Neuron Specialization: Leveraging Intrinsic Task Modularity for Multilingual Machine Translation

Anonymous ACL submission

Abstract

Training a unified multilingual model promotes knowledge transfer but inevitably introduces *negative interference*. Language-specific modeling methods show promise in reducing interference. However, they often rely on heuristics to distribute capacity and struggle to foster cross-lingual transfer via isolated modules. In this paper, we explore intrinsic task modularity within multilingual networks and leverage these observations to circumvent interference under multilingual translation. We show that neurons in the feed-forward layers tend to be activated in a language-specific manner. Meanwhile, these specialized neurons exhibit structural overlaps that reflect language proximity, which progress across layers. Based on these findings, we propose *Neuron Specialization*, an approach that identifies specialized neurons to modularize feed-forward layers and then continuously updates them through sparse networks. Extensive experiments show that our approach achieves consistent performance gains over strong baselines with additional analyses demonstrating reduced interference and increased knowledge transfer.[1]

1 Introduction

Jointly training multilingual data in a unified model with a shared architecture for different languages has been a trend (Conneau et al., 2020; Le Scao et al., 2022) encouraging knowledge transfer across languages, especially for low-resource languages (Johnson et al., 2017; Pires et al., 2019). However, such a training paradigm also leads to *negative interference* due to conflicting optimization demands (Wang et al., 2020). This interference often causes performance degradation for high-resource languages (Li and Gong, 2021; Pfeiffer et al., 2022) and can be further exacerbated by limited model capacity (Shaham et al., 2023).

Modular-based methods, such as Language-specific modeling (Zhang et al., 2020b) and adapters (Bapna and Firat, 2019), aim to mitigate interference by balancing full parameter sharing with isolated or partially shared modules (Pfeiffer et al., 2023). However, they heavily depend on heuristics for allocating task-specific capacity and face challenges in enabling knowledge transfer between modules (Zhang et al., 2020a). Specifically, such methods rely on prior knowledge for managing parameter sharing such as language-family adapters (Chronopoulou et al., 2023) or directly isolate parameters per language, which impedes transfer (Pires et al., 2023).

Research in vision and cognitive science has shown that unified multi-task models may spontaneously develop task-specific functional specializations for distinct tasks (Yang et al., 2019; Dobs et al., 2022), a phenomenon also observed in mixture of experts Transformer systems (Zhang et al., 2023). These findings suggest that through multi-task training, networks naturally evolve towards specialized modularity to effectively manage diverse tasks, with the ablation of these specialized modules adversely affecting task performance (Pfeiffer et al., 2023). Despite these insights, exploiting the inherent structural signals for multi-task optimization remains largely unexplored.

In this work, we explore the intrinsic task-specific modularity within multi-task networks in Multilingual Machine Translation (MMT), treating each language pair as a separate task. We focus on analyzing the intermediate activations in the Feed-Forward Networks (FFN) where most model parameters reside. Our analysis shows that neurons activate in a language-specific way, yet they present structural overlaps that indicate language proximity. Moreover, this pattern evolves across layers in the model, consistent with the transition of multilingual representations from language-specific to language-agnostic (Kudugunta et al., 2019).

[1] We release anonymous code at https://anonymous.4open.science/r/NS-3D93

Building on these observations, we introduce *Neuron Specialization*, a novel method that leverages intrinsic task modularity to reduce interference and enhance knowledge transfer. In general, our approach selectively updates the FFN parameters during back-propagation for different tasks to enhance task specificity. Specifically, we first identify task-specific neurons from pre-trained multilingual translation models, using standard forward-pass validation processes without decoding. We then specifically modularize FFN layers using these specialized neurons and continuously update FFNs via sparse networks.

Extensive experiments on small- (IWSLT) and large-scale EC30 (Tan and Monz, 2023) multilingual translation datasets show that our method consistently achieves performance gains over strong baselines. Moreover, we conduct in-depth analyses to demonstrate that our method effectively mitigates interference and enhances knowledge transfer in high and low-resource languages, respectively. Our main contributions are summarized as follows:

- We identify inherent multilingual modularity by showing that neurons activate in a language-specific manner and their overlapping patterns reflect language proximity.
- Building on these findings, we enhance task specificity through sparse sub-networks, achieving consistent improvements in translation quality over strong baselines.
- We employ analyses to show that our method effectively reduces interference in high-resource languages and boosts knowledge transfer in low-resource languages.

2 Related Work

Multilingual Interference. Multilingual training enables knowledge transfer but also causes *interference*, largely due to optimization conflicts among various languages or tasks (Wang and Zhang, 2022). Methods addressing conflicts between tasks hold promise to reduce interference (Wang et al., 2020), yet they show limited effectiveness in practical applications (Xin et al., 2022). Scaling up model size reduces interference directly but may lead to overly large models (Chang et al., 2023), with risks of overfitting (Aharoni et al., 2019).

Language-Specific Modeling. Modular-based approaches enhance the unified model by adding language-dependent modules such as adapters (Bapna and Firat, 2019) or language-aware layers (Zhang et al., 2020b). Although the unified model serves as a common foundation, these approaches struggle to facilitate knowledge transfer among isolated modules due to a lack of clear inductive biases and thus heavy reliance on heuristics. For instance, Chronopoulou et al. (2023) rely on priori knowledge to control parameter sharing in language family adapters, Bapna and Firat (2019); Pires et al. (2023) isolate modules per language, hindering knowledge sharing.

Additionally, these modular-based methods substantially increase the number of parameters, thereby leading to increased memory demands and slower inference times (Liao et al., 2023a,b). Despite adapters normally being lightweight, they can easily accumulate to a significant parameter growth when dealing with many languages. In contrast, our method leverages the model's intrinsic modularity signals to promote task separation, without adding extra parameters.

Sub-networks in Multi-task Models. The lottery ticket hypothesis (Frankle and Carbin, 2018) states that within dense neural networks, sparse subnetworks can be found with iterative pruning to achieve the original network's performance. Following this premise, recent studies attempt to isolate sub-networks of a pre-trained unified model that captures task-specific features (Lin et al., 2021; He et al., 2023; Choenni et al., 2023a). Nonetheless, unlike our method that identifies intrinsic modularity within the model, these approaches depend on fine-tuning to extract the task-specific sub-networks. This process may not reflect the original model modularity and also can be particularly resource-consuming for multiple tasks.

Specifically, these methods extract the task-specific sub-networks by fine-tuning the original unified multi-task model on specific tasks, followed by employing pruning to retain only the most changed parameters. We argue that this process faces several issues: 1) The sub-network might be an artifact of fine-tuning, suggesting the original model may not inherently possess such modularity. 2) This is further supported by the observation that different random seeds during fine-tuning lead to varied sub-networks and performance instability (Choenni et al., 2023a). 3) The process is highly inefficient for models covering multiple tasks, as it necessitates separate fine-tuning for each task.

3 Neuron Structural Analysis

Recent work aims to identify a subset of parameters within pre-trained multi-task networks that are sensitive to distinct tasks. This exploration is done by either 1) ablating model components to assess impacts on performance, such as Dobs et al. (2022) ablate task-specific filters in vision models by setting their output to zero; or 2) fine-tuning the unified model on task-specific data to extract sub-networks (Lin et al., 2021; He et al., 2023; Choenni et al., 2023b). These approaches, however, raise a fundamental question, namely whether the modularity is inherent to the original model, or simply an artifact introduced by network modifications.

In this paper, we perform a thorough identification of task-specific modularity through the lens of neuron behaviors, without altering the original parameters or architectures. We focus on the neurons — the intermediate activations inside the Feed-Forward Networks (FFN) — to investigate if they indicate task-specific modularity features. As FFN neurons are active (>0) or inactive (=0) due to the $ReLU$ activation function, this binary activation state offers a clear view of their contributions to the network's output. Intuitively, neurons that remain inactive for one task but show significant activation for another may be indicative of specialization for the latter. Analyzing such modularity structures can improve our understanding of fundamental properties in multi-task models and yield insights to advance multi-task learning.

