set-up. The model is built on mBERT (Devlin et al. 2019), a bidirectional Transformer (Vaswani et al. 2017) with 12 layers, each with 12 attention heads, pretrained on the combined Wikipedia dumps of 104 languages, and using a shared WordPiece vocabulary for tokenization. We initialize the model from mBERT, plus random initialization of the task-specific classifier. For each input token j, a weighted sum r_j over all layers $i \in [1..12]$ is computed as follows:

$$r_j = \eta \sum_i \mathbf{U}_{i,j} \cdot \text{softmax}(\lambda)_i \tag{1}$$

where $\mathbf{U}_{i,j}$ is the output of layer i at token position j, λ is a vector of trainable scalar mixing weights that distribute importance across the layers, and η is a trainable scalar that scales the normalized averages. For words that were tokenized into multiple word pieces, only the first word piece is used as input to the task-specific graph-based biaffine attention classifier (Dozat and Manning 2016).

The classifier projects the word encodings r_j through separate arc-head and arc-child feedforward layers with 768 hidden dimensions and Exponential Linear Unit non-linear activation. The resulting outputs $H_{\text{arc-head}}$ and $H_{\text{arc-dep}}$ are then combined using the biaffine attention function with weights $\mathbf{W}_{\text{arc}}$ and bias $\mathbf{b}_{\text{arc}}$ to score all possible dependency arcs:

$$S_{\text{arc}} = H_{\text{arc-head}} \mathbf{W}_{\text{arc}} H^T_{\text{arc-dep}} + \mathbf{b}_{\text{arc}} \tag{2}$$

Similarly, we compute label scores S_{tag} by using another biaffine attention function over two separate tag-head and tag-child feedforward layers with 256 hidden dimensions. The Chu-Liu/Edmonds algorithm (Chu 1965) is then used to select the optimal valid candidate tree.

4.2 Training Procedure

Taking inspiration from Nooralahzadeh et al. (2020) for cross-lingual transfer to low-resource languages, our training procedure is split into two stages: (1) fine-tune on the full English training set (~12.5K samples), without applying any subnetwork restrictions, for 60 epochs, to provide the full model with a general understanding of the task; and (2) fine-tune on the 7 other high-resource languages, to give the model a broad view over a typologically diverse set of languages in order to facilitate cross-lingual transfer to new languages.

For stage 2, in each iteration, we sample a batch from each language and average the losses of all languages to update the model. During this stage, we restrict each example to just the parameters in that language's subnetwork. We perform 1,000 iterations, with a batch of size 20 from each of the 7 languages, for a total of 140K samples.

We use a cosine-based learning rate scheduler with 10% warm-up and the Adam optimizer (Kingma and Ba 2015), with separate learning rates for updating the encoder and the classifier (see Appendix A, Table 9 for details).

4.3 Subnetwork Masks

We represent language-specific subnetworks as masks that are applied to the model in order to ensure that only a subset of the model's parameters are activated (or updated) during fine-tuning and inference. We follow Prasanna, Rogers, and Rumshisky (2020) in using *structured* masks, treating entire attention heads as units which are always fully enabled or disabled. Thus, for language ℓ, its subnetwork is implemented as a binary mask $\xi_\ell \in \{0,1\}^{12\times12}$.

In our experiments, we present two ways of using the masks during fine-tuning: *statically*, in which we find initial masks based on the pretrained model parameters and hold those masks fixed throughout fine-tuning and inference ($\text{SN}_{\text{static}}$); and *dynamically*, in which we update those masks over the course of fine-tuning (SN_{dyna}). In Figure 1, we give a general overview of our training procedure.

4.3.1 Finding Initial Subnetwork Masks. We aim to find a mask for each of the 7 fine-tuning languages that prunes away as many heads as possible without harming performance for that language (i.e., by pruning away heads that are only used by other languages, or that are unrelated to the dependency parsing task). For this, we apply the procedure introduced by Michel, Levy, and Neubig (2019).

For a language ℓ, the procedure starts by fine-tuning the model on ℓ's training set. We then iterate by repeatedly removing the 10% of heads with the lowest importance

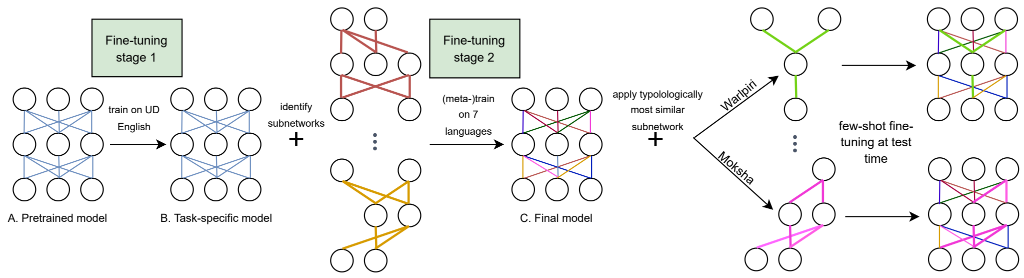

Figure 1
Schematic overview of our two-stage fine-tuning and test procedure. At fine-tuning stage 1, we first fine-tune pretrained mBERT on the task of dependency parsing using English data. We then apply language-specific subnetworks to our task-specific model. At fine-tuning stage 2, we either keep the subnetworks static or dynamically update the found subnetworks during (meta-)training on the task of dependency parsing using the other 7 fine-tuning languages. At test time, we then perform few-shot fine-tuning separately for each test language while applying the subnetwork of the typologically most similar training language.

scores $\mathrm{HI}_\ell^{(i,j)}$ (i=head, j=layer), which is estimated based on the expected sensitivity of the model to mask variable $\xi_\ell^{(i,j)}$:

$$\mathrm{HI}_\ell^{(i,j)} = \mathbb{E}_{x_\ell \sim X_\ell} \left| \frac{\delta \mathcal{L}(x_\ell)}{\delta \xi_\ell^{(i,j)}} \right| \tag{3}$$

where X_ℓ is ℓ's data distribution, x_ℓ is a sample from that distribution, and $\mathcal{L}(x_\ell)$ is the loss with respect to the sample. The procedure stops when performance on the ℓ's development set reaches 95% of the original model performance.

Consistent with findings from Prasanna, Rogers, and Rumshisky (2020), we observed that the subnetworks found by the procedure are unstable across different random initializations. To ensure that the subnetwork we end up with is more robust to these variations, we repeat the pruning procedure with 4 random seeds, and take the union[3] of their results as the true subnetwork (i.e., it includes even those heads that were only *sometimes* found to be important).

4.3.2 Dynamically Adapting Subnetworks. Blevins, Gonen, and Zettlemoyer (2022) showed that multilingual models acquire linguistic knowledge progressively—lower-level syntax is learned prior to higher-level syntax, and then semantics—but that the order in which the model learns to transfer information between specific languages varies. As such, the optimal set of parameters to share may depend on what learning stage the model is in, or on other factors, for example, the domains of the specific training datasets, the amounts of data available, the complexity of the language with respect to the task, and so forth. Thus, we propose a dynamic approach to subnetwork sharing, in which each language's subnetwork mask is trained jointly with the model during fine-tuning. This allows the subnetwork masks to be improved, and also allows for different patterns of sharing at different points during fine-tuning.

For dynamic adaptation, we initialize the identified static subnetworks as described in §4.3.1 using small positive weights. We then allow the model to update the mask weights during fine-tuning. After each iteration, the learned weights are fed to a threshold function that sets the smallest 20% of weights to zero (i.e., 28 heads[4]) to obtain a binary mask again. Given that the derivative of a threshold function is zero, we use a straight-through estimator (Bengio, Léonard, and Courville 2013) in the backward pass, meaning that we ignore the derivative of the threshold function and pass the incoming gradient on as if the threshold function was an identity function.

4.4 Meta-learning with Subnetworks

Meta-learning for multilingual models has been shown to enable both quick adaptation to unseen languages (Langedijk et al. 2022) and mitigation of negative interference (Wang, Lipton, and Tsvetkov 2020), but it does so using techniques that are different from—though compatible with—our subnetwork-sharing approach. Therefore, we

3 Stricter criteria (e.g., the intersection of the 4 subnetworks) resulted in lower performance on the development set.

4 We opted for a number roughly between our largest (13 heads pruned) and smallest (37 heads pruned) language-specific subnetwork found via pruning.

