Memory-efficient NLLB-200: Language-specific Expert Pruning of a Massively Multilingual Machine Translation Model

Yeskendir Koishekenov[*1,2] **Alexandre Berard**[1] **Vassilina Nikoulina**[1]
[1]NAVER LABS Europe
[2]University of Amsterdam
{first.last}@naverlabs.com
yeskendir.koishekenov@student.uva.nl

Abstract

The recently released NLLB-200 is a set of multilingual Neural Machine Translation models that cover 202 languages. The largest model is based on a Mixture of Experts architecture and achieves SoTA results across many language pairs. It contains 54.5B parameters and requires at least four 32GB GPUs just for inference. In this work, we propose a pruning method that enables the removal of up to 80% of experts without further finetuning and with a negligible loss in translation quality, which makes it feasible to run the model on a single 32GB GPU. Further analysis suggests that our pruning metrics can identify language-specific experts.

1 Introduction

The Transformer (Vaswani et al., 2017) has become the dominant modeling paradigm in Natural Language Processing tasks. Many subsequent advances in the field came from increasing the computational budget, training data, and model size. Neural Machine Translation was not an exception, where massively multilingual NMT (Aharoni et al., 2019; Fan et al., 2021; Tang et al., 2020; Zhang et al., 2020) demonstrated promising results, while attempting to overcome the curse of multilinguality (Conneau et al., 2019) by scaling up model size.

However, increasing the parameter size exacerbates the cost of training (Yang et al., 2019; Strubell et al., 2019; Patterson et al., 2021) and hurts the memory footprint and inference latency (Dai et al., 2019; Fan et al., 2021; Wang et al., 2022). Sparsely-gated Mixture-of-Experts (MoE) models are an efficient alternative to dense models (Lepikhin et al., 2020; Fedus et al., 2021; Riquelme et al., 2021). For example, Du et al. (2022) demonstrates that an MoE language model results in a 7x larger model compared to GPT-3, but requires only 30% of its energy for training and half of its FLOPs at inference.

Mixture-of-Experts models are neural networks whose set of parameters is partitioned into experts. Contrary to dense models, where all network parameters are used for every input, an MoE model activates different parts of the network, the experts, depending on the input, which is typically done by a gating mechanism at the token level. MoE models are computationally efficient due to expert parallelism (Fedus et al., 2021) across a large number of GPUs, by having each GPU hold a subset of all experts and communicate with the other GPUs when it needs expert outputs for its local batch.

In NLLB-200[1](Costa-jussà et al., 2022), a load balancing regularizer in the objective function (Shazeer et al., 2017) promotes equal distribution of the tokens across experts. This encourages the model to use all the experts and ensures that all GPUs are used equally for the sake of computational efficiency. However, considering a large number of experts, it does not guarantee that all experts will be equally activated for a particular pair of languages at inference. It raises a research question: *are there language-specific experts in multilingual MoE models?* If this is the case, we may be able to prune such models without loss of translation quality for the language pairs of our interest. Reducing memory usage would be useful for a model like NLLB-200, which normally requires at least four 32GB GPUs at inference.

In this work, we define metrics to assess the importance of each expert and prune the least important experts at inference. We aim to avoid finetuning because of its computational cost. In an ideal scenario, we would like to be able to identify the important experts in an MoE model so that practitioners can deploy large models, such as NLLB-200, on a single GPU. We summarize our

[*]Work done during an internship at NAVER LABS Europe

[1]In what follows, NLLB-200 refers to the 54.5B-parameter MoE NLLB model, unless specified otherwise.

main contributions as follows:

- We propose a pruning strategy that can remove 80% of experts in the NLLB-200 model without further finetuning and with a negligible loss in translation quality;
- We find that the decoder experts can be pruned more aggressively than the encoder experts;
- We show the emergence of language-specific experts in the NLLB-200 model;
- We demonstrate that the important language-specific experts in the decoder are shared between linguistically related languages;
- We release the ids of the pruned experts, along with other experts' gathered statistics so that anyone with a single 32GB GPU can use NLLB-200 at inference.[2]

2 Related work

The concept of Mixture-of-Experts models in machine learning dates back to the works of Jacobs et al. (1991); Jordan and Jacobs (1994). Most recent versions were inspired by Shazeer et al. (2017), who achieved state-of-the-art language modeling and translation results with the largest model at that time. Combined with the Transformer model, MoE models grew in popularity (Lepikhin et al., 2020; Fedus et al., 2021). Beyond natural language processing, MoE models showed a large success in computer vision (Puigcerver et al., 2020), speech recognition (You et al., 2021), multi-modal learning (Mustafa et al., 2022), and diffusion models (Feng et al., 2022; Balaji et al., 2022) to name a few. For a more detailed survey of MoE models, we refer readers to Yuksel et al. (2012) and Fedus et al. (2022).

Despite the recent successes, large MoE models require a lot of memory and the contribution (or roles) of experts is under-explored. Chen et al. (2022) showed that the contributions of experts of a pre-trained MoE model in different tasks such as MNLI, CoLA, and SQuAD are quite different. Moreover, they converted a large sparse MoE model pre-trained on a general task to a single-expert dense model by fine-tuning the most 'professional' expert and dropping the other experts. It demonstrates that experts do not contribute equally to the performance and some are more important than others. Zoph et al. (2022) also studied different expert specializations such as sentinel tokens, punctuation, conjunctions and articles, and even languages. They concluded that experts in the encoder exhibit specialization, in contrast to the decoder, but not by language. According to the authors, their mechanism of token routing and load balancing prevents language specialization.

Kudugunta et al. (2021) train study routing mechanisms at different levels of granularity and show that task-level experts (i.e., per language) can achieve similar performance as token-level experts. However, this work assumes that the model is trained this way, while our own work attempts to prune an existing token-level MoE model at inference without re-training it.

There have been a number of attempts to compress existing massively multilingual NMT models (Costa-jussà et al., 2022; Mohammadshahi et al., 2022b,a). However, to the best of our knowledge, none of them explicitly studied expert pruning and the emergence of language-specific experts in a large MoE model like we do. There has been a related line of works on pruning attention heads in transformer models (Michel et al., 2019; Voita et al., 2019), demonstrating linguistically-interpretable roles of attention heads (Voita et al., 2019; Jo and Myaeng, 2020) and the emergence of language-specific attention heads (Kim et al., 2021b; Held and Yang, 2022). Understanding the role of attention heads helps carefully remove the least important ones without damage to translation quality.

Closest to our work, Kim et al. (2021a) tried to prune a machine translation MoE model by keeping the most activated experts,[3] but did not manage to preserve performance without further fine-tuning.

Even though it has been shown that multilingual NMT models benefit from a larger number of experts (Costa-jussà et al., 2022), to the best of our knowledge, our work is the first to study whether any language-specific experts emerge in a massively multilingual Mixture-of-Expert model for NMT, and how can redundant (or non-relevant) experts be pruned.

3 Background

3.1 Mixture-of-Experts models

Sparsely-gated Mixture-of-Experts (MoE) models activate a subset of their parameters per input token, contrary to dense models, where the entire network is used for each input token. Therefore,

[2] `https://europe.naverlabs.com/research/natural-language-processing/nllb-200-expert-pruning`

[3] Equivalent to our *activity* pruning metric.

the total amount of parameters can be significantly increased because the computation cost per token becomes only proportional to the size of the activated sub-network, not the total model size. An increased number of parameters unlocks significant representational capacity. Allocating different devices for different experts and running them in parallel (i.e., expert parallelism, Fedus et al., 2021), in combination with data parallelism makes MoE computationally efficient and highly scalable (Fedus et al., 2021; Lepikhin et al., 2020).

In the MoE Transformer models proposed by Lepikhin et al. (2020), the FFN sublayers in the dense model are replaced with MoE layers. An MoE layer takes an input token representation x_t and then routes it to the top-k experts selected from a set $\{E_i\}_{i=1}^{N}$ of N experts thanks to a gating network:

$$G_t = softmax(W_g \cdot x_t) \tag{1}$$

Where $W_g \in \mathbb{R}^{N \times d}$ is a learned parameter. The output of the MoE layer is a weighted sum of the outputs of the k selected experts $\mathcal{E}$:

$$y_t = \frac{1}{\sum_{i \in \mathcal{E}} G_{t,i}} \sum_{i \in \mathcal{E}} G_{t,i} E_i(x_t) \tag{2}$$

3.2 NLLB-200

No Language Left Behind (NLLB-200) is a set of massively multilingual NMT models that can translate to and from 202 languages (Costa-jussà et al., 2022), including many very low resources languages. Models of varying sizes have been released. The largest one is a Mixture-of-Experts model and has 54.5B parameters. A dense model of 3.3B models is also available, which has the same architecture as the 54.5B MoE model without the experts. In this work, we will attempt to prune the experts from the 54.5B model while using the 3.3B variant as a *lower-bound* baseline.[4]

In the 54.5B MoE model, every 4th FFN sublayer – in both the encoder and decoder – is replaced by an MoE layer, starting at the 4th layer (this makes 12 layers with experts). Each MoE layer consists of 128 experts (1536 experts in total) with the same architecture as an FFN sublayer, and has its own gating network, following the top-k gating algorithm of Lepikhin et al. (2020) and selecting the top-2 experts per token without any randomization. The model was trained with a linear combination of label-smoothed cross-entropy (Szegedy et al., 2016) with an auxiliary load balancing loss (Shazeer et al., 2017), which encourages tokens to be uniformly distributed across experts.

