Fixing MoE Over-Fitting on Low-Resource Languages in Multilingual Machine Translation

Maha Elbayad* Meta AI elbayadm@meta.com

Anna Sun* Meta AI annaysun@meta.com

Shruti Bhosale Meta AI shru@meta.com

Abstract

Sparsely gated Mixture of Experts (MoE) models have been shown to be a compute-efficient method to scale model capacity for multilingual machine translation. However, for low-resource tasks, MoE models severely over-fit. We introduce in this work effective regularization strategies, namely (1) dropout techniques for MoE layers in Expert Output Masking (EOM) and Final Output Masking (FOM), (2) Conditional MoE Routing (CMR) that learns what tokens require the extra capacity of MoE layers and (3) Curriculum Learning methods that introduce low-resource pairs at later stages of training. All these methods prevent over-fitting and improve the performance of MoE models on low-resource tasks without adversely affecting high-resource tasks. On a massively multilingual machine translation benchmark, our strategies result in about +1 chrF++ improvement in very low resource language pairs.

1 Introduction

Training massively multitask models such as multilingual machine translation models benefit from transfer learning across different tasks. But they also suffer from reduced model capacity per task and potential interference between conflicting tasks. Scaling up models has been shown to be a very effective strategy in many natural language processing tasks such as language modeling, massively multilingual translation and natural language understanding (Brown et al., 2020; Kaplan et al., 2020). Most of these advancements have focused on training increasingly larger dense models. However, dense model scaling is computationally expensive, as a result, various sparse model architectures have been proposed to increase model capacity without incurring additional compute costs; the most commonly used one is the Sparsely-Gated Mixture-of-Experts (MoE) layer (Shazeer et al., 2017; Lepikhin et al., 2020; Du et al., 2021; Hwang et al., 2022; Zoph et al., 2022).

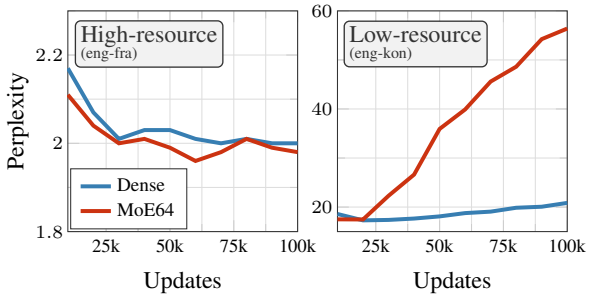


Figure 1: Validation perplexity of dense and MoE (64 experts) models. We show a high-resource direction that does not suffer from over-fitting, when a low resource direction sees extreme over-fitting.

MoE models are a type of conditional compute models (Bengio et al., 2013; Almahairi et al., 2016) that activate a subset of model parameters per input, as opposed to *dense* models that activate all model parameters. MoE models unlock significant representational capacity while maintaining the same inference and training efficiencies in terms of FLOPs as compared to the core dense architecture. As a result, past work has demonstrated improved performance on multitask models such as multilingual machine translation when using MoE models (Lepikhin et al., 2020; Kim et al., 2021; Fedus et al., 2022; Zoph et al., 2022).

But we notice that, on imbalanced datasets, MoE models suffer from over-fitting on low resource tasks i.e., tasks with relatively less training data. Figure 1 illustrates this phenomenon on a multilingual translation benchmark. We see that `eng-fra`, a high-resource translation direction, does not over-fit with either dense or MoE models. On the other hand, `eng-kon`, a low-resource translation direction, extremely over-fits with the MoE model compared to the dense model.

In this work, we introduce four effective strate-

*Equal contribution

Findings of the Association for Computational Linguistics: ACL 2023, pages 14237–14253
July 9-14, 2023

gies to reduce the over-fitting of MoE models on low-resource tasks in a massively multilingual MT benchmark:

1. Dropout techniques for MoE layers: we introduce Expert Output Masking (EOM) and Final Output Masking (FOM), two dropout methods specific to MoE layers that we apply on top of overall dropout.
2. Conditional MoE Routing (CMR): We train an additional gate to decide when to route a token to an MoE layer vs. a shared dense layer.
3. Curriculum Learning (CL): We introduce low-resource pairs that are prone to over-fitting in the later stages of model training.

On a massively multilingual MT benchmark,[1] we experimentally demonstrate the effectiveness of each of these strategies. Particularly, we observe close to +1 chrF++ improvements with EOM, FOM, CMR and CL strategies on very low resource language directions out of English.

2 Background

We first describe the multilingual machine translation (MMT) task setup, the dense backbone architecture, and how we augment it with MoE layers.

Multilingual Machine Translation. We model multilingual neural machine translation as a sequence-to-sequence task, where we condition on an input sequence in the source language with an encoder and generate the output sequence in the expected target language with a decoder (Sutskever et al., 2014). We train to maximize the probability of the translation sequence in the target language given the source sequence, in addition to the source language ℓ_s and the target language ℓ_t.

Model Architecture. Our sequence-to-sequence multilingual machine translation model is based on the Transformer encoder-decoder architecture (Vaswani et al., 2017).

To prime the model for multilingual translation, we prefix the source sequence with the source language ℓ_s and the target sequence with the target language ℓ_t.

[1] 53 languages and 110 translation directions and approximately 1.7B training examples

Sparsely Gated Mixture of Experts. In both Transformer encoder and decoder, we replace every other dense FFN sublayer with an MoE sublayer. The MoE sublayer consists of E feed-forward networks (FFN), denoted with $(\mathrm{FFN}_1, \mathrm{FFN}_2, \ldots, \mathrm{FFN}_E)$. A gating network, consisting of a softmax-normalized linear layer with weights W_g, is attached to each MoE sublayer to decide how to route tokens to experts. Given an input token x_t the output of the MoE sublayer is evaluated as:

$$\mathcal{G}_t = \text{Top-k-Gating}(\mathrm{softmax}(W_g \cdot x_t)), \quad (1)$$

$$\mathrm{MoE}(x_t) = \sum_{e=1}^{E} \mathcal{G}_{te} \cdot \mathrm{FFN}_e(x_t), \quad (2)$$

with $\mathcal{G}_t \in \mathbb{R}^E$ the routing vector computed by the gating network, i.e., for each expert, $\mathcal{G}_{t,e}$ is the contribution of the e$^{\text{th}}$ expert (FFN_e) in the MoE output. We follow the Top-k-Gating algorithm of Lepikhin et al. (2020) and dispatch each token to at most k=2 experts.

The sparse MoE model learns to route input tokens to the corresponding top-2 experts by optimizing a linearly weighted combination of label-smoothed cross entropy, L_{MT}, (ϵ=0.1, Szegedy et al. (2015)) and an auxiliary load balancing loss, L_{MoE} (Shazeer et al., 2017),

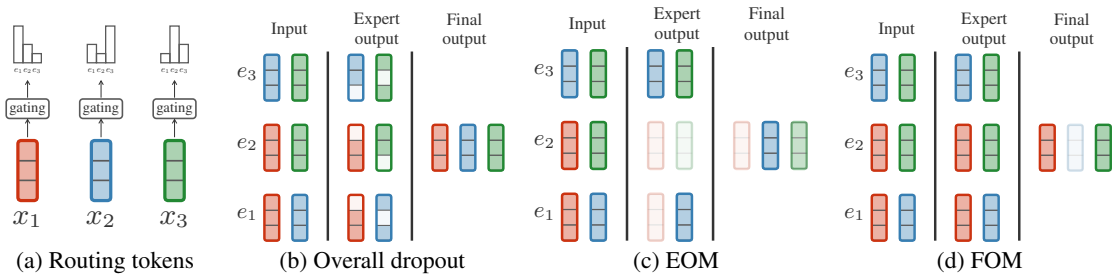
$$L = L_{\text{MT}} + \lambda_{\text{MoE}} L_{\text{MoE}}. \quad (3)$$

This additional loss term (L_{MoE}) pushes the tokens to be uniformly distributed across experts. We set λ_{MoE} to 0.01 in all our experiments. We refer the reader to Lepikhin et al. (2020) for more on the optimization of MoE models.