Algorithm 1 Meta-training procedure

```
Require: Language datasets $\mathcal{T}$; step sizes $\alpha$ and $\beta$; number of updates $k$; number of
    episodes EPS; support/query set size $N$; and subnetworks $\{\xi_\ell \mid \ell \in \mathcal{T}\}$. Train on $\ell \notin \mathcal{T}$
    to yield initial parameters $\theta$.
    for EPS do:
        for $\ell \in \mathcal{T}$ do :                                              (inner loop)
            Yield learner: $\phi_\ell \leftarrow \theta$.copy()
            Mask $\phi_\ell$ using $\xi_\ell$
            Take $N$ samples to form $\mathcal{D}_\ell^{trn} = \{x\}_{n=1}^N \in \mathcal{T}_\ell$ and $\mathcal{D}_\ell^{tst} = \{x\}_{n=1}^N \in \mathcal{T}_\ell$
            Update learner $\phi_\ell$ on the support set ( $\mathcal{D}_\ell^{trn}$):
            for $k$ steps do:
                $\phi_\ell \leftarrow \theta - \alpha\nabla_\theta\mathcal{L}(\phi_\ell, \mathcal{D}_\ell^{trn})$
            end for
            Evaluate on the query set: $\mathcal{L}(\phi_\ell, \mathcal{D}_\ell^{tst})$
        end for
        Meta-update the original model $\theta$:                                   (outer loop)
        $\theta \leftarrow \theta - \beta\sum_{\ell\in\mathcal{T}}\nabla_\theta\mathcal{L}(\phi_\ell, \mathcal{D}_\ell^{tst})$
    end for
```

experiment with the combination of these methods, and test the extent to which their benefits are complementary (as opposed to redundant) in practice.

To integrate our subnetworks within a meta-learning set-up, we just have to apply them in the inner loop of MAML, that is, given a model f parameterized by θ, we train θ by optimizing for the performance of the learner model of a language ℓ masked with the corresponding subnetwork $f_{\phi_\ell} \cdot \xi_l$. See Algorithm 1 for the details of the procedure.[5]

For all meta-learning experiments, we train for 500 episodes with support and query sets of size 20, that is, 10K samples per language are used for meta-training and validation each. We use 20 inner loop updates (k) and we follow Finn, Abbeel, and Levine (2017) in using SGD for updating the learner. All other training details are kept consistent with the non-episodic (NONEP) models (as described in §4.2).

4.5 Few-shot Fine-tuning at Test Time

Because the primary goal of this work is to improve performance in low-resource scenarios, we evaluate our models using a set-up that is appropriate when there is almost no annotated data in the target language: Few-shot fine-tuning. For a given test language, the model is fine-tuned on just 20 examples in that language, using 20 gradient updates. The examples are drawn from the development set, if there is one; otherwise they are drawn from (and removed from) the test set. We use the same hyperparameter values as during training. We report Labeled Attachment Scores (LAS) averaged across 5 random seeds, as computed by the official CoNLL 2018 Shared Task evaluation script.[6]

5 Note that for the meta-update, we use a first-order approximation, replacing $\nabla_\theta\mathcal{L}(\phi_\ell, \mathcal{D}_\ell^{tst})$ by $\nabla_\phi\mathcal{L}(\phi_\ell, \mathcal{D}_\ell^{tst})$. See Finn, Abbeel, and Levine (2017) for more details on first-order MAML.

6 `https://universaldependencies.org/conll18/evaluation.html`.

Table 1
Results on UD Parsing, for both non-episodic (NonEp) and meta-learning (Meta) set-ups. For each of the 6 models, we report Labeled Attachment Score (LAS) averaged across all 82 test languages, as well as the percentage of languages for which that model performed best (e.g., Meta-SN_{dyna} yielded the highest LAS on 28% of test languages). The best performance is denoted by **boldface**.

		Full	SN_{static}	SN_{dyna}	Total
NonEp	LAS	38.49	**41.32**	40.0	
	Best%	0%	22%	8.5 %	30.5%
Meta	LAS	40.68	40.27	**40.89**	
	Best%	14.5%	27%	**28%**	**69.5%**

Since we do not have subnetworks for the test languages—only for the 7 high-resource languages used in stage 2 of fine-tuning (§4.2)—we instead use the subnetwork of the typologically most similar training language. We determine typological similarity by computing the cosine similarity between the language vectors from the URIEL database (`syntax_knn`) (Littell et al. 2017).

4.6 Baselines

To measure the effectiveness of our subnetwork-based methods, we train and evaluate baselines in which no subnetwork masking is applied (but for which all other details of the training and testing set-ups are kept unchanged). We refer to this as **full model training** (Full) to contrast our training approaches that use static or dynamic subnetworks (SN_{static} and SN_{dyna}), and we report these baselines for both the non-episodic (NonEp)[7] and meta-learning (Meta) frameworks. For a fair comparison to existing literature, we also re-train UDify on dependency parsing using only our 8 treebanks for training and perform few-shot fine-tuning at test time as was done for all other models (Udf8).

5. Results

Overall, the results show that our subnetwork-based methods yield improvements over baseline models trained without any subnetwork masking. In Table 1, we see that, based on average LAS scores across all test languages, static subnetworks (SN_{static}) perform best in the non-episodic training set-up, resulting in +2.8% average improvement over the Full baseline, and yielding the highest average LAS of all the models. Dynamic subnetworks (SN_{dyna}), on the other hand, exhibit superior performance in the meta-learning setting, resulting in the model that performed best across all settings for the largest number of languages. In Table 2, we report the full set of results on all 82 test languages.

To gain more insight into the effects of our methods across the test languages, we plot the distribution over performance changes compared with the baseline per method

7 Note that Non-Episodic (NonEp) is used throughout the article to refer to models trained without meta-learning.

and learning framework in Figure 2. We find that static and dynamic subnetworks exhibit opposite trends. NONEP-SN$_{\text{static}}$ achieves large gains (up to +25%), but can also cause more deterioration on other languages (up to −6%). In contrast, the performance change distribution for NONEP-SN$_{\text{dyna}}$ is centered around more modest improvements, but is also the safest option given that it deteriorates performance for the fewest languages. The same trade-off can be observed in the meta-learning framework, except that now META-SN$_{\text{static}}$ results in modest changes compared with META-SN$_{\text{dyna}}$.

Table 2
Average LAS scores across 5 random seeds for all test languages (we do not report standard deviations as they were overall very small [6e-05–0.09]). Within each learning framework (NONEP and META) the best performance is denoted by **boldface**. Subnetwork-based models that substantially improve over their full-model baselines are highlighted, and color-code based on the amount of improvement: 🟧 +3–5%, 🟨 +5–7 %, 🟩 +7–10 %, 🟦 +10–15%, 🟪 +20–25%. Results are grouped according to which high-resource language was the source of their subnetwork mask (i.e., which high-resource language is most typologically similar), and we report average typological similarity between transfer and test languages ($\bar{\theta}$). Lastly, next to results from our Udf8 baseline, we report available scores for UDify trained on 75 languages (Udf75) from Kondratyuk and Straka (2019), but note that these scores are not directly comparable as they come from *zero-shot* testing.

			Non-Episodic (NONEP)			Meta-Learning (META)		
From Arabic ($\bar{\theta} = 0.70$)	Udf75	Udf8	FULL	SN$_{\text{static}}$	SN$_{\text{dyna}}$	FULL	SN$_{\text{static}}$	SN$_{\text{dyna}}$
Guajajara TuDeT	–	32.47	28.61	27.07	**33.62**	25.80	26.11	**30.90**
Kiche IU	–	37.94	40.13	**41.24**	40.04	30.10	21.03	**40.78**
Indonesian GSD	80.10	63.95	63.64	63.96	**64.05**	62.39	62.42	**64.73**
Indonesian PUD	56.90	71.63	70.08	**74.88**	71.54	**74.71**	73.32	71.64
Javanese CSUI	–	56.77	57.80	**61.92**	60.07	62.40	**62.94**	60.84
Maltese MUDT	75.56	29.75	**29.37**	24.67	28.47	16.92	13.43	**30.05**
Mbya Guarani Thomas	–	17.43	16.76	16.30	**17.61**	10.79	11.55	**16.55**
South Levantine Arabic	–	36.77	39.42	**41.93**	41.20	42.05	42.32	**42.37**
Tagalog TRG	40.07	69.06	70.46	65.26	**71.82**	**73.17**	71.58	72.58
Tagalog Ugnayan	–	48.16	48.39	47.38	**49.76**	50.93	46.69	**53.22**
Vietnamese VTB	66.0	38.81	40.79	**44.62**	43.34	**45.24**	43.75	43.67
Wolof WTB	–	23.16	20.72	18.98	**22.55**	17.26	15.56	**24.63**
Average (12)	–	43.83	43.85	44.02	**45.34**	42.17	40.89	**46.00**
From Czech ($\bar{\theta} = 0.83$)	Udf75	Udf8	FULL	SN$_{\text{static}}$	SN$_{\text{dyna}}$	Meta	SN$_{\text{static}}$	SN$_{\text{dyna}}$
Armenian ArmTDP	78.61	50.28	48.22	**58.35**	52.07	57.64	**61.09**	50.66
Armenian BSUT	–	58.80	57.26	**64.75**	59.49	62.95	**67.30**	60.24
Kurmanji MG	20.40	14.19	13.28	**16.40**	14.78	15.57	12.86	**17.28**
Lithuanian ALKSNIS	–	50.75	50.09	**59.98**	57.28	60.81	**61.20**	53.15
Lithuanian HSE	69.34	54.75	53.02	**59.74**	57.28	61.26	**61.38**	55.57
Western Armenian	–	41.59	43.01	**57.0**	49.14	56.93	**58.34**	48.62
Average (6)	–	45.06	44.15	**52.70**	48.34	52.53	**53.70**	47.59
From Estonian ($\bar{\theta} = 0.84$)	Udf75	Udf8	FULL	SN$_{\text{static}}$	SN$_{\text{dyna}}$	Meta	SN$_{\text{static}}$	SN$_{\text{dyna}}$
Apurina UFPA	–	37.25	37.70	**39.66**	37.68	28.18	24.11	**35.75**
Erzya JR	16.38	14.90	16.06	**17.35**	16.39	17.77	**18.66**	15.64
Hungarian Szeged	84.88	52.19	53.38	**62.24**	54.51	61.67	**68.69**	50.20
Karelian KKPP	–	35.06	36.67	**43.69**	38.41	40.58	40.19	**40.93**

