How Effective are State Space Models for Machine Translation?

Hugo Pitorro[*1,3], **Pavlo Vasylenko**[*2,3], **Marcos Treviso**[3], **André F. T. Martins**[2,3,4,5]
[1]TU Munich, [2]Instituto Superior Técnico, Universidade de Lisboa
[3]Instituto de Telecomunicações, [4]Unbabel, [5]ELLIS Unit Lisbon

Abstract

Transformers are the current architecture of choice for NLP, but their attention layers do not scale well to long contexts. Recent works propose to replace attention with linear recurrent layers—this is the case for state space models, which enjoy efficient training and inference. However, it remains unclear whether these models are competitive with transformers in machine translation (MT). In this paper, we provide a rigorous and comprehensive experimental comparison between transformers and linear recurrent models for MT. Concretely, we experiment with RetNet, Mamba, and hybrid versions of Mamba which incorporate attention mechanisms. Our findings demonstrate that Mamba is highly competitive with transformers on sentence and paragraph-level datasets, where in the latter both models benefit from shifting the training distribution towards longer sequences. Further analysis show that integrating attention into Mamba improves translation quality, robustness to sequence length extrapolation, and the ability to recall named entities.

1 Introduction

The inherent design of attention—the underlying mechanism of transformers—leads to quadratic computational costs and challenges in length generalization (Varis and Bojar, 2021). As an alternative, recent works propose to replace attention with linear recurrent approaches, which enjoy efficient training and inference, and obtain competitive results in language modeling tasks (Katharopoulos et al., 2020; Gu et al., 2022; Peng et al., 2023; Sun et al., 2023a; Gu and Dao, 2023).

In machine translation (MT), there is an increasing demand for supporting longer context lengths, such as paragraphs or entire documents (Fernandes et al., 2021; Wang et al., 2023; Kocmi et al., 2023). Given this trend, it has become increasingly important to design models capable of efficiently handling longer sequences. Previous research indicates that models like state space models (SSMs), exemplified by S4 (Gu et al., 2022), still lag behind transformers in MT (Vardasbi et al., 2023). However, it remains unclear whether these findings hold true for recent, more expressive variations of linear recurrent models, such as RetNet (Sun et al., 2023a) and Mamba (Gu and Dao, 2023), especially on settings that involve the use of pretrained models and long context datasets.

In this paper, we provide a rigorous and comprehensive experimental comparison between transformers, RetNet, Mamba, as well as hybrid versions of Mamba that incorporate attention mechanisms (§4). We also compare with pretrained Mamba and Pythia (Biderman et al., 2023) at two parameter scales, ∼400M and 1.4B. Building on existing literature that explores the capabilities of linear recurrent models in language modeling (Arora et al., 2024a; Jelassi et al., 2024), we further investigate the performance of models trained from scratch in recalling context tokens during the translation process (§4.2). Moreover, we extend our analysis by investigating the models' ability to handle long contexts, on paragraph-level datasets (§5), along with measuring their sensitivity to different sequence lengths (§5.2) and inference cost (§5.4). Overall, our main findings are:[1]

- For sentence-level experiments, we show that Mamba exhibits competitive performance compared to transformers, for both trained-from-scratch and pretrained models.
- At the paragraph level, we find that Mamba is sensitive to the training distribution's sequence length and struggles with longer inputs. However, shifting the distribution towards longer sequence lengths helps to close the gap with transformers.
- We observe that integrating attention and state

[*]Equal contribution.

[1]https://github.com/deep-spin/ssm-mt

Proceedings of the Ninth Conference on Machine Translation (WMT), pages 1107–1124
November 15–16, 2024. ©2024 Association for Computational Linguistics

space models creates a strong model in terms of translation quality, robustness to sequence length extrapolation, and ability to recall named entities.

2 Background

In this section, we present an overview of transformers, and the foundation of the linear recurrent models covered in this paper: linear attention (RetNet) and state space models (Mamba).

2.1 Transformers

The key component in the transformer architecture is the attention mechanism, which is responsible for contextualizing information within and across input sequences. Concretely, given query $\boldsymbol{Q} \in \mathbb{R}^{n\times d}$, key $\boldsymbol{K} \in \mathbb{R}^{n\times d}$, and value $\boldsymbol{V} \in \mathbb{R}^{n\times d}$ matrices as input, where n is the sequence length and d the hidden size, the single head *self-attention mechanism* is defined as follows (Vaswani et al., 2017):

$$\boldsymbol{Y} = \mathsf{softmax}\left(\frac{\boldsymbol{Q}\boldsymbol{K}^\top}{\sqrt{d}}\right)\boldsymbol{V} \in \mathbb{R}^{n\times d}. \quad (1)$$

For decoder-only models, a causal mask is used to ignore future tokens. Notably, the $\boldsymbol{Q}\boldsymbol{K}^\top$ operation leads to a $\mathcal{O}\left(n^2\right)$ cost during training, and $\mathcal{O}\left(n\right)$ during inference with caching and causal masking.

2.2 Linear Attention

Denote by $\boldsymbol{q}_i, \boldsymbol{k}_i, \boldsymbol{v}_i, \boldsymbol{y}_i \in \mathbb{R}^d$ respectively the (column) vectors corresponding to the ith rows of the matrices $\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}, \boldsymbol{Y}$ defined above. Katharopoulos et al. (2020) reformulate the attention mechanism by casting the role of the softmax as a similarity function $\mathsf{sim}\left(\boldsymbol{q}, \boldsymbol{k}\right) = \exp\left(\boldsymbol{q}^\top \boldsymbol{k}/\sqrt{d}\right)$:

$$\boldsymbol{y}_i = \frac{\sum_{j=1}^n \mathsf{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j)\boldsymbol{v}_j}{\sum_{j=1}^n \mathsf{sim}(\boldsymbol{q}_i, \boldsymbol{k}_j)}. \quad (2)$$

However, any kernel $k(\boldsymbol{x}, \boldsymbol{y}) : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a suitable candidate for the similarity function (Smola and Schölkopf, 1998; Tsai et al., 2019). In particular, a kernel $k(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{\phi}(\boldsymbol{x})^\top \boldsymbol{\phi}(\boldsymbol{y})$, where $\boldsymbol{\phi} : \mathbb{R}^d \to \mathbb{R}^r$ is a feature map, leads to:

$$\begin{aligned}\boldsymbol{y}_i &= \frac{\sum_{j=1}^n \boldsymbol{\phi}(\boldsymbol{q}_i)^\top \boldsymbol{\phi}(\boldsymbol{k}_j)\boldsymbol{v}_j}{\sum_{j=1}^n \boldsymbol{\phi}(\boldsymbol{q}_i)^\top \boldsymbol{\phi}(\boldsymbol{k}_j)} \\ &= \frac{\sum_{j=1}^n \boldsymbol{v}_j \boldsymbol{\phi}(\boldsymbol{k}_j)^\top \boldsymbol{\phi}(\boldsymbol{q}_i)}{\sum_{j=1}^n \boldsymbol{\phi}(\boldsymbol{k}_j)^\top \boldsymbol{\phi}(\boldsymbol{q}_i)} \\ &= \frac{\boldsymbol{S}^\top \boldsymbol{\phi}(\boldsymbol{q}_i)}{\boldsymbol{z}^\top \boldsymbol{\phi}(\boldsymbol{q}_i)}, \end{aligned} \quad (3)$$

where $\boldsymbol{S} = \sum_{j=1}^n \boldsymbol{\phi}(\boldsymbol{k}_j)\boldsymbol{v}_j^\top \in \mathbb{R}^{r\times d}$ and $\boldsymbol{z} = \sum_{j=1}^n \boldsymbol{\phi}(\boldsymbol{k}_j) \in \mathbb{R}^r$. Notably, if initial states are initialized as $\boldsymbol{S}_0 = \boldsymbol{0}_{r\times d}$ and $\boldsymbol{z}_0 = \boldsymbol{0}_r$, intermediate states can be computed in a recurrent fashion:

$$\begin{aligned}\boldsymbol{S}_i &= \boldsymbol{S}_{i-1} + \boldsymbol{\phi}(\boldsymbol{k}_i)\boldsymbol{v}_i^\top, \\ \boldsymbol{z}_i &= \boldsymbol{z}_{i-1} + \boldsymbol{\phi}(\boldsymbol{k}_i). \end{aligned} \quad (4)$$

Since we can reuse the same $\boldsymbol{S}_i$ and $\boldsymbol{z}_i$ for all queries, this recurrent variant offers a $\mathcal{O}\left(n\right)$ complexity during training and enjoys a $\mathcal{O}\left(1\right)$ complexity for inference.[2]

Retentive Networks (RetNet). Sun et al. (2023a) set $\boldsymbol{\phi}$ as the identity function, i.e., $k(\boldsymbol{q}, \boldsymbol{k}) = \boldsymbol{q}^\top \boldsymbol{k}$, ignore the normalizer in Equation 2, and introduce an exponential decay mask γ, leading to:

$$\begin{aligned}\boldsymbol{S}_i &= \gamma \boldsymbol{S}_{i-1} + \boldsymbol{k}_i \boldsymbol{v}_i^\top, \\ \boldsymbol{y}_i &= \boldsymbol{S}_i^\top \boldsymbol{q}_i. \end{aligned} \quad (5)$$

This formulation effectively biases the attention mechanism to focus on closer token interactions. RetNet also uses XPos (Sun et al., 2023b), a relative positional encoding method, to improve its context extrapolation abilities.