3.1 Identifying Specialized Neurons

We choose multilingual translation as a testbed, treating each translation direction as a distinct task throughout the paper. We start with a pre-trained multilingual model with d_{ff} as its dimension of the FFN layer. We hypothesize the existence of neuron subsets specialized for each task and describe the identification process of an FFN layer as follows.

Activation Recording. Given a validation dataset D_t for the t-th task, we measure activation frequencies in an FFN layer during validation. For each sample $x_i \in D_t$, we record the state of each neuron after $ReLU$, reflecting whether the neuron is active or inactive to the sample. We use a binary vector $a_i^t \in \mathbb{R}^{d_{ff}}$ to store this neuron state information. Note that this vector aggregates neuron activations for all tokens in the sample by taking the neuron union of them. By further merging all of the binary vectors for all samples in D_t, an accumulated vector $a^t = \sum_{x_i \in D_t} a_i^t$ can be derived, which denotes the frequency of each neuron being activated during a forward pass given a task-specific dataset D_t.

Neuron Selection. We identify specialized neurons for each task t based on their activation frequency a^t. A subset of neurons S_k^t is progressively selected based on the highest a^t values until reaching a predefined threshold k, where

$$\sum_{i \in S_k^t} a_{(i)}^t >= k \sum_{i=1}^{d_{ff}} a_{(i)}^t \qquad (1)$$

Here, the value $a_{(i)}^t$ is the frequency of the activation at dimension i, and $\sum_{i=1}^{d_{ff}} a_{(i)}^t$ is the total activation of all neurons for an FFN layer. k is a threshold factor, varying from 0% to 100%, indicating the extent of neuron activation deemed necessary for specialization. A lower k value results in higher sparsity in specialized neurons; $k = 0$ means no neuron will be involved, while $k = 100$ fully engages all neurons, the same as utilizing the full capacity of the original model. This dynamic approach emphasizes the collective significance of neuron activations up to a factor of k. In the end, we repeat these processes to obtain the specialized neurons of all FFN layers for each task.

3.2 Analysis on EC30

In this section, we describe how we identify specialized neurons on EC30 (Tan and Monz, 2023), where we train an MMT model covering all directions. EC30 is a multilingual translation benchmark that is carefully designed to consider diverse linguistic properties and real-world data distributions. It collects high to low-resource languages, resulting in 30 diverse languages from 5 language families, allowing us to connect our observations with linguistic properties easily. See Sections 5 for details on data and models.

3.2.1 Neuron Overlaps Reflect Language Proximity

We identified specialized neurons following Section 3.1, while setting the cumulative activation threshold k at 95%. This implies that the set of specialized neurons covers approximately 95% of the total activations. Intuitively, two similar tasks should have a high overlap between their specialized neuron sets. Therefore, we examined the overlaps among specialized neurons across different

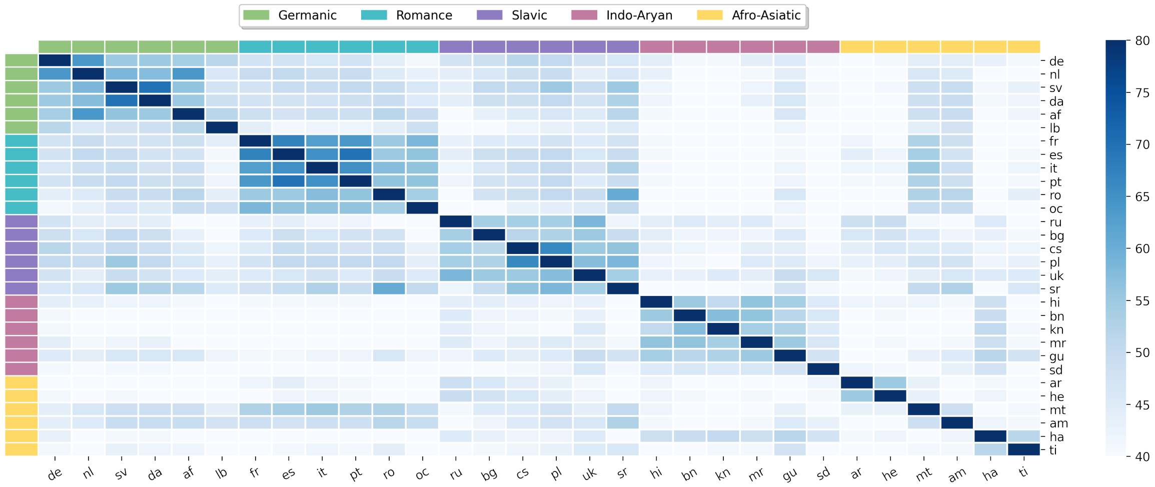

Figure 1: Pairwise Intersection over Union (IoU) scores for specialized neurons extracted from the first decoder FFN layer across all out-of-English translation directions to measure the degree of overlap. Darker cells indicate stronger overlaps, with the color threshold set from 40 to 80 to improve visibility.

tasks by calculating the Intersection over Union (IoU) scores: For task t_i and t_j, with specialized neurons denoted as sets S^i and S^j, their overlap is quantified by $\text{IoU}(S^i, S^j) = \frac{|S^i \cap S^j|}{|S^i \cup S^j|}$.

Figure 1 shows the IoU scores for specialized neurons across different tasks in the first decoder layer. Figures for the other layers can be found in Appendix A.6. We first note a structural separation of neuron overlaps, indicating a preference for language specificity. Notably, neuron overlap across language families is relatively low, a trend more pronounced in encoder layers (Figure 6). Secondly, this structural distinction generally correlates with language proximity as indicated by the clustering pattern in Figure 1. This implies that target languages from the same family are more likely to activate similar neurons in the decoder, even when they use different writing systems, e.g., Arabic (ar) and Hebrew (he). Overlaps also show linguistic traits beyond family ties, exemplified by notable overlaps between Maltese (mt) and languages in the Romance family due to vocabulary borrowing.

3.2.2 The Progression of Neuron Overlaps

To analyze how specialized neuron overlaps across tasks evolve within the model, we visualize the IoU score distribution across layers in Figure 2. For each layer, we compute the pair-wise IoU scores between all possible tasks and then show them in a distribution. Overall, we observe that from shallow to deeper layers, structural distinctions intensify in the decoder (decreasing IoU scores) and weaken in the encoder (increasing IoU scores).

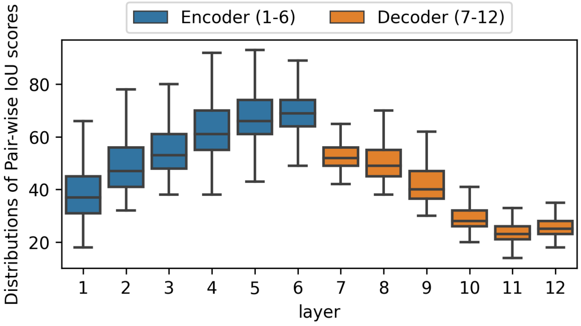

Figure 2: Progression of distribution of IoU scores for specialized neurons across layers on the EC30 dataset. The scores are measured for different source and target languages in the Encoder and Decoder, respectively.

On the one hand, all neuron overlaps increase as we move up the encoder, regardless of whether these tasks are similar or not. This observation may suggest that the neurons in the encoder become more language-agnostic, as they attempt to map different scripts into semantic concepts. As for the Decoder, the model presents intensified modularity in terms of overlaps of specialized neurons. This can be seen by all overlaps becoming much smaller, indicating that the neurons behave more separately. Additionally, we found the progression of neuron overlaps is similar to the evolution of multilingual representation: embedding gets closer in the encoder and becomes more dissimilar in the decoder (Kudugunta et al., 2019). Our observations, highlighting the inherent features of the multilingual translation model, occur without modifying the network's outputs or parameters.

