# Examining Modularity in Multilingual LMs via Language-Specialized Subnetworks

**Rochelle Choenni**
University of Amsterdam
r.m.v.k.choenni@uva.nl

**Ekaterina Shutova**
University of Amsterdam
e.shutova@uva.nl

**Dan Garrette**
Google DeepMind
dhgarrette@google.com

## Abstract

Recent work has proposed explicitly inducing language-wise modularity in multilingual LMs via sparse fine-tuning (SFT) on per-language subnetworks as a means of better guiding cross-lingual sharing. In this paper, we investigate (1) the degree to which language-wise modularity *naturally* arises within models with no special modularity interventions, and (2) how cross-lingual sharing and interference differ between such models and those with explicit SFT-guided subnetwork modularity. In order to do so, we use XLM-R as our multilingual LM. Moreover, to quantify language specialization and cross-lingual interaction, we use a Training Data Attribution method that estimates the degree to which a model's predictions are influenced by in-language or cross-language training examples. Our results show that language-specialized subnetworks do naturally arise, and that SFT, rather than always increasing modularity, can decrease language specialization of subnetworks in favor of more cross-lingual sharing.

## 1 Introduction

Multilingual language models (LMs) can achieve remarkable performance across many languages thanks to phenomena like cross-lingual sharing (Pires et al., 2019), but they still suffer from the "curse of multilinguality" (Conneau et al., 2020) as performance can be hindered by negative cross-language interference (Wang et al., 2020). Recently, new methods have been proposed for mitigating these negative effects by training specialized model components for processing individual languages (Pfeiffer et al., 2022). These approaches, which add explicit **modularity** to the model, are also effective in promoting positive transfer and increasing interpretability (Pfeiffer et al., 2023).

While previous work has focused on developing techniques for explicitly adding modularity to models, we take a step back and ask: To what degree does language-wise modularity *naturally* arise within a model with no targeted modularity interventions? To investigate this question, we make use of a method inspired by the Lottery Ticket Hypothesis (Frankle and Carbin, 2018; Chen et al., 2020): for each language, we identify a **subnetwork**—a subset of model parameters—such that when fine-tuned on in-language data, it performs on par with the full model on that language (Wang et al., 2020; Nooralahzadeh et al., 2020). We then use these subnetworks to quantify language-wise modularity in a model by measuring the degree to which the subnetworks depend solely on in-language training examples when making predictions, which we refer to as *language specialization*. Subnetworks are an appealing method for our study because they do not require the introduction of additional model parameters, which means that we are able to use this approach on a model that has not been explicitly modified to add modularity.

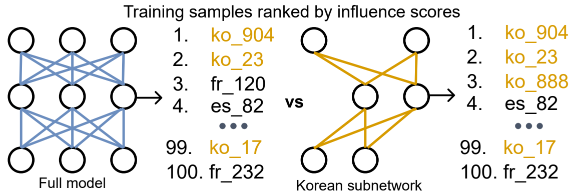



**Figure 1:** We study how in-language training data reliance changes for individual test languages when using a subnetwork compared to the full model at test time. For instance, will a Korean subnetwork rely more on Korean training samples when making a prediction for a Korean test sample? Note that each training example is denoted by its language and a training sample ID (`lang_ID`).

Moreover, subnetworks have also proven to be a popular modularization technique because when used to restrict parameter updates as a form of sparse fine-tuning (**SFT**), they are able to guide cross-lingual sharing toward positive transfer and away from negative interference (Lin et al., 2021; Lu et al., 2022; Xu et al., 2022; Choenni et al.,



*Findings of the Association for Computational Linguistics: NAACL 2024*, pages 287–301
June 16-21, 2024 ©2024 Association for Computational Linguistics

2023a; Hendy et al., 2022). However, less is known about precisely what effects SFT has on the underlying model behavior. Thus, we investigate the following set of questions for XLM-R (Conneau et al., 2020): (1) To what extent does language-wise modularity naturally arise within the model, when it is not explicitly enforced by restricting gradient updates? (2) How do cross-lingual sharing and interference differ between models without modularity interventions versus models with SFT-guided language-wise modularity? (3) How does the degree of language specialization affect model performance? and (4) To what extent does the similarity of language-specific subnetworks dictate cross-language influence?

To quantify cross-language interaction, we follow Choenni et al. (2023b) in using a Training Data Attribution (TDA) method, TracIn (Pruthi et al., 2020), which measures the degree of influence each training example has on a particular model prediction. By examining the influence each language's training set has on the test predictions for individual languages, we can estimate how much influence languages on average exert cross-lingually.

We conduct experiments on three text classification tasks—natural language inference, paraphrasing, and sentiment analysis. For each task, even without special modularity interventions, we are able to identify subnetworks that rely more heavily on in-language data than the full model does. Additionally, we find that SFT does not always increase this modularity, but instead can decrease language specialization within the subnetworks and boost cross-lingual sharing to improve performance. Finally, we provide additional analysis on factors that affect cross-language influence, and find interesting correlations between subnetwork similarity and the amount of positive influence across languages.

## 2 Background and Related work

### 2.1 Modular deep learning

Modular approaches existed before the rise of pre-trained LMs (Shazeer et al., 2016; Andreas et al., 2016), but have recently regained popularity in NLP. The idea is that modular systems will allow us to improve performance in an interpretable way as modularity provides a more intuitive path to compositionality. Various methods have been proposed to implement specialized modules, for instance, by inserting adapter layers into the model (Rebuffi et al., 2017, 2018; Houlsby et al., 2019; Pfeiffer et al., 2022), replacing fine-tuning by prefix-tuning (Li and Liang, 2021), or by SFT with subnetworks (Sun et al., 2020). While the former two aim to create modularity *post-hoc* by injecting task-specific parameters into the existing model, the latter approach aims to induce it into the model as an inductive bias during fine-tuning. In this work, we delve deeper into the effects of SFT to understand whether it is able to produce more modular systems. While some work studies modularity in both vision and language models (Csordás et al., 2020; Zhang et al., 2023; Lepori et al., 2023; Dobs et al., 2022), we are the first to explicitly examine the degree of modularity in multilingual LMs, and to study subnetwork interaction by directly looking at the training data.

### 2.2 Subnetworks and SFT

Frankle and Carbin (2018) showed that subnetworks can be found through pruning methods (Han et al., 2015; Li et al., 2016) that match the performance of the full model. Since then, it has been shown that such subnetworks exist within BERT models (Prasanna et al., 2020; Budhraja et al., 2021; Li et al., 2022), and that both language-neutral and language-specific subnetworks can be found in multilingual LMs (Foroutan et al., 2022). Hence, sparse training gained popularity in multilingual NLP: Nooralahzadeh and Sennrich (2023) show that training *task-specific* subnetworks can help in cross-lingual transfer, Lin et al. (2021) use *language-pair-specific* subnetworks for neural machine translation, and Hendy et al. (2022) use *domain-specific* subnetworks. Finally, Wang et al. (2020); Lu et al. (2022); Choenni et al. (2023a); Xu et al. (2022) use language-specific subnetworks to improve cross-lingual performance on a range of tasks, e.g. speech recognition, dependency parsing and natural language understanding, suggesting that sparse training can reduce negative interference and/or stimulate positive knowledge transfer. While Choenni et al. (2023a) found evidence of the former through fewer gradient conflicts during training (Yu et al., 2020), we are the first to study the effect of SFT on cross-lingual data sharing.