2.3 State Space Models (SSMs)

SSMs (Gu et al., 2020) provide an alternative sequence mixing layer by processing sequences $\boldsymbol{x}_1, ..., \boldsymbol{x}_n$, where each $\boldsymbol{x}_i \in \mathbb{R}^d$, through a linear recurrence. Letting $\boldsymbol{H}_i \in \mathbb{R}^{r\times d}$ denote the "state" at the ith time step, a discrete SSM is defined as follows:[3]

$$\begin{aligned}\boldsymbol{H}_i &= \boldsymbol{A}\boldsymbol{H}_{i-1} + \boldsymbol{b}\boldsymbol{x}_i^\top, \\ \boldsymbol{y}_i &= \boldsymbol{H}_i^\top \boldsymbol{c}, \end{aligned} \quad (6)$$

where $\boldsymbol{A} \in \mathbb{R}^{r\times r}$, $\boldsymbol{b} \in \mathbb{R}^r$, and $\boldsymbol{c} \in \mathbb{R}^r$ are (discrete) parameters.[4] Since the same parameters are used for both relevant and irrelevant inputs, this model is deemed *input-independent*, which, in turn,

[2] In practice, however, this recurrent view is not parallelizable, leading to chunkwise-recurrent variations for training (Hua et al., 2022; Sun et al., 2023a; Yang et al., 2024).

[3] A discretization step is needed in order to obtain discrete parameters. For example, a possible method for this step is the zero-order hold rule, used by Mamba (Gu and Dao, 2023).

[4] The SSM equations are commonly written independenty for each input dimension $j \in [d]$ as

$$\boldsymbol{h}_i^{(j)} = \boldsymbol{A}\boldsymbol{h}_{i-1}^{(j)} + \boldsymbol{b}x_i^{(j)}, \quad y_i^{(j)} = \boldsymbol{c}^\top \boldsymbol{h}_i^{(j)},$$

with $\boldsymbol{A}$, $\boldsymbol{b}$, and $\boldsymbol{c}$ shared across input dimensions. This is equivalent to (6), where the jth-column of $\boldsymbol{H}_i$ equals $\boldsymbol{h}_i^{(j)}$.

makes the model unable to reset or overwrite its hidden states. S4 (Gu et al., 2022) is an instance of this model, which enjoys a $\mathcal{O}(n \log n)$ time complexity during training, and $\mathcal{O}(1)$ during inference. Vardasbi et al. (2023) shows that S4 still underperforms transformers for MT. Finally, note the similarity between Eq. 5 and Eq. 6: RetNets can be seen as state space models with $\boldsymbol{A} = \gamma \boldsymbol{I}$ and data-dependent $\boldsymbol{b}$ and $\boldsymbol{c}$.

Mamba. To make the SSM parameters *data-dependent*, Mamba (Gu and Dao, 2023) introduces a selection mechanism that uses learnable linear projections over $\boldsymbol{x}$ prior to the discretization step, effectively making all parameters dependent on the ith input. This leads to:

$$\begin{aligned} \boldsymbol{H}_i &= \boldsymbol{A}_i \odot \boldsymbol{H}_{i-1} + \boldsymbol{B}_i \odot \boldsymbol{X}_i, \\ \boldsymbol{y}_i &= \boldsymbol{H}_i^\top \boldsymbol{c}_i, \end{aligned} \tag{7}$$

where $\boldsymbol{X}_i = \mathbf{1}_r \boldsymbol{x}_i^\top \in \mathbb{R}^{r \times d}$ is an r-sized stack of the input, $\boldsymbol{A}_i \in \mathbb{R}^{r \times d}$ represents d diagonal matrices of size $r \times r$, $\boldsymbol{B}_i \in \mathbb{R}^{r \times d}$, $\boldsymbol{c}_i \in \mathbb{R}^r$, and $\odot$ is the Hadamard product. Note that, unlike S4, where the same $\boldsymbol{A}$ and $\boldsymbol{B}$ parameters are shared across all hidden dimensions $1 \leq h \leq d$, Mamba defines $\boldsymbol{A}_i$ and $\boldsymbol{B}_i$ with a shape of $(\ldots, d)$, allowing for unique parameters in each hidden dimension. While this formulation makes Mamba more expressive, it disrupts the convolutional approach used for training in S4. To address this, Gu and Dao (2023) propose an efficient IO-aware and parallelizable associative scan algorithm for training (Smith et al., 2023). Nonetheless, the recurrent view can still be used for inference with a $\mathcal{O}(1)$ time complexity.

3 Experimental Setup

We conduct experiments with transformers, RetNet, and Mamba for MT in §4 and §5. In this section, we detail the sentence and paragraph-level datasets used in our experiments, along with the settings for our models, which are trained in two distinct regimes: from scratch, or finetuned from a pretrained checkpoint.

3.1 Datasets

For sentence-level experiments, we focus on WMT14 DE↔EN and WMT16 RO↔EN for consistency with previous works (Vardasbi et al., 2023), but also include WMT16 FI↔EN using the standard training, validation and test splits. For paragraph level, we use the more recent WMT23 dataset (Kocmi et al., 2023), which contains ∼300M training samples and ∼1K test samples incorporating multi-sentence passages. In order to obtain a small high-quality subset for training, we exclude ParaCrawl and CommonCrawl samples from the original dataset and clean the remaining data. Our cleaning process includes three steps. First, we identify and remove samples in incorrect languages via `langdetect`[5]. Second, we eliminate duplicates. Third, we rank the samples using COMETKIWI-22 (Rei et al., 2022b) a state-of-the-art translation quality estimator, and keep only the top 6M samples. We call the refined dataset WMT23-6M. Datasets statistics are shown in Table 1.

DATASET	# SAMPLES	# TOKENS
IWSLT17 (DE↔EN)	200K	45.2 ± 29.5
WMT16 (RO↔EN)	610K	58.9 ± 31.1
WMT16 (FI↔EN)	2.08M	52.8 ± 33.1
WMT14 (DE↔EN)	4.5M	62.1 ± 45.6
WMT23-6M (DE↔EN)	6M	58.4 ± 32.9
WMT23-CAT-5 (DE↔EN)	2M	171.3 ± 134.9
WMT23-CAT-10 (DE↔EN)	1M	312.4 ± 282.3
WMT23 Test (DE→EN)	549	135.1 ± 147.7
WMT23 Test (EN→DE)	557	185.2 ± 188.2
Ted Talks Val. (DE↔EN)	995	268.5 ± 189.6
Ted Talks Test (DE↔EN)	2247	939.2 ± 594.0

Table 1: Sentence and paragraph-level datasets statistics.

3.2 Models

We make a broad selection of models spanning both trained-from-scratch and finetuned versions. In the first setting, we compare standard transformers, linear recurrent models, and also hybrid approaches that integrate attention into Mamba. For finetuned models, we experiment with released Pythia and Mamba checkpoints. We describe each model next.

3.2.1 Standard Models

Transformers. We select two variants of the transformer model as baselines: a base encoder-decoder formulation and a modern decoder-only version. The **Transformer Enc-Dec.** model, as described in the original paper (Vaswani et al., 2017), has 77M parameters, and uses sinusoidal positional embeddings and standard ReLU activations. The second variant, **Transformer++**, is a decoder-only formulation incorporating recent advancements, such as rotary positional embeddings (Su et al., 2024) and the SwiGLU layer (Shazeer,

[5] https://github.com/Mimino666/langdetect

2020). Specifically, we use the LLaMA architecture (Touvron et al., 2023), adjusting the embedding dimension to match the parameter count of the base transformer (79M), consistent with the version employed in (Gu and Dao, 2023).

Linear recurrent models. We select two representative recurrent models, **RetNet** (Sun et al., 2023a) and **Mamba** (Gu and Dao, 2023). Both models are tested with 77M parameters to approximately match the number of parameters in the transformer models.

3.2.2 Hybrid Models

Previous work has shown that incorporating attention into linear recurrent models leads to strong performance in language modeling (Fu et al., 2023; Arora et al., 2024b; De et al., 2024). Therefore, we aim to examine if this is also the case for MT by exploring three hybrid variants, detailed next.

Mamba-MHA. The simplest hybrid formulation involves replacing some of the Mamba layers with attention. Some natural questions then arise: how many attention layers are needed, and where to place them? After careful ablations, detailed in Appendix B, we use two attention layers placed at the middle and at the output of the network, resembling the hybrid version of H3 (Fu et al., 2023).

Mamba-Local. While aiming to achieve robust performance, the introduction of full attention to Mamba disrupts its efficiency gains. Thus, we consider local attention variants such as sliding window attention (Beltagy et al., 2020; Child et al., 2019), employed in recent hybrid models (Arora et al., 2024b; De et al., 2024). We use a window size of 64 based on the average sequence length shown in Table 1 and ablations in Appendix B.

Mamba Enc-Dec. Lastly, inspired by the S4-based encoder-decoder model from Vardasbi et al. (2023), we replace the self-attention mechanism in transformers with a Mamba block and keep the cross-attention component intact. In terms of complexity, since this variant computes attention over the source sentence, it incurs an additional $\mathcal{O}\left(n^2\right)$ cost for training and $\mathcal{O}\left(n\right)$ for inference.

3.2.3 Pretrained Models

In order to fairly evaluate the relative performance between pretrained models, we need to ensure consistency between their pretraining data. Taking this into account, we consider two strong models pretrained on The Pile (Gao et al., 2020): Pythia (Biderman et al., 2023), a modern transformer, and Mamba, a modern SSM. Note, however, that Pythia was pretrained on more tokens than Mamba (see Table 6), hence the comparison might be slightly unfavorable to Mamba. We experiment with two model scales, *small* (S) and *medium* (M). Concretely, we experiment with Pythia 410M and 1.4B, and with Mamba 370M and 1.4B.