4 Neuron Specialization Training

Our neuron structural analysis showed the presence of specialized neurons within the Feed-Forward Network (FFN) layers of a multilingual network. We hypothesize that continuously training the model, while leveraging these specialized neurons' intrinsic modular features, can further enhance task-specific performance. Building on this hypothesis, we propose *Neuron Specialization*, an approach that leverages specialized neurons to modularize the FFN layers in a task-specific manner.

4.1 Vanilla Feed-Forward Network

We first revisit the Feed-Forward Network (FFN) in Transformer (Vaswani et al., 2017). The FFN, crucial to our analysis, consists of two linear layers (fc1 and fc2) with a $ReLU$ activation function. Specifically, the FFN block first processes the hidden state $H \in \mathbb{R}^{n\times d}$ (n denotes number of tokens in a batch) through fc1 layer $W_1 \in \mathbb{R}^{d\times d_{ff}}$. Then the output is passed to $ReLU$ and the fc2 layer W_2, as formalized in Eq 2, with bias terms omitted.

$$\mathrm{FFN}(H) = \mathrm{ReLU}(HW_1)\,W_2. \qquad (2)$$

4.2 Specializing Task-Specific FFN

Next, we investigate continuous training upon a subset of specialized parameters within FFN for each task. Given a pre-trained vanilla multilingual Transformer model with tags to identify the language pairs, e.g., Johnson et al. (2017), we can derive specialized neuron set S_k^t for each layer of a task task[2] t and threshold k following the method outlined in Section 3.1. Then, we derive a boolean mask vector $m_k^t \in \{0,1\}^{d_{ff}}$ from S_k^t, where the i-th element in m_k^t is set to 1 only when $i \in S_k^t$, and apply it to control parameter updates. Specifically, we broadcast m_k^t and perform Hadamard Product with W_1 in each FFN layer as follows:

$$FFN(H) = ReLU(H(m_k^t \odot W_1))W_2. \qquad (3)$$

m_k^t plays the role of controlling parameter update, where the boolean value of i-th element in m_k^t denotes if the i-th row of parameters in W_1 can be updated or not for each layer[3] during continues training. Broadly speaking, our approach selectively updates the first FFN (fc1) weights during back-propagation, tailoring the model more closely towards specific translation tasks and reinforcing neuron separation. Note that while fc1 is selectively updated for specific tasks, other parameters are universally updated to maintain stability, and the same masking is applied to inference to ensure consistency. We provide the pseudocode of our method in Appendix A.3.

5 Experimental Setup

In this section, we evaluate the capability of our proposed method on small (IWSLT) and large-scale (EC30) multilingual machine translation tasks. More details of the datasets are in Appendix A.1.

5.1 Datasets

IWSLT. Following Lin et al. (2021), we constructed an English-centric dataset with eight languages using IWSLT-14, ranging from 89k to 169k in corpus size. We learned a 30k SentencePiece unigram (Kudo and Richardson, 2018) shared vocabulary and applied temperature oversampling with $\tau = 2$ to balance low-resource languages. For a more comprehensive evaluation, we replaced the standard test set with Flores-200 (Costa-jussà et al., 2022), merging *devtest* and *test*, which offers multiple parallel sentences per source text.

EC30. We further validate our methods using the large-scale EC30 dataset (Tan and Monz, 2023), which features 61 million parallel training sentences across 30 English-centric language pairs, representing five language families and various writing systems. We classify these language pairs into low-resource (=100k), medium-resource (=1M), and high-resource (=5M) categories. Following Wu and Monz (2023), we build a 128k size shared SentencePiece BPE vocabulary. Aligning with the original EC30 setups, we use Ntrex-128 (Federmann et al., 2022) as the validation set. Also, we use Flores-200 (merging *devtest* and *test*) as test sets for cross-domain evaluation.

5.2 Systems

We compare our method with strong open-source baselines that share similar motivations in reducing interference for multilingual translation tasks.

Baselines:

- **mT-small.** For IWSLT, we train an mT-small model on Many-to-Many directions as

[2] We treat each translation direction as a distinct task.

[3] Note that m_k^t is layer-specified, we drop layer indexes hereon for simplicity of notation.

Language Size	$\Delta\theta$	Fa 89k	Pl 128k	Ar 139k	He 144k	Nl 153k	De 160k	It 167k	Es 169k	Avg
One-to-Many (O2M / En-X)										
mT-small	-	14.5	9.9	12.0	13.1	17.0	20.6	17.3	18.3	15.4
Adapter$_{LP}$	+67%	+0.1	-0.1	+0.4	**+1.4**	**+0.2**	+0.6	+0.1	**+0.4**	**+0.4**
LaSS	0%	-2.6	0	+0.6	+0.7	-0.2	**+0.7**	-0.2	-0.4	-0.2
Ours	0%	**+0.7**	**+0.1**	**+0.9**	+0.6	+0.1	+0.1	**+0.2**	-0.3	+0.3
Many-to-One (M2O / X-En)										
mT-small	-	19.1	19.4	25.7	30.9	30.6	28.1	29.0	34.0	24.7
Adapter$_{LP}$	+67%	+0.9	+0.6	+0.9	+1.0	+0.8	+1.0	+0.9	+0.3	+0.8
LaSS	0%	+1.2	+0.6	+0.9	+1.4	+1.1	+1.6	+1.6	+0.8	+1.2
Ours	0%	**+1.6**	**+1.2**	**+1.7**	**+2.0**	**+1.9**	**+2.1**	**+1.8**	**+1.4**	**+1.7**

Table 1: Average BLEU improvements over the baseline (mT-small) model on the IWSLT dataset. $\Delta\theta$ denotes the relative parameter increase over the baseline, encompassing all translation directions. The best results are in **bold**.

per (Lin et al., 2021): a 6-layer Transformer with 4 attention heads, d = 512, d_{ff} = 1,024.

- **mT-big.** For EC30, we train a mT-big on Many-to-Many directions following Wu and Monz (2023). It has 6 layers, with 16 attention heads, d = 1,024, and d_{ff} = 4,096.

Adapters. We employ two adapter methods: 1) Language Pair Adapter (**Adapter$_{LP}$**) and 2) Language Family Adapter (**Adapter$_{Fam}$**). We omit Adapter$_{Fam}$ for IWSLT due to its limited languages. Adapter$_{LP}$ inserts adapter modules based on language pairs, demonstrating strong effects in reducing interference while presenting no parameter sharing (Bapna and Firat, 2019). In contrast, Adapter$_{Fam}$ (Chronopoulou et al., 2023) facilitates parameter sharing across similar languages by training modules for each language family. Their bottleneck dimensions are 128 and 512 respectively. See Appendix A.2 for more training details.

LaSS. Lin et al. (2021) proposed LaSS to locate language-specific sub-networks following the lottery ticket hypothesis, i.e., finetuning all translation directions from a pre-trained model and then pruning based on magnitude. They then continually train the pre-trained model by only updating the sub-networks for each direction. We adopt the strongest LaSS configuration by applying sub-networks for both attention and FFNs.

5.3 Implementation and Evaluation

We train our baseline models following the same hyper-parameter settings in Lin et al. (2021) and Wu and Monz (2023). Specifically, we use the Adam optimizer ($\beta1 = 0.9$, $\beta2 = 0.98$, $\epsilon = 10^{-9}$) with 5e-4 learning rate and 4k warmup steps in all experiments. We use 4 NVIDIA A6000 (48G) GPUs to conduct most experiments and implement them based on Fairseq (Ott et al., 2019) with FP16. We list detailed training and model specifications for all systems in Appendix A.2.

We adopt the tokenized BLEU (Papineni et al., 2002) for the IWSLT dataset and detokenized case-sensitive SacreBLEU[4] (Post, 2018) for the EC30 dataset in our main result evaluation section. In addition, we provide ChrF++ (Popović, 2017) and COMET (Rei et al., 2020) in Appendix A.4.