Komi Permyak UH	–	24.19	24.47	**26.19**	25.86	24.96	**26.52**	25.65
Komi Zyrian IKDP	22.12	22.46	22.58	**25.55**	23.62	**24.97**	22.23	24.63
Komi Zyrian Lattice	12.99	14.21	14.17	**16.43**	14.23	14.72	**15.30**	13.62
Livvi KKPP	–	27.31	34.22	**38.00**	32.45	36.52	**37.08**	33.45
Moksha JR	–	15.09	15.20	**20.18**	16.30	18.79	**20.57**	16.65
North Sami Giella	67.13	14.25	14.05	14.69	**14.75**	11.74	11.90	**16.51**
Skolt Sami-Giellagas	–	25.21	26.49	26.20	**27.83**	21.84	18.10	**27.66**
Tatar NMCTT	–	54.73	52.63	**56.56**	55.67	55.79	**58.90**	54.61
Tupinamba TuDeT	–	20.83	21.24	21.65	**22.74**	16.68	15.30	**20.12**
Turkish PUD	46.07	46.47	47.00	**50.91**	49.34	50.58	50.01	**52.63**
Turkish IMST	67.44	34.70	34.90	**40.60**	35.99	40.87	**41.81**	36.32
Average (15)	–	29.26	30.05	**33.33**	31.05	31.04	**31.30**	30.96
From Hindi (θ = 0.74)	Udf75	Udf8	FULL	SN$_{static}$	SN$_{dyna}$	Meta	SN$_{static}$	SN$_{dyna}$
Akuntsu TuDeT	–	25.28	**24.71**	21.86	23.97	21.76	21.29	**25.55**
Bambara CRB	8.60	20.52	21.94	**22.54**	21.47	17.79	18.09	**23.76**
Basque BDT	80.97	45.15	45.81	47.59	**48.60**	**52.81**	52.52	45.71
Beja NSC	–	18.26	18.07	14.79	**19.95**	14.04	8.21	**19.87**
Bengali BRU	–	44.94	43.49	**48.67**	42.87	**58.91**	58.52	47.33
Bhojpuri BHTB	35.90	36.16	36.00	**38.76**	38.24	36.72	**37.70**	35.71
Buryat BDT	26.28	18.45	15.95	16.50	**17.31**	24.26	25.02	**27.79**
Kaapor TuDeT	–	30.11	30.54	**33.29**	29.87	30.77	30.18	**32.75**
Kangri KDTB	–	33.84	30.79	**34.32**	34.20	**36.16**	35.90	35.77
Karo TuDeT	–	18.47	18.47	19.01	**19.32**	17.47	17.38	**18.76**
Kazakh KTB	63.66	46.96	45.35	**50.50**	47.07	53.92	**54.70**	48.56
Makurap TuDeT	–	24.27	25.26	**25.35**	23.98	20.63	20.07	**28.47**
Marathi UFAL	67.72	37.38	37.96	**41.46**	37.72	**51.21**	50.78	39.17
Munduruku TuDeT	–	35.60	**35.73**	32.74	34.25	29.43	28.87	**36.51**
Sanskrit UFAL	18.56	18.40	18.63	19.70	**19.74**	21.58	**22.18**	19.00
Sanskrit Vedic	–	13.02	12.96	**13.09**	12.43	12.51	12.40	**12.71**
Tamil MWTT	–	58.95	63.11	**65.34**	61.86	**72.39**	72.04	64.76
Tamil TTB	71.29	47.51	46.66	**51.48**	48.10	52.00	**53.89**	46.76
Uyghur UDT	48.80	20.05	20.78	21.07	**21.41**	**21.04**	19.93	20.91
Warlpiri UFAL	7.96	51.01	55.91	59.30	**59.40**	42.78	42.67	**59.69**
Yakut YKTDT	–	34.35	30.85	**35.06**	34.73	32.40	32.54	**33.99**
Yupik SLI	–	12.18	12.73	11.31	**13.28**	8.84	9.28	**33.92**
Average (22)	–	31.40	31.44	**33.09**	32.27	**34.63**	34.39	33.80
From Italian (θ = 0.85)	Udf75	Udf8	FULL	SN$_{static}$	SN$_{dyna}$	Meta	SN$_{static}$	SN$_{dyna}$
Akkadian PISANDUB	4.54	21.36	17.33	11.42	**18.44**	7.44	9.30	**19.28**
Akkadian RIAO	–	22.19	21.87	17.99	**23.17**	12.89	9.33	**27.01**
Breton KEB	39.84	52.98	50.33	**61.63**	53.89	63.05	**64.31**	56.67
Galician TreeGal	76.77	75.79	75.81	**77.63**	76.05	**78.41**	77.95	76.09
Greek GDT	92.15	78.74	77.90	**81.46**	80.14	**81.13**	80.79	79.78
Irish IDT	69.28	46.45	47.14	**50.80**	49.04	52.03	**53.10**	49.42
Ligurian GLT	–	25.98	29.43	**49.81**	34.05	44.26	**46.65**	34.30
Manx Cadhan	–	44.76	46.13	44.97	**46.64**	40.52	36.70	**47.31**
Naija NSC	32.16	32.12	32.00	**35.59**	32.09	34.52	33.46	**37.84**
Scottish Gaelic ARCOSG	–	17.09	15.41	**23.28**	18.57	24.49	**25.88**	21.86
Welsh CCG	–	47.86	47.37	51.72	**52.60**	**54.97**	53.18	51.10
Average (11)		42.30	41.88	**46.03**	44.10	44.88	44.60	**45.54**
From Norwegian (θ = 0.91)	Udf75	Udf8	FULL	SN$_{static}$	SN$_{dyna}$	Meta	SN$_{static}$	SN$_{dyna}$
Afrikaans AfriBooms	–	66.64	65.88	63.57	**65.94**	68.28	63.64	**69.75**

Albanian TSA	–	72.45	70.95	**76.06**	73.34	**79.76**	76.65	74.13
Faroese FarPaHC	–	49.75	47.51	50.03	**51.0**	49.24	44.70	**54.13**
Faroese OFT	59.26	62.17	60.95	**70.36**	63.76	**70.12**	69.41	65.87
Gothic PROIEL	79.37	19.53	19.23	**19.67**	18.68	16.65	15.85	**20.24**
Icelandic Modern	–	47.36	45.98	**49.98**	47.44	**53.45**	50.43	51.27
Low Saxon LSDC	–	50.70	47.75	**51.42**	49.88	**50.26**	47.50	50.08
Swiss German UZH	–	47.12	45.98	**52.66**	47.57	**51.93**	51.73	51.16
Average (8)	–	51.97	50.48	**54.22**	52.20	**54.96**	52.49	54.58
From Russian ($\theta = 0.76$)	Udf75	Udf8	FULL	$\text{SN}_{\text{static}}$	SN_{dyna}	Meta	$\text{SN}_{\text{static}}$	SN_{dyna}
Ancient Greek PROIEL	82.11	26.55	23.68	**28.38**	27.72	23.81	**31.24**	26.32
Cantonese HK	32.01	28.26	28.66	**31.17**	30.58	**33.02**	32.87	31.50
Chinese CFL	42.48	44.35	45.48	**49.28**	48.26	49.88	**50.67**	47.50
Chinese HK	49.32	47.75	47.20	**49.94**	48.26	52.31	**52.90**	48.16
Chinese PUD	56.51	43.49	44.74	**46.98**	46.47	45.9	44.92	**46.71**
Serbian SET	91.95	81.05	78.98	**81.57**	79.66	**80.98**	80.96	79.8
Upper Sorbian UFAL	62.82	53.26	49.81	**54.88**	53.84	**54.01**	53.78	51.29
Yoruba YTB	19.09	38.34	38.11	38.17	**38.75**	38.79	38.17	**39.28**
Average (8)	–	45.38	44.58	**47.55**	46.62	47.34	**48.19**	46.32
Total Avg. (82)	–	38.66	38.49	**41.32**	40.0	40.68	40.27	**40.89**

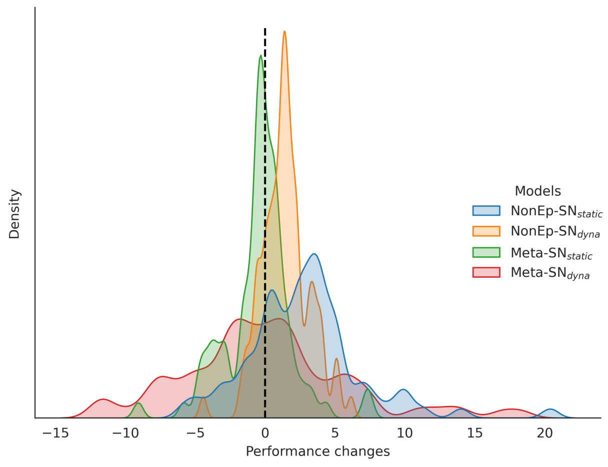

Figure 2
Kernel density estimation plot over the relative performance changes of each model for all test languages when comparing with its corresponding full model training baseline.