3 Fixing over-fitting on low-resource tasks

The motivation behind MoE models is to allow different parameters to model different aspects of the input space. The added expert capacity should help higher resource language pairs that might otherwise be constrained to share the same capacity with many other language pairs. Besides, increasing model capacity should reduce interference, thus benefiting tasks of all resource levels.

Although overall dropout is sufficient to regularize dense models, it is not enough for MoE models (see Figure 4). To address the issue of over-fitting of MoE models on low-resource tasks, we propose a series of architectural changes that improve the performance on low-resource language pairs with



(a) Routing tokens (b) Overall dropout (c) EOM (d) FOM

Figure 2: Illustration of Expert/Final Output Masking (EOM/FOM) in contrast to overall dropout for MoE layers: a color represents a token, and each token is dispatched to two experts. Faded colors correspond to dropped units or masked outputs. Note that EOM and FOM are always combined with overall dropout.

MoE models in Sections 3.1 to 3.3. In Section 3.4, we devise and study a simple but effective curriculum learning strategy as another approach to reduce the over-fitting on low-resource directions.

3.1 MoE Expert Output Masking (EOM).

In this proposed regularization strategy, we mask the *expert output* for a random fraction (p_{eom}) of the input tokens. For input tokens with dropped expert outputs, the first and/or second expert is effectively skipped, as illustrated in Figure 2c. Note that although this masking will zero out some combination weights $\mathcal{G}_{t,e}$ in Equation (2), it will not affect the weights used in the load balancing loss.

3.2 Final Output Masking (FOM).

A simpler alternative to EOM would be to mask the combined expert output for a random fraction of tokens, i.e., the last stage in Figure 2d. We denote with p_{fom} the fraction of tokens masked with this regularization method. Note that this type of masking is more generic as it can be applied to dense models as well.

3.3 Conditional MoE Routing (CMR).

Instead of randomly dropping a proportion of activations or masking expert outputs, we consider the option of letting the model learn which tokens need the extra capacity or specialization of MoE layers, and which tokens are better routed to a limited-capacity shared layer. Inspired by Zhang et al. (2021)'s CLSR-Gate, we design Conditional MoE Routing layers (CMR for short). As depicted in Figure 3, we augment MoE layers with a binary gate that determines the weights associated with two branches of the computational graph: **(1)** a shared dense FFN sublayer ($\text{FFN}_{\text{shared}}$) and **(2)** an MoE layer with its own E expert FFN sublayers. For an input token x_t, the output of CMR is evaluated as follows:

$$g(x_t) = \text{sigmoid}(W_{\text{CMR}} \cdot x_t), \tag{4}$$

$$\text{CMR}(x_t) = (1 - g(x_t)) \cdot \text{FFN}_{\text{shared}}(x_t) \tag{5}$$

$$+ g(x_t) \cdot \text{MoE}(x_t), \tag{6}$$

where W_{CMR} are the weights of the CMR's binary gate. W_{CMR} is trained by optimizing translation accuracy under a budget constraint b. For a mini-batch with T tokens, this amounts to adding the following auxiliary loss term (L_{CMR}) to the loss function in equation (3):

$$L_{\text{CMR}} = \frac{1}{T} \cdot \sum_{t=1}^{T} |g(x_t) - b|\,, \tag{7}$$

$$L = L_{\text{MT}} + \lambda_{\text{MoE}} L_{\text{MoE}} + \lambda_{\text{CMR}} L_{\text{CMR}}. \tag{8}$$

We use the budget parameter b to limit the effective capacity of MoE layers, thus providing a regularizing effect; at b=0, the model is dense, practically pushing all tokens through $\text{FFN}_{\text{shared}}$, and at b=1, the model is free to always route tokens through the high-capacity MoE layer.

To reduce over-fitting, we experiment with zeroing out a fraction of the CMR gates $g(x_t)$ in the mini-batch; we denote this fraction with p_{cmr}. This means that we force $p_{\text{cmr}}\%$ tokens in the mini-batch to only take the route of $\text{FFN}_{\text{shared}}$.

3.4 Curriculum Learning

We next explore alternative methods of regularization by means of Curriculum Learning (CL). We propose to start training with high-resource pairs first, then introduce low-resource pairs, prone to over-fit, later in phases. To derive the phases of the curriculum, we first train a vanilla MoE model (without CL), then we partition the tasks (translation directions) into n bins $\{b_1, \ldots, b_n\}$. If U is the total number of training updates, we introduce

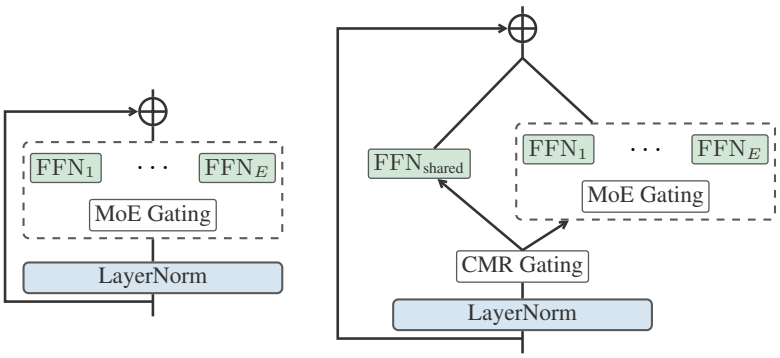


Figure 3: Illustration of Conditional MoE Routing (CMR) showing a residual block in a Transformer layer with regular MoE (left) vs. CMR (right).

Algorithm 1 Partitioning for *step-based* CL

Input: number of bins n, a set of tasks $\mathcal{T}$, the maximum number of updates U, the step corresponding to the best validation perplexity $s_{\text{best}} : \mathcal{T} \rightarrow [0, U]$.
▷ For s_{best}, we take the max if multiple
Output: Partitioning of $\mathcal{T}$ into n bins $\mathbf{b}$=$(b_1, \ldots, b_n)$, characteristic step for each bin $\mathbf{k}$=$(k_1, \ldots, k_n)$.
▷ The bin b_i will be introduced at $U - k_i$.
$s_{\text{max}} = \max_{t \in \mathcal{T}} s_{\text{best}}(t)$, $s_{\text{min}} = \min_{t \in \mathcal{T}} s_{\text{best}}(t)$
$\Delta = \dfrac{s_{\text{max}} - s_{\text{min}}}{n - 1}$.
for $i \in \{1 \ldots n\}$ **do**
    $b_i = \varnothing,\ k_i = s_{\text{max}} - (i - 1)\Delta$.
end for
for $t \in \mathcal{T}$ **do**
    $c_t = \text{argmin}_{1 \leq i \leq n} |s_{\text{best}}(t) - k_i|$
    $b_{c_t} = b_{c_t} \cup \{t\}$
    ▷ assign to the closest bin wrt. its characteristic step.
end for

each bin b_i after $U - k_i$ updates. We compare two partitioning strategies for when and what directions to add at every phase.

1. *Count-based*: we empirically partition based on training example counts.

2. *Step-based*: partition based on the step where we observed a task to start over-fitting. See Algorithm 1.

4 Experimental Setup

4.1 MMT dataset

We construct a multilingual machine translation benchmark consisting of 53 languages and a total of 110 translation directions. Our MMT dataset consists of 45 directions out of English (aggregated as `eng-xx`), 45 directions into English (aggregated as `xx-eng`) and 20 non-English directions (aggregated as `xx-yy`). In terms of resource level, there are 40 high-resource and 70 low-resource directions, out of which 22 are very low-resource.[2] The training data is composed of publicly available bitext in all 110 language directions (primary data in NLLB Team et al. (2022)) and large-scale mined data (Heffernan et al., 2022; NLLB Team et al., 2022) in English-centric directions. There are a total of 2×847M examples in this benchmark. For a detailed listing of the directions, see Appendix A.