3.3 Training and Evaluation

For models trained from scratch, we follow the settings proposed in (Vardasbi et al., 2023), whereas for pretrained models, we follow the finetuning settings used by Mamba (Gu and Dao, 2023). For decoder-only models, we pass a concatenation of the source and target sequences separated by a special token as input. We evaluate all models with BLEU (Post, 2018)[6] and COMET (Rei et al., 2022a).[7] We base our analysis on the latter, given its strong correlation with human judgments on sentence and paragraph-level data (Freitag et al., 2022, 2023). More training details can be found in §A.

4 Sentence-level Translation

We start by evaluating our standard, hybrid, and finetuned models on the sentence-level WMT16 RO↔EN, FI↔EN and WMT14 DE↔EN datasets. Results can be found in Table 2 in terms of BLEU and COMET. Next, we discuss the key findings.

4.1 Discussion

Mamba is competitive when trained from scratch. Mamba, a decoder-only model, not only outperforms a decoder-only transformer (Transformer++) across the board, but also an encoder-decoder transformer (Transf. Enc-Dec) in the larger WMT14 for both DE↔EN language pairs. This creates a contrast with the S4 results obtained by Vardasbi et al. (2023). We hypothesize that Mamba's good results are due to its data-dependent state updates (Eq. 7), which allows for more precise information retention in its hidden state. On the other hand, RetNet's performance is generally subpar compared to other models, likely due to its strong locality bias (induced by γ in Eq. 5), which may hinder performance in MT, a task where the source

[6] SacreBLEU signature: `|1|mixed|no|13a|exp|`

[7] `huggingface.co/Unbabel/wmt22-comet-da`

		WMT16								WMT14			
		RO→EN		EN→RO		FI→EN		EN→FI		DE→EN		EN→DE	
MODEL	SIZE	BLEU	COMET	BLEU	COMET	BLEU	COMET	BLEU	COMET	BLEU	COMET	BLEU	COMET
Trained from scratch													
Transf. Enc-Dec	77M	**29.2**	74.8	22.0	**78.6**	15.3	70.5	**14.8**	**78.2**	27.4	78.6	22.3	77.1
Transformer++	79M	26.4	72.6	21.7	72.7	14.9	69.3	14.2	75.5	26.9	79.0	22.8	77.9
RetNet	77M	26.4	72.4	19.9	76.0	14.5	70.2	11.0	70.2	23.4	74.7	19.6	71.7
Mamba	77M	27.0	73.8	21.4	77.9	16.0	72.7	13.0	75.4	**27.5**	80.2	22.4	77.8
Mamba-MHA	78M	28.5	**75.1**	21.7	78.3	**17.5**	**73.8**	14.3	76.4	27.4	**80.6**	**23.2**	**79.9**
Mamba-Local	78M	25.9	73.9	20.9	76.9	16.3	73.1	13.2	75.4	27.2	80.1	23.2	79.5
Mamba Enc-Dec	82M	28.5	74.4	**22.7**	77.9	17.0	73.6	14.3	77.0	27.2	80.0	21.6	78.8
Finetuned													
Pythia-S	410M	33.4	82.0	24.1	85.8	19.8	80.1	16.5	84.6	30.9	83.6	25.2	84.0
Mamba-S	370M	**34.1**	**83.2**	24.2	86.4	**21.4**	81.5	16.5	85.5	29.8	83.3	25.0	83.2
Pythia-M	1.4B	33.9	**83.2**	**24.9**	**87.1**	20.9	81.7	17.8	**87.1**	**32.2**	**84.5**	**26.7**	**84.9**
Mamba-M	1.4B	33.8	83.1	24.5	86.2	21.3	**82.1**	**18.4**	86.8	31.9	**84.5**	26.5	84.2

Table 2: Sentence-level results in terms of BLEU and COMET for models trained from scratch (top) and models finetuned from a pretrained checkpoint (bottom). **Bold** represents top results; underline represents second-best.

input servers as a prefix to the translation, and it requires "focused attention" during decoding.

Attention benefits Mamba. By including attention layers in Mamba's architecture, we find that Mamba-MHA, which employs only two attention layers, is able to outperform both transformers and Mamba for almost all language pairs. While Mamba-Local retains constant inference complexity via windowed attention, it is not as strong as the full attention variant. Finally, Mamba Enc-Dec's performance is also competitive, falling just short of Mamba-MHA and echoing the S4 encoder-decoder findings of Vardasbi et al. (2023).

Pretraining improves all models. We note a large COMET gap, roughly 4-8 COMET points, between the finetuned models and those trained from scratch for all language pairs. This is expected, since not only are these models bigger, but they also have strong data-driven priors, which are beneficial in downstream tasks (Amos et al., 2024).

Larger models achieve top results. For small models, Mamba outperforms Pythia for RO↔EN and FI↔EN in terms of COMET and BLEU. However, Pythia is superior on the larger DE↔EN datasets. Moving to larger models, we note that Mamba improves COMET scores by ∼1 point on EN↔DE and 0.6-1.3 point on EN↔FI while dropping only 0.1-0.2 on EN↔RO datasets. On the other hand, Pythia improves results consistently for all language pairs with a larger model size, outperforming or matching the results of other models. On average, we find that both their gaps decrease when moving from smaller to medium-sized models but Pythia benefits more in terms of COMET. It is worth noting that Mamba is pretrained on fewer samples than Pythia (see Table 6) and that the impact of pretraining data quality can also play a role in downstream task performance.

4.2 Recall of Named Entities

Following our discussion of sentence-level translation, we now focus on how well these models recall context tokens during translation. Inspired by prior studies investigating the recall of context tokens in language modeling with state space models (Arora et al., 2024a; Jelassi et al., 2024), we conduct a similar experiment for MT. Unlike language modeling, where token patterns often recur within a near context, MT presents a challenge due to the distinct spelling of words across languages. Therefore, we focus on the recall of named entities (NEs) that appear verbatim in both source and target sentences, using NLTK for NE recognition (Bird, 2006).

We assess the models' ability to recall NEs from the WMT16 RO→EN dataset according to their frequency in the training set, as illustrated in Figure 1. The results reveal a clear correlation between NE frequency and their chance to be recalled in the translation process, as more frequent NEs are recalled more often. Notably, we note a disparity in performance with unseen entities, which provides a more illustrative view of recall ability. In this respect, transformers and Mamba perform on par, while RetNet shows inferior results. As before, the hybrid models are promising, with Mamba-

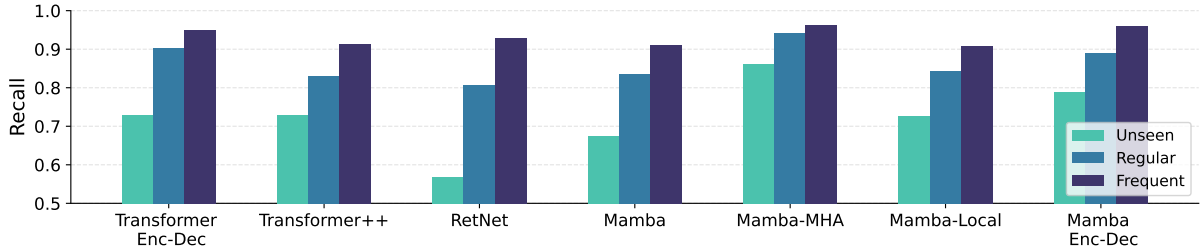

Figure 1: Recall in recovering named entities on the WMT16 RO→EN dataset by their training set frequency: *unseen* entities do not appear in the training data, while *regular* and *frequent* entities appear $[1, 16)$ and $16+$ times.

MHA outperforming all models, followed closely by Mamba Enc-Dec. We include additional analyses for other datasets in the Appendix §C.

5 Paragraph-level translation

While Mamba shows competitive performance with transformers on sentence-level datasets (see Table 2), it was originally designed to handle long sequences. Thus, we now turn our attention to paragraph-level datasets. This allows us to study the models' sensitivity to long sequence lengths along with their robustness in handling sequences that are longer than the ones seen during training.[8]

To this end we focus on the WMT23-6M dataset (§3.1), from which the training and test sets are composed of sentence and paragraph-level passages, respectively. In order to see the impact of training on long sequences, we propose to artificially construct multi-sentence datasets, which we call WMT23-CAT-N. Our procedure is as follows:

1. We first retain the original training samples from WMT23-6M with a probability of 50%.
2. Next, for the remaining part, we concatenate N random training samples.

This approach ensures that we consistently maintain a 50% ratio between single-sentence and multi-sentence samples. For validation, we sample 1-to-10-sentence passages from the TED Talks dataset (Cettolo et al., 2012). Statistics for CAT-N datasets can be found in Table 1. COMET scores on the WMT23 EN↔DE test sets are shown in Table 3. We provide additional BLEU scores in Table 9 in Appendix E. Next, we discuss our key findings.

5.1 Discussion

Concatenation helps. Our strategy of concatenating sentences proves beneficial for almost all models, as COMET scores typically improve with the CAT-5 and CAT-10 datasets, whether models are trained from scratch or finetuned. Among models trained from scratch, Transformer Enc-Dec, Mamba-MHA, and Mamba Enc-Dec show substantial improvements, with Mamba Enc-Dec achieving the best overall results. For finetuned models, concatenation benefits larger models; Mamba-M outperforms Pythia-M in DE→EN but underperforms in EN→DE. Interestingly, for both training regimes, the concatenation strategy can lead to COMET gains of up to 5 points.