Lastly, we do not find strong trends for transfer languages; different magnitudes of performance changes are scattered across all transfer languages. Yet, when transferring from Norwegian, META-$\text{SN}_{\text{static}}$ and META-SN_{dyna} particularly often underperform compared with META-FULL; see Table 2. In contrast, META-SN_{dyna} performs particularly well when transferring from Arabic; similarly, $\text{SN}_{\text{static}}$ performs especially well when transferring from Czech. Thus, the best approach might be dependent on the relationship between the transfer and test languages, or the properties of the transfer language itself.

We note that despite the observed improvements, overall performance remains low for many languages. Yet we would like to point out that we also find instances where our methods might already make the difference in acquiring a usable system compared with state-of-the-art models. For example, even with few-shot fine-tuning Udf75's performance on Faroese OFT only reaches 53.8%, which is much lower than our 70.4% (NONEP-SN$_{\text{static}}$), and for Indonesian PUD it reaches 69.0% versus 74.9% (NONEP-SN$_{\text{static}}$)

6. Analysis

In this section, we provide more insight into the effects of our methods by analyzing performance with respect to four factors: typological relatedness, data-scarcity, robustness to domain transfer, and ability to predict unseen and rare labels. We focus on the best model from each learning framework: NONEP-SN$_{\text{static}}$ and META-SN$_{\text{dyna}}$.

Typological Relatedness. The languages most similar to a low-resource language are often themselves low-resource, meaning that a low-resource language is often quite dissimilar from all the languages that are resource-rich enough to be used for fine-tuning. A method that only works well when a very similar high-resource language is available for fine-tuning will not be as useful in practice. Thus, we want to understand the degree to which our methods depend on similarity to a high-resource fine-tuning language. In Figure 3 (top), we plot each test language's performance improvement against its typological closeness to the nearest high-resource fine-tuning, where that distance is as computed using the cosine similarity between the languages' URIEL features. Interestingly, we find that our models show opposite trends: Whereas NONEP-SN$_{\text{static}}$ works well for typologically similar languages, the biggest gains from META-SN$_{\text{dyna}}$ actually come from less similar languages.

Data Scarcity. Given that language distribution in the mBERT pretraining corpus is very uneven, and 37 of our 70 unique test languages are not covered at all, we want to understand what effect this has on downstream model performance. As shown in Figure 3 (middle), we find that META-SN$_{\text{dyna}}$ provides the most benefit to previously unseen languages. In contrast, more data in pretraining positively correlates with the performance of NONEP-SN$_{\text{static}}$.

Out-of-domain Data. For cross-lingual transfer we often focus on the linguistic properties of source and target languages. However, the similarity of the source and target datasets will also be based on the domains from which they were drawn (Glavaš and Vulić 2021). For example, our training datasets cover only 11 of 17 domains, as annotated by the creators of the UD treebank. While we acknowledge that it is difficult to neatly separate data based on source domain, we test for a correlation between performance and the proportion of out-of-domain data. Interestingly, we find no clear correlation with the percentage of domains from the test language covered by the transfer language. We do, however, find a strong correlation with the domain diversity of the transfer and test language in general for NONEP-SN$_{\text{static}}$, as shown in Figure 3 (bottom), where we plot improvements against number of domain sources our test data is coming from (more sources $\rightarrow$ more diversity). In contrast, we see that META-SN$_{\text{dyna}}$ remains insensitive to this variable.

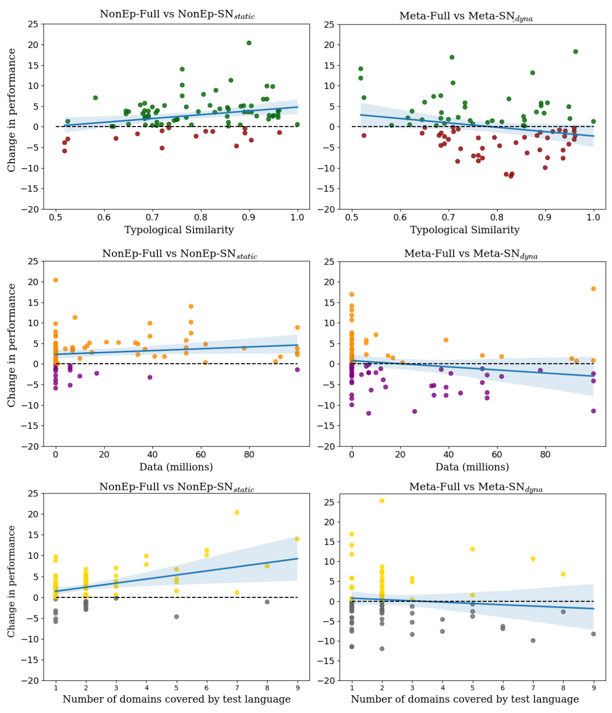

Figure 3
Plots of the relationships between a test language's performance gains and: (top) how typologically similar the language is to the nearest high-resource fine-tuning language, (middle) the amount of in-language data used to pretrain mBERT, and (bottom) the number of domain sources represented in its test data. Note that different colors were only used for visual ease.

Unseen and Rare Labels. Lastly, another problem in cross-lingual transfer, especially when fine-tuning on only a few languages, is that the fine-tuning data may not cover the entire space of possible labels from our test data. In principle, only a model that is able to adequately adapt to unseen and rare labels can truly succeed in cross-lingual transfer. Given that we perform few-shot fine-tuning at test time, we could potentially overcome this problem (Lauscher et al. 2020). Thus, we investigate the extent to which our models succeed in predicting such labels for our test data. We consider a label to be rare when it is covered by our training data, but makes up <0.1% of training instances (23 such labels). There are 169 unseen labels, thus in total, 192 of 233 (82%) of the labels from our test data are rare or unseen during training. In Table 3, we report how often each model correctly predicts instances of unseen and rare labels. We find that models differ greatly, and, in particular, META-SN$_{\text{dyna}}$ vastly outperforms all other models when it comes to both unseen and rare labels. Upon further inspection, we find that two unseen labels are particularly often predicted correctly: sentence particle (`discourse:sp`) and inflectional dependency (`dep:infl`). The former label seems specific to Chinese linguistics and has a wide range of functions (e.g., modifying the modality of a sentence or its proposition, and expressing discourse and pragmatic information). The latter represents inflectional suffixes for the morpheme-level annotations, something that is unlikely to be observed in morphologically poor languages such as English; but, for instance, Yupik has much of its performance boost due to it.

Table 3
Percentages of correctly predicted instances of unseen and rare labels. We also report across how many labels/languages correct predictions were made.

NonEp-	Full	SN_{static}	SN_{dyna}
Unseen	0.04% (3/3)	0.003% (1/1)	0.004% (2/2)
Rare	12.5% (12/50)	6.4% (11/41)	9.9% (8/49)

Meta-	Full	SN_{static}	SN_{dyna}
Unseen	0%	0%	**6.6%** (15/23)
Rare	3.5% (10/39)	3.0% (7/36)	**21.3%** (13/55)

7. Effect of Subnetworks at Training Time

7.1 Interaction Between Subnetworks

We now further investigate the selected subnetworks and their impact during training. Our findings were similar for meta-learning, so we just focus our analysis here on the non-episodic models.

Table 4 shows how using subnetworks affects performance on the training languages. Training with the subnetworks always improves performance, however, this effect is larger when subnetworks are kept static during training. Moreover, for the static subnetworks, the number of heads that are masked out can vary considerably per language; for example, for Arabic we only disable 13 heads compared with 37 for Estonian. Yet, we observe similar effects on performance, obtaining ∼+4% improvement for both languages. To disentangle how much of the performance gain comes from disabling suboptimal heads vs. protection from negative interference by other languages, we re-train NonEp-SN_{static} in two ways using Czech as a test case: (1) we keep updates from Czech restricted to its subnetwork (i.e., we disable the suboptimal heads for Czech), but drop subnetwork masking for the other languages (i.e., we do not protect Czech from negative interference as all other languages can still update the full model); (2) we use subnetworks for all languages *except* Czech (i.e., we protect Czech from the other languages by restricting their updates to their subnetworks only, but still allow Czech to use the full model capacity).

Table 4
Labeled Attachment Scores for Non-Episodic models on each training language. Number of heads disabled by the subnetwork is shown in parentheses.