Segmentation with SentencePiece. To tokenize our text sequences, we train a single SentencePiece (SPM) (Kudo and Richardson, 2018) model for all languages.[3] The vocabulary size of our trained SPM model is 256,000. For more on this SPM model, see NLLB Team et al. (2022).

5 Results

All MoE sub-layers have E=64 experts[4]. All models are trained for 100k updates with an effective batch size of 1M tokens per update. We evaluate

[2]We follow the categorization in NLLB Team et al. (2022); a language is low-resource if there are fewer than 1M publicly available, de-duplicated bitext samples with any other language, very low-resource if fewer than 100K.

[3]202 languages in total, including the ones not part of our MMT dataset.

[4]$E = 64$ is close to the number of languages in the benchmark, i.e., 53

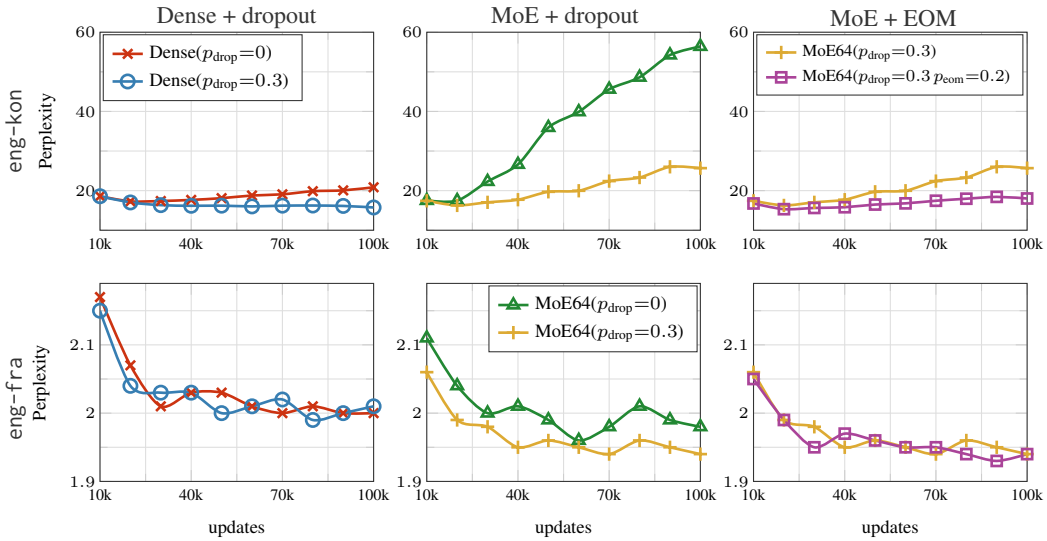


Figure 4: Validation perplexities with Various dropout strategies for a low-resource direction (eng-kon in the top row) and a high-resource direction (eng-fra in the bottom row).

	eng-xx		xx-eng		xx-yy
	all	v.low	all	v.low	all
DENSE 615M					
Dense	41.7	30.4	51.1	44.0	39.4
MoE-64	43.0	30.3	52.6	44.2	39.8
Dense (p_{drop}=0.1)	41.9	31.1	51.8	45.3	39.6
MoE-64 (p_{drop}=0.1)	**43.6**	**32.0**	**53.4**	**45.9**	**41.1**
DENSE 1.3B					
Dense	43.3	31.6	53.5	46.5	41.3
MoE-64	43.3	29.7	52.9	43.7	39.3
Dense (p_{drop}=0.1)†	43.7	**33.1**	**54.4**	**47.9**	**41.9**
MoE-64 (p_{drop}=0.3)†	**44.3**	32.5	**54.4**	47.7	**41.9**

Table 1: Validation set chrF++ of vanilla MoE with and without overall dropout. † indicates best of sweep.

using See Appendix B for additional details. We use the chrF++ metric (Popović, 2017) to compare the model performance[5]. We report averages in each set of directions: eng-xx, xx-eng and xx-yy as *all*. For eng-xx and xx-eng, and when relevant, we breakdown the pairs by resource level: high-resource (*high*), low-resource (*low*) and very low resource (*v.low*).

5.1 Vanilla (un-regularized) MoE

When looking at un-regularized models (without overall dropout), we see in Table 1, that when the backbone dense model has 615M parameters, the MoE model, while computationally similar, shows +1.3, +1.5 and +0.4 chrF++ improvements on eng-xx, xx-eng and xx-yy respectively. When focusing on the very low resource pairs (v.low), the performance actually drops on eng-xx (-0.1 chrF++) signaling an over-fitting issue. When scaling the backbone to 1.3B, we see even more over-fitting on *v.low* directions (-1.9chrF++ in eng-xx and -2.8 chrF++ in xx-eng).

Adding overall dropout[6] significantly improves the performance of MoE models in both the 615M and 1.3B variants. Importantly, when increasing the dropout to 0.1 for the small MoE (615M), we see that the relative decline of -0.1 chrF++, turns into an improvement of +0.9 chrF++ for eng-xx v.low pairs. Once we scale the computational cost per update (1.3B), tuned overall dropout does not fix the over-fitting of very low-resource pairs.

In Figure 4, we observe in the case of eng-kon, a very low-resource pair, that the model continues to face significant over-fitting when trained for 100k updates. This is unsurprising, as iterating over a small training set with large capacity causes over-fitting. Training for more steps is important for high-resource pairs, but we want to avoid negatively affecting low-resource pairs in the process.

5.2 Regularizing MoEs

For the rest of this paper, we use the 1.3B variant as our backbone, to which we add MoE layers with E=64 experts.

Experimental Setup. We consider the MoE model with an overall dropout rate of 0.3

[5]We use sacreBLEU to compute chrF++, signature: nrefs:1|case:mixed|eff:yes|nc:6|nw:2|space:no|version:2.1.0

[6]sweeping over $p_{drop} \in \{0.1, 0.2, 0.3\}$

	eng-xx				xx-eng				xx-yy
	all	high	low	v.low	all	high	low	v.low	all
Baseline p_{drop}=0.3 †	44.3	**56.0**	39.5	32.5	54.4	63.9	50.6	47.7	41.9
EOM (p_{drop}=0.3, p_{eom}=0.1)†	**44.7**	55.9	**40.1**	**33.4**	54.8	**64.3**	51.0	48.3	**42.5**
FOM (p_{drop}=0.2, p_{fom}=0.3) †	44.4	55.7	39.8	33.1	**55.0**	**64.3**	**51.3**	**48.8**	**42.5**
Gating Dropout (p_{drop}=0.3, p_{gd}=0.2) (Liu et al., 2022) †	44.4	55.7	39.8	33.0	54.8	64.1	51.0	48.5	42.3
CMR top-1 (p_{drop}=0.3, p_{cmr}=0.1, b=0.6) †	44.2	55.8	39.5	33.2	54.9	**64.3**	51.1	48.7	42.3
CMR top-2 (p_{drop}=0.3, p_{cmr}=0.2) †	**46.2**	**56.2**	**41.8**	**35.7**	**55.1**	**64.7**	**51.5**	**49.2**	**42.8**

Table 2: Comparison of Various Regularization Strategies applied to an MoE-64 baseline. In each column, we bold the best results out of the first six rows (computationally comparable), and we bold results from the last row (CMR top-2) if they outperform the other models. † signals that this model is best of sweep.

(p_{drop}=0.3), best performing after a sweep of $p_{\text{drop}} \in \{0.1, 0.2, 0.3\}$ to be our baseline.[7]

In each of the sweeps below, we choose the best variant based on the average chrF++ score on the validation set.

For EOM and FOM, we sweep over the values of $(p_{\text{drop}}, p_{\text{eom/fom}}) \in \{0.1, 0.2, 0.3\}^2$.