Memory usage. The 3.3B and 54.5B models are Transformers with an embedding dimension of 2048, an FFN dimension of 8192, 16 attention heads, 24 encoder layers, and 24 decoder layers. When storing their parameters in half precision, the 3.3B dense model and 54.5B MoE model take respectively 6.2GiB and 101.5GiB of memory. Each expert has 33.6M parameters, representing 51.6B parameters in total or 96GiB of memory. While the 3.3B model can easily run on a single GPU, the 54.5B model requires at the very least 4 32GB GPUs to run. To maximize efficiency, decoding with the MoE model has to be done with expert parallelism (Fedus et al., 2021), with each GPU holding a full copy of the "dense" parameters (2.9B or 5.5GiB) and $1/N^{\text{th}}$ of the experts per layer, where N is the number of GPUs.[5] Because of the memory usage of beam search decoding and memory fragmentation, batched decoding actually requires more GPUs in practice (e.g., 6 or 8), or to offload the encoder and decoder to the CPU when they are not used.[6]

4 Our Approach

We experiment with different experts' pruning metrics and strategies that allow us to select the most relevant experts per language or language pair, and thus significantly reduce the memory usage at inference time of NLLB-200.

4.1 Expert pruning metrics

The pruning metric should quantify the contribution of a given expert to the translation. Intuitively, experts that were more involved in translation should be considered more important.

Activity. We define the *Top 1 activity*, $top_1(e)$, of an expert e as the fraction of tokens routed to this expert as the first choice (i.e., the frequency at which this expert was ranked first by the gating mechanism). We also consider the *Top 2 activity*

[4]If the pruned models' performance is worse than the 3.3B baseline, there is no point in using the MoE model, which is larger and more cumbersome to use.

[5]This brings the memory usage to 118GiB (or 29.5GiB per GPU) when decoding on 4 GPUs.

[6]Memory usage can be divided by almost two by encoding the full test set with the encoder and then moving the encoder to CPU and decoder to GPU.

variant, $top_2(e)$, with the fraction of tokens routed to this expert as their first or second choice.

Using only *activity* as an importance metric can be sub-optimal as it does not take into account the gating value assigned to this expert by the model.

Load Balancing. We experiment with the *load balancing* pruning metric, similar to the load balancing loss used by Costa-jussà et al. (2022) to train the MoE model. It is defined as the product of the *activity* and the average gate value: $LB(e) = top_1(e) \times mean(e)$.

Importance. Following the definition of attention head confidence by Voita et al. (2019), we define the *confidence* of an expert, $conf(e)$, as its average gate value when it is *ranked first*. Then, we can define the "vanilla" *importance* of an expert as the product of its' *activity* and *confidence*.[7]

$$imp_{vanilla}(e) = top_1(e) \times conf(e) \qquad (3)$$

We define *importance* as an improved version of *vanilla importance* with an exponential to smooth the confidence values:

$$imp(e) = top_1(e) \times \exp\left(conf(e)\right) \qquad (4)$$

4.2 Expert statistics granularity

To compute the pruning metrics defined above, for each expert $e \in \{1, \dots, 1536\}$[8] we collect the gate statistics, $top_1(e)$, $top_2(e)$, $mean(e)$ and $conf(e)$, by decoding the validation sets for all language directions.[9] However, these statistics can be aggregated at different granularity levels. Depending on how these statistics are aggregated, we hope to see language-specific experts emerge. In our experiments, we consider three different granularities:

- *global*: we aggregate the statistics across all language pairs to keep the overall best experts;
- *language-pair*: we collect gate statistics for each language pair and thus keep a (potentially) different set of experts for each language pair;
- *language-specific*: we aggregate encoder-side statistics per source language and decoder-side statistics per target language, which will let us keep a single set of encoder/decoder experts per source/target language.

4.3 Expert pruning algorithm

Using the pruning metrics defined in Section 4.1, there are different expert pruning strategies that we can adopt. The pruning metric values are normalized to sum to one in each layer, and experts are sorted from most important to least important.

Fixed per layer. First, the simplest way is to retain a fixed amount of top experts in each layer. For example, 75% pruning retains 384 out of 1536 experts, which corresponds to 32 experts per layer. In the *balanced* setting, the number of experts per layer is the same in the encoder and decoder (e.g., 32 per layer). In the *unbalanced* setting, we keep a different number of experts in the encoder and decoder (e.g., 40 per encoder layer and 24 per decoder layer).

Global threshold. The pruning metrics we defined let us easily prune experts per layer, but not globally. To select *globally best* experts (with no *a priori* on the number of experts per layer) we search for a global threshold θ such that:

$$\sum_{k=1}^{12} min(n_k \mid \sum_{i=1}^{n_k} \phi(e_i^k) \geq \theta) = count \qquad (5)$$

Where ϕ is the pruning metric; k the layer id (out of 12 layers with experts); e_i^k the i^{th} expert in the sorted list of experts for that layer; and *count* the desired total number of experts to retain (e.g., 384 for 75% pruning). Experts $\{e_i^k\}_{i=1}^{n_k}$ are then retained and the rest are pruned.[10] In our experiments, we make sure to keep at least 4 experts per layer.[11]

Our intuition behind this pruning method is to define a constant probability mass (or "importance" mass) each layer should have. Keeping only a couple of experts in a layer is fine if they are collectively used a majority of the time. Conversely, some layers may need more experts if expert usage is more uniformly distributed.

Figure 1 illustrates how experts are distributed among layers with this approach at 75% pruning and with the top_1 metric. We see that the decoder requires much fewer experts per layer than the encoder to reach the same activity threshold.

[7]Using confidence alone as a pruning metric has demonstrated very poor performance in our preliminary experiments, and therefore was not retained for the follow up study.

[8]12 layers with 128 experts each = 1536 experts

[9]We always use beam search with a beam size of 4.

[10]We iterate from 0 to 1 by increments of 0.001 until we find a value of θ which satisfies this equation.

[11]To be able to decode on up to 4 GPUs and to limit the risk of degenerate behavior because some layers have too few experts. Also since NLLB-200 uses top-2 gating, we need at least 2 experts per layer.

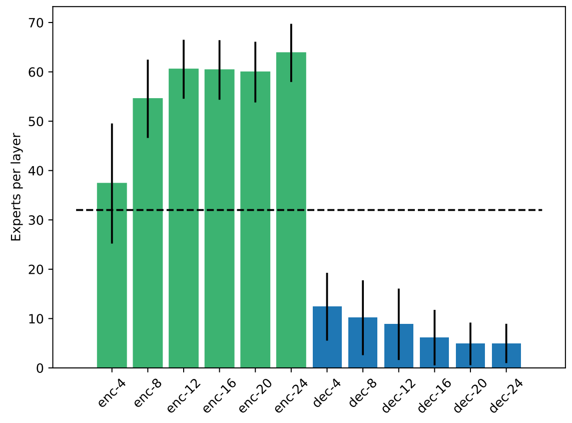

Figure 1: Average number of experts per layer after pruning 75% of experts with the global threshold algorithm (average activity threshold: 0.69). Pruning is done per language direction and the values are averaged over the 870 directions of the valid set.

We also experiment with a variant of this method, which we call **Enc/Dec thresholds**, with a fixed amount in the encoder and decoder (e.g., 192 and 192) and thresholds that are defined independently in the encoder and decoder.

5 Experiments

5.1 Evaluation settings

In our experiments, we use the FLORES-200 benchmark (Costa-jussà et al., 2022), which consists of translations of 3001 English sentences (from 842 distinct Wikipedia articles) to all other 201 languages. The multi-parallel nature of this dataset makes it possible to evaluate performance in all 40 602 language directions. As our final test benchmark, we take a representative subsample of 53 languages out of 202, which were also used as an ablation dataset by Costa-jussà et al. (2022). In our intermediate experiments, we work with a smaller subset of 30 out of 53 languages, with 10 languages per resource type (high, low, very low) and covering the same fourteen language families as the full subset of 53 languages. More details on the languages considered in our experiments as well as the amount of resources available per category are provided in Tables 8 and 14 in Appendix.

To evaluate translation quality we use two metrics: chrF++[12] (Popović, 2015) and spBLEU[13] (Costa-jussà et al., 2022). BLEU is heavily tokenization-dependant and its implementations do not include tokenizers for most of the NLLB-200 languages. spBLEU overcomes this issue by tokenizing the references and model outputs with a multilingual SentencePiece tokenizer (SPM-200, Costa-jussà et al., 2022). We report chrF++ results in the main paper and spBLEU results in Appendix. We use FLORES-200 dev (which we call *valid*) for collecting MoE gate statistics and comparing different pruning algorithms and rates, and FLORES-200 devtest (which we call *test*) for reporting final results and comparing with the 3.3B and 54.5B baselines.