6 Results and Analyses

6.1 Small-Scale Results on IWSLT

We show results on IWSLT in Table 1. For Many-to-One (M2O) directions, our method achieves an average +1.7 BLEU gain over the baseline, achieving the best performance among all approaches for all languages. The Adapter$_{LP}$, with a 67% increase in parameters over the baseline model, shows weaker improvements (+0.8) than our method. As for One-to-Many (O2M) directions, we observed weaker performance improvements for all methods. While the gains are modest (averaging +0.3 BLEU), our method demonstrates consistent improvements across various languages in general.

Scaling up does not always reduce interference. Shaham et al. (2023); Chang et al. (2023) have found scaling up the model capacity reduces interference, even under low-resource settings. We

[4] nrefs:1|case:mixed|eff:no|tok:13a|smooth:exp|version:2.3.1

Methods	$\Delta\theta$	High (5M)			Med (1M)			Low (100K)			All (61M)		
		O2M	M2O	Avg	O2M	M2O	Avg	O2M	M2O	Avg	O2M	M2O	Avg
mT-big	-	28.1	31.6	29.9	29.7	31.6	30.6	18.9	26.0	22.4	25.5	29.7	27.7
Adapter$_{Fam}$	+70%	+0.7	+0.3	+0.5	+0.7	+0.3	+0.5	+1.1	+0.5	+0.8	+0.8	+0.4	+0.6
Adapter$_{LP}$	+87%	+1.6	+0.6	+1.1	+1.6	+0.4	+1.0	+0.4	+0.4	+0.4	+1.2	+0.5	+0.8
LaSS	0%	**+2.3**	+0.8	+1.5	**+1.7**	+0.2	+1.0	-0.1	-1.8	-1.0	+1.3	-0.3	+0.5
Random	0%	+0.9	-0.5	+0.2	+0.5	-0.7	-0.2	-0.3	-1.5	-0.9	+0.5	-0.9	-0.2
Ours-Enc	0%	+1.2	+1.1	+1.1	+1.0	+1.0	+1.0	+0.7	+0.8	+0.8	+1.0	+1.0	+1.0
Ours-Dec	0%	+1.2	+1.1	+1.1	+0.9	+1.1	+1.0	+0.7	+1.1	+0.9	+0.9	+1.1	+1.0
Ours	0%	+1.8	**+1.4**	**+1.6**	+1.4	**+1.1**	**+1.3**	**+1.4**	**+0.9**	**+1.2**	**+1.5**	**+1.1**	**+1.3**

Table 2: Average SacreBLEU improvements on the EC30 dataset over the baseline (mT-big), categorized by High, Medium, and Low-resource translation directions. 'Random' denotes continually updating the model with randomly selected task-specific neurons. 'Ours-Enc' and 'Ours-Dec' indicate Neuron Specialization applied solely to the Encoder and Decoder, respectively, while 'Ours' signifies the method applied to both components.

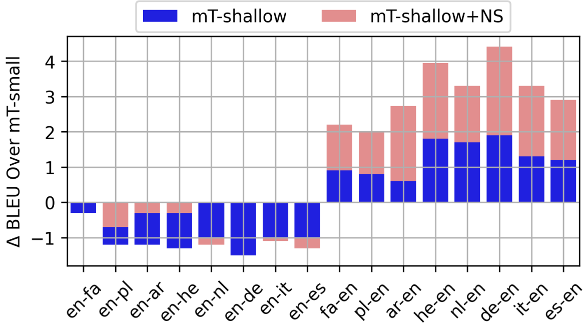

Figure 3: BLEU gains of shallower models over mT-small on IWSLT show improved X-En performance at the expense of En-X. Applying Neuron Specialization reduces EN-X degradation and amplifies X-En gains.

then investigate the trade-off between performance and model capacity by employing mT-shallow, a shallower version of mT-small with three fewer layers (with $\Delta\theta = -39\%$ for parameters, see Table 6 for details). Surprisingly, in Figure 3, we show that reducing parameters improved Many-to-One (X-En) performance but weakened One-to-Many (En-X) results. This result indicates that scaling up the model capacity does not always reduce interference, but may show overfitting to have performance degradation. Furthermore, we show that implementing Neuron Specialization with mT-shallow enhances Many-to-One (X-En) performance in all directions while lessening the decline in One-to-Many (En-X) translation quality in general.

6.2 Large-Scale Results on EC-30

Similar to what we observed in the small-scale setting, we find notable improvements when we scale up on the EC30 dataset. As shown in Table 2, we show consistent improvements across high-, medium-, and low-resource languages, with an average gain of +1.3 SacreBLEU over the baseline. LaSS, while effective in high-resource O2M pairs, presents limitations with negative impacts (-1.0 score) on low-resource languages, highlighting difficulties in sub-network extraction for low-resource languages. In contrast, our method achieves stable and consistent gains across all resource levels. The Adapter$_{LP}$, despite increasing parameters by 87% compared to the baseline, falls short of our method in boosting performance. Additionally, we show that applying Neuron Specialization in either the encoder or decoder delivers similar gains, with both combined offering stronger performance.

Model	$\triangle\theta$	$\triangle T_{subnet}$	$\triangle$ Memory
Adapter$_{LP}$	+87%	n/a	1.42 GB
LaSS	0%	+33 hours	9.84 GB
Ours	0%	+5 minutes	3e-3 GB

Table 4: Efficiency comparison on EC30 dataset regarding extra trainable parameters ($\triangle\theta$: relative increase over the baseline), extra processing time for subnet extraction ($\triangle T_{subnet}$), and extra memory ($\triangle$ Memory).

Efficiency Comparisons. We compare the efficiency on three aspects (Table 4). For trainable parameter increase, introducing lightweight language pair adapters accumulates a significant +87% parameter growth over the baseline. Next, compared to LaSS, which is fine-tuned to identify subnetworks and demands substantial time (33 hours with 4 Nvidia A6000 GPUs), our approach efficiently locates specialized neurons in just 5 minutes. Considering memory costs, essential for handling numerous languages in deployment environments, our method proves more economical, pri-

Lang Size	De 5m	Es 5m	Cs 5m	Hi 5m	Ar 5m	Lb 100k	Ro 100k	Sr 100k	Gu 100k	Am 100k	High Avg	Low Avg
One-to-Many												
Bilingual	36.3	24.6	28.7	43.9	23.7	5.5	16.2	17.8	12.8	4.1	31.8	11.3
mT-big	-4.7	-1.5	-3.6	-4.4	-4.7	+9.0	+8.9	+6.2	+13.9	+3.1	-3.7	+8.2
Ours	-2.0	-0.2	-1.7	-2.4	-3.0	+10.8	+10.0	+8.2	+16.4	+3.7	-1.9	+9.8
Many-to-One												
Bilingual	39.1	24.5	32.6	35.5	30.8	8.7	19.5	21.3	7.0	8.7	32.7	13.0
mT-big	-1.5	+0.9	+0.2	-1.8	-2.3	+13.7	+11.9	+10.3	+18.2	+12.5	-1.1	+13.3
Ours	-0.3	+1.7	+1.8	-0.2	-0.3	+15.3	+12.4	+11.3	+19.6	+14.1	+0.3	+14.5

Table 3: SacreBLEU score comparisons for Multilingual baseline and Neuron Specialization models against Bilingual ones on the EC30 dataset, limited to 5 high- and low-resource languages due to computational constraints. Red signifies negative interference, Blue denotes positive synergy, with darker shades indicating better effects.

marily requiring storage of 1-bit masks for the FFN neurons instead of extensive parameters.

Random Mask. We also incorporate the experiments using random masks with Neuron Specialization Training, to validate whether our Specialized Neuron Identification process can capture useful task-specific modularity. We randomly sample 70% neurons to be task-specific and then conduct the same Neuron Specialization Training step. Our results indicate that the random masks strategy sacrifices performance on low-resource tasks (average -0.9 score) to enhance the performance of high-resource O2M directions (+0.9 score). This indicates the effectiveness of our identification method in locating intrinsic task-specific neurons.