For CMR, and in order to keep the compute equivalent to the baseline MoE, we use top-1 instead of the top-2 gating used in previous experiments. We fix p_{drop}=0.3 and sweep over the CMR parameters (p_{cmr}, b). We also train a CMR top-2 model, although not compute-equivalent to the baseline MoE, it provides insight into performance under a large compute budget. For CMR top-2, we fix p_{drop}=0.3 and sweep over the values of $p_{\text{cmr}} \in \{0.1, 0.2, 0.3\}$. We set λ_{CMR} to 0.1 in all our CMR experiments.

We additionally compare our methods to Gating Dropout (Liu et al., 2022), a method in which we route tokens with probability p_{gd} to the local experts, thus skipping the `All-to-All` communication between GPUs. We sweep over the values of $(p_{\text{drop}}, p_{gd}) \in \{0.1, 0.2, 0.3\}^2$. To generate translation hypotheses, we use beam search with a width of 4 and a length penalty of 1.0. For each model, we report chrF++ averages on the validation set (FLORES-200 dev - NLLB Team et al. (2022)) in 3 groups of directions: `eng-xx`, `xx-eng` and `xx-yy`, broken down w.r.t. to resource levels: high, low and very low (v.low) for `eng-xx` and `xx-eng`.

Results. In terms of alleviating the over-fitting issue, the last column of Figure 4 shows that EOM leads to better regularization and less over-fitting on low-resource tasks compared to overall dropout. In terms of translation quality, we observe in Table 2 gains of +0.4 chrF++ across all pairs into English and +0.6 chrF++ across non-English pairs for MoE EOM compared to the MoE baseline. For out of English, the largest gains are observed on low and very low-resource languages; +0.6 and 0.9 chrF++ respectively.

With FOM, we see in Table 2 gains over the baseline MoE of +0.1 chrF++ across `eng-xx` pairs, +0.6 chrF++ across `xx-eng` pairs and +0.6 chrF++ across `xx-yy` pairs. For into English, the largest gains are observed on low and very low-resource languages; +0.7 and 1.1 chrF++. Compared to the best EOM model, FOM under-performs slightly on `eng-xx` (-0.3 chrF++) but outperforms on `xx-eng` (+0.2 chrF++); when averaging over all pairs, the two models achieve the same chrF++ score of 48.4.

We look in Table 3 at the impact of the budget b and the dropout p_{cmr}. We observe that p_{cmr} is a

	eng-xx		xx-eng		xx-yy
	all	v.low	all	v.low	all
top-1 b=0.4	**44.2**	32.8	54.4	48.0	42.1
+ p_{cmr}=0.1	43.9	**33.0**	**54.9**	**48.6**	**42.3**
top-1, b=0.8	44.5	32.9	**54.3**	47.4	**42.2**
+ p_{cmr}=0.1	**44.6**	**33.5**	**54.3**	**47.7**	**42.2**
top-1 p_{cmr}=0.1					
+ b=0.2	43.8	32.7	54.5	48.5	42.2
+ b=0.4	43.9	33.0	**54.9**	48.6	**42.3**
+ b=0.6	44.2	33.2	**54.9**	**48.7**	**42.3**
+ b=0.8	**44.6**	**33.5**	54.3	47.7	42.2
top-2 b=0.8	44.6	33.1	54.3	47.2	41.9
+ p_{cmr}=0.2	**46.2**	**35.7**	**55.1**	**49.2**	**42.8**

Table 3: Sweep over hyperparameters for MoE-64 CMR: The budget b, the CMR gate dropout p_{cmr}. We bold the best results in each column.

[7] Initial experiments separating the dropout rates of shared and MoE blocks showed that the best values align.

	eng-xx		xx-eng		xx-yy
	all	v.low	all	v.low	all
MoE-64	44.3	32.5	54.4	47.7	41.9
+ CL (count-based)	43.7	32.5	54.0	47.1	41.1
+ CL (step-based)	**44.7**	**33.3**	**54.6**	**47.9**	**42.2**
MoE-64 EOM	**44.7**	**33.4**	**54.8**	48.3	**42.5**
+CL (step-based)	44.3	33.1	54.7	**48.4**	42.2

Table 4: Results of Curriculum Learning applied to a vanilla MoE model and an MoE model with EOM.

necessary ingredient in CMR top-2; in the last two rows of Table 3, adding p_{cmr} improves the performance across the board, particularly in en-xx and xx-en very low directions (+2.6 and +2.0 chrF++, respectively). With top-1, p_{cmr} is less critical as it barely affects the overall performance, but does help on eng-xx and xx-eng very low pairs. In the middle section of Table 3, we note that CMR top-1 is not sensitive to the exact value of b, but, at low budget b (less capacity), model performance significantly drops on eng-xx across all pairs. Pairs in xx-eng, on the other hand, favor a mid-range budget value.

In Table 2 for CMR top-1, we see +0.4 chrF++ across all pairs into English, and +0.4 chrF++ across non-English pairs. Improvements are larger for out of English low and very low-resource languages, with +0.5 and +1.0 chrF++ respectively. For CMR top-2, we see +1.9 chrF++ across all pairs out of English and +0.9 chrF++ across non-English pairs. The improvements are largest for low and very low-resource languages, with +2.3 and +3.2 chrF++ out of English, and +0.9 and +1.5 into English. CMR top-2 is computationally more expensive by 23% because of the additional shared FFN layer at the level of each MoE layer in the model.

We find that Gating Dropout performs better than the baseline MoE, but is outperformed by all of our proposed methods. Overall, these results demonstrate that EOM, FOM, and CMR strategies help improve on top of vanilla MoE.

5.3 CL

Experimental Setup. To derive the phases of the curriculum, we train a vanilla MoE model with p_{drop}=0.3 (our baseline), then, based on observed over-fitting patterns, we partition the tasks in our MMT dataset. For both count and step-based curricula, we introduce pairs in n=3 phases over U=100k. For count-based curriculum, we partition language pairs into bins w.r.t. the training examples available for the task ($\mathcal{D}_t$): b_1 if $|\mathcal{D}_t| \geq 5e6$, b_2 if $8e5 \leq |\mathcal{D}_t| < 5e6$, and b_3 if $|\mathcal{D}_t| < 8e5$. With that we use $(k_1, k_2, k_3) = (100k, 40k, 20k)$.[8] For step-based curriculum, we follow Algorithm 1 with $n = 5$ and merge the first 3 buckets resulting in 3 bins introduced at $(k_1, k_2, k_3) = (100k, 40k, 20k)$. See Appendix C for the exact partitioning.

To combine a stronger dropout regularization with Curriculum Learning methods, we next apply our best CL strategy (*step-based*) to an MoE model with EOM (p_{eom}=0.1).

Results. We show the results of our CL experiments in Table 4. For the baseline MoE-64, by using *step-based* CL, we improve the accuracy on very low-resource directions by 0.8 chrF++ in eng-xx and 0.2 chrF++ in xx-eng . Across all resource levels, we improve the accuracy in eng-xx and xx-eng by 0.4 and 0.2 chrF++. On non-English directions *step-based* CL improves the quality by 0.3 chrF++. The *count-based* CL hurts the model performance in all tasks except from very low-resource eng-xx directions.

For MoE EOM, training with *step-based* CL actually hurts performance across all tasks except for xx-eng very low-resource. We hypothesize that over-fitting on our MMT dataset is already reduced by EOM, thus, adding a curriculum on top of that is not needed and has a negligible impact on translation quality.

6 Related work

Improved routing in MoE models. Recent works have proposed alternatives to the commonly used top-2 gating of Lepikhin et al. (2020): Hash layers (Roller et al., 2021) use random fixed routing and Lewis et al. (2021) view routing as a linear assignment problem and drop the load balancing loss. Zuo et al. (2022) suggest to randomly select experts. Fedus et al. (2022) opt for top-1 routing, and Yang et al. (2021) split experts into different groups and applies k top-1 routing in each. In this work, we only use Top-2 gating[9] but our techniques are orthogonal to the routing method.

