



Exploring Linear Variant Transformers and k -NN Memory Inference for Long-Form ASR

Carlos Carvalho¹, Jinchuan Tian², William Chen², Yifan Peng², Alberto Abad¹, Shinji Watanabe²

¹INESC-ID & Instituto Superior Tecnico, Portugal

²Carnegie Mellon Portugal, USA

carlos.mf.carvalho@inesc-id.pt

Abstract

While transformer-based models excel in short-form (SF) automatic speech recognition, the quadratic complexity of the self-attention mechanism introduces significant challenges in long-form (LF) audio. To address this, we compare strong linear transformer variants—Fastformer, SummaryMixing, BiMamba, and E-Branchformer with Flash Attention (EBranch-FA). For the latter, we also explore rotary positional encodings. Additionally, we propose a new challenging LF benchmark derived from the LibriHeavy corpus, featuring development and test sets with varying average durations to enable comprehensive evaluation across different temporal scales. Furthermore, we propose a memory system, KNN-MAN, for inference, which can be applied to any existing encoder-decoder models, without additional training. For example, with BiMamba, we reduce the word error rate from 18.8% to 17.5% in our LF Test Clean set derived from LibriSpeech.

Index Terms: ASR, long-form, memory-augmented

1. Introduction

Automatic speech recognition (ASR) has greatly benefited from transformer-based architectures, achieving state-of-the-art results on short-form (SF) utterances [1–5]. However, scaling these systems to long-form (LF) audio—such as lectures, meetings, and podcasts—remains challenging [6–8]. A key limitation is the SF bias inherent in most ASR systems, which are primarily trained on utterances up to 30 seconds [9, 10]. Additionally, the quadratic complexity of self-attention in transformer-based architectures [11] hinders scalability. Given these challenges, developing robust LF ASR systems that maintain strong performance on SF audio is essential.

Existing approaches include segmentation-based methods [12], simulated LF training or decoding (e.g., window-based strategies [13]), and architecture-based solutions [14–16], with the latter enabling true end-to-end ASR without changing SF pipelines. Among architectural innovations, several linear transformer variants have been proposed in the literature [17–19]. Also, architectures such as Fastformer [20] and SummaryMixing [16] and neural state space models (SSMs) like Mamba [15] have shown promise for LF ASR [21, 22]. However, these latter systems have primarily been evaluated on audio durations of up

to one minute [16, 22]. Furthermore, it is valuable to compare these architectural improvements with transformer variants like E-Branchformer [5], with FlashAttention [23]-based efficient implementation (yielding linear complexity, EBranch-FA), and rotary positional embeddings (RoPE) [24].

This work compares Fastformer, SummaryMixing, Mamba (1 and 2) [25], and EBranch-FA to assess their performance on LF audio. For EBranch-FA, we also explore the impact of RoPE-based positional embeddings, by introducing NTK-by-Parts (NbP) [26], an improved version of RoPE. To achieve this comparison, we create a LF partition of the LibriSpeech (LS) dataset [27] named LibriLong (LL), inspired from [8, 28], by concatenating all continuous segments (end time of segment S_n matches start time of segment S_{n+1}). This allows for fine-tuning and evaluation of the models on LF data. Leveraging EBranch-FA, one of the strongest models identified in this work, we further fine-tune them on both SF and LF data. This training approach further enhances LF performance while maintaining competitive results on SF data.

While LL is a good starting point to evaluate LF ASR systems, the maximum average duration for development and test set partitions is still short (around 50s). For this reason, we introduce LongLibriHeavy (LLH), a benchmark derived from the LibriHeavy (LH) corpus [29], comprising 36k training hours which is also created from the concatenation of continuous segments, similar to LL. The development and test sets feature varying durations (16s, 30s, 1min, 3min, and 7min), enabling comprehensive evaluation across different sequence lengths. Also, the main advantage of LLH over existing benchmarks [6–8] is its compatibility with LS-based models and its provision of 36k hours for training with both SF and LF. We evaluate LLH using our strongest model fine-tuned on both SF and LF data, as mentioned above, alongside foundation models like Whisper [9] and OWSM-CTC [30].

In addition to the benchmark, we propose an architectural modification to enhance the performance of encoder-decoder-based ASR systems for LF speech, inspired by memory-augmented networks (MANs) [31]. While MANs are typically explored in the text field [32–35], their application to LF ASR remains limited [28]. Specifically, we propose a k -nearest neighbor (k -NN) memory-augmented system (KNN-MAN), integrated between the encoder and decoder of any ASR system. This memory module operates exclusively during inference and is designed to especially improve zero-shot performance on LF audio by effectively leveraging contextual acoustic information. We demonstrate that our KNN-MAN provides additional gains on any of the encoder-decoder models mentioned above.