### 2.3 Training Data Attribution

TDA methods aim to identify a set of training examples that most informed a particular test prediction. Typically, the influence of training point $z_{train}$ on test point $z_{test}$ is formalized as the change in the loss that would be observed for $z_{test}$ if sample $z_{train}$ was omitted during training (Koh and Liang, 2017). Thus, we can use it as a measure of how important $z_{train}$ is for making a prediction for $z_{test}$. TDA methods have been used in NLP for unveiling data artifacts (Han and Tsvetkov, 2022), e.g., to detect outlier data (Han et al., 2020), enable instance-specific data filtering (Lam et al., 2022), or to correct erroneous model predictions (Meng et al., 2020; Guo et al., 2021). Following Choenni et al. (2023b), we instead employ it to study cross-lingual data sharing in LMs. To understand how much influence languages exert cross-lingually, Choenni et al. (2023b) quantify cross-language influence during multilingual fine-tuning by the percentage that each language's training data contributes to the most influential training samples for each test language. While they study the effects of full model fine-tuning, we employ their framework to study modularity in LMs by testing the data reliance behavior of language-specific subnetworks and the effect that SFT has on this.

## 3 Methods

### 3.1 Identifying Subnetworks

Subnetworks are represented by masks that can be applied to the model to ensure that only a subset of the model's parameters are activated (or updated during training). We follow Prasanna et al. (2020) in using *structured* masks, treating entire attention heads as units which are always fully enabled or disabled. Thus, for a language $\ell$, its subnetwork is implemented as a binary mask $\xi_\ell \in \{0,1\}^{H \times L}$, where $H$ and $L$ correspond to the number of attention heads and layers. We aim to find masks for languages that prune away as many heads as possible without harming model performance on a given language (i.e., by pruning away heads that are only used by other languages, or that are unrelated to the task). For this, we apply the procedure introduced by Michel et al. (2019). Starting from a model that is fine-tuned for a task in language $\ell$, we iterate by repeatedly removing the 10% of heads with the lowest importance scores $\text{HI}_\ell^{(i,j)}$ ($i$=head, $j$=layer), which is estimated based on the expected sensitivity of the model to the mask variable $\xi_\ell^{(i,j)}$:

$$\text{HI}_\ell^{(i,j)} = \mathbb{E}_{x_\ell \sim X_\ell} \left| \frac{\partial \mathcal{L}(x_\ell)}{\partial \xi_\ell^{(i,j)}} \right| \quad (1)$$

where $X_\ell$ is $\ell$'s data distribution, $x_\ell$ is a sample from that distribution, and $\mathcal{L}(x_\ell)$ is the loss with respect to the sample. Pruning stops when we reach 95% of the original model performance.

### 3.2 TracIn: Tracing Influence

Pruthi et al. (2020) propose TracIn, a simple TDA method to approximate influence of a training sample over training. They do this by computing the influence of a training sample $z_i$ on the prediction for a test sample $z_{test}$ as follows:

$$\mathcal{I}(z_i, z_{test}) = \sum_{e=1}^{E} \nabla_\theta \mathcal{L}(z_i, \theta_e) \cdot \nabla_\theta \mathcal{L}(z_{test}, \theta_e) \quad (2)$$

where $\theta_e$ is the checkpoint of the model at each training epoch. The intuition behind this method is to approximate the total reduction in the test loss $\mathcal{L}(z_{test}, \theta)$ during the training process when the training sample $z_i$ is used. This gradient product method reduces the problem to the dot product between the gradient of the training loss and the gradient of the test loss. As dominating gradients are a known problem in multilingual NLP (Wang et al., 2020), we also adopt the simple normalization trick from Barshan et al. (2020), i.e., substituting the dot product operation with cosine similarity, thus normalizing by the norm of the training gradients. Lastly, following Pruthi et al. (2020), we reduce computational costs by pre-computing low-memory sketches of the loss gradients of the training points using random projections, and reuse them to compute randomized unbiased estimators of the influence on different test points (Woodruff et al., 2014). See Appendix A for more details.

## 4 Experimental setup

### 4.1 Tasks and datasets

**Natural language inference** The Cross-Lingual Natural Language Inference (XNLI) dataset (Conneau et al., 2018) contains premise-hypothesis pairs labeled with their relationship: 'entailment', 'neutral' or 'contradiction'. The dataset contains parallel data of which the original pairs come from English and were translated to other languages. We use English, French, German, Russian and Spanish portions of the dataset.

**Paraphrasing** Cross-Lingual Paraphrase Adversaries from Word Scrambling (PAWS-X) (Yang et al., 2019) requires the model to decide if two sentences are paraphrases of one another. PAWS-X contains translated data from PAWS (Zhang et al., 2019). Part of the development and test sets was translated from English by professionals and the training data was translated automatically. We experiment with English, French, German, Korean and Spanish.

**Sentiment analysis** The Multilingual Amazon Review Corpus (MARC) (Keung et al., 2020) contains Amazon reviews written by users in various languages. Each record in the dataset contains the review text and title, and a star rating. The corpus is balanced across 5 star rating, so that each star rating constitutes 20% of the reviews in each language. Note that this is a non-parallel dataset. We experiment with Chinese, English, French, German and Spanish.

### 4.2 Training techniques

**Full model fine-tuning** We fine-tune the full XLM-R model (Conneau et al., 2020) on the concatenation of 2K samples from 5 languages, i.e. 10K samples for each task. As computational costs of TracIn increase with training size, we use a minimal required number of training examples to obtaining reasonably high performance. Thus, we simplify the task to get a better trade-off between the number of training examples and performance. For XNLI, we follow Han et al. (2020) by performing binary classification "entailment or not" ; for MARC, we collapse 1 and 2 stars into a negative and 4 and 5 stars into a positive review category. Training converges at epoch 4 for XNLI, and at epoch 5 for PAWS-X and MARC, obtaining 78%, 83%, and 90% accuracy on their development sets respectively, for more details see Appendix B.

**Sparse fine-tuning (SFT)** We sample language-specific batches in random order, and each time restrict parameter updates to only those parameters that are enabled within the respective language's identified subnetwork. We use the subnetworks during fine-tuning by restricting the model both in the forward and backward pass.[1] We ensure that we sample each language equally often. All other fine-tuning details remain the same as for full model fine-tuning.