The role of threshold factor. We explore the impact of our sole hyper-parameter k (neuron selection threshold factor) on performance. The results indicate that performance generally improves with an increase in k, up to a point of 95% (around 25% sparsity), beyond which the performance starts to drop. See Appendix A.5 for more detailed results.

6.3 The Impact of Reducing Interference

In this section, we evaluate to what extent our Neuron Specialization method mitigates interference and enhances cross-lingual transfer. Similar to Wang et al. (2020), we train bilingual models that do not contain interference or transfers, and then compare results between bilingual models, the conventional multilingual baseline model (mT-big), and our neuron specialization (ours). We train Transformer-big and Transformer-based models for high- and low-resource tasks, see Appendix A.2.

In Table 3, we show that the conventional multilingual model (mT-big) facilitates clear positive transfer for low-resource languages versus bilingual setups, leading to +8.2 (O2M) and +13.3 (M2O) score gains but incurs negative interference for high-resource languages (-3.7 and -1.1 scores).

Our method reduces interference for high-resource settings, leading to +1.8 and +1.4 SacreBLEU gains over mT-big in O2M and M2O directions. Moreover, our Neuron Specialization enhances low-resource language performance with average gains of +1.6 (O2M) and +1.2 (M2O) SacreBLEU over the mT-big, demonstrating its ability to foster cross-lingual transfer. Despite improvements, our approach still trails behind bilingual models for most high-resource O2M directions, indicating that while interference is largely reduced, room for improvement still exists.

7 Conclusions

In this paper, we have identified and leveraged *intrinsic task-specific modularity* within multilingual networks to mitigate interference. We showed that FFN neurons activate in a language-specific way, and they present structural overlaps that reflect language proximity, which progress across layers. We then introduced *Neuron Specialization* to leverage these natural modularity signals to structure the network, enhancing task specificity and improving knowledge transfer. Our experimental results, spanning various resource levels, show that our method consistently outperforms strong baseline systems, with additional analyses demonstrating reduced interference and increased knowledge transfer. Our work deepens the understanding of multilingual models by revealing their intrinsic modularity, offering insights into how multi-task models can be optimized without extensive modifications.

Limitations

This study primarily focuses on Multilingual Machine Translation, a key method in multi-task learning, using it as our primary testbed. However, the exploration of multilingual capabilities can be extended beyond translation to include a broader range of Multilingual Natural Language Processing tasks. These areas remain unexplored in our current research and are considered promising directions for future work.

Additionally, our analysis is limited to the feed-forward network (FFN) components within the Transformer architecture, which, although they constitute a significant portion of the model's parameters, represent only one facet of its complex structure. Future investigations could yield valuable insights by assessing the modularity of other Transformer components, such as the attention mechanisms or layer normalization modules, to provide a more comprehensive understanding of the system's overall functionality.

Lastly, we conducted our identification methods of specialized neurons primarily on Feed-Forward Networks that use ReLU as the activation function. This is because neurons after the ReLU naturally present two states: active (>0) and inactive (=0), which offers a clear view of their contributions to the network outputs, thus being inherently interpretable. Recent work on Large Language Models has also explored the binary activation states of FFN neurons, particularly focused on when neurons are activated, and their roles in aggregating information (Voita et al., 2023). We leave the exploration of FFN neurons using other activation functions such as the GELU (Hendrycks and Gimpel, 2016), to future work.

Broader Impact

Recognizing the inherent risks of mistranslation in machine translation data, we have made efforts to prioritize the incorporation of high-quality data, such as two open-sourced Multilingual Machine Translation datasets: IWSLT and EC30. Additionally, issues of fairness emerge, meaning that the capacity to generate content may not be equitably distributed across different languages or demographic groups. This can lead to the perpetuation and amplification of existing societal prejudices, such as biases related to gender, embedded in the data.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. 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.

Ali Araabi and Christof Monz. 2020. Optimizing transformer for low-resource neural machine translation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 3429–3435.

Ankur Bapna and Orhan Firat. 2019. Simple, scalable adaptation for neural machine translation. 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 1538–1548.

Tyler A Chang, Catherine Arnett, Zhuowen Tu, and Benjamin K Bergen. 2023. When is multilinguality a curse? language modeling for 250 high-and low-resource languages. *arXiv preprint arXiv:2311.09205*.

Rochelle Choenni, Dan Garrette, and Ekaterina Shutova. 2023a. Cross-lingual transfer with language-specific subnetworks for low-resource dependency parsing. *Computational Linguistics*, 49(3):613–641.

Rochelle Choenni, Ekaterina Shutova, and Dan Garrette. 2023b. Examining modularity in multilingual lms via language-specialized subnetworks. *arXiv preprint arXiv:2311.08273*.

Alexandra Chronopoulou, Dario Stojanovski, and Alexander Fraser. 2023. Language-family adapters for low-resource multilingual neural machine translation. In *Proceedings of the The Sixth Workshop on Technologies for Machine Translation of Low-Resource Languages (LoResMT 2023)*, pages 59–72.

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.

Marta R Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, et al. 2022. No language left behind: Scaling human-centered machine translation. *arXiv preprint arXiv:2207.04672*.

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.

Angela Fan, Shruti Bhosale, Holger Schwenk, Zhiyi Ma, Ahmed El-Kishky, Siddharth Goyal, Mandeep Baines, Onur Celebi, Guillaume Wenzek, Vishrav Chaudhary, et al. 2021. Beyond english-centric multilingual machine translation. *The Journal of Machine Learning Research*, 22(1):4839–4886.

Christian Federmann, Tom Kocmi, and Ying Xin. 2022. Ntrex-128–news test references for mt evaluation of 128 languages. In *Proceedings of the First Workshop on Scaling Up Multilingual Evaluation*, pages 21–24.

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

Dan He, Minh Quang Pham, Thanh-Le Ha, and Marco Turchi. 2023. Gradient-based gradual pruning for language-specific multilingual neural machine translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 654–670.

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*.

Melvin Johnson, Mike Schuster, Quoc V Le, Maxim Krikun, Yonghui Wu, Zhifeng Chen, Nikhil Thorat, Fernanda Viégas, Martin Wattenberg, Greg Corrado, et al. 2017. Google's multilingual neural machine translation system: Enabling zero-shot translation. *Transactions of the Association for Computational Linguistics*, 5:339–351.

Taku Kudo and John Richardson. 2018. Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 66–71.

Sneha Kudugunta, Ankur Bapna, Isaac Caswell, and Orhan Firat. 2019. Investigating multilingual nmt representations at scale. 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 1565–1575.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, Matthias Gallé, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model.

Xian Li and Hongyu Gong. 2021. Robust optimization for multilingual translation with imbalanced data. *Advances in Neural Information Processing Systems*, 34:25086–25099.

Baohao Liao, Yan Meng, and Christof Monz. 2023a. Parameter-efficient fine-tuning without introducing new latency. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4260, Toronto, Canada. Association for Computational Linguistics.

Baohao Liao, Shaomu Tan, and Christof Monz. 2023b. Make pre-trained model reversible: From parameter to memory efficient fine-tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*.

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.

Myle Ott, Sergey Edunov, Alexei Baevski, Angela Fan, Sam Gross, Nathan Ng, David Grangier, and Michael Auli. 2019. fairseq: A fast, extensible toolkit for sequence modeling. *arXiv preprint arXiv:1904.01038*.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. Bleu: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318.

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 Ponti. 2023. Modular deep learning. *Transactions on Machine Learning Research*. Survey Certification.

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.

Telmo Pires, Robin Schmidt, Yi-Hsiu Liao, and Stephan Peitz. 2023. Learning language-specific layers for multilingual machine translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14767–14783.

Maja Popović. 2017. chrf++: words helping character n-grams. In *Proceedings of the second conference on machine translation*, pages 612–618.

Matt Post. 2018. A call for clarity in reporting bleu scores. In *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191.

Ricardo Rei, Craig Stewart, Ana C Farinha, and Alon Lavie. 2020. Comet: A neural framework for mt evaluation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2685–2702.