Our contributions are as follows: (1) we compare linear transformer variants—Mamba, SummaryMixing, Fastformer, and EBranch-FA—conducting zero-shot and fine-tuning eval-

Work supported by Portuguese national funds through Fundação para a Ciência e a Tecnologia (FCT), with references UIDB/50021/2020 and 2022/12328/BD, as well as by the Portuguese Recovery and Resilience Plan (RRP) through project C644865762-00000008 (Accelerat.AI) - Experiments of this work also used the Bridges2 system at PSC and Delta system at NCSA through allocations CIS210014 and IRI120008P from the Advanced Cyberinfrastructure Coordination Ecosystem: Services & Support (ACCESS) program, supported by National Science Foundation grants #2138259, #2138286, #2138307, #2137603, and #2138296.

uations on LF data, including RoPE and the novel NbP [26] in EBranch-FA; (2) we introduce LLH, a new LF benchmark¹; and (3) we propose a novel KNN-MAN that enhances LF performance for ASR, particularly in zero-shot scenarios.

2. Related Work

2.1. Linear Variants of Self-Attention

Fastformer [20] is a linear variant of self-attention that approximates the attention computation. Let $\mathbf{Q}, \mathbf{K} \in \mathbb{R}^{T \times D}$ represent the standard transformations in the transformer, where $\mathbf{Q} = [\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_T]$, $\mathbf{K} = [\mathbf{k}_1, \mathbf{k}_2, \dots, \mathbf{k}_T]$, T is the sequence length and D is the dimensionality of the vectors. The query \mathbf{Q} is summarized into a single query vector \mathbf{q} as follows:

$$a_t = \text{softmax} \left(\frac{\mathbf{w}_q^\top \mathbf{q}_t}{\sqrt{D}} \right), \quad \mathbf{q} = \sum_{t=1}^T a_t \mathbf{q}_t, \quad (1)$$

where $\mathbf{w}_q \in \mathbb{R}^D$ is a learnable vector used to compute the attention scores a_t . Finally, \mathbf{q} is multiplied element-wise with each vector in \mathbf{K} , producing a global context-aware key matrix.

SummaryMixing [16] is a linear alternative to self-attention that summarizes input information into a global vector which is combined with local features to produce context-aware representations. Formally,

$$\bar{\mathbf{s}} = \frac{1}{T} \sum_{t=1}^T u(\mathbf{x}_t), \quad \mathbf{Z} = c(f(\mathbf{X}), \bar{\mathbf{s}}), \quad (2)$$

where $\bar{\mathbf{s}}$ is the summary vector, computed as the average of input vectors \mathbf{x}_t transformed by the summary function u . Additionally, f is a local transformation function, c is a combiner function and \mathbf{Z} is the final context-aware representation.

Mamba 2 [25] refines the SSM framework of Mamba 1 by introducing a scalar-structured selective SSM (SSD). This framework is described by:

$$\mathbf{h}_t = \bar{\mathbf{A}}_t \mathbf{h}_{t-1} + \bar{\mathbf{B}}_t \mathbf{x}_t, \quad \mathbf{y}_t = \mathbf{C}_t \mathbf{h}_t, \quad (3)$$

where \mathbf{h}_t is the hidden state at time t , and \mathbf{y}_t is the output. The matrices $\bar{\mathbf{A}}_t$ and $\bar{\mathbf{B}}_t$ are discrete matrices that serve as input-dependent parameters, allowing Mamba 2 to dynamically adapt to sequences while retaining efficiency. Additionally, \mathbf{C}_t is also an input-dependent parameter. A hardware-aware parallel scan algorithm unrolls Equation (3) by convolving the input sequence \mathbf{X} with a fixed structured kernel of $\bar{\mathbf{A}}, \bar{\mathbf{B}}$, and \mathbf{C} , enabling efficient processing of temporal dependencies. To incorporate global context, a bidirectional Mamba (BiMamba 2) architecture is employed. Notably, Mamba 2 supports significantly larger state dimensions than Mamba 1, enabling richer temporal representations.

EBranch-FA is a linear variant of E-Branchformer [5] that leverages FlashAttention [23] to optimize attention computation by utilizing different parts of GPU memory. Specifically, FlashAttention improves self-attention complexity by computing softmax incrementally over input blocks and reducing memory overhead by recomputing the attention matrix during the backward pass instead of storing it.