### 4.3 Evaluation

**Computing influence scores** We use 500 random test samples from each language and compute influence scores between each test sample and all 10K training instances. For each test sample, we retrieve the top $m$=100 training instances with the largest *positive* and the largest *negative* influence scores and refer to them as the set of most positively and negatively influential samples respectively. Note that we use $m$=100 as it was previously found to be optimal on the exact same tasks (Choenni et al., 2023b).[2] Moreover, negative cosine similarity between gradients have been referred to as gradient conflicts (Yu et al., 2020), and were shown to be indicative of negative interference in the multilingual setting (Wang et al., 2020)[3]. In addition, we ensure that the model was able to predict the correct label for all test instances that we compute influence scores for such that we only study the training samples that influenced the model to make a correct prediction. Also, as we train on parallel data for XNLI and PAWS-X, the content in our training data is identical across languages, giving each language an equal opportunity to be retrieved amongst the most influential samples.

**Quantifying cross-language influence** After obtaining an influence score ranking over our training set for each test sample, we compute how much each training language contributed to the prediction for the test samples in other languages. We then compare the resulting rankings produced using the full model and an identified subnetwork, see Figure 1. As there can be small differences in performance between the subnetworks and the full model, throughout all experiments, we compare cross-language influence for test samples that both models were able to correctly classify.

## 5 Naturally arising modularity

In this section, we study whether modularity has naturally arisen within a model after multilingual full model fine-tuning. As such, the subnetworks are only applied at test time.

### 5.1 How specialized are subnetworks?

To study the degree to which modularity has naturally arisen after full model fine-tuning, we look for subnetworks that naturally specialize in their

[1] We implement this during backpropagation by multiplying the gradients by the binary subnetwork mask, and passing the masked gradients to the optimizer. In the forward pass, we simply disable the attention heads.

[2] Note that we carefully follow the experimental set-up from Choenni et al. (2023b), i.e., we use the same tasks, data and model for our experiments.

[3] When gradients point in opposite directions, the model will update in a suboptimal direction for both examples, hence resulting in negative interference.

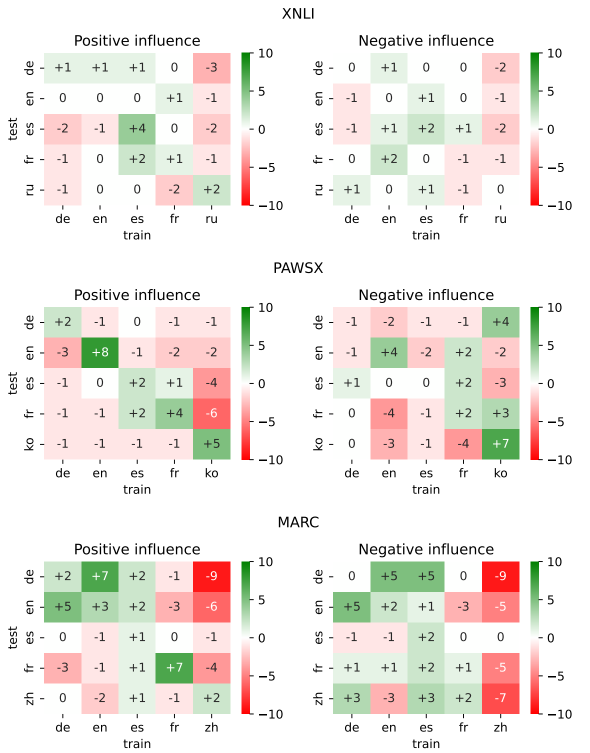



**Figure 2: (After full model fine-tuning)** The effect of using the identified language-specific subnetwork for each test language compared to the full model at test time. On the $x$-axis we have the training language and on the $y$-axis the test language. The values denote the change (%) in influence from the training on the test language. Results are averaged over all 500 test samples per language.

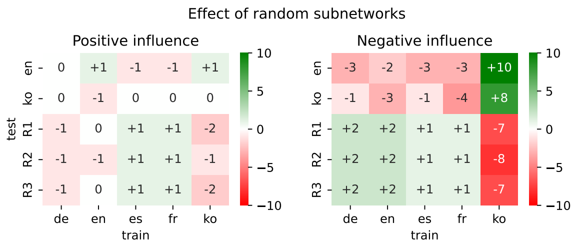



**Figure 3: (After full model fine-tuning)** The effect on cross-language influence when using random (R) and suboptimal (English and Korean) subnetworks on German as a test language for PAWS-X.

respective languages. We quantify language specialization as the extent to which the subnetworks rely solely on in-language training data when making test-time predictions. Thus, for each test language, we use the pruning procedure explained in Section 3.1 to identify a subnetwork within the fine-tuned model. We then compute influence scores on the fine-tuned model, applying the subnetwork mask corresponding to the language of the test example. Finally, we compare the model's reliance on in-language data when using these subnetworks against its reliance when no subnetwork mask is applied (i.e. when predicting with the full model).

**Results** In Figure 2 we show, per task and test language, the change in contribution (%) to the top 100 most positively and negatively influential samples when using the subnetworks compared to the full model. On the diagonals, we clearly see that for all languages across all tasks, using the subnetwork does mostly result in more positive influence from the respective language (from +1 to +8%). This indicates that we are able to identify language-specialized subnetworks that are more biased toward relying on in-language data, and thus suggests that some form of modularity naturally exists within the model. For baseline results from the full model and more details on the subnetworks, see Appendices C and D respectively. Also, importantly, our results using 500 test samples per language on the full model are similar to those on the same tasks from Choenni et al. (2023b), who performed extensive analysis on the quality of the influence scores.

The effects are less clear when looking at negative influence; here we see that using a language's subnetwork can also decrease negative influence coming from in-language data (e.g. Chinese for MARC). Finally, results from XNLI are overall weaker than for the other tasks. This is in line with results from the full model that showed that, for XNLI, the model relies to the least extent on in-language data, hence we can expect language-specificity to be less strong for these subnetworks. Moreover, for English, we find no difference in language specialization. This can be explained by the fact that the German and Russian subnetworks share 100% of their capacity with English, making its subnetwork less distinct (see Appendix D).

**Cross-language influence** We have shown that language-specialized subnetworks rise. We now analyze how cross-language influence differs within such subnetworks compared to the full model. For MARC, we see that the increase in positive self-influence (diagonal) can be smaller than the increase in positive influence from related languages. In particular, we see that using a German subnetwork strongly increases positive influence from the most typologically similar training language, i.e., English (+7%), and vice versa (+5%). While the change in positive influence from related languages is stronger than that of the respective subnetwork's language, the subnetwork still relies more on in-

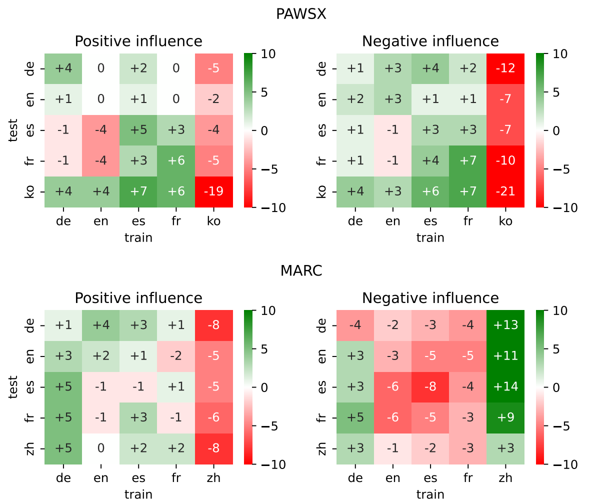



**Figure 4: (After SFT)** The effect of using the identified language-specific subnetwork for each test language compared to using the full model at test time. On the $x$-axis we have the training language and on the $y$-axis the test language. The values denote the change (%) in influence from the training on the test language. Results are averaged over all 500 test samples per language