5.2 Results

In the first set of experiments, we work with a subset of 30 languages. Table 1 compares different expert pruning metrics and strategies under a 75% pruning rate. The experts are selected per language pair, and the scores are averaged per resource type (high, low, very low). The first part of the table reports two baselines: an upper bound corresponding to the full (unpruned) 54.5B MoE model, and a lower bound being the 3.3B dense model (same architecture without experts).

Pruning metric The second part of Table 1 compares the chrF++ performance of different pruning metrics (spBLEU score are reported in Appendix Table 9). From these results, we can see that the *top-1 activity* and *importance* metrics are the most effective at identifying important experts. Further experiments with global threshold pruning (third part of Table 1) confirm the slightly better performance of the *importance* metric which we keep as the default for the next experiments.

Pruning algorithm Table 1 also compares the pruning algorithms described in Section 4.3 (*fixed per layer* and *global threshold*). Note that with *fixed per layer*, we can either allocate the same expert budget in the encoder and decoder (balanced setting) or have more experts in the encoder (unbalanced setting).

First, we see that the *global threshold* strategy gives the best results overall, with the same average chrF++ as the full unpruned model. However, *global threshold* is not very practical for several reasons. First, it identifies a different amount of experts per layer for each language pair, which leads to variable memory usage across language pairs. It also requires recreating and reloading the model when decoding multiple directions, which is very

[12]SacreBLEU signature for chrF++: `nrefs:1|case:mixed|eff:yes|nc:6|nw:2|space:no|version:2.3.1`

[13]SacreBLEU signature for spBLEU: `nrefs:1|case:mixed|eff:no|tok:flores200|smooth:exp|version:2.3.1`

Method	Metric	High→High	High→Low	High→V. low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model (Costa-jussà et al., 2022)		44.54	38.20	30.08	40.49	35.19	27.61	35.27	30.68	24.75	34.06
54.5B MoE model (Costa-jussà et al., 2022)		45.90	39.19	30.24	**42.29**	**36.35**	28.18	**36.55**	**32.16**	24.93	35.07
Fixed per layer (balanced)	Top 1	45.52	38.75	30.13	41.51	35.50	27.92	36.09	31.68	24.90	34.64
	Top 2	44.38	37.92	29.60	40.56	34.86	27.48	35.24	30.97	24.54	33.93
	Load balancing	44.48	38.06	29.64	40.67	34.95	27.56	35.29	31.04	24.59	34.01
	Importance (vanilla)	42.87	34.73	28.40	40.92	34.17	27.46	34.96	29.71	23.99	33.00
	Importance	45.59	38.76	30.18	41.50	35.41	27.87	36.15	31.69	24.96	34.66
Global threshold	Top 1	46.01	39.28	30.44	41.91	36.18	28.21	36.40	31.97	25.06	35.03
	Importance	**46.10**	**39.31**	**30.46**	41.99	36.25	**28.29**	36.47	32.09	**25.10**	**35.09**
Fixed per layer (unbalanced)	Importance	45.79	39.00	30.33	41.80	35.76	28.12	36.36	31.93	**25.10**	34.89
Enc/Dec thresholds (balanced)		45.57	38.73	30.07	41.52	35.36	27.81	36.13	31.62	24.88	34.61
Enc/Dec thresholds (unbalanced)		45.88	38.97	30.28	41.92	35.85	28.10	36.39	31.84	25.06	34.90

Table 1: chrF++ valid scores on 30 languages of different pruning algorithms and metrics, with 75% pruning (i.e., 384 experts are kept in total). The unbalanced approaches keep 240 encoder experts and 144 decoder experts.

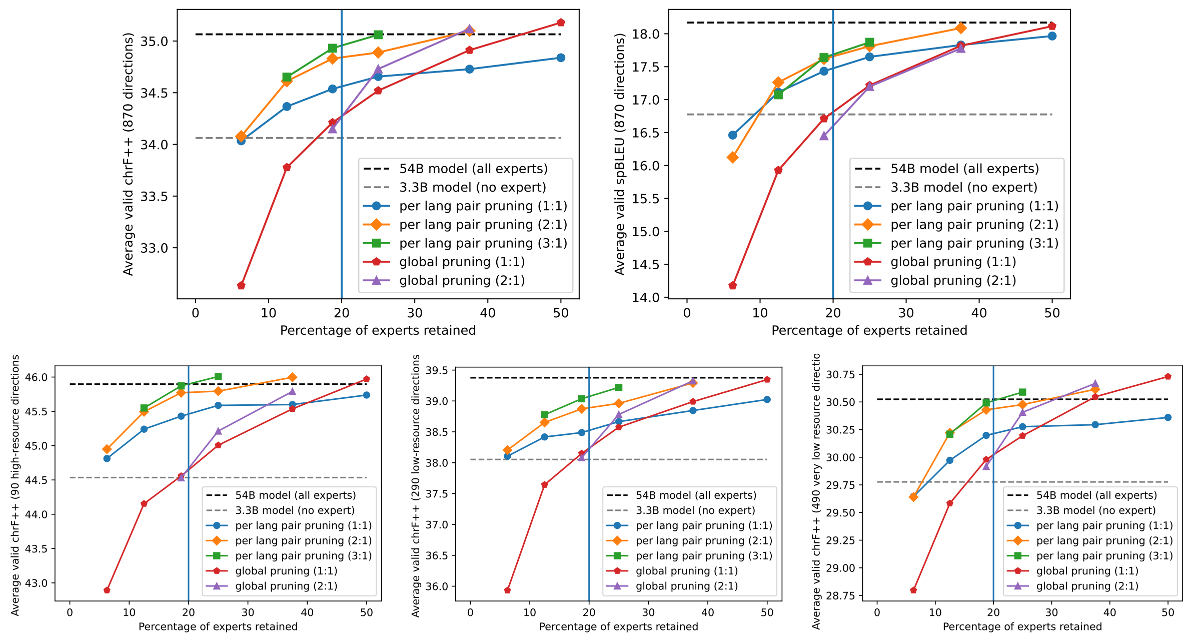

Figure 2: chrF++ and spBLEU valid scores on 30 languages for different resource types as a function of the percentage of experts retained. Pruning is done per language pair with the *importance* metric and with a fixed number of experts per layer.

slow. Finally, we found that it was more sensitive to over-generation and hallucinations (which we elaborate on in Section A in Appendix) at higher pruning rates. The *enc/dec thresholds* approach does not suffer from all the limitations of *global threshold*, but it is not better than *fixed per layer* either. Therefore, for simplicity, we pick the *fixed per layer* approach for our next experiments.

Balanced versus unbalanced pruning When retaining 25% of experts (384 out of 12×128), *global threshold* keeps on average 335 encoder experts and 49 decoder experts. The number of selected experts in the encoder and decoder for different language resource types is shown in Table 16 in Appendix. Following this observation that encoder experts seem more important than decoder ones, we experiment with different encoder/decoder ratios. **1:1** is the balanced setting. **2:1** and **3:1** are unbalanced with respectively twice and three times as many encoder experts as decoder experts. Figure 2 shows that **3:1** performs the best across almost all pruning rates and resource types.

Pruning with global statistics. Figure 2 and Figure 4 in Appendix also show that the same experts can be pruned across all language pairs (with statistics aggregated over all directions) with no loss in performance at 50% pruning. Statistics at the language-direction granularity let us safely prune up to 80% of the experts (in the unbalanced setting), which makes the model small enough to fit on a single GPU.

Test results and language-specific pruning. Finally, we validate our results over the test set on 53 languages (2 756 directions). We use the *fixed per layer* approach with a **3:1** ratio, which showed

Method	Enc experts	Dec experts	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	44.18	38.30	31.45	38.24	34.60	27.93	35.93	32.02	26.47	35.81
54.5B MoE model	768	768	**45.41**	38.98	**31.89**	**39.72**	**35.40**	**28.83**	**37.29**	**33.23**	**26.95**	**36.81**
Fixed per layer (lang-pair)	216	72	45.37	39.06	31.79	39.20	35.03	28.47	37.05	33.16	26.63	36.59
Fixed per layer (global)	216	72	43.20	37.60	31.68	37.37	33.94	28.40	35.38	31.97	26.84	35.34
Fixed per layer (lang)	216	72	45.35	**39.10**	31.82	39.18	35.10	28.51	37.02	33.19	26.62	36.61

Table 2: chrF++ test scores on 53 languages, with the *importance* metric for 80% pruning (1-GPU decoding).

Method	Enc experts	Dec experts	High→High	High→Low	High→V. low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	45.54	38.84	32.72	39.18	34.87	29.07	38.39	34.11	29.21	34.64
54.5B MoE model	768	768	**46.68**	39.36	**33.56**	**40.53**	35.49	**30.07**	**40.46**	**35.49**	**30.16**	**35.74**
Fixed per layer (lang)	216	72	46.67	**39.59**	33.33	40.19	**35.50**	29.67	39.94	35.29	29.50	35.46

Table 3: chrF++ test scores on all 202 languages, with the *importance* metric for 80% pruning (1-GPU decoding).