Finetuning outperforms training from scratch. Finetuned models consistently achieve higher COMET scores, with larger models attaining the top results overall. Similar to sentence-level experiments, Pythia outperforms Mamba when trained on the original, WMT23-6M dataset. However, both Pythia and Mamba benefit from our concatenation strategy. While these results indicate that our concatenation strategy helps in translating long inputs, it remains unclear whether performance on short inputs is compromised or if the models can handle longer inputs than those seen during training. We investigate these issues next.

5.2 Sensitivity to Input Length

Based on the previous observations, we notice that performance between models varies considerably after being exposed to different sequence lengths during training. In this subsection, we investigate how robust each model is to length distribution shifts between training and test. Results are shown in Figure 2 for both training regimes on the WMT23 DE→EN dataset. Results are consistent for EN→DE, shown in Figure 6, Appendix D.

Discussion. When training on WMT23-6M, we observe a decline in performance for all models on long sequences, affecting both trained-from-scratch

[8]We dropped RetNet and Mamba-Local as they already achieve poor results on long *sentence-level* inputs (see Fig. 5).

		DE→EN			EN→DE		
MODEL	SIZE	ORIG.	CAT5	CAT10	ORIG.	CAT5	CAT10
Trained from scratch							
Transf. Enc-Dec	77M	72.4	74.6	69.6	65.2	<u>70.3</u>	<u>70.3</u>
Transformer++	79M	70.7	73.6	72.8	64.8	69.1	68.8
Mamba	77M	70.0	73.3	72.3	63.3	67.5	67.8
Mamba-MHA	78M	72.7	74.2	<u>74.5</u>	67.0	68.6	69.7
Mamba Enc-Dec	82M	70.7	73.8	**75.6**	65.3	**71.0**	70.1
Finetuned							
Pythia-S	410M	77.4	78.4	79.0	76.7	<u>77.8</u>	77.1
Mamba-S	370M	77.2	78.2	78.3	72.4	74.2	73.1
Pythia-M	1.4B	76.2	78.6	79.4	75.8	77.4	**79.0**
Mamba-M	1.4B	74.6	**79.6**	<u>79.5</u>	73.4	77.5	77.3

Table 3: Paragraph-level results in terms of COMET for models trained from scratch (top) and models finetuned from a pretrained checkpoint (bottom) on WMT23 EN↔DE test set, according to the training dataset: original WMT23-6M, WMT23-CAT-5 and WMT23-CAT-10. **Bold** represents top results; <u>underline</u> represents second-best.

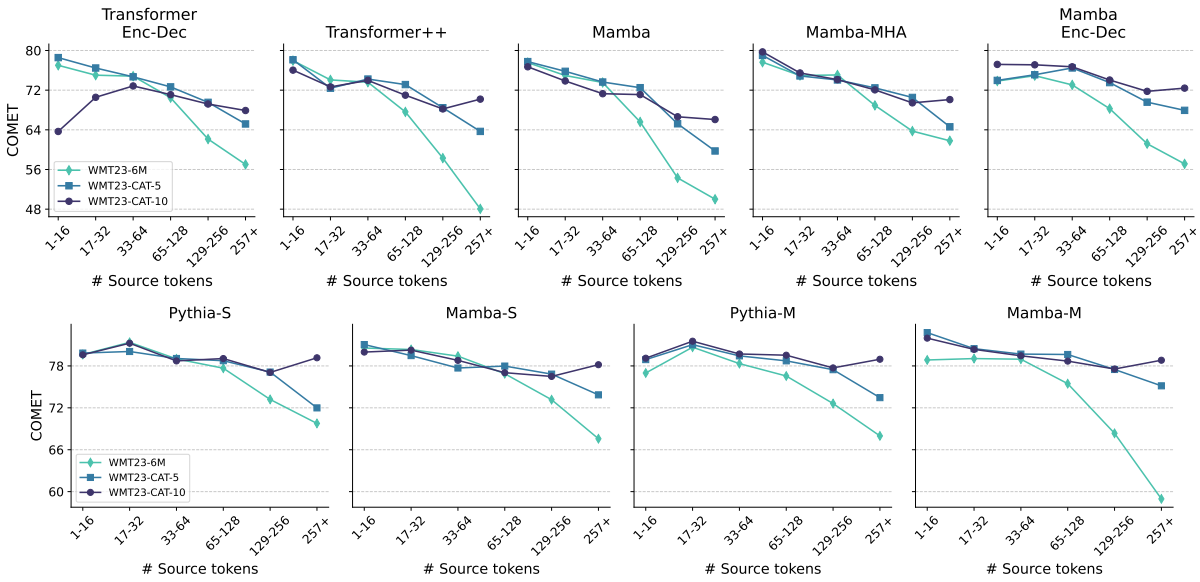

Figure 2: Sensitivity to input length, measured by the number of sources tokens, on the WMT23 DE→EN datset, for models trained from scratch (top) and finetuned from a pretrained checkpoint (bottom).

and finetuned variants. Interestingly, this degradation is evident in Mamba, as expected due to its finite hidden state capacity. However, it is also challenging for transformers despite their relative positional embeddings. Moreover, both finetuned and hybrid models exhibit more consistent performance across different sequence lengths, even on the original sentence-level dataset, suggesting a more robust capability for dealing with long-context inputs.

Overall, our concatenation approach has largely mitigated the performance issues with long inputs present in models trained on WMT23-6M, as models trained on CAT datasets produce higher-quality translations for longer sequences. This improvement is uniform across all models, with CAT-10 yielding consistently better translations in the longest bin (257+ tokens). However, the CAT-10 dataset seems to reduce translation quality for shorter samples in some models, though this effect is minimal or absent in hybrid and finetuned models. Next, we further examine the ability to extrapolate to even longer sentences than those seen during training.

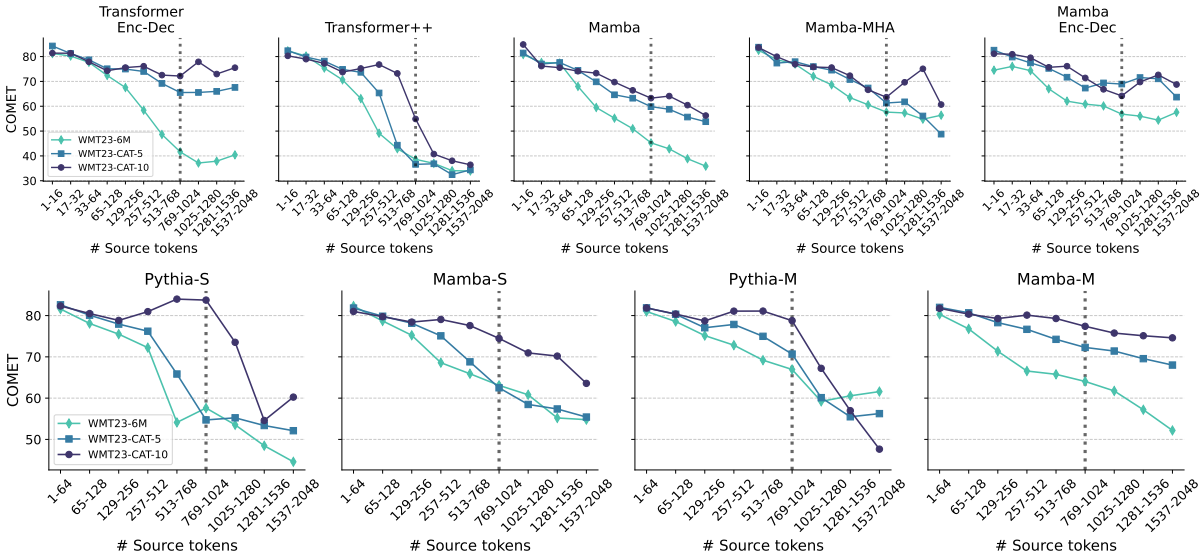

Figure 3: Sensitivity to input length, measured by the number of sources tokens, on the Ted Talks DE→EN dataset, for models trained from scratch (top) and finetuned from a pretrained checkpoint (bottom). The dashed vertical line represents the bin containing the longest sentence in the training set.

5.3 Extrapolation to Longer Sequences

Following the previous discussion, to further explore the impact of sequence length on our models, we create a new test set sampled from TED Talks DE→EN passages that is larger (2200 samples) and contains even longer sequences (up to 2048 tokens) than WMT23. Details on this dataset can be found in Table 1. The source length distribution can be seen in Figure 7. After evaluating our models in this dataset, we plot COMET scores per sentence length in Figure 3, where we include a dashed vertical line representing the bin containing the longest sentences the model has been exposed to during training.

Discussion. When training from scratch, we highlight the sharp decline in translation quality decline in the Transformer++ model when considering samples larger than those it has been exposed to during training, this finding is consistent with the generalization task in (Jelassi et al., 2024). In contrast, Transformer Encoder-Decoder and Mamba exhibit a steady decline overall with the first being robust to generalization problems when trained with larger-context datasets. Additionally, the hybrid models prove to excel at generalization, providing good translation quality even when trained with the WMT23-6M dataset. With the finetuned models, we also see decreasing translation quality over longer sequences which is consistent with previous experiments. Nonetheless, Mamba models show a more robust trend when generalizing to unseen lengths. In particular, the larger Mamba-M, when trained on the WMT23-CAT-10 dataset, exhibits a much lower performance degradation on longer samples in comparison to other finetuned models.