Language	Full	SN_{static}	SN_{dyna}
Arabic	68.6	**72.9** (13)	69.1 (28)
Czech	75.4	**81.2** (13)	77.9 (28)
Estonian	65.4	**69.2** (37)	68.3 (28)
Hindi	74.4	**77.2** (21)	75.2 (28)
Italian	85.0	**87.7** (23)	86.1 (28)
Norwegian	73.2	**79.8** (24)	73.6 (28)
Russian	79.5	**81.6** (27)	80.4 (28)

Table 5
Labeled Attachment Scores for the baseline model FULL on each fine-tuning language $\ell \in \mathcal{T}$ when either using a subnetwork for the fine-tuning language ℓ only (selection) or using a subnetwork for all fine-tuning languages in $\mathcal{T} \setminus \{\ell\}$ (protection). Percentage of improvement over the FULL baseline is shown in parentheses.

Language	FULL	Selection	Protection
Arabic	68.6	71.2 (+2.5)	72.2 (+3.6)
Czech	75.4	79.5 (+4.1)	80.1 (+4.7)
Estonian	65.4	67.9 (+2.5)	67.9 (+2.5)
Hindi	74.4	76.8 (+2.4)	76.7 (+2.3)
Italian	85.0	86.8 (+1.8)	87.3 (+2.3)
Norwegian	73.2	80.2 (+7.1)	80.3 (+7.2)
Russian	79.5	80.5 (+1.0)	80.4 (+0.9)

We find that (1) disabling suboptimal heads for Czech only, results in 79.5 LAS on Czech (+4.1% improvement compared to baseline), while (2) just protection from the other languages, results in 80.3 LAS (+4.7% improvement); see Table 5 for results on the other training languages. This indicates that protection from negative interference has a slightly larger positive effect on the training language in this case. Still, a combination of both (i.e., using subnetworks for all fine-tuning languages) results in the best performance in most cases (81.2 LAS for Czech, a +5.9% improvement, as reported in Table 4). This suggests that the interaction between the subnetworks is a driving factor behind the selective sharing mechanism that resolves language conflicts. We confirm that similar trends were found for the other languages.

This, however, also means that if the quality of one subnetwork is suboptimal, it is still likely to negatively affect other languages. Moreover, analyzing the subnetworks can provide insights on language conflicts. For instance, using a subnetwork for only Czech or Arabic results in the biggest performance gains for Norwegian (+7.1% and +7.3% compared with the FULL baseline), indicating that, in this set-up, Norwegian suffers more from interference.

7.2 Gradient Conflicts and Similarity

In multilingual learning, we aim to maximize knowledge transfer between languages while minimizing negative transfer between them. In this study, our main goal is the latter. To evaluate the extent to which our methods succeed in doing this, we explicitly test whether we are able to mitigate negative interference by adopting the gradient conflict measure from Yu et al. (2020). They show that *conflicting gradients* between dissimilar tasks, defined as a negative cosine similarity between gradients, is predictive of negative interference in multi-task learning. Similar to Wang, Lipton, and Tsvetkov (2020), we deploy this method in the multilingual setting: We study how often gradient conflicts occur between batches from different languages. For batches from each language, we compute the gradient of the loss with respect to the parameters of the full model during backpropagation. To get a stable estimate, we use gradient accumulation for 50 episodes/iterations before computing conflicts. Gradient conflicts are then computed between each language pair, $\binom{7}{2}$ pairs in total, and the percentage of total conflicts is computed across all language pairs.

Table 6
We report the percentage of gradient conflicts and average cosine similarity between gradients over the last 50 iterations/episodes for our non-episodic and meta-trained models. We report average results over 4 random seeds.

	Conflicts	Cosine Sim.
NONEP-FULL	42%	0.03
NONEP-SN$_{\text{static}}$	**26%**	0.05
NONEP-SN$_{\text{dyna}}$	38%	**0.07**
META-FULL	55%	−0.04
META-SN$_{\text{static}}$	54%	−0.02
META-SN$_{\text{dyna}}$	**44%**	**0.12**

At the same time, Lee et al. (2021) argue that lower cosine similarity between language gradients indicates that the model starts memorizing language-specific knowledge that at some point might cause catastrophic forgetting of the pretrained knowledge. This suggests that, ideally, our approach would find a good balance between minimizing gradient conflicts and maximizing the cosine similarity between the language gradients.

We quantitatively find that both subnetwork-based methods indeed reduce the percentage of gradient conflicts between languages. Over the last 50 iterations, we find that NONEP-SN$_{\text{static}}$ has reduced conflicts by 16% and NONEP-SN$_{\text{dyna}}$ by 4% compared with the NONEP-FULL baseline as reported in Table 6. In the meta-learning set-up, we found an opposite trend where META-SN$_{\text{static}}$ reduces conflicts by 1% and META-SN$_{\text{dyna}}$ by 11% over the last 50 iterations compared to META-FULL. This partly explains why NONEP-SN$_{\text{static}}$ and META-SN$_{\text{dyna}}$ are found to be the best performing models: They suffer the least from gradient conflicts. Interestingly, we do not find that our meta-trained models suffer less from gradient conflicts than the non-episodic models. In fact, while we found that, on average, META-FULL improves over NONEP-FULL (recall Table 1), its training procedure suffers from 13% more conflicts, meaning that we do not find meta-learning in itself to be a suitable method for reducing gradient conflicts, but our subnetwork-based methods are.

At the same time, the average cosine similarity between gradients increases when using both subnetwork methods compared with the FULL model baselines. We compute the Pearson correlation coefficient between the relative decrease in percentage of gradient conflicts and increase in cosine similarity over training iterations compared with the baselines. We test for statistical significance (p-value <0.02), and average results over 4 random seeds. We get statistically significant positive correlation scores of 0.08, 0.16, 0.33, and 0.58 for NONEP-SN$_{\text{static}}$, NONEP-SN$_{\text{dyna}}$, META-SN$_{\text{static}}$, and META-SN$_{\text{dyna}}$, respectively. This indicates that our subnetwork-based methods try to minimize negative interference while simultaneously maximizing knowledge transfer.

8. Ablations

To ensure that each of the aspects of our set-up are indeed contributing to the improvements shown in our experiments, we retrained models with specific aspects ablated.

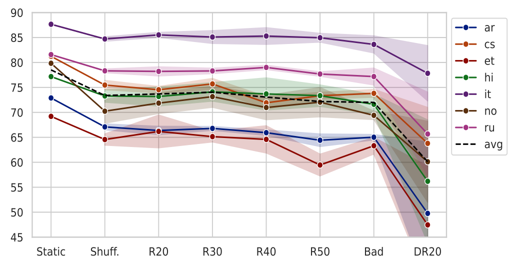

Figure 4
Effect of training with masks randomly generated under different constraints (across 3 seeds): Shuffled, masking *n* heads, only select bad heads, and start dynamic training from a random subnetwork (DR20).

8.1 Random Ablations

Random Mask Initialization – Static. In these experiments, we verify that there is value in using the iterative pruning procedure to generate subnetwork masks (as opposed to the value coming entirely from the mere fact that masks were used).

First, we re-trained NONEP-SN$_{\text{static}}$, but swapped out the subnetwork masks derived from iterative pruning with masks containing the same number of enabled heads, but that were randomly generated (Shuffle). Second, given that the number of masked heads might be more important than which exact heads are being masked out, we experiment with masking 20, 30, 40, and 50 random heads. We find that using the random masks results, on average, in ∼5% performance decreases on the training languages compared with using the subnetworks initialized using importance pruning; see Figure 4. In addition, we see that randomly masking out more heads results in further negative effects on performance.

Lastly, given that for many languages our subnetworks mask out very few heads (e.g., 13 for Arabic and Czech), we also try swapping these out with "intentionally bad" masks, where we randomly choose 20 heads to mask out, but do not allow any of the heads selected by the real pruning procedure to be chosen (Bad). From this, we see that preventing the right heads from being selected for masking does result in lower performance versus pure random selection (R20).

Random Mask Initialization – Dynamic. In these experiments, we verify that there is value in using the iterative pruning procedure to initialize subnetwork masks that will then by dynamically updated during fine-tuning.

We retrained NONEP-SN$_{\text{dyna}}$ 3 times using randomly initialized subnetworks. Figure 4 (DR20) shows that average performance across all test languages drops substantially (∼10%), making this method considerably worse than any of our other random baselines. We hypothesize that this is because the model is able to correct for any random static subnetwork, but that with dynamic masking, the subnetworks keep changing, which deprives the model of the chance to properly re-structure its information. This also gives us a strong indication that the improvements we observe are not merely an effect of regularization (Bartoldson et al. 2020).

Random Transfer Language. To test the effectiveness of our typology-based approach to selecting which high-resource fine-tuning language's subnetwork should be used for a given test language, we experimented with just picking one of the high-resource languages at random, and found that this performed worse overall, resulting in lower scores for 78 of 82 test languages.