2.2. RoPE-Based Embeddings

RoPE encodes positional information by rotating the query and key vectors, $\mathbf{q} \in \mathbb{R}^D$ and $\mathbf{k} \in \mathbb{R}^D$, through a frequency-based transformation. This rotation enables the derivation of relative positional information during the self-attention dot product [24]. The transformation is generalized as:

$$J_{\mathbf{W}}(\mathbf{x}_m, m, \theta_d) = J_{\mathbf{W}}(\mathbf{x}_m, g(m), l(\theta_d)), \quad (4)$$

where m is the token position in the sequence, and θ_d is the frequency parameter for the d -th hidden dimension. For RoPE $g(m) = m$ and $l(\theta_d) = \theta_d$, while other variants may define $g(m)$ and $l(\theta_d)$ differently.

We define λ_d as the wavelength of the RoPE embedding at the d -th hidden dimension:

$$\lambda_d = \frac{2\pi}{\theta_d} = 2\pi b \frac{2d}{D}, \quad (5)$$

where b is a base scaling factor. The wavelength λ_d represents the number of tokens required for a full rotation (2π). While RoPE effectively encodes absolute positional information, its uniform treatment of dimensions can lead to inefficiencies for longer sequences.

To address this, NbP introduces selective interpolation [26], categorizing dimensions based on the ratio $r_d = \frac{T}{\lambda_d}$ and adjusting interpolation as follows:

$$\gamma(r_d) = \begin{cases} 0 & \text{if } r_d < \alpha, \\ 1 & \text{if } r_d > \beta, \\ \frac{r_d - \alpha}{\beta - \alpha} & \text{otherwise.} \end{cases} \quad (6)$$

This approach selectively interpolates mid-range dimensions while avoiding extrapolation for others, enhancing the adaptability of RoPE. The NbP interpolation modifies RoPE from Equation (4) using:

$$g(m) = m, \quad l(\theta_d) = (1 - \gamma(r_d)) \frac{\theta_d}{\phi} + \gamma(r_d) \theta_d, \quad (7)$$

α and β are tuned parameters and ϕ is a scaling parameter.

3. KNN-MAN for LF Inference

We enhance LF performance during test time by incorporating KNN-MAN between the encoder and decoder of the ASR system. More specifically, let the encoder hidden states be represented as $\{\mathbf{h}_1, \mathbf{h}_2, \dots, \mathbf{h}_T\}$. The external memory $\mathbf{M}_t \in \mathbb{R}^{E \times D}$ at time step t stores the past E hidden states, dynamically updated during inference. For each hidden state \mathbf{h}_t , we perform a k -NN search within \mathbf{M}_{t-1} to retrieve the k most relevant past representations. The similarity between \mathbf{h}_t and each memory vector $\mathbf{m} \in \mathbf{M}_t$ is computed using cosine similarity. Let $\{\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_k\}$ be the top- k matched vectors. These vectors are aggregated via an unweighted average and then combined with the current hidden state \mathbf{h}_t through a simple weighted addition:

$$\bar{\mathbf{m}}_t = \frac{1}{k} \sum_{i=1}^k \mathbf{m}_i, \quad \tilde{\mathbf{h}}_t = \mathbf{h}_t + \rho \bar{\mathbf{m}}_t, \quad (8)$$

where ρ is the weight given to the retrieved vector $\bar{\mathbf{m}}_t$.

After the update, $\tilde{\mathbf{h}}_t$ is written into memory \mathbf{M}_t , following the first in, first out policy. This process continues iteratively until all encoder outputs are processed. More importantly, this approach enables the model to dynamically incorporate acoustic long-range dependencies *without fine-tuning*.

¹<https://github.com/Miamoto/LongLibriHeavy>

Table 1: Mean, minimum, and maximum durations (in seconds) for each data split in LibriSpeech (LS), LibriLong (LL), and LibriLongHeavy (LLH). Additionally, the Total Duration (TD) in hours is provided for each set.

Data	Mean (s)	Min. (s)	Max. (s)	TD (h)
LS 360	12.6	1.1	29.7	363.6
LS 960	12.3	0.8	29.7	961.1
LS Test Clean	7.4	1.3	35.0	5.4
LS Test Other	6.5	1.3	34.5	5.3
LL 360	52.0	30.0	480.6	362.3
LL Test Clean	52.8	30.1	272.4	8.2
LL Test Other	43.1	30.1	124.8	2.5
LLH-Small	49.8	2.1	535.2	276.5
LLH-Medium	48.6	2.0	736.40	3005.7
LLH-Large	48.9	2.0	873.4	32623.4
LLH-Dev	369.2	180.5	567.7	10.1
LLH-Test	393.3	180.7	592.6	10.1