5.4 Inference Cost

In §2 we covered the theoretical time complexity of our models in training and inference time. Here, we examine the wallclock time and memory usage of Pythia and Mamba in a realistic setting where inputs are WMT23 DE→EN test samples, and outputs continue to be generated until they reach exactly $L \in \{512, 1024\}$ tokens. Table 4 shows that Mamba's memory usage is significantly lower than Pythia's, with gaps of $\sim$ 3-5x overall. The wallclock time difference is not as notable but still substantial, especially for larger models, where Mamba-M is 2x faster than Pythia-M for $L = 1024$. In other words, Mamba-M throughputs $\sim$806 tokens/s while Pythia-M outputs $\sim$405 tokens/s, aligning with (Gu and Dao, 2023).[9]

6 Related Works

Linear recurrent models for MT. Our work is closely related to (Vardasbi et al., 2023), which compares SSMs and transformers. Furthermore, they also experiment with hybrid architectures composed of S4 and attention layers, an approach that has since become common (Arora et al., 2024b; De

[9]Computed as batch-size $\times$ L/wallclock-time.

	512		1024	
MODEL	T (S)	M (GB)	T (S)	M (GB)
Pythia-S	11.52	2.472	25.80	3.934
Mamba-S	10.38	0.839	20.59	1.607
Pythia-M	14.88	4.789	40.41	7.841
Mamba-M	10.29	0.913	20.31	1.668

Table 4: Average time (T) and maximum allocated memory (M) of 30 inference runs with batch size 16 on WMT23 DE→EN.

et al., 2024; Glorioso et al., 2024). In this work, we experiment with more recent linear recurrent models and their respective hybrid versions while also including larger and pretrained variants. Our analysis further includes investigating each model's ability to recall named entities, along with measuring translation performance across different sequence lengths on paragraph-level datasets. In contrast to Vardasbi et al. (2023)'s results showing that S4 lags behind transformer baselines in MT tasks, we observe that Mamba, a modern SSM, is competitive with transformers on sentence and paragraph-level datasets, whether trained from scratch or fine-tuned from a pretrained checkpoint, especially in the first setting when equipped with attention mechanisms.

Linear recurrent models' limitations. Recent works show that Mamba struggles in tasks that involve recalling context tokens (Arora et al., 2024a; Jelassi et al., 2024), such as the synthetic Multi-Query Associative Recall task. In MT, however, context tokens (source and translation prefix) are not often replicated in the output (translation). In this work, we study this phenomenon with named entities and analyze the recall ability of transformers and linear recurrent models in §4.2.

Sentence concatenation Kondo et al. (2022); Varis and Bojar (2021) analyze transformers' generalization towards sequence length. They show that transformers are susceptible to the training distribution of context length and that concatenating multiple sentences can improve the translation of longer sentences. Specifically, Kondo et al. (2022) augment the original data with samples containing concatenations of two random sentences, while Varis and Bojar (2021) concatenate up to six sentences. While these studies focused on sentence-level translation with sequence lengths up to 120 tokens, in this work, we extend the analysis to much longer sequences and test on paragraph-level data from the WMT2023 dataset.

7 Conclusion

We set out to evaluate recent linear recurrent models, particularly RetNet and Mamba, in MT tasks while thoroughly comparing them to transformer baselines and hybrid models, which combine Mamba and attention. We find that Mamba models are competitive with transformers, both when they are trained from scratch and when they are finetuned from a pretrained checkpoint; however, the performance delta is smaller in the latter regime. Our paragraph-level experiments reveal that models are hindered by the mismatch in the training and test length distributions; however, a simple concatenation approach helps to mitigate the issue. We find that hybrid models are only slightly affected by this issue while also being competitive or outperforming transformers. Finally, we note that Mamba models also exhibit a faster runtime, consume less memory, and extrapolate better to longer inputs than decoder-only transformers.

Acknowledgments

We thank Haau-Sing Li, Saul Santos, Patrick Fernandes, Sweta Agrawal and Nuno Guerreiro for their useful and constructive comments. This work was supported by the Portuguese Recovery and Resilience Plan through project C645008882-00000055 (Center for ResponsibleAI), by the EU's Horizon Europe Research and Innovation Actions (UTTER, contract 101070631), by the project DECOLLAGE (ERC-2022-CoG 101088763), and by Fundação para a Ciência e Tecnologia through contract UIDB/50008/2020.

Limitations

We point out some limitations of the presented study. First, one limitation is that we refrain from pretraining the hybrid models due to the high associated compute costs. To this effect, while our trained-from-scratch results are promising, validating them with a larger scale and strong language priors would strengthen our claim of their good performance. Secondly, our experiments (§5.3) appear to indicate larger models are more robust to sequence length issues. Nonetheless, we limited our study to models with parameter scales between 370M and 1.4B since, in preliminary sentence-level experiments, translation quality gains plateaued at the latter scale.

In another direction, we mainly rely on automated metrics for evaluating translation quality,

which might not fully capture the accuracy of the translation. We alleviate this fault by considering the recollection of NEs in translations (§4.2). Furthermore, our experiments in §5.2 do not have a notion of translation difficulty, which might help explain the differences between models and associated datasets in different length buckets (albeit sentence length and difficulty may be connected).

Potential Risks

Translation biases and error modes inherent in transformed-based LLMs could also be manifested in the linear recurrent models studied in this paper. Careful evaluation and mitigation strategies, such as detecting and overcoming hallucinations (Guerreiro et al., 2023; Dale et al., 2023), can alleviate these risks and ensure models' responsible use. It should also be noted that although SSMs are potentially more energy efficient than transformer-based models, they still pose energy consumption concerns, particularly due to the large size of the models.

References

Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. 2024. In-context language learning: Architectures and algorithms. In *Forty-first International Conference on Machine Learning*.

Ido Amos, Jonathan Berant, and Ankit Gupta. 2024. Never train from scratch: Fair comparison of long-sequence models requires data-driven priors. In *The Twelfth International Conference on Learning Representations*.

Simran Arora, Sabri Eyuboglu, Aman Timalsina, Isys Johnson, Michael Poli, James Zou, Atri Rudra, and Christopher Re. 2024a. Zoology: Measuring and improving recall in efficient language models. In *The Twelfth International Conference on Learning Representations*.

Simran Arora, Sabri Eyuboglu, Michael Zhang, Aman Timalsina, Silas Alberti, James Zou, Atri Rudra, and Christopher Re. 2024b. Simple linear attention language models balance the recall-throughput tradeoff. In *Forty-first International Conference on Machine Learning*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer.

Stella Biderman, Hailey Schoelkopf, Quentin Anthony, Herbie Bradley, Kyle O'Brien, Eric Hallahan, Mohammad Aflah Khan, Shivanshu Purohit, USVSN Sai Prashanth, Edward Raff, Aviya Skowron, Lintang Sutawika, and Oskar Van Der Wal. 2023. Pythia: a suite for analyzing large language models across training and scaling. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org.

Steven Bird. 2006. NLTK: The Natural Language Toolkit. In *Proceedings of the COLING/ACL 2006 Interactive Presentation Sessions*, pages 69–72, Sydney, Australia. Association for Computational Linguistics.

Mauro Cettolo, Marcello Federico, Luisa Bentivogli, Jan Niehues, Sebastian Stüker, Katsuhito Sudoh, Koichiro Yoshino, and Christian Federmann. 2017. Overview of the IWSLT 2017 evaluation campaign. In *Proceedings of the 14th International Conference on Spoken Language Translation*, pages 2–14, Tokyo, Japan. International Workshop on Spoken Language Translation.

Mauro Cettolo, Christian Girardi, and Marcello Federico. 2012. WIT3: Web inventory of transcribed and translated talks. In *Proceedings of the 16th Annual Conference of the European Association for Machine Translation*, pages 261–268, Trento, Italy. European Association for Machine Translation.

Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. 2019. Generating long sequences with sparse transformers.

David Dale, Elena Voita, Loic Barrault, and Marta R. Costa-jussà. 2023. Detecting and mitigating hallucinations in machine translation: Model internal workings alone do well, sentence similarity Even better. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 36–50, Toronto, Canada. Association for Computational Linguistics.

Soham De, Samuel L. Smith, Anushan Fernando, Aleksandar Botev, George Cristian-Muraru, Albert Gu, Ruba Haroun, Leonard Berrada, Yutian Chen, Srivatsan Srinivasan, Guillaume Desjardins, Arnaud Doucet, David Budden, Yee Whye Teh, Razvan Pascanu, Nando De Freitas, and Caglar Gulcehre. 2024. Griffin: Mixing gated linear recurrences with local attention for efficient language models.

Patrick Fernandes, Kayo Yin, Graham Neubig, and André F. T. Martins. 2021. Measuring and increasing context usage in context-aware machine translation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6467–6478, Online. Association for Computational Linguistics.

Markus Freitag, Nitika Mathur, Chi-kiu Lo, Eleftherios Avramidis, Ricardo Rei, Brian Thompson, Tom Kocmi, Frederic Blain, Daniel Deutsch, Craig Stewart, Chrysoula Zerva, Sheila Castilho, Alon Lavie, and George Foster. 2023. Results of WMT23 metrics shared task: Metrics might be guilty but references

are not innocent. In *Proceedings of the Eighth Conference on Machine Translation*, pages 578–628, Singapore. Association for Computational Linguistics.