Regularizing MoE models. Zoph et al. (2022) tried increasing the dropout within the expert (dubbed expert dropout) but saw marginal improvement in quality. They also proposed an additional

[8] That means b_1 is introduced at step 0, b_2 at step $U - 40k$, and b_3 at step $U - 20k$

[9] We did use top-1 gating for CMR to maintain a comparable computational cost with the baseline

regularization loss for MoE layers to resolve training instabilities. Kim et al. (2021) randomize the priority of tokens within a mini-batch as a regularization method. Liu et al. (2022) propose *gating dropout* to reduce cross-machine communication in MoE layers. Xie et al. (2022) propose routing tokens to expert clusters and a cluster-level expert dropout.

Conditional compute. Another line of research in the space of MoE models focuses on designing alternative strategies to learn balanced routing e.g., Lewis et al. (2021) formulated token-to-expert allocation as a linear assignment problem and Roller et al. (2021) assign tokens to experts using hash functions.

language-specific parameters. A common solution to relax parameter sharing in MMT models is to use light-weight language-specific adapters (Rebuffi et al., 2017; Bapna and Firat, 2019). Their size, however, scales linearly in the number of languages. Baziotis et al. (2022) introduce hyper-adapters to generate the adapters themselves. To make these language-specific parameters optional, Zhang et al. (2021) propose CLSR to dynamically select language-specific or shared paths. These paths are simple linear projections and do not incorporate routing. Similar to our own CMR's budget loss, CLSR optimizes the MMT cross-entropy while constraining the use of the language-specific capacity. Another approach similar to CMR is Residual-MoE (Rajbhandari et al., 2022). It is a hybrid dense and MoE model but it does not learn weights for each component. (Rajbhandari et al., 2022) also introduces Pr-MoE for pyramidal MoE where they increase the number of experts in the later layers to make the MoE models more parameter-efficient.

Curriculum Learning Curriculum learning (Bengio et al., 2009; Lu et al., 2020) is motivated by the learning behavior of humans in which training samples are introduced by increasing levels of difficulty. The most common curriculum in MT models consist of pre-training on the more abundant monolingual data before finetuning on MT aligned bitexts (Liu et al., 2020; Tang et al., 2020; Xue et al., 2021). In bilingual MT, recent works explored fixed curricula that shard training samples based on some difficulty criteria like sentence-length (Kocmi and Bojar, 2017) or the confidence of a baseline model (Zhang et al., 2018). Platanios et al. (2019) proposed a heuristic that decides which samples are shown to the model based on the estimated sample's difficulty and current model competence. Kumar et al. (2019) use reinforcement learning to learn the curriculum automatically and Zhou et al. (2020) propose uncertainty-aware curriculum learning. In data sampling, which can be viewed as a sort of curriculum learning, Wang et al. (2018) propose dynamic sentence sampling to assign lower weights to well-learned sentences.

7 Conclusion

In massively multilingual settings with imbalanced datasets, MoE models over-fit significantly more than dense models on low-resource directions. This work introduce multiple effective strategies for regularizing MoE models and achieving better performance across all language pairs, especially low-resource pairs. With EOM and FOM, we propose dropout methods to further regularize MoE models. We introduce in CMR a novel architecture to balance the capacity between MoEs and shared dense paths. Finally, we design curricula for introducing low-resource languages later during training. These strategies lead to less over-fitting on low-resource tasks, leading to improvements in translation quality.

8 Limitations

The first limitation of this work is that it lacks a study on how the proposed regularization methods work at other scales; although we looked in Section 5.1 at two variants based on the compute budget (615M and 1.3B), we only tested our methods on the 1.3B variant with a fixed number of experts E=64. These methods can potentially show larger improvements on larger models (larger backbone or more experts) and marginal impacts on smaller models that do not suffer from severe over-fitting. The second limitation of this work is that our methods are only validated on a single multilingual MT benchmark. Some of these techniques proved to be generalizable to a much larger benchmarks (NLLB Team et al., 2022), and we leave testing these techniques on other tasks like language modeling to future work. Another limitation of this work, and most other works on multilingual machine translation, is the evaluation metrics and how to aggregate them. We report in this paper chrF++ scores and we average across three subsets of directions and three resource levels. This makes it difficult to

highlight the impact in some challenging directions on which our methods can lead to ±3chrF++ differential in quality. We did not report other metrics for the sake of brevity, and since we are not comparing to previously published results, chrF++ is a reliable metric for comparing and contrasting our methods.

Acknowledgments

We would like to thank Philipp Koehn for his help with framing this paper. We would also like to thank James Cross, Onur Çelebi, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Guillaume Wenzek, Bapi Akula, Loic Barrault, Gabriel Mejia Gonzalez, Prangthip Hansanti, Kaushik Ram Sadagopan, Pierre Andrews, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk for their input on this work.

References

Solomon Teferra Abate, Michael Melese, Martha Yifiru Tachbelie, Million Meshesha, Solomon Atinafu, Wondwossen Mulugeta, Yaregal Assabie, Hafte Abera, Binyam Ephrem Seyoum, Tewodros Abebe, et al. 2018. Parallel corpora for bi-directional statistical machine translation for seven ethiopian language pairs. In Proceedings of the First Workshop on Linguistic Resources for Natural Language Processing, pages 83–90.

Ahmed Abdelali, Francisco Guzman, Hassan Sajjad, and Stephan Vogel. 2014. The AMARA corpus: Building parallel language resources for the educational domain. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 1856–1862, Reykjavik, Iceland. European Language Resources Association (ELRA).

David Adelani, Dana Ruiter, Jesujoba Alabi, Damilola Adebonojo, Adesina Ayeni, Mofe Adeyemi, Ayodele Esther Awokoya, and Cristina España-Bonet. 2021. The effect of domain and diacritics in Yoruba–English neural machine translation. In Proceedings of the 18th Biennial Machine Translation Summit (Volume 1: Research Track), pages 61–75, Virtual. Association for Machine Translation in the Americas.

Amjad Almahairi, Nicolas Ballas, Tim Cooijmans, Yin Zheng, Hugo Larochelle, and Aaron Courville. 2016. Dynamic capacity networks. In Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48, ICML'16, page 2091–2100. JMLR.org.

Antonios Anastasopoulos, Alessandro Cattelan, Zi-Yi Dou, Marcello Federico, Christian Federmann, Dmitriy Genzel, Franscisco Guzmán, Junjie Hu, Macduff Hughes, Philipp Koehn, Rosie Lazar, Will Lewis, Graham Neubig, Mengmeng Niu, Alp Öktem, Eric Paquin, Grace Tang, and Sylwia Tur. 2020. TICO-19: the translation initiative for COvid-19. In Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020, Online. Association for Computational Linguistics.

Paul Azunre, Lawrence Adu-Gyamfi, Esther Appiah, Felix Akwerh, Salomey Osei, Cynthia Amoaba, Salomey Afua Addo, Edwin Buabeng-Munkoh, Nana Boateng, Franklin Adjei, and Bernard Adabankah. 2021a. English-akuapem twi parallel corpus.

Paul Azunre, Salomey Osei, Salomey Addo, Lawrence Asamoah Adu-Gyamfi, Stephen Moore, Bernard Adabankah, Bernard Opoku, Clara Asare-Nyarko, Samuel Nyarko, Cynthia Amoaba, et al. 2021b. English-twi parallel corpus for machine translation. arXiv preprint arXiv:2103.15625.

Lei Jimmy Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. 2016. Layer normalization. CoRR, abs/1607.06450.

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.

Christos Baziotis, Mikel Artetxe, James Cross, and Shruti Bhosale. 2022. Multilingual machine translation with hyper-adapters. In EMNLP.

Yoshua Bengio, Nicholas Léonard, and Aaron C. Courville. 2013. Estimating or propagating gradients through stochastic neurons for conditional computation. CoRR, abs/1308.3432.

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In Proceedings of the 26th annual international conference on machine learning, pages 41–48.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. Advances in neural information processing systems, 33:1877–1901.

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 Bosma, Zongwei Zhou, Tao Wang, Yu Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathy Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc V. Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. 2021. Glam: Efficient scaling of language models with mixture-of-experts. CoRR, abs/2112.06905.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research, 23(120):1–39.