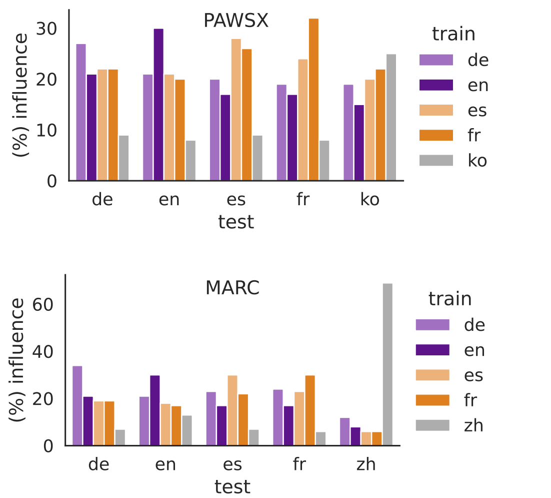



**Figure 5:** The positive influence (%) from each training language on each test language in absolute numbers. The values are retrieved from the subnetworks after SFT. Note that the $y$-axes are not on the same scale.

language data when looking at absolute numbers. For German, the full model was relying for 33% on in-language data, which using its subnetwork increased to 35% (+2%). Yet, English initially only contributed 17% to German, which after using its subnetwork increased to 24% (+7%) (see Appendix C for the full model results). We suspect that we observe the effect of positive knowledge transfer through cross-lingual sharing here. Similar to the full model, when subnetworks have exploited most useful in-language data, they start benefitting more from exploiting other languages' data instead.

### 5.2 Random and suboptimal subnetworks

As baselines to our identified subnetworks, we study whether evidence for language specialization can also be found within random and suboptimal subnetworks for PAWS-X. *Random*: we shuffle the binary subnetwork masks with 3 random seeds, and recompute scores from them. Note that we do this only for German—we saw the weakest increase in language-specificity for German (+2%, see Figure 2), thus it should be the easiest to get similar results from a random subnetwork. *Suboptimal*: we pick the subnetwork from the most similar and distant language to German, i.e., English and Korean, and recompute influence scores for German (i.e., testing the effect of applying the subnetwork from a language $A$ to a language's $B$'s input.).

**Results** In Figure 3, we find that using random subnetworks overall causes little change to the score distributions as compared to the full model. In particular, we find that in none of the cases the influence of German increases. Also, it is evident that the behavior from the suboptimal subnetworks is different from the random subnetworks. For instance, we find that using either the correct English or Korean subnetworks result in a strong increase of negative interference from Korean (+10 and 8%). Yet, when we use the random subnetworks we instead observe a strong tendency for Korean to decrease in negative influence. These results show that our identified subnetworks encode meaningful differences compared to randomly selected ones.

## 6 How does SFT affect modularity?

In Section 5, we studied whether modularity had naturally arisen in the model in the form of language-specialized subnetworks. We now study the effect that SFT has on these subnetworks, i.e., does it further encourage modularity within the model? Thus, instead of only applying the subnetworks at test time, as was done in the previous section, we now use the same identified subnetworks, but apply them both during SFT and at test time. We then recompute influence scores between test and training samples, and observe the change in language specialization compared to full model fine-tuning. This way, we test whether SFT, compared to full model fine-tuning, causes the subnetworks to further specialize on in-language data.

Given that the subnetworks found for XNLI had the smallest effect on cross-language data reliance, and we did not find a distinct English subnetwork, we conduct further experiments on PAWS-X and MARC (that contain parallel and non-parallel data respectively) to reduce computational costs. Also, we confirm that SFT improves performance on both tasks (see Appendix E). For PAWS-X, we obtain an average test accuracy of 74.8% when using subnetworks after full model fine-tuning and 78.4% after SFT (+3.6%). For MARC we see an average improvement of +1.2% when using SFT.

**Results** In Figure 4, we see the change in language influence compared to using the full model. We find that the in-language data reliance of some subnetworks tends to decrease after SFT (i.e., Korean for PAWS-X and Chinese, French, and Spanish for MARC). This is surprising given that SFT is generally seen as a modularization technique. Whilst it is important to note that all subnetworks still mostly rely on in-language data as shown by the absolute numbers reported in Figure 5, our results suggest that the benefit of SFT cannot fully be attributed to language specialization of the subnetworks. Instead, cross-lingual sharing, guided through subnetwork interaction, is likely a contributing factor as well. Finally, as our results suggest that SFT does not necessarily strengthen language specialization, it sheds doubt on SFT as a method for creating more modular systems.

### 6.1 SFT with random subnetworks

As a baseline to our previous findings, we now test whether any randomly found subnetwork could in principle be taught to specialize in a language when we use SFT as a training method. Thus, for each language, we shuffle the language-specific subnetworks to obtain a random subnetworks with the same sparsity level. We then use these random subnetworks, both during SFT and at test time, and repeat the procedure from Section 6.

**Results** Surprisingly, in Figure 6, we see that random subnetworks to a much larger extent rely on in-language data than the identified subnetworks used in Section 6. In particular, we see that the model barely relies on cross-lingual sharing for English (+64% compared to the full model, which results in 97% reliance on English data when using the subnetwork). Yet, we also find that these highly specialized subnetworks perform considerably worse, on average only obtaining ±56%

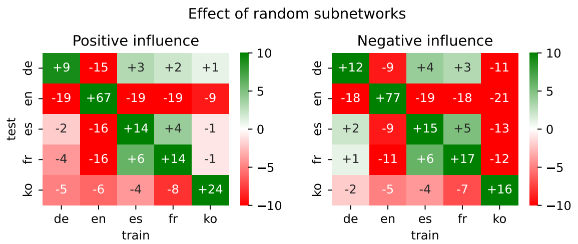



**Figure 6: (After SFT)** The effect that SFT with *random* subnetworks has for PAWS-X on the amount of language specialization that the subnetworks acquire compared to full model fine-tuning.

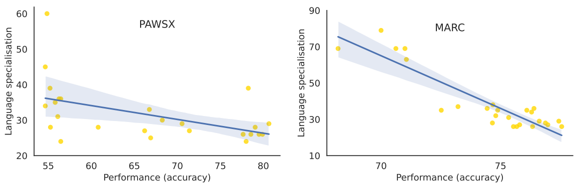



**Figure 7:** The correlation between language specialization and performance accuracy for PAWS-X and MARC. We compute scores for all languages and model checkpoints.

across languages. Given that random subnetworks do not contain the necessary information to process the language, we hypothesize that (1) during SFT they need to learn both the task and language, which causes them to focus on in-language data first, and (2) cross-lingual sharing will only happen once the in-language data has been fully exploited. Our results show that any subnetwork can in principle learn to specialize in one language, but that this might be suboptimal.