Markus Freitag, Ricardo Rei, Nitika Mathur, Chi-kiu Lo, Craig Stewart, Eleftherios Avramidis, Tom Kocmi, George Foster, Alon Lavie, and André F. T. Martins. 2022. Results of WMT22 metrics shared task: Stop using BLEU – neural metrics are better and more robust. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 46–68, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Daniel Y Fu, Tri Dao, Khaled Kamal Saab, Armin W Thomas, Atri Rudra, and Christopher Re. 2023. Hungry hungry hippos: Towards language modeling with state space models. In *The Eleventh International Conference on Learning Representations*.

Leo Gao, Stella Biderman, Sid Black, Laurence Golding, Travis Hoppe, Charles Foster, Jason Phang, Horace He, Anish Thite, Noa Nabeshima, Shawn Presser, and Connor Leahy. 2020. The pile: An 800gb dataset of diverse text for language modeling.

Paolo Glorioso, Quentin Anthony, Yury Tokpanov, James Whittington, Jonathan Pilault, Adam Ibrahim, and Beren Millidge. 2024. Zamba: A compact 7b ssm hybrid model.

Albert Gu and Tri Dao. 2023. Mamba: Linear-time sequence modeling with selective state spaces.

Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. 2020. Hippo: Recurrent memory with optimal polynomial projections. In *Advances in Neural Information Processing Systems*, volume 33, pages 1474–1487. Curran Associates, Inc.

Albert Gu, Karan Goel, and Christopher Re. 2022. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations*.

Nuno M. Guerreiro, Elena Voita, and André Martins. 2023. Looking for a needle in a haystack: A comprehensive study of hallucinations in neural machine translation. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 1059–1075, Dubrovnik, Croatia. Association for Computational Linguistics.

Weizhe Hua, Zihang Dai, Hanxiao Liu, and Quoc Le. 2022. Transformer quality in linear time. In *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 9099–9117. PMLR.

Samy Jelassi, David Brandfonbrener, Sham M. Kakade, and eran malach. 2024. Repeat after me: Transformers are better than state space models at copying. In *Forty-first International Conference on Machine Learning*.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: fast autoregressive transformers with linear attention. In *Proceedings of the 37th International Conference on Machine Learning*, ICML'20. JMLR.org.

Tom Kocmi, Eleftherios Avramidis, Rachel Bawden, Ondřej Bojar, Anton Dvorkovich, Christian Federmann, Mark Fishel, Markus Freitag, Thamme Gowda, Roman Grundkiewicz, Barry Haddow, Philipp Koehn, Benjamin Marie, Christof Monz, Makoto Morishita, Kenton Murray, Makoto Nagata, Toshiaki Nakazawa, Martin Popel, Maja Popović, and Mariya Shmatova. 2023. Findings of the 2023 conference on machine translation (WMT23): LLMs are here but not quite there yet. In *Proceedings of the Eighth Conference on Machine Translation*, pages 1–42, Singapore. Association for Computational Linguistics.

Seiichiro Kondo, Naoya Ueda, Teruaki Oka, Masakazu Sugiyama, Asahi Hentona, and Mamoru Komachi. 2022. Japanese named entity recognition from automatic speech recognition using pre-trained models. In *Proceedings of the 36th Pacific Asia Conference on Language, Information and Computation*, pages 102–108, Manila, Philippines. Association for Computational Linguistics.

Bo Peng, Eric Alcaide, Quentin Anthony, Alon Albalak, Samuel Arcadinho, Stella Biderman, Huanqi Cao, Xin Cheng, Michael Chung, Leon Derczynski, Xingjian Du, Matteo Grella, Kranthi Gv, Xuzheng He, Haowen Hou, Przemyslaw Kazienko, Jan Kocon, Jiaming Kong, Bartłomiej Koptyra, Hayden Lau, Jiaju Lin, Krishna Sri Ipsit Mantri, Ferdinand Mom, Atsushi Saito, Guangyu Song, Xiangru Tang, Johan Wind, Stanisław Woźniak, Zhenyuan Zhang, Qinghua Zhou, Jian Zhu, and Rui-Jie Zhu. 2023. RWKV: Reinventing RNNs for the transformer era. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14048–14077, Singapore. Association for Computational Linguistics.

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.

Ricardo Rei, José G. C. de Souza, Duarte Alves, Chrysoula Zerva, Ana C Farinha, Taisiya Glushkova, Alon Lavie, Luisa Coheur, and André F. T. Martins. 2022a. COMET-22: Unbabel-IST 2022 submission for the metrics shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 578–585, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Ricardo Rei, Marcos Treviso, Nuno M. Guerreiro, Chrysoula Zerva, Ana C Farinha, Christine Maroti, José G. C. de Souza, Taisiya Glushkova, Duarte Alves, Luisa Coheur, Alon Lavie, and André F. T.

Martins. 2022b. CometKiwi: IST-unbabel 2022 submission for the quality estimation shared task. In *Proceedings of the Seventh Conference on Machine Translation (WMT)*, pages 634–645, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics.

Noam Shazeer. 2020. Glu variants improve transformer.

Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. 2023. Simplified state space layers for sequence modeling. In *The Eleventh International Conference on Learning Representations*.

Alex J Smola and Bernhard Schölkopf. 1998. *Learning with kernels*, volume 4. Citeseer.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

Jianlin Su, Murtadha Ahmed, Yu Lu, Shengfeng Pan, Wen Bo, and Yunfeng Liu. 2024. Roformer: Enhanced transformer with rotary position embedding. *Neurocomput.*, 568(C).

Yutao Sun, Li Dong, Shaohan Huang, Shuming Ma, Yuqing Xia, Jilong Xue, Jianyong Wang, and Furu Wei. 2023a. Retentive network: A successor to transformer for large language models.

Yutao Sun, Li Dong, Barun Patra, Shuming Ma, Shaohan Huang, Alon Benhaim, Vishrav Chaudhary, Xia Song, and Furu Wei. 2023b. A length-extrapolatable transformer. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 14590–14604, Toronto, Canada. Association for Computational Linguistics.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023. Llama: Open and efficient foundation language models.

Yao-Hung Hubert Tsai, Shaojie Bai, Makoto Yamada, Louis-Philippe Morency, and Ruslan Salakhutdinov. 2019. Transformer dissection: An unified understanding for transformer's attention via the lens of kernel. 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 4344–4353, Hong Kong, China. Association for Computational Linguistics.

Ali Vardasbi, Telmo Pessoa Pires, Robin Schmidt, and Stephan Peitz. 2023. State spaces aren't enough: Machine translation needs attention. In *Proceedings of the 24th Annual Conference of the European Association for Machine Translation*, pages 205–216, Tampere, Finland. European Association for Machine Translation.

Dusan Varis and Ondřej Bojar. 2021. Sequence length is a domain: Length-based overfitting in transformer models. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Longyue Wang, Chenyang Lyu, Tianbo Ji, Zhirui Zhang, Dian Yu, Shuming Shi, and Zhaopeng Tu. 2023. Document-level machine translation with large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 16646–16661, Singapore. Association for Computational Linguistics.

Songlin Yang, Bailin Wang, Yikang Shen, Rameswar Panda, and Yoon Kim. 2024. Gated linear attention transformers with hardware-efficient training. In *Forty-first International Conference on Machine Learning*.

A Implementation and Training Details

All experiments were carried on Nvidia RTX A6000 GPUS with 48GB VRAM, and the training framework is constructed around PyTorch Lightning.[10] To train and generate translations in batches, we use a left-padding strategy. However, for Mamba, additional functionality is required to avoid processing padding tokens. To address this, we zero out inputs before and after convolution at the positions of the padding tokens and sacrifice some efficiency by using the slow path in Mamba.[11] Notably, during inference, the slow path affects only the initial processing of the prompt and does not impact the actual generation. Moreover, we added Dropout (Srivastava et al., 2014) to Mamba blocks, which was missing in the original implementation. Specifically, dropout is applied after the last linear projection of the Mamba block. Additionally, following the findings in (Vardasbi et al., 2023), we calculate cross-entropy loss only for target tokens. During training, we use greedy decoding and select the top model using BLEU as the validation metric, as it is faster to compute in comparison to COMET. For inference, we use beam search with a beam size of 5. Due to the time-consuming nature of our experiments, we report the results of a single run for all experiments. The overall model structure and hyperparameters across both training regimes, from-scratch (§A.1) and finetuning (§A.2), are shown in Table 5. Furthermore, all models were trained with `bfloat16` precision.

MODEL	SIZE	LR	L	H	D	FFN
Trained from scratch						
Transf. Enc-Dec	77M	4e-4	6-6	8	512	2048
Transf.++	79M	4e-4	12	8	496	1984
RetNet	77M	1e-3	12	4	512	1024
Mamba	77M	1e-3	24	-	610	-
Mamba-MHA	78M	7e-4	24	8	624	-
Mamba-Local	78M	7e-4	24	8	624	-
Mamba Enc-Dec	82M	7e-4	8-6	8	512	2048
Finetuned						
Pythia-S	410M	1e-5	24	16	1024	4096
Mamba-S	370M	3e-4	24	-	1024	-
Pythia-M	1.4B	1e-5	24	16	2048	8192
Mamba-M	1.4B	3e-4	24	-	2048	-

Table 5: Detailing the full set of hyperparameters for the different models. Encoder-Decoder models have their number of layers separated by each module. LR represents the Learning Rate; L represents the number of layers; H is the number of Attention Heads; D is the model dimension; FFN is the size of the inner feed-forward network.

MODEL	SIZE	TRAINING TOKENS	CONTEXT TOKENS
Pythia-S	410M	300B	2048
Pythia-M	1.4B	300B	2048
Mamba-S	370M	7B	2048
Mamba-M	1.4B	26B	2048

Table 6: Pre-training details. All models were pretrained on The Pile (Gao et al., 2020).