Asmelash Teka Hadgu, Gebrekirstos G. Gebremeskel, and Abel Aregawi. 2021. HornMT: Machine translation benchmark dataset for languages in the horn of africa. https://github.com/asmelashteka/HornMT.

Kevin Heffernan, Onur Çelebi, and Holger Schwenk. 2022. Bitext mining using distilled sentence representations for low-resource languages. In EMNLP.

Changho Hwang, Wei Cui, Yifan Xiong, Ziyue Yang, Ze Liu, Han Hu, Zilong Wang, Rafael Salas, Jithin Jose, Prabhat Ram, et al. 2022. Tutel: Adaptive mixture-of-experts at scale. arXiv preprint arXiv:2206.03382.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. CoRR, abs/2001.08361.

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andrés Felipe Cruz-Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and efficient moe training for multitask multilingual models. CoRR, abs/2109.10465.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings.

Tom Kocmi and Ondřej Bojar. 2017. Curriculum learning and minibatch bucketing in neural machine translation. In Proceedings of the International Conference Recent Advances in Natural Language Processing, RANLP 2017, pages 379–386, Varna, Bulgaria. INCOMA Ltd.

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, EMNLP 2018: System Demonstrations, Brussels, Belgium, October 31 - November 4, 2018, pages 66–71. Association for Computational Linguistics.

Gaurav Kumar, George Foster, Colin Cherry, and Maxim Krikun. 2019. Reinforcement learning based curriculum optimization for neural machine translation. arXiv preprint arXiv:1903.00041.

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. CoRR, abs/2006.16668.

Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. 2021. Base layers: Simplifying training of large, sparse models. In International Conference on Machine Learning, pages 6265–6274. PMLR.

Pierre Lison and Jörg Tiedemann. 2016. Opensubtitles2016: Extracting large parallel corpora from movie and tv subtitles.

Rui Liu, Young Jin Kim, Alexandre Muzio, Barzan Mozafari, and Hany Hassan Awadalla. 2022. Gating dropout: Communication-efficient regularization for sparsely activated transformers. arXiv preprint arXiv:2205.14336.

Yinhan Liu, Jiatao Gu, Naman Goyal, Xian Li, Sergey Edunov, Marjan Ghazvininejad, Mike Lewis, and Luke Zettlemoyer. 2020. Multilingual denoising pre-training for neural machine translation. Transactions of the Association for Computational Linguistics, 8:726–742.

Jiasen Lu, Vedanuj Goswami, Marcus Rohrbach, Devi Parikh, and Stefan Lee. 2020. 12-in-1: Multi-task vision and language representation learning. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 10437–10446.

Laurette Marais, Ilana Wilken, Nina Van Niekerk, and Karen Calteaux. 2021. Mburisano covid-19 multilingual corpus. https://hdl.handle.net/20.500.12185/536.

Arya D. McCarthy, Rachel Wicks, Dylan Lewis, Aaron Mueller, Winston Wu, Oliver Adams, Garrett Nicolai, Matt Post, and David Yarowsky. 2020. The Johns Hopkins University Bible corpus: 1600+ tongues for typological exploration. In Proceedings of the 12th Language Resources and Evaluation Conference, pages 2884–2892, Marseille, France. European Language Resources Association.

Jamshidbek Mirzakhalov, Anoop Babu, Duygu Ataman, Sherzod Kariev, Francis Tyers, Otabek Abduraufov, Mammad Hajili, Sardana Ivanova, Abror Khaytbaev, Antonio Laverghetta Jr, et al. 2021. A large-scale study of machine translation in the turkic languages. arXiv preprint arXiv:2109.04593.

Toshiaki Nakazawa, Hideki Nakayama, Chenchen Ding, Raj Dabre, Shohei Higashiyama, Hideya Mino, Isao Goto, Win Pa Pa, Anoop Kunchukuttan, Shantipriya Parida, Ondřej Bojar, Chenhui Chu, Akiko Eriguchi, Kaori Abe, Yusuke Oda, and Sadao Kurohashi. 2021. Overview of the 8th workshop on Asian translation. In Proceedings of the 8th Workshop on Asian Translation (WAT2021), pages 1–45, Online. Association for Computational Linguistics.

NLLB Team, Marta R. Costa-jussà, James Cross, Onur Çelebi, Maha Elbayad, Kenneth Heafield, Kevin Heffernan, Elahe Kalbassi, Janice Lam, Daniel Licht, Jean Maillard, Anna Sun, Skyler Wang, Guillaume

Wenzek, Al Youngblood, Bapi Akula, Loic Barrault, Gabriel Mejia-Gonzalez, Prangthip Hansanti, John Hoffman, Semarley Jarrett, Kaushik Ram Sadagopan, Dirk Rowe, Shannon Spruit, Chau Tran, Pierre Andrews, Necip Fazil Ayan, Shruti Bhosale, Sergey Edunov, Angela Fan, Cynthia Gao, Vedanuj Goswami, Francisco Guzmán, Philipp Koehn, Alexandre Mourachko, Christophe Ropers, Safiyyah Saleem, Holger Schwenk, and Jeff Wang. 2022. No language left behind: Scaling human-centered machine translation. arXiv preprint arXiv:2207.04672.

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. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations), pages 48–53, Minneapolis, Minnesota. Association for Computational Linguistics.

Emmanouil Antonios Platanios, Otilia Stretcu, Graham Neubig, Barnabas Poczos, and Tom M Mitchell. 2019. Competence-based curriculum learning for neural machine translation. arXiv preprint arXiv:1903.09848.

Maja Popović. 2017. chrf++: words helping character n-grams. In Proceedings of the Second Conference on Machine Translation, Volume 2: Shared Task Papers, pages 612–618, Copenhagen, Denmark. Association for Computational Linguistics.

Alexandre Rafalovitch and Robert Dale. 2014. United Nations general assembly resolutions: A six-language parallel corpus. In Proceedings of the MT Summit XII, pages 292–299, Ottawa, Canada.

Samyam Rajbhandari, Conglong Li, Zhewei Yao, Minjia Zhang, Reza Yazdani Aminabadi, Ammar Ahmad Awan, Jeff Rasley, and Yuxiong He. 2022. Deepspeed-moe: Advancing mixture-of-experts inference and training to power next-generation ai scale. arXiv preprint arXiv:2201.05596.

Sylvestre-Alvise Rebuffi, Hakan Bilen, and Andrea Vedaldi. 2017. Learning multiple visual domains with residual adapters. Advances in neural information processing systems, 30.

Stephen Roller, Sainbayar Sukhbaatar, Arthur Szlam, and Jason E Weston. 2021. Hash layers for large sparse models. In Advances in Neural Information Processing Systems.

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. In Proceedings of International Conference on Learning Representations (ICLR).

Raivis Skadiņš, Jörg Tiedemann, Roberts Rozis, and Daiga Deksne. 2014. Billions of parallel words for free: Building and using the EU bookshop corpus. In Proceedings of the Ninth International Conference on Language Resources and Evaluation (LREC'14), pages 1850–1855, Reykjavik, Iceland. European Language Resources Association (ELRA).

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks. In Proc. of NeurIPS.

Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. 2015. Rethinking the inception architecture for computer vision. CoRR, abs/1512.00567.

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. CoRR, abs/2008.00401.

J. Tiedemann. 2012. Parallel data, tools and interfaces in OPUS. In LREC.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In Advances in neural information processing systems, pages 5998–6008.

Rui Wang, Masao Utiyama, and Eiichiro Sumita. 2018. Dynamic sentence sampling for efficient training of neural machine translation. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), pages 298–304, Melbourne, Australia. Association for Computational Linguistics.

Yuan Xie, Shaohan Huang, Tianyu Chen, and Furu Wei. 2022. Moec: Mixture of expert clusters. arXiv preprint arXiv:2207.09094.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In International Conference on Machine Learning, pages 10524–10533. PMLR.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pages 483–498, Online. Association for Computational Linguistics.