Encoder \ Decoder	En→Fr	En→Ur	Ast→Ur	Ur→Fr	Ur→Ast	Fr→Ast	Ast→Ko	Fr→Ko	Ko→En	Ko→Ast	Fr→En	Ast→En
En→Fr	NA	0.18	0.17	**0.71**	0.31	0.32	0.17	0.17	0.19	0.31	0.17	0.17
En→Ur	**1.00**	NA	**0.68**	0.20	0.24	0.22	0.31	0.30	0.26	0.23	0.21	0.23
Ast→Ur	0.35	0.35	NA	0.22	0.25	0.22	0.37	0.35	0.21	0.22	0.18	0.20
Ur→Fr	0.36	0.36	0.40	NA	0.39	0.38	0.19	0.19	0.22	0.39	0.16	0.19
Ur→Ast	0.36	0.36	0.40	**1.00**	NA	**0.83**	0.27	0.25	0.24	**0.87**	0.17	0.22
Fr→Ast	0.42	0.42	0.45	0.44	0.44	NA	0.25	0.24	0.24	**0.80**	0.20	0.22
Ast→Ko	0.35	0.35	**1.00**	0.40	0.40	0.45	NA	**0.78**	0.15	0.24	0.15	0.16
Fr→Ko	0.42	0.42	0.45	0.44	0.44	**1.00**	0.45	NA	0.14	0.23	0.15	0.13
Ko→En	0.33	0.33	0.41	0.48	0.48	0.44	0.41	0.44	NA	0.26	**0.55**	**0.70**
Ko→Ast	0.33	0.33	0.41	0.48	0.48	0.44	0.41	0.44	**1.00**	NA	0.17	0.22
Fr→En	0.42	0.42	0.45	0.44	0.44	**1.00**	0.45	**1.00**	0.44	0.44	NA	**0.61**
Ast→En	0.35	0.35	**1.00**	0.40	0.40	0.45	**1.00**	0.45	0.41	0.41	0.45	NA

Table 4: The Jaccard similarity of selected 25% important experts between different language pairs in the encoder (lower triangle) and decoder (upper triangle). Pruning is done per language pair with the *importance* metric. The same number of experts were chosen for the encoder and decoder with thresholding.

the best results on the validation set at 80% (minimum rate for 1-GPU decoding). Tables 2 and 11 report these test scores with three different levels of granularity: *global*, *language-pair-specific* or *language-specific* (as described in Section 4.2). Table 10 in the Appendix reports valid scores with the same settings.

Pruning important experts chosen per language pair gives 0.8 chrF++ more on average than the 3.3B dense model, and 0.2 chrF++ less than the full MoE model. Global pruning on the other hand performs worse than both the MoE and dense models, which confirms the importance of having a language-specific pruning strategy.

While choosing important experts for each language pair is effective, it is not very practical: with L languages, this generates $L \times (L-1)$ different configurations. A more practical approach is to prune encoder experts per source language and decoder experts per target language (i.e., *language-specific* pruning). This pruning strategy performs exactly as well as pruning per language direction and is more convenient. Following this observation, we extract per-language gate statistics on all 202 languages.[14] Then, we apply 80% per-layer pruning with the *importance* metric (at the language granularity) and decode the test set in all 40 602 directions. Tables 3 and 12 report the chrF++ and spBLEU scores. Table 13 reports average score deltas with the unpruned model (and standard deviation per resource type). To facilitate future research and give the opportunity for anyone with a 32GB GPU to run the NLLB-200 model, we release the detailed gate statistics and the ids of the selected experts. We also share the scores for each direction and the decoding outputs of our best pruning approaches.

6 Discussion

6.1 Inference speed and compute budget

Table 5 reports the inference speed of different models: the 3.3B dense model, the full MoE model, and the MoE model with 80% pruning. We see that with 80% pruning, the MoE model requires a single 32GB V100 and performs approximately as fast as the full model on 4 GPUs. If 4 GPUs are available, 80% pruning can double the inference speed of the

[14]By decoding 25 random line pairs per language direction, resulting in 5 025 lines per source language and per target language. To speed up this process, we do teacher forcing instead of beam-search decoding, which we found to perform as well.

Model	Batch size	GPUs	WPS	Time (s)
54.5B	16k	8	195	105
		4	156	131
80% pruning	16k	4	299	79
	4k	1	172	135
3.3B	4k	1	246	86

Table 5: Inference speed benchmark for the 3.3B dense baseline model, the full MoE model, and its pruned version with 36 experts per encoder layer and 12 per decoder layer. We decode the FLORES valid set from 29 languages into English and average the decoding time or words per second.

MoE model.

Table 15 in Appendix gives a breakdown of the number of GPU hours used for this work.

6.2 Similarity of selected experts

Section 5.2 shows that only a fraction of all experts is necessary to translate between two given languages. We analyze the experts selected by our pruning method, to verify whether we can claim that there are indeed language-specific experts. In order to do so, we select experts with our proposed *importance* metric and prune them per language pair at a 75% rate with the *Enc/dec thresholds* method, so that both the encoder and decoder have the same number of experts. We then compute the Jaccard similarity of selected encoder/decoder experts between different language pairs sharing the same source or target language. The lower and upper triangles of Table 4 show this similarity in the encoder and decoder respectively. We see that the encoder experts are independent of the target language (even though pruning is based on statistics collected at the lang-pair granularity level). This is an expected result, and it is due to the model design, where the target language code is introduced on the decoder side only: the encoder representation is not impacted by the target language. We note that the similarity between different source languages is also quite high (30-50%). The similarity between important decoder experts for the same target language is in the 68-87% range; and in the 13-39% range for different target languages. These observations combined with the results in Section 5.2 suggest the emergence of language-specific experts in the NLLB-200 model.

6.3 Similarity of languages based on the importance metric

Finally, we compare expert statistics across different languages, to better understand whether knowledge transfer happens at the expert level between similar languages. We gather importance metrics for each expert in the decoder for each language and concatenate the values of all MoE layers to have one feature vector of dimension 768. Then we do hierarchical clustering and show it as a dendrogram in Figure 3, where we highlight different language subgroupings with different colors. We can see that some clusters contain linguistically related languages, such as Yue Chinese, Korean and Japanese; Russian and Belarussian; or Portuguese, Asturian, and French. We run a similar analysis on the encoder experts and also observe meaningful language clustering, but less clear (Appendix Figure 7).

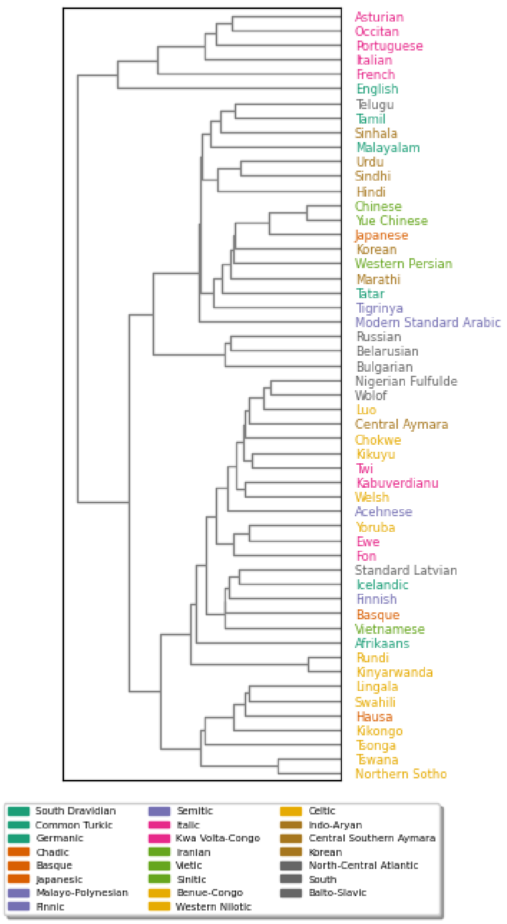

Figure 3: Hierarchical clustering of languages based on the *importance* metric of experts in the decoder. Different colors represent different language subgroupings.

6.4 Discrepancy between chrF++ and spBLEU scores

We observed that our pruning method results in slightly higher performance drop according to spBLEU, than with chrF++. We hypothesize that it is due to a rare but visible phenomenon of overgeneration (and sometimes hallucinations). In the majority of cases, the translation is accurate initially but subsequently includes repetitions, paraphrasing, or slight hallucinations. The spBLEU metric penalizes this behavior more than chrF++, which could account for the variation in scores observed. More details on this are in Section A in Appendix.

7 Conclusion

In this paper, we study expert pruning in the NLLB-200 Mixture-of-Experts MT model. We propose expert pruning metrics based on gate statistics collected while decoding. We study several pruning strategies and demonstrate that it is possible to prune up to 80% of experts with a negligible loss in performance, which makes it possible to decode on a single 32GB GPU. We compare pruning at three levels of granularity: per language direction, per language, or global. Language-specific and language-pair pruning perform the same but the former is the most convenient. Global pruning (i.e., pruning always the same experts regardless of the source and target languages) performs surprisingly well but worse than language-specific pruning, which suggests that there are indeed some language-specific experts. This latter hypothesis is confirmed by our analysis of the selected experts.