A.1 Training from Scratch

Regarding tokenization, we leverage the HuggingFace *tokenizers* library[12] and construct a separate BPE tokenizer (Sennrich et al., 2016) per dataset. The total vocabulary size is 32000 tokens. We carried out a hyperparameter search to select appropriate dropout values, learning rates and architectural decisions, with the latter two detailed in Table 5. We employ a dropout of 0.3 for WMT16 EN↔RO, 0.1 for WMT14 EN↔DE, WMT16 EN↔FI and the different variations of WMT23. Other hyperparameters were kept intact. Concretely, we use the Inverse Square Root learning rate scheduler (Vaswani et al., 2017) with 4000 warmup steps and a weight decay of 0.001.

A.2 Finetuning Pretrained Checkpoints

We employ pretrained models and corresponding tokenizers from the Huggingface library. Table 6 shows the number of tokens and the size of the context window used during pretraining. For finetuning, in all experiments, we use a dropout of 0.1 with the exception of WMT16 EN↔RO and Pythia-S + EN↔FI, where dropout varies from 0.1 to 0.3 for the former and 0 for the latter. Moreover, we use weight decay only in Mamba-M, with a value of $2 \cdot 10^{-4}$. Additionally, learning rates and models' attributes are shown in Table 5.

A.3 Inference Cost

For the inference cost experiments, we measure overall wallclock time using cuda events and cuda synchronization from `torch.cuda` module. The overall reported time measures the entire generation pipeline, including the use of beam search. Moreover, we use

[10] https://lightning.ai/docs/pytorch/

[11] https://github.com/state-spaces/mamba/issues/216

[12] https://github.com/huggingface/tokenizers

	512		1024	
MODEL	T (S)	M (GB)	T (S)	M (GB)
Transformer++	12.33	5.862	34.00	10.711
Mamba	11.71	0.562	29.37	0.554
Mamba-MHA	12.77	1.250	25.28	1.536
Mamba Enc-Dec	7.46	0.394	14.36	0.394

Table 7: Average time (T) and maximum allocated memory (M) of 30 inference runs with batch size 16 on WMT23 DE→EN.

	DE→EN		EN→DE	
	BLEU	COMET	BLEU	COMET
Mamba-MHA				
Interleaved	30.81	77.98	24.40	72.48
L1,11	30.52	78.10	**24.99**	73.76
L11,23	**30.81**	**78.30**	24.40	**73.94**
Mamba-Local				
Interleaved - w64	28.85	76.76	23.61	72.10
L11,23 - w16	29.37	77.19	24.12	72.88
L11,23 - w32	28.24	76.44	23.20	72.22
L11,23 - w64	29.40	77.56	24.41	72.98
L11,23 - w128	30.49	77.98	24.85	73.58

Table 8: Hybrid models ablations with BLEU and COMET scores on the IWSLT17 dataset. Different window sizes are denoted as w$\{16, 32, 64, 128\}$. *Interleaved* refers to alternating Mamba and attention layers. L*1,11* and L*11,23* refer to placing attention in layers 2 - $N/2$ and $N/2$ - N, respectively.

`torch.cuda.max_memory_allocated` to measure memory usage.

We additionally include the profiling measurements for the trained-from-scratch models in Table 7. Crucially, we advise that these metrics are rough estimates since the models are not optimized to perform at their best capacity. To this end, we do not include the Transformer Encoder-Decoder as the implementation used is not efficient.

B Hybrid Models Ablation

Building on the shortcomings of linear models (Akyürek et al., 2024; Arora et al., 2024a; Jelassi et al., 2024), we designed hybrid models to complement SSMs with attention mechanisms. In this section, we ablate the design choices leading to the construction of our hybrid models. These experiments were conducted using the IWSLT17 DE↔EN dataset (Cettolo et al., 2017). Results are shown in Table 8.

Since our Mamba-MHA model replaces a set of Mamba layers with attention modules, we ablated various configurations to determine the optimal number and placement of attention layers. Our analysis of COMET scores indicated that incorporating two attention layers significantly boosted performance, aligning with findings in previous studies (Fu et al., 2023). The placement of these layers had a minimal effect, leading us to select the configuration with layers at positions $N/2$ and N for further experiments due to its consistently higher COMET scores.

In the case of Mamba-Local, which uses a sliding window attention, we explored various window sizes. Our experiments revealed that performance generally improved with window size in a linear way. Ultimately, a 128-token window nearly matched full attention performance, and two layers of 64-token windowed attention provided a good balance between performance and efficiency for our experiments.

C Named Entity Recall Experiments

Following up on the discussion from §4.2, we extend our evaluation of NE recall accuracy to the WMT14 DE↔EN dataset and two paragraph-level datasets, WMT23-6M and WMT23-CAT-5, both in the DE↔EN translation direction. The results, detailed in Figure 4, offer further insights into the models' recall accuracy performance across other datasets and context length settings.

Sentence-Level (WMT14 DE↔EN). The NE recall results on the WMT14 DE↔EN dataset align closely with those obtained in WMT16 RO→EN, shown in Figure 1; we still observe Mamba's recall accuracy to be closer to that of the transformer models, while the hybrid models continue to (slightly) outperform their unmodified counterparts. Note, however, that overall, the gap between models is narrower, as also reflected in their close results in terms of BLEU.

Paragraph-Level Datasets. When assessing the WMT23-6M and WMT23-CAT-5 DE↔EN datasets, contrary to the WMT16 RO↔EN experiments, the Transformer Encoder-Decoder model outperforms all other models in recalling unseen entities. Additionally, while the hybrid models remain comparable to the Transformer++ model, Mamba's performance declines. This presents a striking contrast to the sentence-level experiments, suggesting that transformers may have an advantage in NE recall when shifting to longer contexts. Nonetheless, the transition from the 6M dataset to the CAT-5 dataset

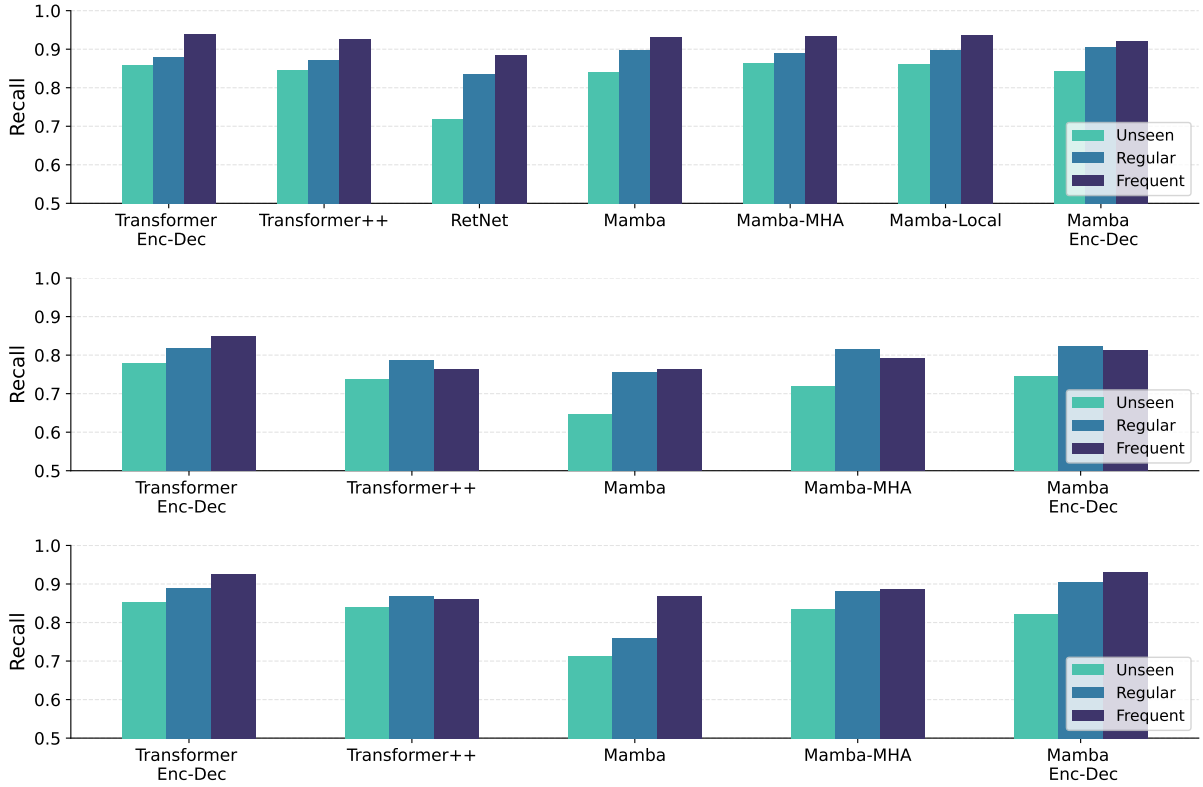

Figure 4: Recall in recovering named entities on the WMT14 (top), WMT23-6M (middle) and WMT23-CAT-5 (bottom) DE→EN datasets, by their training set frequency: *unseen* entities do not appear in the training data, while *regular* and *frequent* entities appear $[1, 16)$ and $16+$ times.