Uri Shaham, Maha Elbayad, Vedanuj Goswami, Omer Levy, and Shruti Bhosale. 2023. Causes and cures for interference in multilingual translation. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Toronto, Canada. Association for Computational Linguistics.

Shaomu Tan and Christof Monz. 2023. Towards a better understanding of variations in zero-shot neural machine translation performance. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 13553–13568.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *Advances in neural information processing systems*, 30.

Elena Voita, Javier Ferrando, and Christoforos Nalmpantis. 2023. Neurons in large language models: Dead, n-gram, positional. *arXiv preprint arXiv:2309.04827*.

Qian Wang and Jiajun Zhang. 2022. Parameter differentiation based multilingual neural machine translation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 11440–11448.

Zirui Wang, Yulia Tsvetkov, Orhan Firat, and Yuan Cao. 2020. Gradient vaccine: Investigating and improving multi-task optimization in massively multilingual models. In *International Conference on Learning Representations*.

Di Wu and Christof Monz. 2023. Beyond shared vocabulary: Increasing representational word similarities across languages for multilingual machine translation. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore. Association for Computational Linguistics.

Derrick Xin, Behrooz Ghorbani, Justin Gilmer, Ankush Garg, and Orhan Firat. 2022. Do current multi-task optimization methods in deep learning even help? *Advances in neural information processing systems*, 35:13597–13609.

Guangyu Robert Yang, Madhura R Joglekar, H Francis Song, William T Newsome, and Xiao-Jing Wang. 2019. Task representations in neural networks trained to perform many cognitive tasks. *Nature neuroscience*, 22(2):297–306.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2020a. Share or not? learning to schedule language-specific capacity for multilingual translation. In *International Conference on Learning Representations*.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020b. Improving massively multilingual neural machine translation and zero-shot translation. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1628–1639.

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. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4066–4083, Toronto, Canada. Association for Computational Linguistics.

A Appendix

A.1 Dataset details

IWSLT We collect and pre-processes the IWSLT-14 dataset following Lin et al. (2021). We refer readers to Lin et al. (2021) for more details.

EC30 We utilize the EC30, a subset of the EC40 dataset (Tan and Monz, 2023) (with 10 extremely low-resource languages removed in our experiments) as our main dataset for most experiments and analyses. We list the Languages with their ISO and scripts in Table 5, along with their number of sentences. In general, EC30 is an English-centric Multilingual Machine Translation dataset containing 61 million sentences covering 30 languages (excluding English). It collected data from 5 representative language families with multiple writing scripts. In addition, EC30 is well balanced at each resource level, for example, for all high-resource languages, the number of training sentences is 5 million. Note that the EC30 is already pre-processed and tokenized (with Moses tokenizer), thus we directly use it for our study.

A.2 Model and Training Details

We list the configurations and hyper-parameter settings of all systems for the main training setting (EC30) in Table 6. As for global training settings, we adopt the pre-norm and share the decoder input output embedding for all systems. We use cross entropy with label smoothing to avoid overfitting (smoothing factor=0.1) and set early stopping to 20 for all systems. Similar to Fan et al. (2021), we prepend language tags to the source and target sentences to indicate the translation directions for all multilingual translation systems.

Bilingual models. For bilingual models of low-resource languages, we adopt the suggested hyper-parameter settings from Araabi and Monz (2020), such as $d_{ff} = 512$, number of attention head as 2, and dropout as 0.3. Furthermore, We train separate dictionaries for low-resource bilingual models to avoid potential overfitting instead of using the large 128k shared multilingual dictionary.

	Germanic			Romance			Slavic			Indo-Aryan			Afro-Asiatic		
	ISO	Language	Script	ISO	Language	Script	ISO	Language	Script	ISO	Language	Script	ISO	Language	Script
High	de	German	Latin	fr	French	Latin	ru	Russian	Cyrillic	hi	Hindi	Devanagari	ar	Arabic	Arabic
(5m)	nl	Dutch	Latin	es	Spanish	Latin	cs	Czech	Latin	bn	Bengali	Bengali	he	Hebrew	Hebrew
Med	sv	Swedish	Latin	it	Italian	Latin	pl	Polish	Latin	kn	Kannada	Devanagari	mt	Maltese	Latin
(1m)	da	Danish	Latin	pt	Portuguese	Latin	bg	Bulgarian	Cyrillic	mr	Marathi	Devanagari	ha	Hausa*	Latin
Low	af	Afrikaans	Latin	ro	Romanian	Latin	uk	Ukrainian	Cyrillic	sd	Sindhi	Arabic	ti	Tigrinya	Ethiopic
(100k)	lb	Luxembourgish	Latin	oc	Occitan	Latin	sr	Serbian	Latin	gu	Gujarati	Devanagari	am	Amharic	Ethiopic

Table 5: Details of EC30 Training Dataset. Numbers in the table represent the number of sentences, for example, 5m denotes exactly 5,000,000 number of sentences. The only exception is Hausa, where its size is 334k (334,000).

Models	Dataset	Num. trainable params	Num. Layer	Num. Attn Head	dim	d_{ff}	max tokens	update freq	dropout
mT-shallow	IWSLT	47M	3	8	512	1,024	2,560	4	0.1
mT-small	IWSLT	76M	6	8	512	1,024	2,560	4	0.1
bilingual-low	EC30	52M	6	2	512	1,024	2,560	1	0.3
bilingual-high	EC30	439M	6	16	1,024	4096	2,560	10	0.1
mT-big	EC30	439M	6	16	1,024	4,096	7,680	21	0.1
LaSS	EC30	439M	6	16	1,024	4,096	7,680	21	0.1
Neuron Specialization	EC30	439M	6	16	1,024	4,096	7,680	21	0.1

Table 6: Configuration and hyper-parameter settings for all models in this paper. Num. Layer and Attn Head denote the number of layers and attention heads, respectively. dim represents the dimension of the Transformer model, d_{ff} means the dimension of the feed-forward layer. bilingual-low and -high represent the bilingual models for low and high-resource languages.

For bilingual models of high-resource languages, we adopt the 128k shared multilingual dictionary and train models with the Transformer-big architecture as the multilingual baseline (mT-big). The detailed configurations can be found in Table 6.

Language Pair Adapters. We implement Language Pair Adapters (Bapna and Firat, 2019) by ourselves based on Fairseq. The Language Pair Adapter is learned depending on each pair, e.g., we learn two modules for en-de, namely en on the Encoder side and the de on the Decoder side. Note that, except for the unified pre-trained model, language pair adapters do not share any parameters with each other, preventing potential knowledge transfers. We set its bottleneck dimension as 128 for all experiments of IWSLT and EC30.

- **IWSLT.** For the IWSLT dataset that contains 8 languages with 16 language pairs/translation directions, the size mT-small base model is 76M. Language Pair Adapters insert 3.2M additional trainable parameters for one language pair, thus resulting in 51.2M added parameters for all language pairs, leading to 67% relative parameter increase over the baseline model.

- **EC30.** For the EC30 dataset that contains 30 languages with 60 language pairs/translation directions, the size mT-big base model is 439M. Language Pair Adapters insert 6.4M extra trainable parameters for one language pair, thus resulting in 384M added parameters for all language pairs, leading to 87% relative parameter increase over the baseline model.

Language Family Adapters. The Language Family Adapter (Chronopoulou et al., 2023) is learned depending on each language family, e.g., for all 6 Germanic languages in the EC30, we learn two modules for en-Germanic, namely the en adapter on the Encoder side and the Germanic adapter on the Decoder side. We set its bottleneck dimension as 512 for all experiments for the EC30.

- **EC30.** For the EC30 dataset that contains 30 languages with 60 language pairs/translation directions, the size mT-big base model is 439M. Language Family Adapters insert 25.3M additional trainable parameters for one family (on EN-X directions), thus resulting in 303.6M added parameters for all families on both EN-X and X-En directions, leading to 69% relative parameter increase over the baseline model.