## 7 Further analysis

In Section 6, we show that SFT only sometimes causes our identified subnetworks to rely more on in-language data, yet unlike random subnetworks, do seem to encode meaningful information. To understand where the performance improvements from SFT come from, we perform further analysis on how language specialization correlates with performance, and how subnetwork similarity affects cross-language influence.

### 7.1 Correlation between language specialization and performance

We find that SFT only decreases performance on French for PAWS-X (Table 2, Appendix E), which happens to also be the subnetwork that showed the strongest increase in language specialization after SFT (+6%) in Section 6. To test to what degree subnetwork performance benefits from language

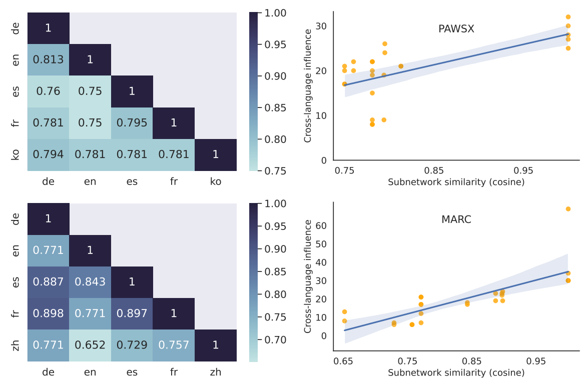



**Figure 8:** (**Left**) The cosine similarity between the flattened binary subnetwork masks for each language pair. (**Right**) Positive cross-language influence as a function of structural (cosine) similarity between subnetworks.

specialization, we study the correlation between the two using data from all model checkpoints.

**Results** In Figure 7, we see that, for both tasks, stronger language specialization is negatively correlated with model performance. This finding further supports our hypothesis that the strength of SFT really comes from cross-lingual sharing that happens between the subnetworks rather than from the language specialization of the subnetworks themselves. Intuitively, this makes sense as SFT forces the model to squeeze information into the smaller subsets of model parameters, which has to improve performance on a set of training languages, and as such, requires better cross-lingual sharing.

### 7.2 Correlation between subnetwork similarity and cross-language influence

SFT allows for cross-lingual interaction through subnetwork overlap in which the model parameters are shared between languages. This sharing mechanism is motivated by the idea that similar languages are encoded by similar subnetworks (and thus naturally dictating cross-lingual sharing by their overlap). To test this hypothesis we study the correlation between subnetwork similarity and cross-language influence between language pairs. We measure similarity by the cosine similarity between the flattened binary subnetwork masks.

**Results** In Figure 8 (Left) we report the cosine similarity between the subnetworks of each language pair and (Right) the correlation between such subnetwork similarity and positive cross-language influence (in absolute numbers). From this, we find that for both tasks, subnetwork similarity is positively correlated with positive cross-language influence. Yet, we did not find a strong correlation between negative cross-language influence and subnetwork overlap. This is a promising finding, as it suggests that positive and negative influence do not necessarily have to go hand-in-hand. Thus, future work should investigate how we can further exploit subnetwork overlap to increase positive influence without increasing negative influence as well. Moreover, it is evident that for MARC the subnetworks show on average more overlap than for PAWS-X. Thus as the capacity within subnetworks from MARC have to be shared with more languages, it can explain why their language specialization is less strong as seen in Figure 4. Future work should test whether SFT is still effective when using many more training languages (in which case subnetwork overlap will inevitably be higher).

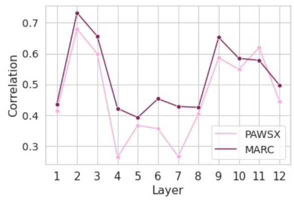



**Figure 9:** The correlation between positive cross-language influence and the subnetwork similarity computed based on individual model layers.

**Layer-wise analysis** To further analyze how subnetwork similarity affects cross-language influence, we now test how layer-wise subnetwork similarity correlates with performance. In Figure 9, we see that similarity between certain layers is much more indicative of cross-language influence, and moreover, that both tasks follow very similar patterns despite ending up with vastly different subnetworks. This suggests that while language-specific subnetworks are also task-specific, there may be general language-specific properties across task-specific subnetworks that we can identify and exploit to better guide cross-lingual sharing.

## 8 Conclusion

We studied to what degree modularity, in the form of language-specialized subnetworks, naturally arises within multilingual LMs. We demonstrate the existence of such subnetworks using TracIn to monitor the change in reliance on in-language data at test time when using subnetworks compared to the full model. Moreover, we studied the effects that SFT has on modularization, and find that it does not cause all subnetworks to become more specialized. Yet, in all cases, our identified subnetworks show vastly different behavior from random ones, indicating that we are able to uncover meaningful language-specific model behavior. Finally, we find that subnetwork similarity, particularly in specific model layers, correlates with positive, but not negative, cross-language influence. Future work should focus on further exploiting subnetworks and their interaction to better control cross-lingual sharing.

## 9 Limitations

One limitation of TDA methods in general is that the experiments are computationally expensive to run. While using the random projection method, explained in Appendix A, somewhat mitigates the problem, it still prevents us from studying a wider range of LMs and/or larger models. Similarly, due to the computational costs, we are restricted to relatively easy tasks as (1) we can not use a large fine-tuning dataset and (2) TracIn operates on the sequence-level, i.e., it estimates how much a full training instance contributed to a prediction, making this method mostly suitable for classification and regression tasks. Given that the tasks are relatively simple, this might also limit the benefit of SFT over full model fine-tuning, hence the subnetwork behavior we see after SFT might be weaker than if we had studied more complicated tasks and/or tasks that generally require more language-specific information (e.g., masked language modelling or dependency parsing).

## Acknowledgements

We would like to thank Ian Tenney, Cindy Wang, and Tolga Bolukbasi for their feedback. This project was in part supported by a Google PhD Fellowship for the first author.

## References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Learning to Compose Neural Networks for Question Answering. In *Proceedings of NAACL-HLT*, pages 1545–1554.

Elnaz Barshan, Marc-Etienne Brunet, and Gintare Karolina Dziugaite. 2020. Relatif: Identifying explanatory training samples via relative influence. In *International Conference on Artificial Intelligence and Statistics*, pages 1899–1909. PMLR.

Aakriti Budhraja, Madhura Pande, Pratyush Kumar, and Mitesh M Khapra. 2021. On the Prunability of Attention Heads in Multilingual BERT. *arXiv preprint arXiv:2109.12683*.

Tianlong Chen, 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.

Rochelle Choenni, Dan Garrette, and Ekaterina Shutova. 2023a. Cross-Lingual Transfer with Language-Specific Subnetworks for Low-resource Dependency Parsing. *Computational Linguistics*.