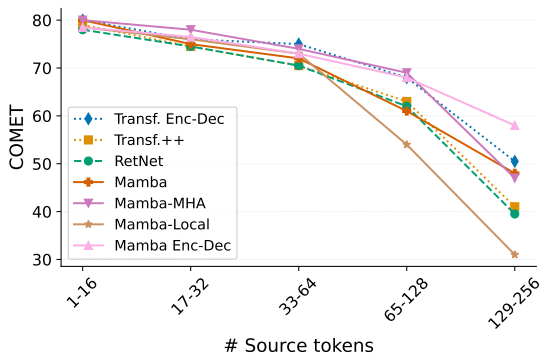

Figure 5: COMET scores per sequence length on WMT14 DE→EN for trained-from-scratch models.

leads to recall improvements across all models, particularly for unseen entities. This indicates that the additional context provided during training in the CAT-5 dataset aids the recall of named entities.

D Exploring Length-related Issues

D.1 Preliminary Sentence-level Experiments

Before experimenting with paragraph-level data, we analyze how our trained-from-scratch models perform on different sequence lengths. To this end, we study their sensitivity to input length when trained and tested on WMT14 DE→EN. The results are shown in Figure 5. While all models show a deterioration in performance as sequence length increases, this effect is more pronounced for Transformer++, RetNet, and Mamba-Local, with a significant drop in performance for samples longer than 64 tokens.

D.2 Sensitivity to Input Length

Following the discussion in §5.2, we further investigate the sensitivity of our models to input length using the WMT23 EN→DE test set, with results shown in Figure 6. Notably, our takeaways remain broadly the same: concatenating samples in the training data is indeed helpful when handling longer sequences, and models trained on the WMT23-CAT-10 dataset are much better in the longer bin (257+) with minimal translation quality degradation in shorter samples. However, when considering each of the training datasets' histograms in Figure 7, we can observe that models have been exposed to the longest samples during training, even if in low quantities. This implies that the previous experiments do not represent an extrap-

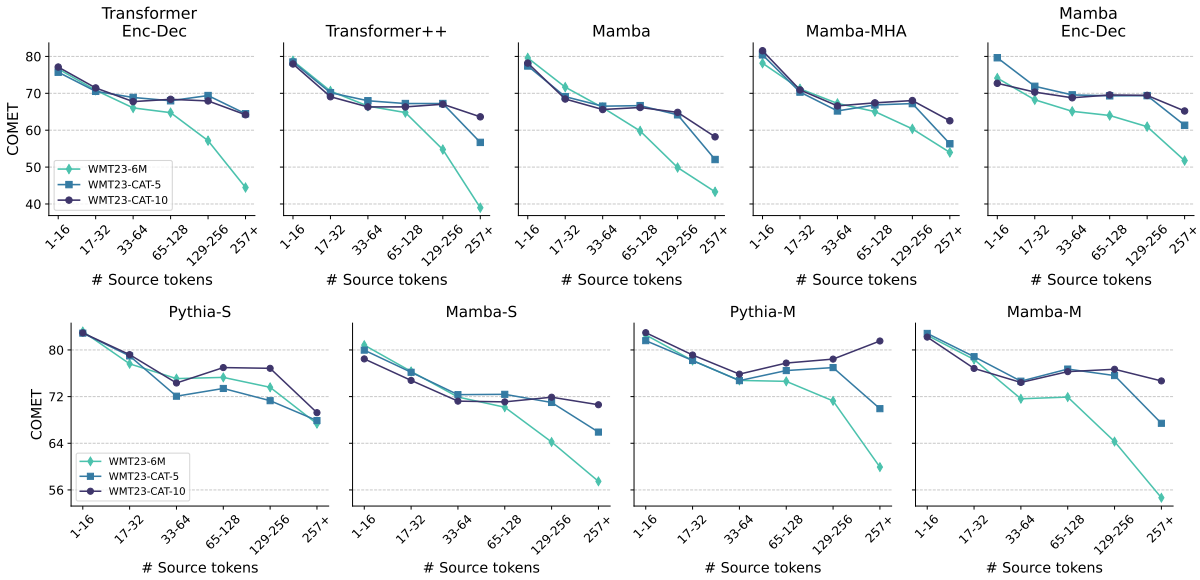

Figure 6: Sensitivity to input length, measured by the number of sources tokens, on the WMT23 EN→DE datset, for models trained from scratch (top) and finetuned from a pretrained checkpoint (bottom).

olation setting, where inference is done on longer sequence lengths than those seen during training. We cover extrapolation to longer sequences next.

E Full Paragraph-Level Results

For completeness, we report paragraph-level results in terms of BLEU and COMET for all language pairs and models in Table 9.

F AI assistants

We have used Github Copilot[13] during code development, and ChatGPT[14] during paper writing for paraphrasing or polishing original contents.

[13] https://github.com/features/copilot
[14] https://chat.openai.com/

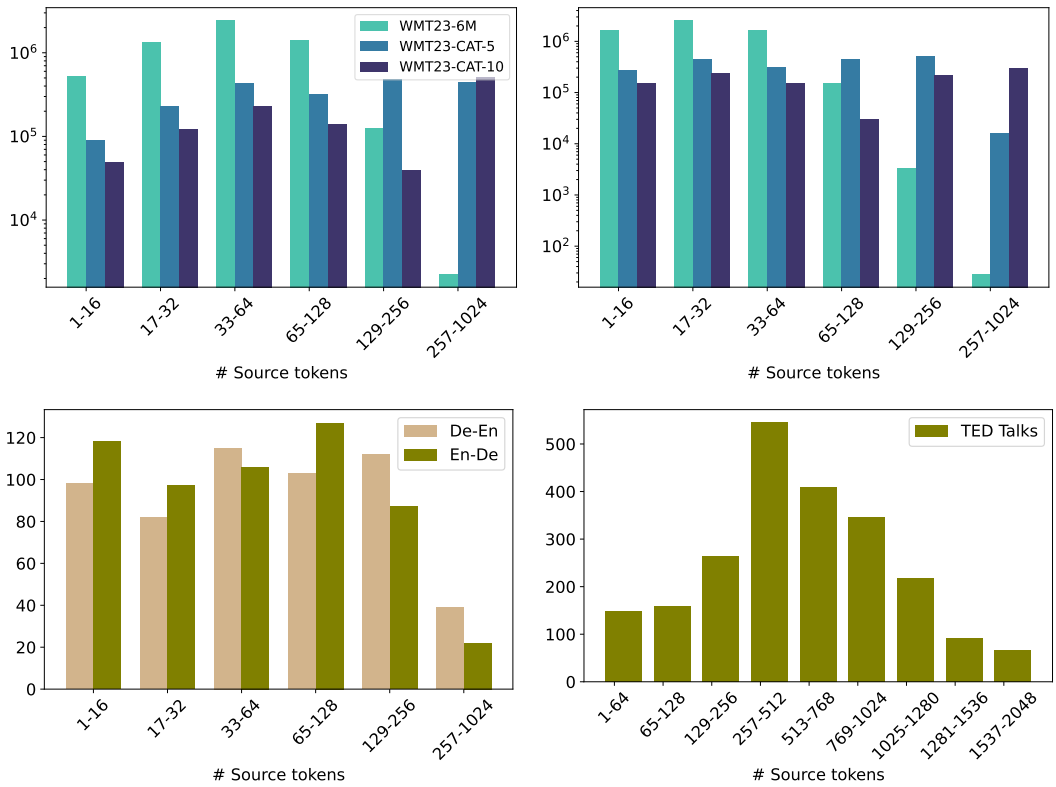

Figure 7: Distribution of source length in 1) the training datasets: WMT23 DE→EN (top left), WMT23 EN→DE (top right), and 2) the test datasets: WMT23 DE↔EN (bottom left), our custom TED Talks DE→EN (bottom right).

MODEL	TRAINING DATA	DE→EN		EN→DE	
		BLEU	COMET	BLEU	COMET
Trained from scratch					
Transformer Enc-Dec	WMT23-6M	25.4	72.4	22.4	65.2
Transformer++		21.6	70.7	20.2	64.8
Mamba		19.0	70.0	15.8	63.3
Mamba-MHA		**23.9**	**72.7**	**23.2**	**67.0**
Mamba Enc-Dec		22.7	70.7	21.5	65.3
Transformer Enc-Dec	WMT23-CAT-5	**30.8**	**74.6**	**29.9**	70.3
Transformer++		28.9	73.6	28.1	69.1
Mamba		26.1	73.3	23.8	67.5
Mamba-MHA		29.5	74.2	23.5	68.6
Mamba Enc-Dec		27.3	73.8	29.1	**71.0**
Transformer Enc-Dec	WMT23-CAT-10	28.3	69.6	29.3	**70.3**
Transformer++		29.8	72.8	29.1	68.8
Mamba		25.9	72.3	25.5	67.8
Mamba-MHA		27.8	74.5	25.9	69.7
Mamba Enc-Dec		**31.4**	**75.6**	**30.1**	70.1
Finetuned					
Mamba-S	WMT23-6M	21.8	77.2	21.4	72.4
Pythia-S		23.9	**77.4**	**25.9**	**76.7**
Mamba-M		20.7	74.6	22.5	73.4
Pythia-M		**26.0**	76.2	25.2	75.8
Mamba-S	WMT23-CAT-5	24.3	78.2	23.3	74.2
Pythia-S		**27.0**	78.4	**28.6**	**77.8**
Mamba-M		26.4	**79.6**	27.5	77.5
Pythia-M		25.8	78.6	27.5	77.4
Mamba-S	WMT23-CAT-10	25.6	78.3	22.5	73.1
Pythia-S		26.8	79.0	29.3	77.1
Mamba-M		32.5	**79.5**	27.5	77.3
Pythia-M		**33.4**	79.4	**33.9**	**79.0**

Table 9: Paragraph-level results in terms of BLEU and COMET on the WMT23 EN↔DE test set.