8.2 Unstructured Pruning

Our approach relies on the assumption that attention heads function independently. However, attention head interpretability studies have sometimes given mixed results on their function in isolation (Prasanna, Rogers, and Rumshisky 2020; Clark et al. 2019; Htut et al. 2019). Moreover, related works commonly focus on unstructured methods (Lu et al. 2022; Nooralahzadeh et al. 2020). Thus, we compare our strategy of masking whole attention heads against versions of NONEP-SN$_{\text{static}}$ and NONEP-SN$_{\text{dyna}}$ that were retrained using subnetwork masks found using the most popular unstructured method, **magnitude pruning**. In magnitude pruning, instead of disabling entire heads during the iterative pruning procedure, as described in §4.3.1, we prune the 10% of parameters with the lowest magnitude weights across all heads. Again, we check the development set score in each iteration and keep pruning until reaching <95% of the original performance. Note that we exclude the embedding and MLP layers.[8]

We find that for both the static and dynamic strategies, unstructured pruning performs worse overall, resulting in lower scores for 76% of test languages, and is especially harmful for dynamic subnetworks (SN$_{\text{static}}$: 40.4 vs. 39.9, and SN$_{\text{dyna}}$: 39.0 vs. 36.7 average LAS). We hypothesize that it might be more difficult to learn to adapt the unstructured masks as there are more weights to learn (weights per head $\times$ heads per layer $\times$ layers).

8.3 Effect of Selected Training Languages

Given that the selection of training languages can have an important effect on the overall performance, we now also perform a set of ablations to test the robustness of our findings with respect to the choice of training languages.

Fine-tuning Stage 1. As presented above, we fine-tune mBERT first using English to learn the task of dependency parsing. While English is still the most commonly used source language for cross-lingual transfer, it is important to understand how this choice may affect downstream performance. Therefore, we also tried using three other languages in place of English for this step: (1) Chinese (GSD) as it uses a different script, (2) Turkish (PENN) as it has a different word order (SOV), and (3) German (GSD) as it has no dominant word order and it was found to be the best source language for transfer by Turc et al. (2021) (together with Russian).

In Table 7, we show, for each of the three languages, how much the average performance changes in comparison with using English. For both Chinese and Turkish, we find that the average performance across test languages slightly decreases for all NONEP models. Even though the decreases are only minor, it indicates that Chinese and Turkish do not transfer as well to our test languages as English. This is not completely surprising as more of our test languages are written in the Latin script and, like English,

8 We recognize that the interaction between the MLPs and attention heads is important, but by focusing on the attention heads, we keep results comparable to importance pruning.

Table 7
Average change in LAS across all test languages for the NONEP models trained with different languages for fine-tuning stage 1 compared with the original results obtained using English for fine-tuning stage 1. Note that we remove the datasets pertaining to languages used during fine-tuning when comparing—for example, Turkish datasets are removed from our test languages when we use the models fine-tuned on Turkish in our comparison.

Language	FULL	SN_{static}	SN_{dyna}
Chinese	–0.89	–0.46	–0.78
Turkish	–0.95	–0.62	–0.35
German	–0.22	+0.17	+0.07

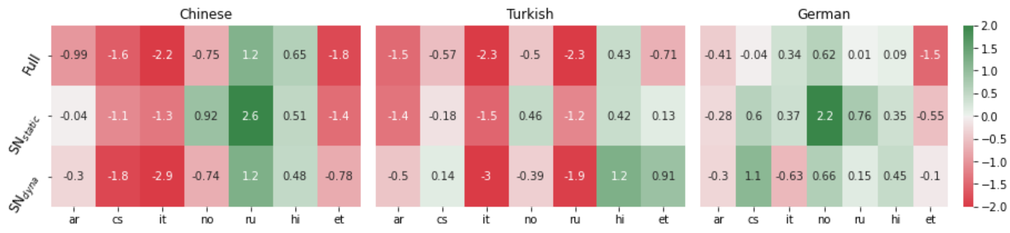

Figure 5
Average change in LAS for languages grouped by their typologically most similar training language. We show the average change in LAS for each language used during fine-tuning stage 1 compared with using English.

use SVO word ordering. Yet, similar to Turc et al. (2021), we find that German is the best source language as it increases our average results when using both static and dynamic subnetworks compared with using English. Interestingly, in Figure 5, we see that all languages are able to increase average performance for the languages most closely related to Hindi, which could indicate that English has some properties that are particularly badly suited for transfer to this set of languages. At the same time, swapping out English with any of our three new languages causes an average decrease in performance on test languages that are most closely related to Arabic.

Fine-tuning Stage 2. The set of 7 languages we used above for the second stage of fine-tuning was chosen to be comparable to previous studies, but that set of languages is dominated by the Indo-European language family, which may result in poor generalization to other language families. Thus, we also re-trained our NONEP models on a completely different set of 7 languages, which were chosen from among those languages with relatively large treebanks ($\geq$ 100K tokens), but selected in order to maximize diversity with respect to: (1) language family, (2) word order, and (3) data domain. This yielded the following set of languages: Belarusian (IE, Slavic, no dominant order), Chinese (Sino-Tibetan, SVO), Finnish (Uralic, SVO), Hebrew (Afro-Asiatic, SOV), Indonesian (Austronesian, SVO), Irish (Celtic, VSO), and Turkish (Turkic, SOV). These 7 languages cover 7 language families, 4 word orderings, and 14 data domains. Note that to limit the scope of this experiment, and to keep the results comparable to our original findings, we now again use English for fine-tuning stage 1.

We find that average results across all test languages[9] are very similar using this different set of training languages. More concretely, for our FULL, SN$_{\text{static}}$, and SN$_{\text{dyna}}$ (NONEP) models we only get +0.12, −1.09, and +0.02 average differences in LAS scores compared with our original results.[10] Thus, our methods seem to be fairly robust with respect to the choice of training languages, and more diversity in training languages does not automatically result in better performance. One artifact that could influence this is the fact that we have much less training data for some of these selected languages (e.g., Irish and Indonesian [see Appendix A, Table 10]), so the quality of the retrieved subnetworks could be worse than those found for our more high-resource training languages. Thus, it could be possible that with more training data, this same set of training languages would result in higher performance gains.

9. Conclusion

We present and compare two methods, namely, static and dynamic subnetworks, that successfully help us guide selective sharing in multilingual training across two learning frameworks: non-episodic learning and meta-learning. We show that through the use of subnetworks, we can obtain considerable performance gains on cross-lingual transfer to low resource languages compared to full model training baselines for dependency parsing. Moreover, we quantitatively show that our subnetwork-based methods are able to reduce negative interference. Finally, we extensively analyze the behavior of our best performing models and show that they possess different strengths, obtaining relatively large improvements on different sets of test languages with often opposing properties. Given that our META-SN$_{\text{dyna}}$ model performs particularly well on data-scarce and typologically distant languages from our training languages, this is an interesting approach to further explore in future work on low-resource languages. In particular, it would be interesting to investigate methods to integrate the strengths of NONEP-SN$_{\text{static}}$ and META-SN$_{\text{dyna}}$ into one model.

Lastly, we test our results only on the task of dependency parsing, which is somewhat different from other NLP tasks as it has an annotation scheme explicitly designed to be applied across languages universally. However, we would like to point out that many NLP tasks are implicitly multilingual as well since most tasks do not involve a language-specific annotation scheme. For instance, in Named Entity Recognition (NER), the goal is to classify named entities into predefined categories such as "person", "location", "organization", and so forth. When performing NER for other languages, we still select from the same categories. Moreover, negative interference is a general problem, first addressed in multi-task learning (Ruder 2017), and later studied in multilingual NLP (Wang, Lipton, and Tsvetkov 2020), that seems to occur whenever we attempt to learn multiple tasks/languages within one model. In multilingual NLP, languages will compete for the limited model capacity regardless of the task we are trying to solve. It was already shown that across a wide range of NLP tasks—NER, POS tagging, question answering, and natural language inference—negative interference occurs in multilingual models, and resolving such language conflicts can improve overall cross-lingual performance (Wang, Lipton, and Tsvetkov 2020). From our analysis of gradient conflicts, we find that similar negative interference issues can be found for the task

9 For a fair comparison, we removed test languages included in our new training set, e.g., Indonesian, so we average over 74 test languages instead. This was done for every experiment, where applicable.
10 We did not find clear patterns for the individual languages on which performance improvements are obtained.

of dependency parsing, and are mitigated by our subnetwork-based methods. Thus, as training with subnetworks appears to be a general approach to mitigating negative interference, we expect it to bring the same benefits to other NLP tasks for which this problem occurs. Moreover, we would like to point out that other studies have already shown the effectiveness of various other types of subnetworks for different tasks, for example, for Neural Machine Translation (Lin et al. 2021; Hendy et al. 2022) and cross-lingual speech recognition (Lu et al. 2022), making it less likely that the effectiveness of our methods are limited to dependency parsing only.

10. Limitations

One problem in multilingual NLP is that performance increases tend to happen for a specific set of languages at a time rather than across all languages simultaneously. This makes it hard to compare models and determine the state-of-the-art performance. Moreover, it is hard to determine the usefulness of a new method as average scores are not very informative when your test languages have a detrimental effect on this—for instance, taking out a few low performing languages would already boost our average performance substantially.