Rochelle Choenni, Dan Garrette, and Ekaterina Shutova. 2023b. How do languages influence each other? Studying cross-lingual data sharing during LLM fine-tuning. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Édouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2020. Unsupervised Cross-lingual Representation Learning at Scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451.

Alexis Conneau, Ruty Rinott, Guillaume Lample, Adina Williams, Samuel Bowman, Holger Schwenk, and Veselin Stoyanov. 2018. XNLI: Evaluating Cross-lingual Sentence Representations. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2475–2485.

Róbert Csordás, Sjoerd van Steenkiste, and Jürgen Schmidhuber. 2020. Are Neural Nets Modular? Inspecting Functional Modularity Through Differentiable Weight Masks. In *International Conference on Learning Representations*.

Katharina Dobs, Julio Martinez, Alexander JE Kell, and Nancy Kanwisher. 2022. Brain-like functional specialization emerges spontaneously in deep neural networks. *Science advances*, 8(11):eabl8913.

Negar Foroutan, Mohammadreza Banaei, Rémi Lebret, Antoine Bosselut, and Karl Aberer. 2022. Discovering Language-neutral Sub-networks in Multilingual Language Models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 7560–7575.

Jonathan Frankle and Michael Carbin. 2018. The Lottery Ticket Hypothesis: Finding Sparse, Trainable Neural Networks. In *International Conference on Learning Representations*.

Han Guo, Nazneen Rajani, Peter Hase, Mohit Bansal, and Caiming Xiong. 2021. FastIF: Scalable Influence Functions for Efficient Model Interpretation and Debugging. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10333–10350.

Song Han, 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.

Xiaochuang Han and Yulia Tsvetkov. 2022. ORCA: Interpreting Prompted Language Models via Locating Supporting Data Evidence in the Ocean of Pretraining Data. *arXiv preprint arXiv:2205.12600*.

Xiaochuang Han, Byron C Wallace, and Yulia Tsvetkov. 2020. Explaining Black Box Predictions and Unveiling Data Artifacts through Influence Functions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5553–5563.

Amr Hendy, Mohamed Abdelghaffar, Mohamed Afify, and Ahmed Y Tawfik. 2022. Domain Specific Subnetwork 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.

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

Phillip Keung, Yichao Lu, György Szarvas Szarvas, and Noah A Smith. 2020. The Multilingual Amazon Reviews Corpus. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Pang Wei Koh and Percy Liang. 2017. Understanding Black-box Predictions via Influence Functions. In *International conference on machine learning*, pages 1885–1894. PMLR.

Tsz Kin Lam, Eva Hasler, and Felix Hieber. 2022. Analyzing the Use of Influence Functions for Instance-Specific Data Filtering in Neural Machine Translation. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 295–309.

Michael A Lepori, Thomas Serre, and Ellie Pavlick. 2023. Break it down: evidence for structural compositionality in neural networks. *arXiv preprint arXiv:2301.10884*.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. 2016. Pruning Filters for Efficient ConvNets. In *International Conference on Learning Representations*.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. 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 4582–4597.

Yanyang Li, 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.

Zehui Lin, 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.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled Weight Decay Regularization. In *International Conference on Learning Representations*.

Yizhou Lu, 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. IEEE.

Yuxian Meng, Chun Fan, Zijun Sun, Eduard Hovy, Fei Wu, and Jiwei Li. 2020. Pair the Dots: Jointly Examining Training History and Test Stimuli for Model Interpretability. *arXiv preprint arXiv:2010.06943*.

Paul Michel, 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.

Farhad Nooralahzadeh, 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. Association for Computational Linguistics.

Farhad Nooralahzadeh and Rico Sennrich. 2023. Improving the cross-lingual generalisation in visual question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 13419–13427.

Jonas Pfeiffer, Naman Goyal, Xi Lin, Xian Li, James Cross, Sebastian Riedel, and Mikel Artetxe. 2022. Lifting the Curse of Multilinguality by Pre-training Modular Transformers. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3479–3495.

Jonas Pfeiffer, Sebastian Ruder, Ivan Vulić, and Edoardo Maria Ponti. 2023. Modular deep learning. *arXiv preprint arXiv:2302.11529*.

Telmo Pires, 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.

Sai Prasanna, 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)*.

Garima Pruthi, Frederick Liu, Satyen Kale, and Mukund Sundararajan. 2020. Estimating training data influence by tracing gradient descent. *Advances in Neural Information Processing Systems*, 33:19920–19930.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning Multiple Visual Domains with Residual Adapters. *Advances in neural information processing systems*, 30.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2018. Efficient Parametrization of Multi-Domain Deep Neural Networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8119–8127.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2016. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. In *International Conference on Learning Representations*.

Tianxiang Sun, 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.

Ian Tenney, Dipanjan Das, and Ellie Pavlick. 2019. BERT Rediscovers the Classical NLP Pipeline. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4593–4601.

Zirui Wang, 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.

David P Woodruff et al. 2014. Sketching as a tool for numerical linear algebra. *Foundations and Trends® in Theoretical Computer Science*, 10(1–2):1–157.

Runxin Xu, Fuli Luo, Baobao Chang, Songfang Huang, and Fei Huang. 2022. S4-Tuning: A Simple Cross-lingual Sub-network Tuning Method-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.

Yinfei Yang, Yuan Zhang, Chris Tar, and Jason Baldridge. 2019. PAWS-X: A Cross-lingual Adversarial Dataset for Paraphrase Identification. 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 3687–3692.

Tianhe Yu, 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.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase Adversaries from Word Scrambling. 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 1298–1308.

Zhengyan Zhang, Zhiyuan Zeng, Yankai Lin, Chaojun Xiao, Xiaozhi Wang, Xu Han, Zhiyuan Liu, Ruobing Xie, Maosong Sun, and Jie Zhou. 2023. Emergent Modularity in Pre-trained Transformers. *Findings of ACL*, pages arXiv–2305.

# A Low-memory sketches using random projections

LMs have a large number of parameters which makes the inner product computations in the first-order approximation of the influence expensive, especially when computing influence scores for a large number of test points. Thus, following Pruthi et al. (2020), we speed up the computations by using random projections, a method that allows us to pre-compute low-memory sketches of the loss gradients of the training points (Woodruff et al., 2014) which can be stored and re-used to compute randomized unbiased estimators of the influence on different test points. To do so, we choose a random matrix $G \in \mathcal{R}^{d \times p}$, where $d \ll p$ is a user-defined dimension for the random projections, whose entries are sampled i.i.d. from $\mathcal{N}(0, \frac{1}{d})$ such that $E[G^T G] = \mathcal{I}$. Similarly, for the fully connected layers with a weight matrix $W \in \mathcal{R}^{m \times n}$, it is also possible to obtain a random projection of the gradient with respect to $W$ into $d$ dimensions. To do so, we use two independently chosen random projection matrices $G1 \in \mathcal{R}^{\sqrt{d} \times m}$ and $G2 \in \mathcal{R}^{\sqrt{d} \times n}$, where $E[G_1 G_1^T] = E[G_2 G_2^T] = I$, and compute:

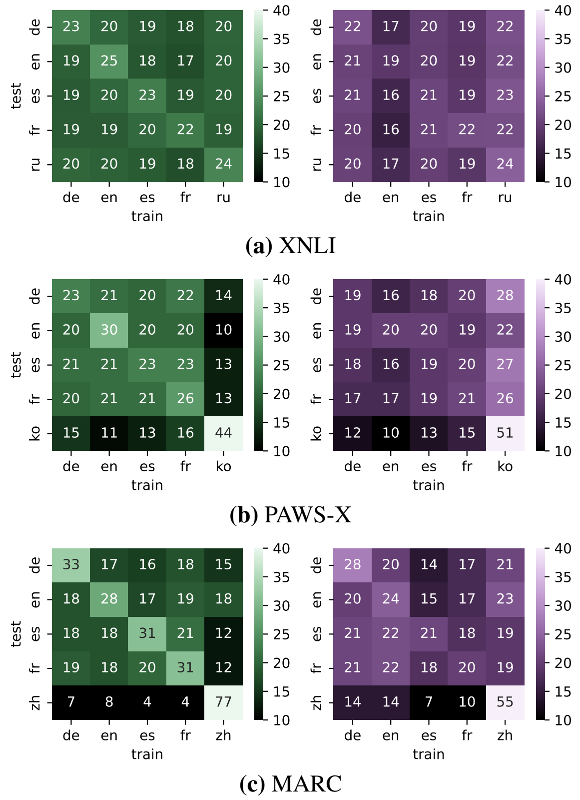
$$G_1 \nabla_y f(y) x^T G_2^T \in \mathcal{R}^{\sqrt{d} \times \sqrt{d}} \tag{3}$$

, which can be flattened into a $d$-dimensional vector. See Appendix E and F from Pruthi et al. (2020) for more details. Note that throughout our experiments we set $d = 256$.

# B Fine-tuning details

For each task, we add a simple classifier on top of the pretrained XLM-R base model (Conneau et al., 2020). The classifier consists of one hidden layer and uses `tanh` activation. We then feed the hidden representation corresponding to the <S> token for each input sequence to the classifier for prediction. Moreover, following Choenni et al. (2023b), we use AdamW (Loshchilov and Hutter, 2017) as an optimizer, and use learning rates of 2e-5, 9e-6, and 2e-5 for XNLI, PAWS-X and MARC respectively.

# C Baseline results




(a) XNLI

(b) PAWS-X

(c) MARC

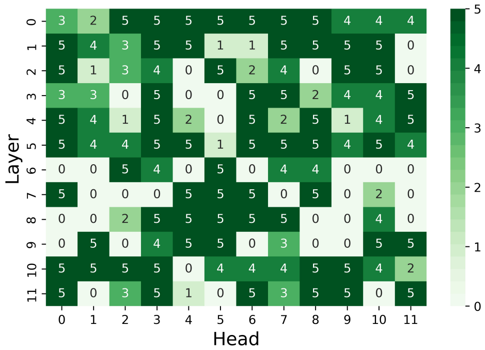
**Figure 10:** Percentage that each training language contributes to the top 100 training samples for each test language when using the full model. Results are averaged over all 500 test samples per language.

# D Details on the identified subnetworks

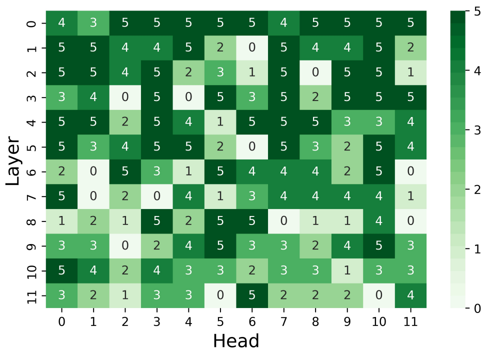
In Figure 11, we show the overlap in attention heads of the identified subnetworks that we found for each of our 5 training languages. While we find that all subnetworks have similar sparsity levels (see Table 1 for the absolute number of disabled attention heads per task and language), we also see that across all tasks, some heads are not used by any of the languages (indicated by 0). This finding suggests that the model capacity does not have to be a limiting factor within this model, as more language-specific parameters could be assigned if needed. In contrast, many heads, especially in the lower layers of the models for PAWS-X and in the higher layers for XNLI, are fully shared across all languages. Given that paraphrasing relies more on lower-level syntactic information than NLI, this is in line with previous findings that suggest that syntax is processed in lower layers while semantics in processed in the higher ones (Tenney et al., 2019). Moreover, in Figures 12, 13 and 14, we see for XNLI, PAWSX-X and MARC the amount of subnetwork overlap between each language pair both in absolute values and as a percentage of the language's full subnetwork capacity.

| | de | en | es | fr | ko | ru | zh |
|---|---|---|---|---|---|---|---|
| PAWS-X | 42 | 56 | 56 | 56 | 42 | - | - |
| XNLI | 70 | 42 | 56 | 42 | - | 56 | - |
| MARC | 56 | 42 | 42 | 56 | - | - | 84 |

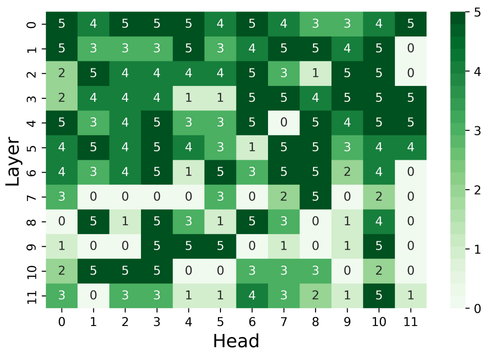
**Table 1:** The number of disabled attention heads in the identified subnetwork of each language and task.




**(a)** XNLI




**(b)** PAWS-X




**(c)** MARC

**Figure 11:** The overlap of heads enabled by each language's subnetwork per task. 5 indicates that the head is shared across all languages and 0 that it is not used by any of the languages.