An Yang, Junyang Lin, Rui Men, Chang Zhou, Le Jiang, Xianyan Jia, Ang Wang, Jie Zhang, Jiamang Wang, Yong Li, et al. 2021. M6-t: Exploring sparse expert models and beyond. arXiv preprint arXiv:2105.15082.

Biao Zhang, Ankur Bapna, Rico Sennrich, and Orhan Firat. 2021. Share or not? learning to schedule language-specific capacity for multilingual translation. In International Conference on Learning Representations.

Xuan Zhang, Gaurav Kumar, Huda Khayrallah, Kenton Murray, Jeremy Gwinnup, Marianna J Martindale, Paul McNamee, Kevin Duh, and Marine Carpuat. 2018. An empirical exploration of curriculum learning for neural machine translation. arXiv preprint arXiv:1811.00739.

Yikai Zhou, Baosong Yang, Derek F. Wong, Yu Wan, and Lidia S. Chao. 2020. Uncertainty-aware curriculum learning for neural machine translation. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 6934–6944, Online. Association for Computational Linguistics.

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. arXiv preprint arXiv:2202.08906.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Jianfeng Gao, and Tuo Zhao. 2022. Taming sparsely activated transformer with stochastic experts. In International Conference on Learning Representations.

A Training data

We list in Table 5 the amount of data (bitexts) used to train our models. Figure 5 shows the data distribution over language pairs sorted by the example count per pair. The highest resource language pair has 180M examples (English-French), and the lowest resource language pair has 40K examples (Hindi-Tamil).

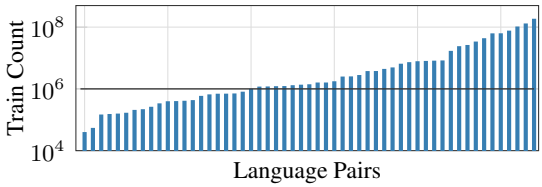


Figure 5: Training data across all language pairs in our MMT dataset.

B Training details

We use Fairseq (Ott et al., 2019) to train Transformer encoder-decoder models with dimension 1024, FFN dimension 8192, 16 attention heads, 24 encoder layers and 24 decoder layers. Dense 615M models have 614,918,144 parameters, and MoE models corresponding with the dense 615M backbone have 6,961,431,552 parameters. Dense 1.3B models have 1,372,055,552 parameters, and MoE models corresponding to the dense 1.3B backbone have have 26,753,140,736 parameters. The total compute across all the experiments reported, including sweeps, is 461,631 GPU hours. We train with seed=2 for all experiments. We apply Layer-normalization (Ba et al., 2016) at the beginning of each Transformer sub-layer (Pre-LN), as opposed to after the residual connection (Post-LN). This is because Pre-LN is more stable in practice compared to Post-LN (Xiong et al., 2020). All models are trained for 100k updates with an effective batch size of 1M tokens per update. We optimize with Adam (Kingma and Ba, 2015) using $(\beta_1, \beta_2, \epsilon) = (0.9, 0.98, 10^{-6})$. We linearly increase the learning rate up to 0.004 through 8000 warmup updates, then follow the inverse square root learning rate schedule. For Top-2-Gating, we set the expert capacity to $2 \times T/E$, i.e., we enforce that each expert processes, at most, $2 \times T/E$ tokens, where T is the number of tokens in the mini-batch and E is the number of experts. During generation, we set the capacity to T so that all tokens can be routed to whichever expert they choose.

Code	Language
ace_Latn	Acehnese
afr	Afrikaans
ara_Arab	Arabic
ast	Asturian
ayr	Aymara
bel	Belarussian
bul	Bulgarian
cjk	Chokwe
cym	Welsh
eus	Basque
ewe	Ewe
fas	Persian
fin	Finnish
fon	Fon
fra	French
fuv	Fula
hau	Hausa
hin	Hindi
isl	Icelandic
ita	Italian
jpn	Japanese
kea	Kabuverdianu
kik	Kikuyu
kin	Kinyarwanda
kon	Kongo
kor	Korean
lav	Latvian
lin	Lingala
luo	Luo
mal	Malayalam
mar	Marathi
nso	Northern Sotho
oci	Occitan
por	Portuguese
run	Rundi
rus	Russian
sin	Sinhalese
snd	Sindhi
swh	Swahili
tam	Tamil
tat_Cyrl	Tatar
tel	Telugu
tir	Tigrinya
tsn	Tswana
tso	Tsonga
twi	Twi
urd	Urdu
vie	Vietnamese
wol	Wolof
yor	Yoruba
yue	Yue Chinese
zho_Hans	Chinese

Direction	#primary	#mined
eng-ace_Latn	36,591	1,148,759
eng-afr	1,449,916	5,840,012
eng-ara_Arab	36,340,863	39,447,939
eng-ast	526	874,884
eng-ayr	69,185	610,749
eng-bel	47,166	892,477
eng-bul	26,706,641	35,742,011
eng-cjk	33,038	660,404
eng-cym	149,598	4,239,464
eng-ewe	534,793	739,132
eng-fas	4,402,104	19,527,935
eng-fin	34,784,117	27,243,736
eng-fon	36,752	299,065
eng-fra	37,993,938	141,929,009
eng-fuv	18,242	189,675
eng-hau	345,481	4,598,698
eng-hin	1,688,720	24,497,780
eng-isl	1,096,312	6,744,150
eng-ita	44,712,431	82,724,756
eng-kea	4,727	146,254
eng-kik	98,740	119,396
eng-kin	376,914	2,423,473
eng-kon	188,251	213,799
eng-lav	3,867,869	10,699,069
eng-lin	666,273	555,208
eng-luo	129,000	670,367
eng-mal	585,452	7,703,121
eng-mar	335,259	6,143,242
eng-nso	526,097	644,586
eng-oci	5,915	585,817
eng-run	454,678	1,138,461
eng-rus	30,271,773	71,205,569
eng-sin	461,857	3,288,143
eng-snd	95,718	2,434,012
eng-tam	680,297	7,223,944
eng-tel	253,718	7,880,705
eng-tir	83,980	1,128,918
eng-tso	711,883	881,110
eng-twi	508,746	1,220,279
eng-urd	875,172	2,873,007
eng-vie	3,689,843	39,782,690
eng-wol	9,233	147,746
eng-yor	397,793	2,099,168
eng-yue	54,534	0
eng-zho_Hans	228,658	33,684,682
ara_Arab-sin	402,450	0
eus-por	432,823	0
fra-hau	168,631	0
fra-kon	147,886	0
fra-lin	397,535	0
fra-swh	664,013	0
hin-tam	39,992	0
jpn-kor	1,009,697	0
rus-tat_Cyrl	263,496	0
swh-tsn	697,681	0

Corpus	Reference
AAU Ethiopian Languages	Abate et al. (2018)
DGT	Tiedemann (2012)
ECB	Tiedemann (2012)
EMEA	Tiedemann (2012)
English-Twi	Azunre et al. (2021b,a)
EU Bookshop	Skadiņš et al. (2014)
GlobalVoices	Tiedemann (2012)
HornMT	Hadgu et al. (2021)
InfoPankki v1	Tiedemann (2012)
QCRI Educational Domain	Abdelali et al. (2014)
JHU Bible	McCarthy et al. (2020)
Mburisano	Marais et al. (2021)
MENYO-20k	Adelani et al. (2021)
MultiIndicMT	Nakazawa et al. (2021)
NLLB-SEED	NLLB Team et al. (2022)
OpenSubtitles v2018	Lison and Tiedemann (2016)
Tanzil	Tiedemann (2012)
Tatoeba	Tiedemann (2012)
Tico19 v20201028	Anastasopoulos et al. (2020)
United Nations Resolutions	Rafalovitch and Dale (2014)
Turkic Interlingua (TIL)	Mirzakhalov et al. (2021)
Wikimedia v20210402	Tiedemann (2012)

Table 5: List of languages and Data counts between primary (pre-existing publicly available parallel data) and mined (Heffernan et al., 2022) for the 110 directions of our MMT dataset. 45 languages are paired with English for a total of 90 English-centric directions. The remaining 20 directions are non-English centric. We also list on the rightmost table the sources of the training data in our MMT dataset following NLLB Team et al. (2022).