LaSS. When reproducing LaSS (Lin et al., 2021), we adopt the code from their official Github page[5] with the same hyper-parameter setting as they suggested in their paper. For the IWSLT dataset, we finetune the mT-small for each translation direction with dropout=0.3, we then identify the language-specific parameters for attention and feed-forward modules (the setting with the strongest improvements in their paper) with a pruning rate of 70%. We continue to train the sparse networks while keeping the same setting as the pre-training phase as they suggested. Note that we observed different results as they reported in the paper, even though we used the same code, hyper-parameter settings, and corresponding Python environment and package version. We also found that He et al. (2023) reproduced LaSS results in their paper, which shows similar improvements (around +0.6 BLUE gains) over the baseline of our reproductions. As for an improved method over LaSS proposed by He et al. (2023), we do not reproduce their method since no open-source code has been released.

A.3 Pseudocode of Neuron Specialization

We provide the pseudocode of our proposed method, *Neuron Specialization*. We present the process of Specialized Neuron Identification in Algorithm. 1 and Neuron Specialization Training in Algorithm. 2.

A.4 Result Details using ChrF++ and COMET

For our main experiments in the EC30, we further provide the ChrF++ (Popović, 2017) and COMET (Rei et al., 2020) scores as extra results, as shown in Table 7 and Table 8, respectively. Similar to what we observed in Section 6.2, our Neuron Specialization presents consistent performance improvements over the baseline model while outperforming other methods such as LaSS and Adapters.

A.5 Sparsity versus Performance

For the Neuron Specialization, we dynamically select specialized neurons via a cumulative activation threshold k in Equation 1, which is the only hyper-parameter of our method. Here, we discuss the impact of k on the final performance and its relationship to the sparsity. As mentioned in Section 3.1, a smaller factor k results in more sparse specialized neuron selection, which makes the fc1 weight more sparse as well in the Neuron Specialization Training process. In Figure 4, we show that increase k leads to higher improvements in general, and the optimal performance is about when k=95%. Such observation follows the intuition since when k is too low, model capacity will be largely reduced.

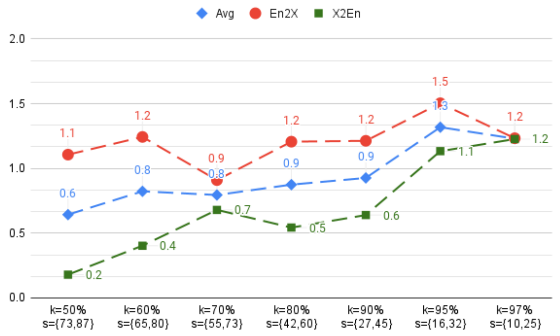

Figure 4: Improvements of Neuron Specialization method over the mT-large baseline on EC30. The x-axis indicates the factor k and the dynamic sparsity of the fc1 layer, with displayed values ranging from minimum to maximum sparsity achieved. The y-axis indicates the SacreBLEU improvements over the mT-large model.

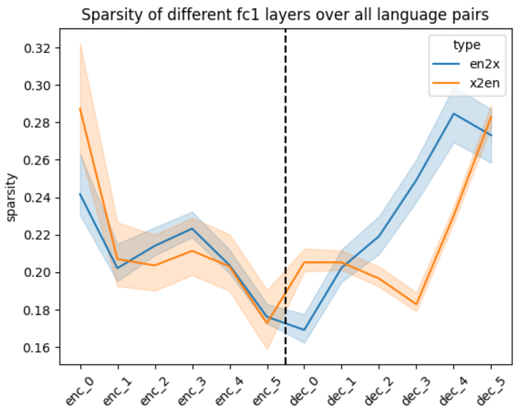

Figure 5: Sparsity progression of Neuron Specialization when $k = 95$ on the EC30. We observe that the sparsity becomes smaller in the Encoder and then goes up in the Decoder. Note that this figure is based on the natural signals extracted from the untouched pre-trained model, and will be leveraged later in the process of Neuron Specialization Training. This intrinsic pattern naturally follows our intuition that specialized neurons progress from language specific to agnostic the in Encoder, and vice versa in the Decoder.

Furthermore, in Figure 5, we show that the spar-

[5] https://github.com/NLP-Playground/LaSS

sity of the network presents an intuitive structure: the sparsity decreases in the Encoder and increases in the Decoder. This implies the natural signal within the pre-trained multilingual model that neurons progress from language-specific to language-agnostic in the Encoder, and vice versa in the Decoder. Such observation is natural because it is reflected by the untouched network, similar to what we observed in the Progression of Neuron overlaps in Section 3.2.2.

A.6 Visualization Details

We provide the additional Pairwise Intersection over Union (IoU) scores for specialized neurons in the first Encoder layer (Figure 6), last Encoder layer (Figure 7), and last Decoder layer (Figure 8). The figures show that the Neurons gradually changed from language-specific to language-agnostic in the Encoder, and vice versa in the Decoder.

Algorithm 1 Specialized Neuron Identification

```
Input: A pre-trained multi-task model $\theta$ with dimensions $d$ and $d_{ff}$; a validation dataset $D$ with $T$ tasks, where $D = \{D_1, ..., D_T\}$; and an accumulation threshold factor $k \in [0\%, 100\%]$ as the only hyper-parameter.
Output: A set of selected specialized neurons $S_k^t$ for each task $t$.
for task $t$ in $T$ do
    Step 1: Activation Recording
    Initialize activation vector $A_t = \mathbf{0} \in \mathbb{R}^{d_{ff}}$
    for sample $x_i$ in $D_t$ do
        Record activation state $a_i^t \in \mathbb{R}^{d_{ff}}$
        $A_t = A_t + a_i^t$                                  ▷ Accumulate activation states
    end for
    $a^t = \frac{A_t}{|D_t|}$                                ▷ Compute average activation state for task $t$
    Step 2: Neuron Selection
    Initialize selected neurons set $S_k^t = \emptyset$
    while selection condition not met do                     ▷ Refer to Eq. 1 for condition
        Select neurons based on $a^t$ and add them to $S_k^t$
    end while
end for
```

Algorithm 2 Neuron Specialization Training

```
Input: A pre-trained multi-task model $\theta$ with dimensions $d$ and $d_{ff}$. Corpora data $C$ with $T$ tasks that contain both training and validation data. A set of selected specialized neurons $S_k^t$ for each task $t$.
Output: A new specialized network $\theta^{new}$. Note that only the fc1 weight matrix will be trained task-specifically, the other parameters are shared across tasks. In addition, $\theta^{new}$ does not contain more trainable parameters than $\theta$ due to the sparse network feature.
Derive boolean mask $m^t \in \{0, 1\}^{d_{ff}}$ from $S_k^t$ for each layer
while $\theta^{new}$ not converge do
    for task $t$ in $T$ do
        $W_1^T = m^t \cdot W_1^\theta$                   ▷ We perform this for all layers, refer to EQ. 3
        Train $\theta^{new}$ using $C^t$       ▷ All parameters will be updated, yet fc1 layers are task specific
    end for
end while
```

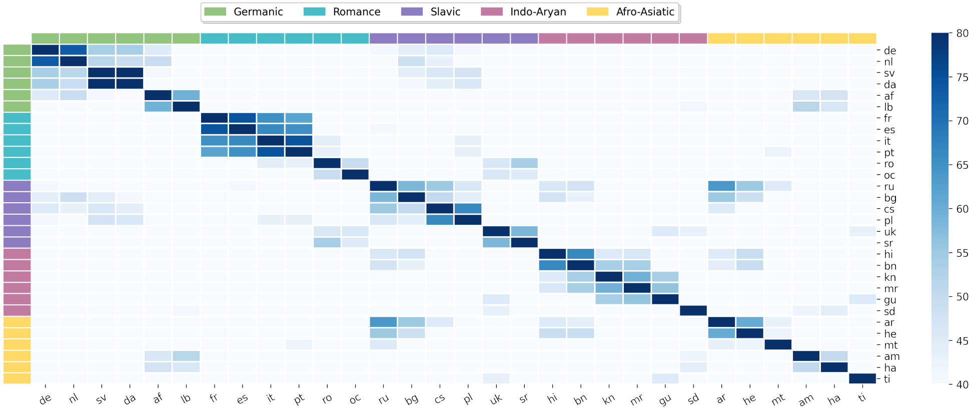

Figure 6: Pairwise Intersection over Union (IoU) scores for specialized neurons extracted from the **first encoder** FFN layer across all X-En language pairs to measure the degree of overlap between language pairs. Darker cells indicate stronger overlap, with the color threshold set from 40 to 80 to improve visibility.