This also makes it more complicated to choose training languages. Changing the training languages can positively influence our performance at test time, especially if they are more similar to a large number of our test languages. However, when we want our model to generalize beyond our chosen set of test languages, it can be misleading to tailor the training set-up to the test data. Thus, while we do show that our methods generally improve performance when using two completely different sets of training languages, further experiments on finding an "optimal" set of training languages are omitted from this study. In addition, meta-learning is notorious for being hard to optimize; for example, slight changes in learning rates can have a detrimental effect on performance (Antoniou, Edwards, and Storkey 2019). This also means that different training languages can require different hyperparameter settings to work, which further complicates the search for an optimal training set.

Another limitation is that while we use a diverse set of test languages, our approach relies on the pretrained mBERT model, which means that it is unsuited to low-resource languages whose scripts are not seen during pretraining. Finding useful ways to circumvent this problem would be a good direction for follow-up work.

Lastly, given that we fine-tune on only 8 languages, the smallest typological distance between the training languages and a test language is often still relatively large. This makes the motivation for typology-informed subnetwork transfer at test time less satisfactory. In future work, it should be further investigated what the effect is of using more similar training and test language pairs for subnetwork transfer.

A. Training and Data Details

Table 8
Number of sentences in the UD treebanks for our training languages.

	Family	TB	Train	Val.	Test
ar	Afro-Asiatic	PADT	6,075	909	680
cs	Slavic	PDT	68,495	9,270	10,148
en	German.	EWT	12,543	2,002	2,077
hi	Indic	HDTB	13,304	1,659	1,684
it	Roman.	ISDT	13,121	564	482
et	Urallic	EDT	24,633	3,125	3,214
no	German.	Norsk	14,174	1,890	1,511
ru	Slavic	SynTag	48,814	6,584	6,491

Table 9
Final selection of learning rates. For all non-episodic models, we use the same learning rates (NONEP). Similarly, we found the same optimal hyperparameter values for all outer-loop learning rates of the meta-trained models (Meta-All). Moreover, the hyperparameter selection is performed based on 4 validation languages: Bulgarian, Japanese, Telugu, and Persian.

	Inner/Test LR	
	mBERT	decoder
NONEP	{**1e-04**, 5e-05, 1e-05}	{**1e-03**, 5e-04, 1e-04}
Unstructured	{1e-04, **5e-05**, 1e-05}	{1e-03, **5e-04**, 1e-04}
META-FULL	{1e-04, 5e-05, **1e-05**}	{1e-03, **5e-04**, 1e-04}
META-SN$_{\text{static}}$	{1e-04, 5e-05, **1e-05**}	{1e-03, **5e-04**, 1e-04}
META-SN$_{\text{dyna}}$	{**1e-04**, 5e-05, 1e-05}	{**1e-03**, 5e-04, 1e-04}
	Outer LR	
Meta-All	{**1e-04**, 5e-05, 1e-05 }	{**1e-03**, 5e-04, 1e-04}

Table 10
Number of sentences in the UD treebanks for our new training languages used in Section 8.3. The set covers 7 languages from 7 language families and 4 word orderings (i.e., SVO, SOV, VSO, and no dominant order), and they cover 14 data domains.

	Family	TB	Train	Val.	Test
be	IE, Slavic	HSE	22,853	1,301	1,077
fi	Uralic	TDT	12,217	1,364	1,555
ga	Celtic	IDT	4,005	451	454
he	Afro-Asiatic	HTB	5,241	484	491
id	Austronesian	GSD	4,482	559	557
tr	Turkic	Penn	14,850	622	924
zh	Sino-Tibetan	GSD	3,997	500	500

All models use the same UDify architecture with the dependency tag and arc dimensions set to 256 and 768, respectively. At fine-tuning stage 1, we train for 60 epochs following the procedure of Langedijk et al. (2022) and Kondratyuk and Straka (2019). The Adam optimizer is used with the learning rates of the decoder and BERT layers set to 1e-3 and 5e-5, respectively. Weight decay of 0.01 is applied, and we use a gradual unfreezing scheme, freezing the BERT layer weights for the first epoch. For more details on the training procedure and hyperparameter selection, see Langedijk et al. (2022). For fine-tuning on seperate languages to find the subnetworks, we apply the same procedure.

Moreover, we need $\sim$3 hours for pretraining and, depending on the training set size, $\sim$4 hours per language for fine-tuning and finding a subnetwork (note that this step is run in parallel for all languages and only needs to be performed once for all models trained with subnetworks). We then only require $\sim$1 hour for non-episodic training or $\sim$6 hours for meta-training. All models are trained on a NVIDIA TITAN RTX.

Acknowledgments
This project was in part supported by a Google PhD Fellowship for the first author. We would like to thank Tim Dozat and Vera Axelrod for their thorough feedback and insights.

References

Ansell, Alan, Edoardo Maria Ponti, Jonas Pfeiffer, Sebastian Ruder, Goran Glavaš, Ivan Vulić, and Anna Korhonen. 2021. MAD-G: Multilingual adapter generation for efficient cross-lingual transfer. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 4762–4781. https://doi.org/10.18653/v1/2021.findings-emnlp.410

Antoniou, Antreas, Harri Edwards, and Amos Storkey. 2019. How to train your MAML. In *Seventh International Conference on Learning Representations*.

Arivazhagan, Naveen, 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. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 3874–3884.

Bartoldson, Brian, Ari Morcos, Adrian Barbu, and Gordon Erlebacher. 2020. The generalization-stability tradeoff in neural network pruning. In *Advances in Neural Information Processing Systems*, volume 33, pages 20852–20864.

Bengio, Yoshua, Nicholas Léonard, and Aaron Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. *arXiv preprint arXiv:1308.3432*. https://doi.org/10.48550/arXiv.1308.3432

Blevins, Terra, Hila Gonen, and Luke Zettlemoyer. 2022. Analyzing the mono-and cross-lingual pretraining dynamics of multilingual language models. *arXiv preprint arXiv:2205.11758*. https://doi.org/10.48550/arXiv.2205.11758

Budhraja, Aakriti, Madhura Pande, Pratyush Kumar, and Mitesh M. Khapra. 2021. On the prunability of attention heads in multilingual BERT. *arXiv preprint arXiv:2109.12683*. https://doi.org/10.48550/arXiv.2109.12683

Chen, Tianlong, Jonathan Frankle, Shiyu Chang, Sijia Liu, Yang Zhang, Zhangyang Wang, and Michael Carbin. 2020. The lottery ticket hypothesis for pre-trained BERT networks. In *Advances in Neural Information Processing Systems*, volume 33, pages 15834–15846.

Chi, Ethan A., John Hewitt, and Christopher D. Manning. 2020. Finding universal grammatical relations in multilingual BERT. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5564–5577. https://doi.org/10.18653/v1/2020.acl-main.493

Choenni, Rochelle and Ekaterina Shutova. 2022. Investigating language relationships in multilingual sentence encoders through the lens of linguistic typology. *Computational Linguistics*, 48(3):635–672. https://doi.org/10.1162/coli_a_00444

Choudhary, Chinmay. 2021. Improving the performance of UDify with linguistic typology knowledge. In *Proceedings of the*

Third Workshop on Computational Typology and Multilingual NLP, pages 38–60. https://doi.org/10.18653/v1/2021.sigtyp-1.5

Chu, Yoeng Jin. 1965. On the shortest arborescence of a directed graph. *Scientica Sinica*, 14:1396–1400.

Clark, Kevin, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT Look at? An analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286. https://doi.org/10.18653/v1/W19-4828

Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of NAACL-HLT*, pages 4171–4186.

Dozat, Timothy and Christopher D. Manning. 2016. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*. https://doi.org/10.48550/arXiv.1611.01734

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

Foroutan, Negar, Mohammadreza Banaei, Remi Lebret, Antoine Bosselut, and Karl Aberer. 2022. Discovering language-neutral sub-networks in multilingual language models. *arXiv preprint arXiv:2205.12672*.

Frankle, Jonathan and Michael Carbin. 2018. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*.

Glavaš, Goran and Ivan Vulić. 2021. Climbing the tower of treebanks: Improving low-resource dependency parsing via hierarchical source selection. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 4878–4888. https://doi.org/10.18653/v1/2021.findings-acl.431

Gu, Jiatao, Yong Wang, Yun Chen, Kyunghyun Cho, and Victor O. K. Li. 2020. Meta-learning for low-resource neural machine translation. In *2018 Conference on Empirical Methods in Natural Language Processing, EMNLP 2018*, pages 3622–3631. https://doi.org/10.18653/v1/D18-1398

Han, Song, Jeff Pool, John Tran, and William Dally. 2015. Learning both weights and connections for efficient neural network. In *Advances in Neural Information Processing Systems*, volume 28, pages 1135–1143.

Held, William and Diyi Yang. 2022. Shapley head pruning: Identifying and removing interference in multilingual transformers. *arXiv preprint arXiv:2210.05709*. https://doi.org/10.48550/arXiv.2210.05709

Hendy, Amr, Mohamed Abdelghaffar, Mohamed Afify, and Ahmed Y. Tawfik. 2022. Domain specific sub-network for multi-domain neural machine translation. In *Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing*, pages 351–356.