8 Risks and Limitations

In our work, we rely on a single Mixture-of-Experts NMT model which is NLLB-200. There is a risk that our conclusions may only hold for this particular model and are specific to the way this model was trained. We believe that our findings still can be of interest to any person willing to use the NLLB-200 model because: (1) It was the only publicly-available MoE NMT model at the time of submission; (2) It is the only model covering 202 languages and reaching SoTA results for most of those languages.

Moreover, we did not try to finetune the pruned model, which could potentially improve the results (but requires a large number of GPUs) and therefore change some of our conclusions.

This work has similar risks as the original NLLB-200 models regarding the misuse of potentially wrong translations. Note that, as observed by Mohammadshahi et al. (2022b), pruning could amplify the biases already present in the full model.

Acknowledgement

This work was completed during a research internship at NAVER LABS Europe. Yeskendir Koishekenov is also supported by ELLIS Amsterdam and Qualcomm AI Research.

References

Roee Aharoni, Melvin Johnson, and Orhan Firat. 2019. Massively multilingual neural machine translation. *arXiv preprint arXiv:1903.00089*.

Yogesh Balaji, Seungjun Nah, Xun Huang, Arash Vahdat, Jiaming Song, Karsten Kreis, Miika Aittala, Timo Aila, Samuli Laine, Bryan Catanzaro, et al. 2022. ediffi: Text-to-image diffusion models with an ensemble of expert denoisers. *arXiv preprint arXiv:2211.01324*.

Tianyu Chen, Shaohan Huang, Yuan Xie, Binxing Jiao, Daxin Jiang, Haoyi Zhou, Jianxin Li, and Furu Wei. 2022. Task-specific expert pruning for sparse mixture-of-experts. *arXiv preprint arXiv:2206.00277*.

Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Unsupervised cross-lingual representation learning at scale. *arXiv preprint arXiv:1911.02116*.

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*.

Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V Le, and Ruslan Salakhutdinov. 2019. Transformer-xl: Attentive language models beyond a fixed-length context. *arXiv preprint arXiv:1901.02860*.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2022. GLaM: Efficient scaling of language models with mixture-of-experts. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR.

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. *J. Mach. Learn. Res.*, 22(107):1–48.

William Fedus, Jeff Dean, and Barret Zoph. 2022. A review of sparse expert models in deep learning. *arXiv preprint arXiv:2209.01667*.

William Fedus, Barret Zoph, and Noam Shazeer. 2021. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity.

Zhida Feng, Zhenyu Zhang, Xintong Yu, Yewei Fang, Lanxin Li, Xuyi Chen, Yuxiang Lu, Jiaxiang Liu, Weichong Yin, Shikun Feng, et al. 2022. Ernie-vilg 2.0: Improving text-to-image diffusion model with knowledge-enhanced mixture-of-denoising-experts. *arXiv preprint arXiv:2210.15257*.

William Held and Diyi Yang. 2022. Shapley head pruning: Identifying and removing interference in multilingual transformers. *arXiv preprint arXiv:2210.05709*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Jae-young Jo and Sung-Hyon Myaeng. 2020. Roles and utilization of attention heads in transformer-based neural language models. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 3404–3417.

Michael I Jordan and Robert A Jacobs. 1994. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214.

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021a. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465*.

Zae Myung Kim, Laurent Besacier, Vassilina Nikoulina, and Didier Schwab. 2021b. Do multilingual neural machine translation models contain language pair specific attention heads? *arXiv preprint arXiv:2105.14940*.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference. *arXiv preprint arXiv:2110.03742*.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

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

Ali Mohammadshahi, Vassilina Nikoulina, Alexandre Berard, Caroline De Brun, James Henderson, and Laurent Besacier. 2022a. Small-100: Introducing shallow multilingual machine translation model for low-resource languages. *ArXiv*, abs/2210.11621.

Ali Mohammadshahi, Vassilina Nikoulina, Alexandre Berard, Caroline De Brun, James Henderson, and Laurent Besacier. 2022b. What do compressed multilingual machine translation models forget? *ArXiv*, abs/2205.10828.

Basil Mustafa, Carlos Riquelme, Joan Puigcerver, Rodolphe Jenatton, and Neil Houlsby. 2022. Multimodal contrastive learning with limoe: the language-image mixture of experts. *arXiv preprint arXiv:2206.02770*.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*.

Maja Popović. 2015. chrf: character n-gram f-score for automatic mt evaluation. In *Proceedings of the Tenth Workshop on Statistical Machine Translation*, pages 392–395.

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, Brussels, Belgium. Association for Computational Linguistics.

Joan Puigcerver, Carlos Riquelme, Basil Mustafa, Cedric Renggli, André Susano Pinto, Sylvain Gelly, Daniel Keysers, and Neil Houlsby. 2020. Scalable transfer learning with expert models. *arXiv preprint arXiv:2009.13239*.

Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. 2021. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. *arXiv preprint arXiv:1906.02243*.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. 2016. Rethinking the inception architecture for computer vision. In

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826.

Yuqing Tang, Chau Tran, Xian Li, Peng-Jen Chen, Naman Goyal, Vishrav Chaudhary, Jiatao Gu, and Angela Fan. 2020. Multilingual translation with extensible multilingual pretraining and finetuning. *arXiv preprint arXiv:2008.00401*.

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, David Talbot, Fedor Moiseev, Rico Sennrich, and Ivan Titov. 2019. Analyzing multi-head self-attention: Specialized heads do the heavy lifting, the rest can be pruned. *arXiv preprint arXiv:1905.09418*.

Hongyu Wang, Shuming Ma, Li Dong, Shaohan Huang, Dongdong Zhang, and Furu Wei. 2022. Deepnet: Scaling transformers to 1,000 layers. *arXiv preprint arXiv:2203.00555*.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *Advances in neural information processing systems*, 32.

Zhao You, Shulin Feng, Dan Su, and Dong Yu. 2021. Speechmoe: Scaling to large acoustic models with dynamic routing mixture of experts. *arXiv preprint arXiv:2105.03036*.

Seniha Esen Yuksel, Joseph N Wilson, and Paul D Gader. 2012. Twenty years of mixture of experts. *IEEE transactions on neural networks and learning systems*, 23(8):1177–1193.

Biao Zhang, Philip Williams, Ivan Titov, and Rico Sennrich. 2020. Improving massively multilingual neural machine translation and zero-shot translation. *arXiv preprint arXiv:2004.11867*.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. St-moe: Designing stable and transferable sparse expert models, 2022. *URL https://arxiv.org/abs/2202.08906*.

A Discrepancy between chrF++ and spBLEU scores

The spBLEU scores (Figure 2 top right, or Figure 4 and Tables 9 and 11) do not show exactly the same trend as chrF++. The gap between the full models and their pruned versions is slightly higher. This is likely caused by a rare but visible phenomenon of over-generation (and sometimes hallucinations). Table 7 shows some examples of such over-generation (with **3:1** *fixed per layer* lang-pair pruning at 80%). Most of the time, the translation is correct, but then continues with repetitions of itself, paraphrasing, or slight hallucinations. This behavior is more penalized by spBLEU than chrF++, which may explain the difference in scores. For instance, when duplicating the FLORES valid English-French translation output of the 54.5B model (i.e., concatenating each output sentence with itself), we see a spBLEU drop of 47% and a chrF++ drop of only 13%. The *global threshold* method is more sensitive to this phenomenon. For instance, 80% pruning leads to a 1.75 spBLEU drop (vs 0.53 for the *fixed per layer* method). We report in Table 6 the difference in length ratio (reported by SacreBLEU, Post, 2018) between the pruned models and the full model. We observe that *global threshold* at 80% has an average length ratio delta with the full model of 0.16 (meaning it generates longer outputs), while *fixed per layer* has 0.04. We hypothesize that this over-generation issue may be mitigated by identifying experts that are specialized in generating the end-of-sequence symbol, but this is the subject of future work.

Method	Enc experts	Dec experts	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	0.02±0.02	0.04±0.03	0.09±0.06	0.02±0.03	0.04±0.04	0.11±0.06	0.06±0.05	0.07±0.07	0.15±0.08	0.06±0.07
Fixed per layer	288	96	0.03±0.03	0.01±0.02	0.04±0.03	0.04±0.06	0.01±0.03	0.04±0.03	0.06±0.06	0.03±0.03	0.06±0.04	0.04±0.04
	216	72	0.04±0.03	0.02±0.02	0.05±0.04	0.05±0.07	0.01±0.04	0.05±0.05	0.07±0.07	0.04±0.04	0.07±0.04	0.04±0.05
	144	48	0.05±0.05	0.03±0.04	0.07±0.05	0.07±0.09	0.05±0.10	0.08±0.06	0.09±0.07	0.07±0.09	0.10±0.06	0.07±0.07
Global threshold	384		0.07±0.07	0.07±0.10	0.10±0.07	0.13±0.11	0.11±0.17	0.14±0.09	0.13±0.12	0.11±0.14	0.14±0.09	0.11±0.11
	288		0.10±0.10	0.12±0.17	0.15±0.20	0.19±0.22	0.15±0.21	0.18±0.24	0.20±0.25	0.13±0.15	0.19±0.22	0.16±0.20
	192		0.10±0.10	0.12±0.15	0.12±0.09	0.17±0.15	0.17±0.23	0.16±0.12	0.16±0.14	0.17±0.19	0.15±0.11	0.15±0.15
Enc/dec thresholds	216	72	0.05±0.04	0.03±0.03	0.06±0.04	0.08±0.09	0.03±0.04	0.06±0.05	0.09±0.07	0.06±0.06	0.08±0.04	0.06±0.06

Table 6: Valid length ratio statistics on 30 languages. We compute the length ratio deltas w.r.t. the full unpruned model and report the mean and standard deviation in each language category. Values smaller than zero mean that the model generates longer sequences on average. We prune with the *importance* metric, using statistics at the language pair granularity. Length ratios are obtained with SacreBLEU and the 'flores200' tokenization.