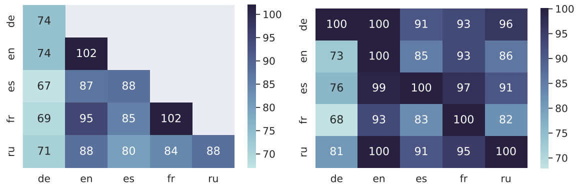



**Figure 12:** The absolute number of overlapping attention heads between each language pairs' subnetworks for XNLI. **(Left)** The percentage of overlap in heads between each language pairs' subnetworks. Note that values are not symmetric between language pairs as each language's subnetwork can have a different sparsity level. For instance, for German on the $y$-axis, it shows that 100% of the enabled heads are shared with English. Yet, 73% of the enabled heads for English are shared with German, given that English has more heads enabled. **(Right)**

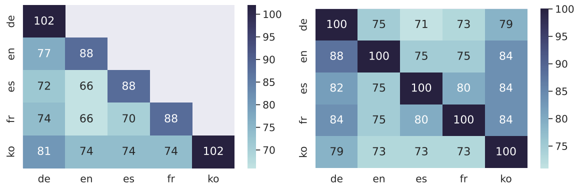



**Figure 13:** The absolute number of overlapping attention heads between each language pairs' subnetworks for PAWS-X. **(Left)** The percentage of overlap in heads between each language pairs' subnetworks. Note that values are not symmetric between language pairs as each language's subnetwork can have a different sparsity level. For instance, for German on the $y$-axis, it shows that 75% of the enabled heads are shared with English. Yet, 88% of the enabled heads for English are shared with German, given that English has fewer heads enabled. **(Right)**

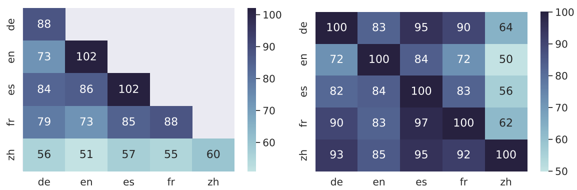



**Figure 14:** The absolute number of overlapping enabled heads between each language pairs' subnetworks for MARC. **(Left)** The percentage of overlap in heads between each language pairs' subnetworks. Note that values are not symmetric between language pairs as each language's subnetwork can have a different sparsity level. **(Right)**

## E Additional results

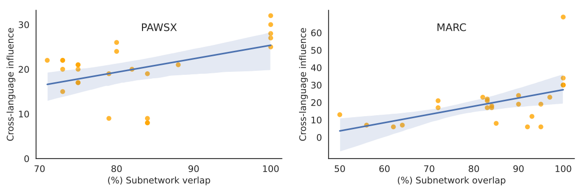



**Figure 15:** The correlation between the percentage of overlap in heads between each language pairs' subnetworks and their amounts of cross-language interference (in absolute numbers).

| | PAWS-X | | MARC | |
|---|---|---|---|---|
| | Full | SFT | Full | SFT |
| de | 68.0 | 78.8 | 75.3 | 76.4 |
| en | 78.6 | 83.0 | 75.1 | 75.8 |
| es | 78.2 | 80.5 | 76.6 | 77.4 |
| fr | 82.1 | 79.8 | 76.2 | 77.6 |
| ko | 67.1 | 69.9 | – | – |
| zh | – | – | 69.5 | 71.1 |

**Table 2:** The performance effect of SFT compared to full model fine-tuning. We report the performance of the language-specific subnetworks when used on the test samples from the respective languages when using either one of the fine-tuning techniques. Note that we do not optimize for obtaining SOTA performance in this study e.g., we train on relatively little data to make our TracIN experiments computationally feasible.

# F Additional experiments

## F.1 What happens within subnetworks during full model fine-tuning versus SFT?

In Sections 5 and 6 we used the sum of influence scores over model checkpoints to compute influence scores. We now conduct the same experiments, but instead study how cross-language influence changes over time while using the different fine-tuning strategies. To do so, we now analyze the influence scores (and their corresponding rankings) from each checkpoint separately.

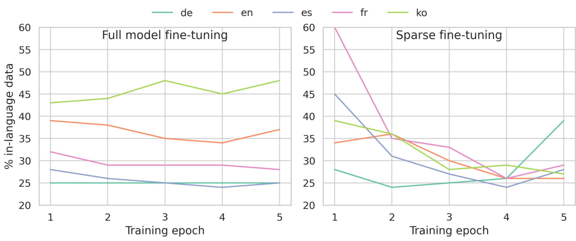



**Figure 16:** The change in language specialization of subnetworks over training epochs for PAWS-X.

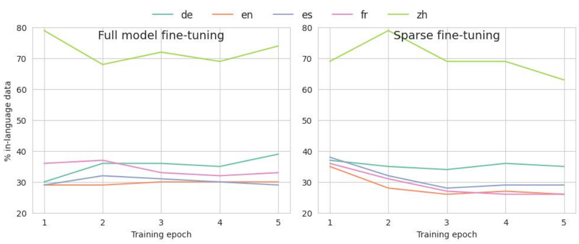



**Figure 17:** The change in language specialization for each test language over training epochs for MARC. We see that the patterns for full model fine-tuning are similar to PAWS-X, yet for sparse fine-tuning they differ considerably.

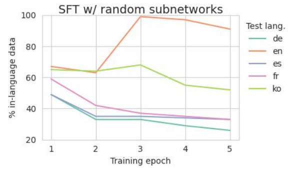



**Figure 18:** The language specialization effect of SFT with random subnetworks on PAWS-X over training epochs.

**Results** In Figure 16 we see that while both fine-tuning techniques converge to similar maximum levels of cross-lingual sharing (∼25% reliance on in-language data) for PAWS-X, SFT allows for *all* training languages to start sharing more data. Whereas for full model fine-tuning, we instead see that Korean and English are left behind. The same trend was found for MARC, see Figure 17. Also, in Figure 18, we find that using random subnetworks for SFT on PAWS-X, similarly to full model fine-tuning, results in Korean and English staying more isolated from the other three languages. This suggests that when we use random subnetworks for SFT, the model can not benefit from better cross-lingual sharing in the same way as when we identify the subnetworks via pruning. In line with results in Sections 6.1 and 7.2, we conclude that the subnetworks meaningfully overlap to enable better cross-lingual interaction during SFT.

## F.2 Composing subnetworks at test time

As an additional analysis, we study whether we can compose two languages' identified subnetworks into a language-pair specific subnetwork that, when applied at test time, will enforce more cross-lingual reliance on each other's training data. For merging two subnetworks we both tried taking the union and the intersect of the respective binary subnetwork masks. Note that we apply the composed subnetwork only at test time to a model that was trained with SFT (using the initial identified subnetworks).

**Results** We find that we can only successfully enforce cross-lingual sharing through subnetwork composition for two languages, if those individual language's subnetworks already stimulated cross-lingual sharing between the pair. For instance, in Figure 4, we saw that both the Spanish and French subnetworks (PAWS-X) and the German and English ones (MARC) resulted in more sharing between the pairs. In Figure 19, we show that taking the intersections of those language pairs' subnetworks can further strengthen this behavior (taking their union resulted in sharing to a lesser extent)

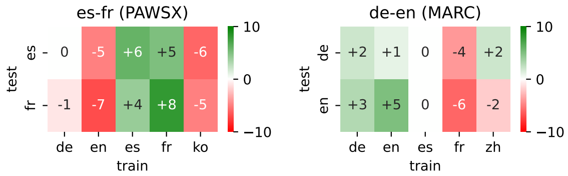



**Figure 19:** The effect on the contribution of positive influence from each training language when composing two language's subnetworks by their intersect and applying them at test time (compared to full model fine-tuning).

Trying to control sharing behavior by composing two language-specific subnetworks that individually did not lead to more sharing between the pair did not yield any clear positive results. This shows that while SFT can better cross-lingual sharing, there is still much room for improvement when it comes to creating a truly modular system that enables compositionality.