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

Htut, Phu Mon, Jason Phang, Shikha Bordia, and Samuel R. Bowman. 2019. Do attention heads in BERT track syntactic dependencies? *arXiv preprint arXiv:1911.12246*. https://doi.org/10.48550/arXiv.1911.12246

Jiao, Xiaoqi, Yichun Yin, Lifeng Shang, Xin Jiang, Xiao Chen, Linlin Li, Fang Wang, and Qun Liu. 2020. TinyBERT: Distilling BERT for natural language understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 4163–4174. https://doi.org/10.18653/v1/2020.findings-emnlp.372

Kingma, Diederik P. and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*.

Kondratyuk, Dan and Milan Straka. 2019. 75 languages, 1 model: Parsing universal dependencies universally. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2779–2795. https://doi.org/10.18653/v1/D19-1279

Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2019. ALBERT: A lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Langedijk, Anna, Verna Dankers, Phillip Lippe, Sander Bos, Bryan Cardenas

Guevara, Helen Yannakoudakis, and Ekaterina Shutova. 2022. Meta-learning for fast cross-lingual adaptation in dependency parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8503–8520. `https://doi.org/10.18653/v1/2022.acl-long.582`

Lauscher, Anne, Vinit Ravishankar, Ivan Vulić, and Goran Glavaš. 2020. From zero to hero: On the limitations of zero-shot language transfer with multilingual transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4483–4499. `https://doi.org/10.18653/v1/2020.emnlp-main.363`

Le, Hang, Juan Pino, Changhan Wang, Jiatao Gu, Didier Schwab, and Laurent Besacier. 2021. Lightweight adapter tuning for multilingual speech translation. In *The Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP 2021)*, volume 2, pages 817–824. `https://doi.org/10.18653/v1/2021.acl-short.103`

Lee, Seanie, Hae Beom Lee, Juho Lee, and Sung Ju Hwang. 2021. Sequential reptile: Inter-task gradient alignment for multilingual learning. In *International Conference on Learning Representations*.

Li, Hao, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016. Pruning filters for efficient convNets. *arXiv preprint arXiv:1608.08710*.

Li, Yanyang, Fuli Luo, Runxin Xu, Songfang Huang, Fei Huang, and Liwei Wang. 2022. Probing structured pruning on multilingual pre-trained models: Settings, algorithms, and efficiency. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1852–1865. `https://doi.org/10.18653/v1/2022.acl-long.130`

Lin, Zehui, Liwei Wu, Mingxuan Wang, and Lei Li. 2021. Learning language specific sub-network for multilingual machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 293–305. `https://doi.org/10.18653/v1/2021.acl-long.25`

Littell, Patrick, David R. Mortensen, Ke Lin, Katherine Kairis, Carlisle Turner, and Lori Levin. 2017. URIEL and Lang2Vec: Representing languages as typological, geographical, and phylogenetic vectors. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, volume 2, pages 8–14. `https://doi.org/10.18653/v1/E17-2002`

Lu, Yizhou, Mingkun Huang, Xinghua Qu, Pengfei Wei, and Zejun Ma. 2022. Language adaptive cross-lingual speech representation learning with sparse sharing sub-networks. In *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 6882–6886. `https://doi.org/10.1109/ICASSP43922.2022.9747671`

Michel, Paul, Omer Levy, and Graham Neubig. 2019. Are sixteen heads really better than one? In *Advances in Neural Information Processing Systems*, volume 32, pages 14037–14047.

Naseem, Tahira, Regina Barzilay, and Amir Globerson. 2012. Selective sharing for multilingual dependency parsing. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*, pages 629–637.

Nivre, Joakim, Marie-Catherine De Marneffe, Filip Ginter, Yoav Goldberg, Jan Hajic, Christopher D. Manning, Ryan McDonald, Slav Petrov, Sampo Pyysalo, Natalia Silveira, et al. 2016. Universal dependencies v1: A multilingual treebank collection. In *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16)*, pages 1659–1666.

Nooralahzadeh, Farhad, Ioannis Bekoulis, Johannes Bjerva, and Isabelle Augenstein. 2020. Zero-shot cross-lingual transfer with meta learning. In the *2020 Conference on Empirical Methods in Natural Language Processing*, pages 4547–4562. `https://doi.org/10.18653/v1/2020.emnlp-main.368`

Nooralahzadeh, Farhad and Rico Sennrich. 2022. Improving the cross-lingual generalisation in visual question answering. *arXiv preprint arXiv:2209.02982*. `https://doi.org/10.48550/arXiv.2209.02982`

Pfeiffer, Jonas, Ivan Vulić, Iryna Gurevych, and Sebastian Ruder. 2020. MAD-X: An adapter-based framework for multi-task cross-lingual transfer. In *Proceedings of the*

2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pages 7654–7673. https://doi.org/10.18653/v1/2020.emnlp-main.617

Pires, Telmo, Eva Schlinger, and Dan Garrette. 2019. How multilingual is multilingual BERT? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4996–5001. https://doi.org/10.18653/v1/P19-1493

Prasanna, Sai, Anna Rogers, and Anna Rumshisky. 2020. When BERT plays the lottery, all tickets are winning. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3208–3229. https://doi.org/10.18653/v1/2020.emnlp-main.259

Ruder, Sebastian. 2017. An overview of multi-task learning in deep neural networks. *arXiv preprint arXiv:1706.05098*. https://doi.org/10.48550/arXiv.1706.05098

Sanh, Victor, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*. https://doi.org/10.48550/arXiv.1910.01108

Sun, Tianxiang, Yunfan Shao, Xiaonan Li, Pengfei Liu, Hang Yan, Xipeng Qiu, and Xuanjing Huang. 2020. Learning sparse sharing architectures for multiple tasks. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 8936–8943. https://doi.org/10.1609/aaai.v34i05.6424

Tran, Ke M. and Arianna Bisazza. 2019. Zero-shot dependency parsing with pre-trained multilingual sentence representations. In *Proceedings of the 2nd Workshop on Deep Learning Approaches for Low-Resource NLP (DeepLo 2019)*, pages 281–288. https://doi.org/10.18653/v1/D19-6132

Turc, Iulia, Kenton Lee, Jacob Eisenstein, Ming-Wei Chang, and Kristina Toutanova. 2021. Revisiting the primacy of English in zero-shot cross-lingual transfer. *arXiv preprint arXiv:2106.16171*. https://doi.org/10.48550/arXiv.2106.16171

Üstün, Ahmet, Arianna Bisazza, Gosse Bouma, and Gertjan van Noord. 2020. UDapter: Language adaptation for truly universal dependency parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2302–2315. https://doi.org/10.18653/v1/2020.emnlp-main.180

van der Goot, Rob, Ahmet Üstün, Alan Ramponi, Ibrahim Sharaf, and Barbara Plank. 2021. Massive choice, ample tasks (MaChAmp): A toolkit for multi-task learning in NLP. In *16th Conference of the European Chapter of the Association for Computational Linguistics: System Demonstrations (EACL)*, pages 176–197. https://doi.org/10.18653/v1/2021.eacl-demos.22

Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30, pages 6000–6010.

Voita, Elena, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. In *57th Annual Meeting of the Association for Computational Linguistics*, pages 5797–5808. https://doi.org/10.18653/v1/P19-1580

Wang, Zirui, Zachary C. Lipton, and Yulia Tsvetkov. 2020. On negative interference in multilingual models: Findings and a meta-learning treatment. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4438–4450. https://doi.org/10.18653/v1/2020.emnlp-main.359

Wu, Qianhui, Zijia Lin, Guoxin Wang, Hui Chen, Börje F. Karlsson, Biqing Huang, and Chin-Yew Lin. 2020. Enhanced meta-learning for cross-lingual named entity recognition with minimal resources. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 9274–9281. https://doi.org/10.1609/aaai.v34i05.6466

Xu, Runxin, Fuli Luo, Baobao Chang, Songfang Huang, and Fei Huang. 2022. S4-Tuning: A simple cross-lingual sub-network tuning method. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 530–537. https://doi.org/10.18653/v1/2022.acl-short.58

Yang, Mu, Andros Tjandra, Chunxi Liu, David Zhang, Duc Le, John H. L. Hansen, and Ozlem Kalinli. 2022. Learning ASR pathways: A Sparse multilingual ASR model. *arXiv preprint arXiv:2209.05735*. https://doi.org/10.48550/arXiv.2209.05735

Yu, Tianhe, Saurabh Kumar, Abhishek Gupta, Sergey Levine, Karol Hausman, and Chelsea Finn. 2020. Gradient surgery for multi-task learning. In *Advances in Neural Information Processing Systems*, volume 33, pages 5824–5836.

Zeman, Daniel, Jan Hajic, Martin Popel, Martin Potthast, Milan Straka, Filip Ginter, Joakim Nivre, and Slav Petrov. 2018. CoNLL 2018 shared task: Multilingual parsing from raw text to universal dependencies. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual parsing from raw text to universal dependencies*, pages 1–21. https://doi.org/10.18653/v1/K18-2001

Zhang, Zhengyan, Fanchao Qi, Zhiyuan Liu, Qun Liu, and Maosong Sun. 2021. Know what you don't need: Single-shot meta-pruning for attention heads. In *AI Open*, volume 2, pages 36–42. https://doi.org/10.1016/j.aiopen.2021.05.003