Reference	However, very few Royal Navy ships were based near the likely invasion routes as the admirals were afraid they would be sunk by German air attack.
Translation	However, the Royal Navy had few ships in the vicinity of the invasion routes, as the admirals feared that they would be sunk by German air attack. The Royal Navy's ships were too small to be able to operate in the vicinity of the invasion routes.
Reference	However, the discovery of his tomb in 1922 made him a celebrity. While many tombs of the past were robbed, this tomb was left virtually undisturbed.
Translation	However, his tomb became famous when it was discovered in 1922. Although many tombs in the past have been excavated, this one has remained largely undisturbed. The tomb was built in the middle of the 19th century. The tomb was built in the middle of the 20th century. The tomb was built in the middle of the 20th century
Reference	The translation engines have improved dramatically, and now often give more or less correct translations (and more seldom gibberish), but some care is due, as they still may have gotten it all wrong.
Translation	Translation engines have improved dramatically, and now often produce more or less accurate translations (as well as more or less uncommon ones), but some errors can still be corrected, so some caution is warranted. For example, in the case of the English language, the use of the word "translate" in the context of a translation of a document, such as a translation of a document into a foreign language, is not recommended. The use of the word "translate" in the context of a translation of a document is discouraged.

Table 7: Examples of over-generation when translating from Tigrinya (`tir_Ethi`) to English (`eng_Latn`) with *fixed per layer* (**3:1**) 80% pruning at the language direction granularity.

Code	Language	Resource Type	Family	Subgrouping
tsn_Latn	Tswana	High	Atlantic-Congo	Benue-Congo
vie_Latn	Vietnamese	High	Austroasiatic	Vietic
rus_Cyrl	Russian	High	Indo-European	Balto-Slavic
eng_Latn	English	High	Indo-European	Germanic
fra_Latn	French	High	Indo-European	Italic
por_Latn	Portuguese	High	Indo-European	Italic
jpn_Jpan	Japanese	High	Japonic	Japanesic
kor_Hang	Korean	High	Koreanic	Korean
fin_Latn	Finnish	High	Uralic	Finnic
tir_Ethi	Tigrinya	Low	Afro-Asiatic	Semitic
nso_Latn	Northern Sotho	Low	Atlantic-Congo	Benue-Congo
yor_Latn	Yoruba	Low	Atlantic-Congo	Benue-Congo
mal_Mlym	Malayalam	Low	Dravidian	South Dravidian
tam_Taml	Tamil	Low	Dravidian	South Dravidian
bel_Cyrl	Belarusian	Low	Indo-European	Balto-Slavic
cym_Latn	Welsh	Low	Indo-European	Celtic
urd_Arab	Urdu	Low	Indo-European	Indo-Aryan
luo_Latn	Luo	Low	Nilotic	Western Nilotic
tat_Cyrl	Tatar	Low	Turkic	Common Turkic
cjk_Latn	Chokwe	Very low	Atlantic-Congo	Benue-Congo
kik_Latn	Kikuyu	Very low	Atlantic-Congo	Benue-Congo
fuv_Latn	Nigerian Fulfulde	Very low	Atlantic-Congo	North-Central Atlantic
wol_Latn	Wolof	Very low	Atlantic-Congo	North-Central Atlantic
ace_Latn	Acehnese	Very low	Austronesian	Malayo-Polynesian
ayr_Latn	Central Aymara	Very low	Aymaran	Central Southern Aymara
snd_Arab	Sindhi	Very low	Indo-European	Indo-Aryan
ast_Latn	Asturian	Very low	Indo-European	Italic
kea_Latn	Kabuverdianu	Very low	Indo-European	Italic
yue_Hant	Yue Chinese	Very low	Sino-Tibetan	Sinitic
arb_Arab	Modern Standard Arabic	High	Afro-Asiatic	Semitic
swh_Latn	Swahili	High	Atlantic-Congo	Benue-Congo
eus_Latn	Basque	High	Basque	Basque
bul_Cyrl	Bulgarian	High	Indo-European	Balto-Slavic
lvs_Latn	Standard Latvian	High	Indo-European	Balto-Slavic
afr_Latn	Afrikaans	High	Indo-European	Germanic
isl_Latn	Icelandic	High	Indo-European	Germanic
hin_Deva	Hindi	High	Indo-European	Indo-Aryan
pes_Arab	Western Persian	High	Indo-European	Iranian
ita_Latn	Italian	High	Indo-European	Italic
zho_Hans	Chinese	High	Sino-Tibetan	Sinitic
hau_Latn	Hausa	Low	Afro-Asiatic	Chadic
kin_Latn	Kinyarwanda	Low	Atlantic-Congo	Benue-Congo
kon_Latn	Kikongo	Low	Atlantic-Congo	Benue-Congo
lin_Latn	Lingala	Low	Atlantic-Congo	Benue-Congo
run_Latn	Rundi	Low	Atlantic-Congo	Benue-Congo
tso_Latn	Tsonga	Low	Atlantic-Congo	Benue-Congo
ewe_Latn	Ewe	Low	Atlantic-Congo	Kwa Volta-Congo
fon_Latn	Fon	Low	Atlantic-Congo	Kwa Volta-Congo
twi_Latn	Twi	Low	Atlantic-Congo	Kwa Volta-Congo
tel_Telu	Telugu	Low	Dravidian	South
mar_Deva	Marathi	Low	Indo-European	Indo-Aryan
sin_Sinh	Sinhala	Low	Indo-European	Indo-Aryan
oci_Latn	Occitan	Very low	Indo-European	Italic

Table 8: Set of 53 languages used in the experiments. We show the lang code, name, resource type, family, and subgrouping of each language. The 30 languages used in the intermediate experiments are in the top half of the table.

Method	Metric	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model (Costa-jussà et al., 2022)		27.22	20.77	11.29	23.10	17.85	9.44	19.07	14.65	7.84	16.78
54.5B MoE model (Costa-jussà et al., 2022)		28.98	22.29	11.87	25.19	19.49	10.24	20.79	16.55	8.36	**18.17**
Fixed per layer (balanced)	Top 1	28.39	21.82	11.64	24.22	18.67	9.92	20.05	15.98	8.13	17.62
	Top 2	27.06	20.89	11.20	23.08	17.87	9.48	19.08	15.18	7.83	16.82
	Load balancing	27.16	21.04	11.30	23.17	17.98	9.60	19.14	15.24	7.88	16.92
	Importance (vanilla)	25.92	18.27	10.51	23.78	17.64	9.69	19.20	14.43	7.73	16.33
	Importance	28.45	21.86	11.66	24.25	18.62	9.90	20.13	16.02	8.19	17.65
Global threshold	Top 1	28.33	21.50	11.26	23.54	18.16	9.26	19.72	15.45	7.69	17.18
	Importance	28.43	21.56	11.28	23.52	18.37	9.40	19.74	15.54	7.69	17.25
Fixed per layer (unbalanced)	Importance	28.63	22.08	11.76	24.47	18.94	10.03	20.19	16.21	8.23	17.81
Enc/Dec thresholds (balanced)		28.47	21.87	11.65	24.25	18.61	9.90	20.11	15.97	8.15	17.64
Enc/Dec thresholds (unbalanced)		28.72	22.08	11.74	24.57	18.99	10.01	20.26	16.14	8.20	17.83

Table 9: spBLEU valid scores on 30 languages of different pruning algorithms and metrics, with 75% pruning (i.e., 384 experts are kept in total). The unbalanced approaches keep 240 encoder experts and 144 decoder experts.

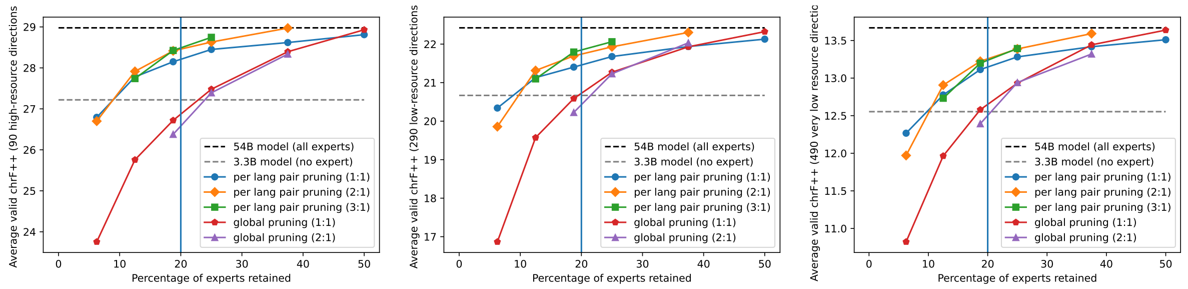

Figure 4: spBLEU valid scores on 30 languages for different resource types as a function of the percentage of experts retained. Pruning is done at the language pair granularity with the *importance* metric and with a fixed number of experts per layer.