C Curriculum Learning

***Count-based* CL.** We empirically partition based on training example counts. We first train our baseline model (MoE-64 (p_{drop}=0.3) without CL, then we look at possible correlations between the number of steps before over-fitting and the count of training examples. In Figure 6 we plot these data points with the counts on the y-axis and the start-of-over-fitting step on the x-axis. The horizontal red lines indicate where the *count-based* curriculum thresholds were set in order to partition language pairs into bins.

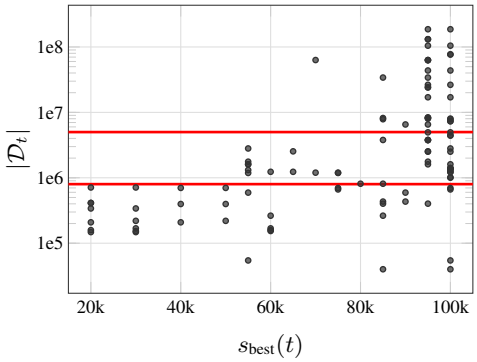


Figure 6: For the baseline MoE model, we plot the steps corresponding to the best validation perplexity (s_{best} on the x-axis) against the number of training examples ($|\mathcal{D}_t|$ on the y-axis).

We list in Table 6 the tasks in each bin for the baseline MoE model.

***Step-based* CL.** We partition based on the step where we observed a task to start over-fitting. Following Algorithm 1, we partition the tasks into n bins. In our experiments, we started with n=5 resulting in a Δ of 20k steps. However, we merged the first three bins with characteristic steps $k_1 = 100k$, $k_2 = 80k$ and $k_3 = 60k$ to remain comparable with *count-based* CL.

bin b_i	#tasks	k_i	Language pairs
b_1	17×2	100k	eng-afr, eng-ara_Arab, eng-bul, eng-fas, eng-fin, eng-fra, eng-hin, eng-isl, eng-ita, eng-lav, eng-mal, eng-mar, eng-rus, eng-tam, eng-tel, eng-vie, eng-zho_Hans
b_2	24×2	40k	eng-ace_Latn, eng-ast, eng-ayr, eng-bel, eng-cjk, eng-cym, eng-ewe, eng-hau, eng-kin, eng-lin, eng-luo, eng-nso, eng-oci, eng-run, eng-sin, eng-snd, eng-tir, eng-tso, eng-twi, eng-urd, eng-yor, fra-swh, jpn-kor, swh-tsn
b_3	14×2	20k	eng-fon, eng-fuv, eng-kea, eng-kik, eng-kon, eng-wol, eng-yue, ara_Arab-sin, eus-por, fra-hau, fra-kon, fra-lin, hin-tam, rus-tat_Cyrl

Table 6: *Count-based* CL bins for the baseline MoE model (p_{drop}=0.3). Step represents the number of steps the language pairs in this bin are trained

bin b_i	#tasks	k_i	Language pairs
b_1	86	100k	ace_Latn-eng, afr-eng, ara_Arab-eng, ara_Arab-sin, ast-eng, bel-eng, bul-eng, cym-eng, eng-afr, eng-ara_Arab, eng-ast, eng-bel, eng-bul, eng-cym, eng-ewe, eng-fas, eng-fin, eng-fra, eng-hau, eng-hin, eng-isl, eng-ita, eng-kea, eng-kin, eng-lav, eng-luo, eng-mal, eng-mar, eng-nso, eng-oci, eng-run, eng-rus, eng-sin, eng-snd, eng-tam, eng-tel, eng-tir, eng-tso, eng-twi, eng-urd, eng-vie, eng-yor, eng-zho_Hans, eus-por, ewe-eng, fas-eng, fin-eng, fra-eng, fra-hau, fra-swh, hau-eng, hin-eng, hin-tam, isl-eng, ita-eng, jpn-kor, kin-eng, kor-jpn, lav-eng, lin-eng, lin-fra, luo-eng, mal-eng, mar-eng, nso-eng, por-eus, rus-eng, sin-ara_Arab, sin-eng, snd-eng, swh-fra, swh-tsn, tam-eng, tam-hin, tat_Cyrl-rus, tel-eng, tir-eng, tsn-swh, tso-eng, twi-eng, urd-eng, vie-eng, wol-eng, yor-eng, yue-eng, zho_Hans-eng
b_2	12	40k	ayr-eng, cjk-eng, eng-ace_Latn, eng-ayr, eng-kik, eng-lin, fra-lin, fuv-eng, kik-eng, oci-eng, run-eng, rus-tat_Cyrl
b_3	12	20k	eng-cjk, eng-fon, eng-fuv, eng-kon, eng-wol, eng-yue, fon-eng, fra-kon, hau-fra, kea-eng, kon-eng, kon-fra

Table 7: *Step-based* CL bins for the baseline MoE-64 (p_{eom}=0.1)

bin b_i	#tasks	k_i	Language pairs
b_1	95	100k	ace_Latn-eng, afr-eng, ara_Arab-eng, ara_Arab-sin, ast-eng, ayr-eng, bel-eng, bul-eng, cym-eng, eng-ace_Latn, eng-afr, eng-ara_Arab, eng-ast, eng-bel, eng-bul, eng-cym, eng-ewe, eng-fas, eng-fin, eng-fra, eng-hau, eng-hin, eng-isl, eng-ita, eng-kea, eng-kik, eng-kin, eng-lav, eng-luo, eng-mal, eng-mar, eng-nso, eng-oci, eng-run, eng-rus, eng-sin, eng-snd, eng-tam, eng-tel, eng-tir, eng-tso, eng-twi, eng-urd, eng-vie, eng-yor, eng-zho_Hans, eus-por, ewe-eng, fas-eng, fin-eng, fra-eng, fra-hau, fra-lin, fra-swh, fuv-eng, hau-eng, hau-fra, hin-eng, hin-tam, isl-eng, ita-eng, jpn-kor, kin-eng, kor-jpn, lav-eng, lin-eng, lin-fra, luo-eng, mal-eng, mar-eng, nso-eng, oci-eng, por-eus, run-eng, rus-eng, rus-tat_Cyrl, sin-ara_Arab, sin-eng, snd-eng, swh-fra, swh-tsn, tam-eng, tam-hin, tat_Cyrl-rus, tel-eng, tir-eng, tsn-swh, tso-eng, twi-eng, urd-eng, vie-eng, wol-eng, yor-eng, yue-eng, zho_Hans-eng
b_2	5	40k	eng-ayr, eng-cjk, eng-lin, eng-wol, eng-yue
b_3	10	20k	cjk-eng, eng-fon, eng-fuv, eng-kon, fon-eng, fra-kon, kea-eng, kik-eng, kon-eng, kon-fra

Table 8: *Step-based* CL bins for the baseline MoE-64 EOM (p_{drop}=0.3,, p_{eom}=0.1)

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
section 9

☐ A2. Did you discuss any potential risks of your work?
Not applicable. Left blank.

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

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

B ☑ Did you use or create scientific artifacts?

We released the code for our different models and techniques

☑ B1. Did you cite the creators of artifacts you used?
We cite the toolkit we use: Fairseq (appendix B) as well as all the different data sources in appendix A (Table 5)

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ 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)?
Not applicable. Left blank.

☐ 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?
Not applicable. All of the data we used is public and open for non-commercial use

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☑ 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.
For data we provide detailed counts in appendix A.

C ☑ Did you run computational experiments?

Section 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?
Appendix B.

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?
Section 5 & Appendix B

☑ 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?
We mention in section 5 that we train models with one seed i.e. single run.

☑ 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.)?
For tokenization see last paragraph of 4.1. For evaluation we use SacreBLEU, see signatures in the footnotes of page 4

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.