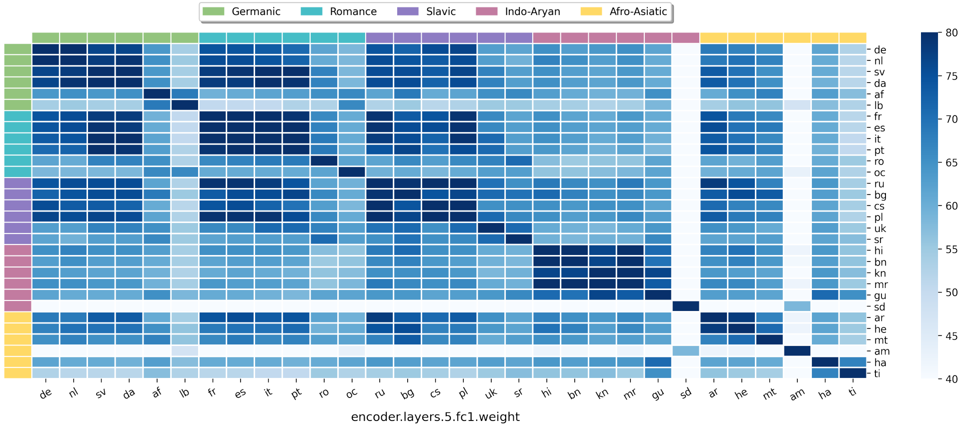

Figure 7: Pairwise Intersection over Union (IoU) scores for specialized neurons extracted from the **last encoder** FFN layer across all One-to-Many language pairs to measure the degree of overlap between language pairs. Darker cells indicate stronger overlap, with the color threshold set from 40 to 80 to improve visibility.

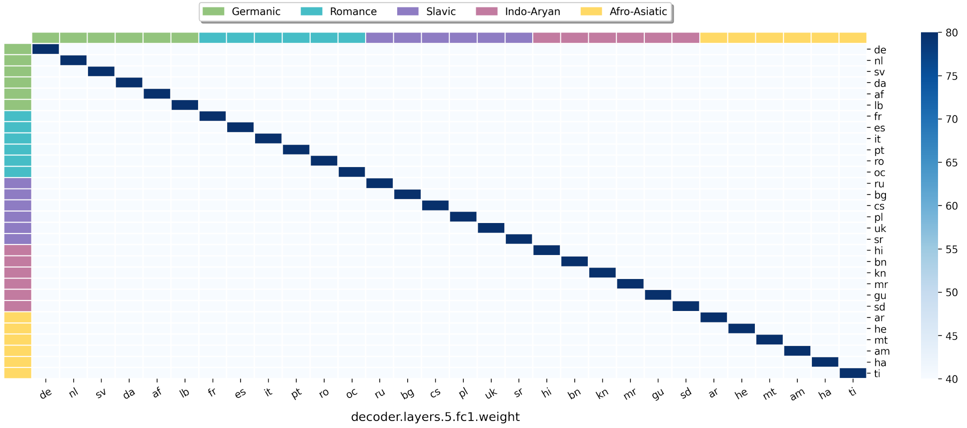

Figure 8: Pairwise Intersection over Union (IoU) scores for specialized neurons extracted from the **last decoder** FFN layer across all X-En language pairs to measure the degree of overlap between language pairs. Darker cells indicate stronger overlap, with the color threshold set from 40 to 80 to improve visibility.

Methods	$\Delta\theta$	High (5M)			Med (1M)			Low (100K)			All (61M)		
		O2M	M2O	Avg	O2M	M2O	Avg	O2M	M2O	Avg	O2M	M2O	Avg
mT-big	-	52.4	57.6	55.0	53.9	56.6	55.3	42.5	50.0	46.3	49.6	54.7	52.2
Adapter$_{LP}$	+87%	+1.3	+0.2	+0.8	+1.1	+0.1	+0.6	+0.3	+0.3	+0.3	+0.9	+0.2	+0.5
Adapter$_{Fam}$	+70%	+0.6	+0.2	+0.4	+0.7	+0.3	+0.5	+1.1	+0.4	+0.8	+0.8	+0.3	+0.5
LaSS	0%	**+1.7**	+0.8	+1.2	**+1.3**	+0.3	+0.8	-0.3	-1.5	-0.9	+0.9	-0.2	+0.5
Random	0%	+0.7	-0.4	+0.2	+0.4	-0.5	-0.1	-0.5	-1.2	-0.9	+0.2	-0.7	-0.3
Ours-Enc	0%	+1.0	+0.9	+1.0	+0.7	+0.9	+0.8	+0.6	+0.9	+0.8	+0.8	+0.9	+0.8
Ours-Dec	0%	+0.9	+0.9	+0.9	+0.6	+1.0	+0.8	+0.5	+1.2	+0.9	+0.7	+1.0	+0.9
Ours	0%	+1.3	**+1.1**	**+1.2**	+1.1	**+0.9**	**+1.0**	**+1.2**	**+0.8**	**+1.0**	**+1.2**	**+0.9**	**+1.1**

Table 7: Average **ChrF++** improvements on the EC30 dataset over the baseline (mT-big), categorized by High, Medium, and Low-resource translation directions. 'Ours-Enc' and 'Ours-Dec' indicate neuron specialization applied solely to the Encoder and Decoder, respectively, while 'Ours' signifies the method applied to both components. The best results are highlighted in **bold**.

Methods	$\Delta\theta$	High (5M)			Med (1M)			Low (100K)			All (61M)		
		O2M	M2O	Avg	O2M	M2O	Avg	O2M	M2O	Avg	O2M	M2O	Avg
mT-big	-	83.4	83.9	83.65	81.1	80.1	80.6	73.8	73.4	73.6	79.1	79.1	79.1
Adapter$_{LP}$	+87%	+0.9	+0.2	+0.5	+0.6	+0.2	+0.4	0	+0.1	0	+0.5	+0.2	+0.4
Adapter$_{Fam}$	+70%	+0.4	+0.1	+0.3	+0.4	+0.2	+0.3	+0.7	+0.3	+0.5	+0.5	+0.2	+0.4
LaSS	0%	**+1.5**	+0.8	**+1.2**	**+0.9**	+0.6	**+0.8**	-0.2	-1.0	-0.6	+0.7	+0.1	+0.4
Random	0%	+0.2	-0.1	+0.1	-0.1	-0.2	-0.2	-0.8	-0.9	-0.9	-0.2	-0.4	-0.3
Ours-Enc	0%	+1.0	+0.8	+0.9	+0.5	+0.9	+0.7	+0.3	**+0.9**	+0.6	+0.6	+0.8	+0.7
Ours-Dec	0%	+0.9	+0.8	+0.9	+0.5	**+1.0**	**+0.8**	+0.3	**+0.9**	+0.6	+0.6	**+1.0**	+0.8
Ours	0%	+1.4	**+1.0**	**+1.2**	**+0.9**	+0.7	**+0.8**	**+0.8**	+0.7	**+0.8**	**+1.0**	+0.8	**+0.9**

Table 8: Average **COMET** improvements on the EC30 dataset over the baseline (mT-big), categorized by High, Medium, and Low-resource translation directions. 'Ours-Enc' and 'Ours-Dec' indicate neuron specialization applied solely to the Encoder and Decoder, respectively, while 'Ours' signifies the method applied to both components. The best results are highlighted in **bold**.