Method	Enc experts	Dec experts	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	44.54	38.20	30.08	40.49	35.19	27.61	35.27	30.68	24.75	34.06
54.5B MoE model	768	768	**45.90**	39.19	30.24	**42.29**	**36.35**	**28.18**	**36.55**	**32.16**	24.93	**35.07**
Fixed per layer (lang-pair)	216	72	45.87	39.16	30.41	41.75	35.89	28.11	36.34	31.97	25.08	34.93
Fixed per layer (global)	216	72	44.56	37.80	29.91	40.61	34.78	27.85	35.51	31.04	25.09	34.10
Fixed per layer (lang)	216	72	45.84	**39.22**	**30.46**	41.72	35.96	28.17	36.29	32.03	**25.11**	34.96
Enc/dec thresholds (lang-pair)	216	72	45.89	39.19	30.39	41.77	36.02	28.21	36.28	31.97	25.07	34.95
Global threshold (lang-pair)	288		45.82	38.96	30.06	41.44	35.98	27.92	35.90	31.83	24.72	34.71

Table 10: chrF++ valid scores on 30 languages, with the *importance* metric for 80% pruning (1-GPU decoding) at three different levels of granularity (global, per language or per language direction).

Method	Enc experts	Dec experts	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	26.72	18.69	12.62	21.08	15.65	10.12	19.52	14.03	9.27	17.71
54.5B MoE model	768	768	**28.42**	**20.11**	**13.31**	**22.81**	**16.93**	**10.99**	**21.35**	**15.75**	**9.91**	**19.12**
Fixed per layer (lang-pair)	216	72	28.01	19.81	12.81	21.94	16.33	10.37	20.63	15.27	9.27	18.56
Fixed per layer (global)	216	72	24.15	17.15	12.26	18.34	13.89	9.77	17.41	12.78	8.88	16.03
Fixed per layer (lang)	216	72	27.87	19.82	12.78	21.79	16.37	10.35	20.51	15.28	9.19	18.50

Table 11: spBLEU test scores on 53 languages, with the *importance* metric for 80% pruning (1-GPU decoding) at three different levels of granularity (global, per language or per language direction).

Method	Enc experts	Dec experts	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	26.86	18.35	14.18	20.91	15.15	11.48	20.09	14.38	11.42	15.95
54.5B MoE model	768	768	**28.61**	**19.49**	**15.41**	**22.66**	**16.22**	**12.69**	**22.71**	**16.18**	**12.71**	**17.48**
Fixed per layer (lang)	216	72	28.27	19.26	15.08	22.02	15.84	12.24	21.90	15.62	12.05	16.97

Table 12: spBLEU test scores on all 202 languages, with the *importance* metric for 80% pruning (1-GPU decoding) at the language granularity.

Method	Enc experts	Dec experts	High→High	High→Low	High→V. Low	Low→High	Low→Low	Low→V. low	V. low→High	V. low→Low	V. low→V. low	Average
3.3B dense model	6	6	1.14±1.23	0.52±1.15	0.84±2.17	1.34±1.30	0.62±1.23	1.00±2.24	2.07±1.46	1.38±1.62	0.95±2.23	1.10±1.83
Fixed per layer (lang)	216	72	0.01±0.58	-0.23±0.55	0.23±1.55	0.34±0.58	-0.01±0.60	0.39±1.61	0.53±0.60	0.19±0.57	0.66±1.68	0.29±1.17
3.3B dense model	6	6	1.75±1.35	1.13±1.25	1.24±2.07	1.76±1.42	1.07±1.21	1.21±1.89	2.62±1.67	1.80±1.58	1.29±1.95	1.53±1.74
Fixed per layer (lang)	216	72	0.33±0.92	0.22±0.56	0.34±1.25	0.65±1.08	0.38±0.53	0.44±1.18	0.81±1.00	0.56±0.52	0.66±1.16	0.51±0.99

Table 13: Test chrF++ deltas (first part) and spBLEU deltas (second part) with the unpruned MoE model on all 202 languages. The pruned version uses the *importance* metric with 80% pruning at the language granularity. Each column reports the average score for a given language category, as well as the standard deviation. A positive value means that this model is worse than the full 54.5B model. The last column reports the average score and standard deviation over all 202×201 directions.

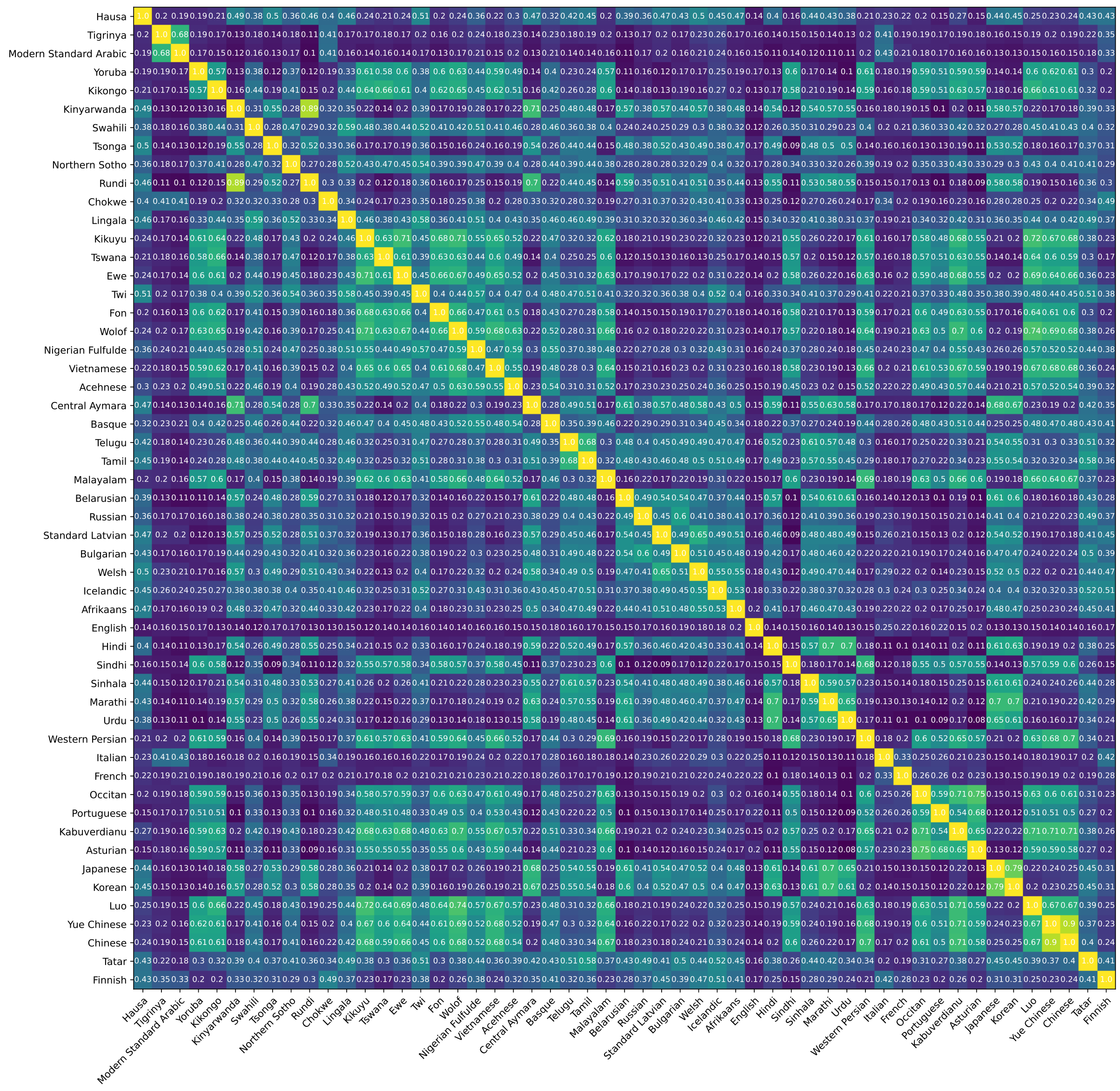

Figure 5: Jaccard similarity of selected 25% decoder experts for different languages. Pruning was done per language with the *importance* metric and *enc/dec threshold* pruning. Languages are sorted by language family.

Resource Type	Criterion	Language count
Very low	$\lvert L\rvert \leq 100k$	11
Low	$100k \leq \lvert L\rvert \leq 1m$	22
High	$1m \leq \lvert L\rvert$	20

Table 14: Distribution of languages in the 53-language subset, based on the amount of available data $|L|$. The 30-language subset has 10 languages of each resource type. Line counts are published by Costa-jussà et al. (2022) here: `https://tinyurl.com/535f7ust`

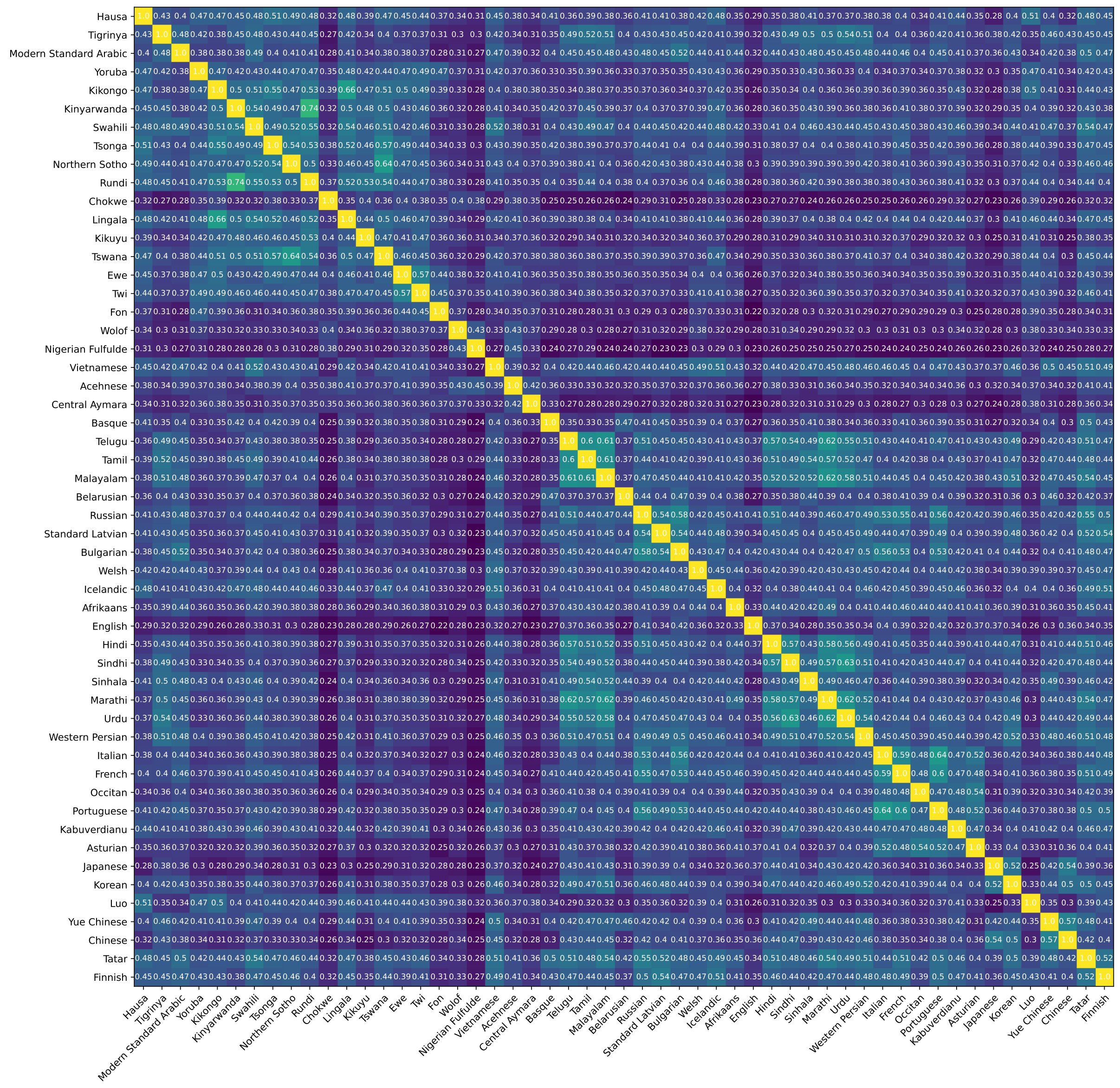

Figure 6: Jaccard similarity of selected 25% encoder experts for different languages. Pruning was done per language with the *importance* metric and *enc/Dec threshold* pruning. Languages are sorted by language family.

Model	Hours	GPU hours
3.3B	480	440
54.5B (full)	4 740	3 840
54.5B (pruned)	15 900	5 700
Total	21 120	9 980

Table 15: Time spent decoding with each type of model in this work. This includes failed or non-discussed experiments. The "hours" column measures the total time spent by the decoding script, including model creation and loading (note that the GPUs were reserved but idle during that time). "GPU hours" measures the time actually spent decoding (i.e., with the GPU active).

Language pair resource type	Encoder	Decoder
High→High	320	64
High→Low	344	44
High→V. low	348	36
Low→High	319	65
Low→Low	343	41
Low→V. low	346	38
V. low→High	314	70
V. low→Low	338	46
V. low→V. low	340	44
Average	335	49

Table 16: Average number of experts in the encoder and decoder for different language resource type language pairs with *global threshold* 75% pruning and the *importance* metric.

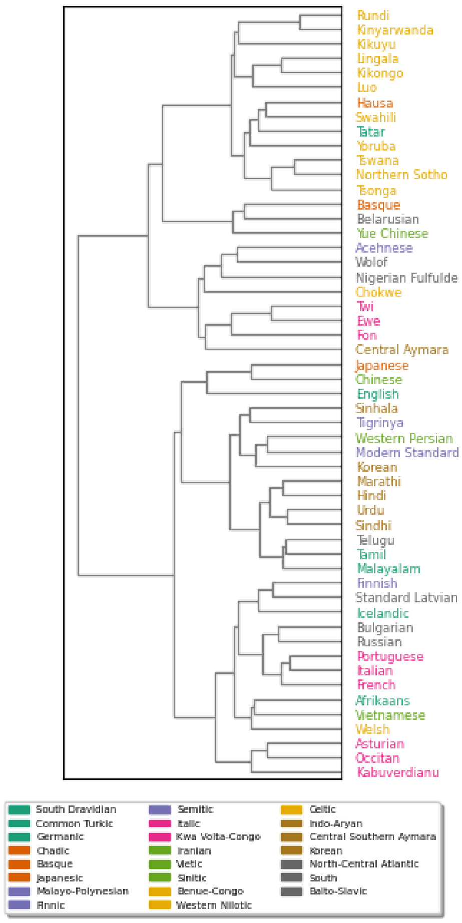

Figure 7: Hierarchical clustering of languages based on the *importance* metric of encoder experts. Different colors represent different language subgroupings.

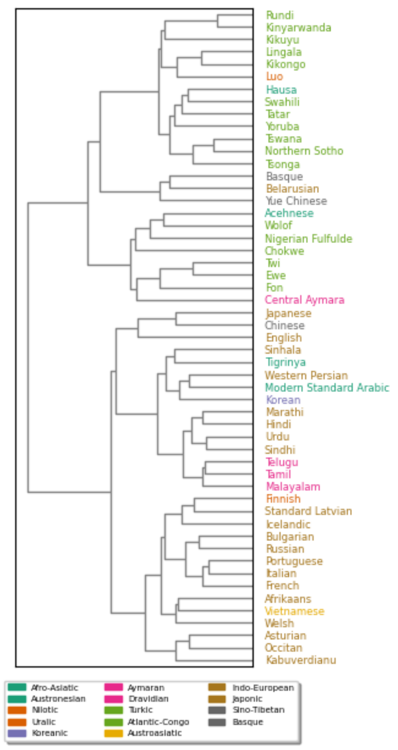

Figure 8: Hierarchical clustering of languages based on the *importance* metric of encoder experts. Different colors represent different language families.

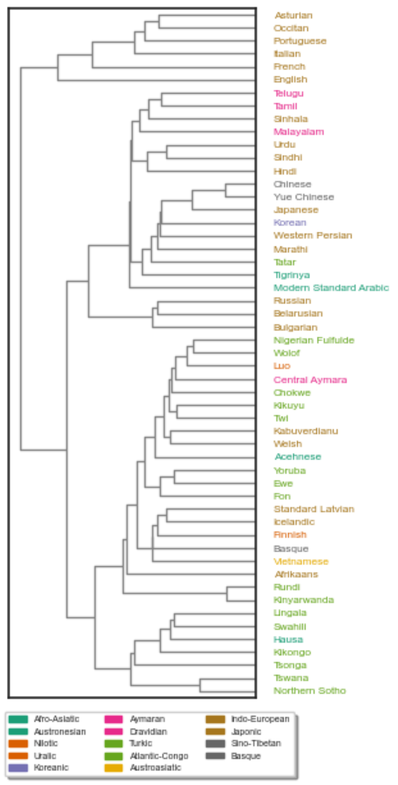

Figure 9: Hierarchical clustering of languages based on the *importance* metric of decoder experts. Different colors represent different language families.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
8

☑ A2. Did you discuss any potential risks of your work?
8

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☑ B1. Did you cite the creators of artifacts you used?
Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
No response.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
No response.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
No response.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
No response.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
No response.

C ☑ Did you run computational experiments?

5

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
6

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4,5

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
Appendix,

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
5

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?
Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
No response.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
No response.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
No response.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
No response.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
No response.