4. LibriLong and LibriLongHeavy

Our experiments rely on the LibriSpeech [27] and LibriHeavy [29] corpora. To create the LF datasets—LibriLong (LL) and the new LibriLongHeavy (LLH) benchmark—we concatenate continuous segments from each corpus. For LibriLong, we use the speaker information from each LibriSpeech set to access the LibriVox MP3 audio² and retrieve the respective “sentence-aware” segmentation files, which split audio at silence intervals aligned with sentence boundaries [27]. All continuous segments are joined sequentially (end time of segment S_n matches start time of segment S_{n+1}), and new sequences are created when no such alignment exists. After the concatenation process, all utterances shorter than 30 seconds are excluded, forming LL 360 (from LS 960), LL Test Clean, and LL Test Other subsets, depicted in Table 1. Overall, this strategy reduces total hours from original sets by excluding short segments (below 30s). However, it can increase audio duration, as seen in LL Test Clean compared to LS Test Clean, due to the inclusion of all original segments in the concatenation process from the respective speakers, which are not fully represented in LS Test Clean [27].

LLH (Table 1) is a new benchmark for both SF and LF, distinct from the original LibriHeavy created through LibriLight [36]. Its concatenation process mirrors LibriLong: we retrieve segmentation files from LibriHeavy, order them, and concatenate continuous segments, removing overlapping segments (where S_{n+1} ends before or at S_n). Development and test sets are built by selecting the longest and most challenging segments from the concatenated small, medium, and large sets, ensuring no speaker or book overlaps. The most challenging utterances (highest WER) are detected using the OWSM-CTC model [30]. Additionally, a mapping file was created for each respective set. Each line in this file records the new utterance ID alongside the ordered original utterance IDs used for concatenation. This enables flexible data generation with varying average durations (e.g., 16s, 30s, 1min, and 7min) for comprehensive ASR evaluation. It is important to note that LLH-{Small, Medium, Large} training sets, while not used in this work, are designed to support future research.

²<https://www.openslr.org/12>

5. Experiments

5.1. Experimental Setup

ESPnet2 [37] is the primary toolkit used to implement and evaluate our work. More specifically, we follow the LibriSpeech 100 recipe in ESPnet for data preparation, training, decoding, and evaluation. All systems are based on encoder-decoder architectures, where SummaryMixing, BiMamba, Fastformer or E-Branchformer refer to the encoder, and a standard transformer decoder is used consistently across all systems. For SummaryMixing, we adapt the implementation from SpeechBrain [38] and for BiMamba, we use the code from [39].

For LS 960 training, the EBranch-FA follows the ESPnet LS 960 architecture but reduces the layers to 10, resulting in approximately 100M parameters. All other systems are designed to match this parameter count while retaining the same decoder architecture. BiMamba 2 has 28 blocks and is configured with a state space size of 4096, while Fastformer and SummaryMixing use 11 layers. Furthermore, for LS, all systems are trained on a single NVIDIA RTX A6000 with 48GB memory. For the LL, BiMamba, SummaryMixing and Fastformer fit on the same hardware, while the EBranch-FA requires a single NVIDIA A100 with 80GB memory. Furthermore, in preliminary studies, when training with RoPE and NbP, we found that the model did not converged using the default learning rate. To address this, we adopted the piecewise-linear learning rate schedule from [30] to LS 960 training, increasing the learning rate to $2.0e-04$ in the first 15K steps and then to $2.0e-03$ over the next 30K steps. This stabilization method was not required during fine-tuning on LL data, as we were already utilizing pre-trained LS models. It is important to note that RoPE and NbP are applied to both the encoder and decoder of the E-Branchformer ASR system. For both LF and LF+SF training, we fine-tune the models for just 30 epochs. Following [26], we set $\alpha = 1$ and $\beta = 32$ (Equation (7)).

For inference, we use the official code³ when using Whisper and apply the English Whisper normalizer to both OWSM v4 Base and OWSM-CTC v4 [40]. Finally, for the proposed KNN-MAN, we explore hyperparameters such as memory size E (e.g., 100 and 1000), the number of top- k matched vectors (e.g., 4 and 8) mentioned in Section 3, and the weight ρ (e.g., 0.1) from Equation (8). Notice that the inference time of KNN-MAN can increase as the memory size E grows very large.

5.2. Results

Linear Self-Attention Variants. From Table 2 we can observe that EBranch-FA (NbP) and Fastformer perform the strongest on SF data. Also, NbP proves effective compared to vanilla RoPE. For zero-shot LF, both RoPE and NbP show slight degradations for EBranch-FA. Furthermore, BiMamba2 may be the least effective option in SF, but it serves as a viable alternative for LF. Notably, SummaryMixing stands out as the most robust zero-shot option for LF. Fine-tuning on LL 360 significantly improves LF performance, breaking the previous trend. For instance, EBranch-FA with NbP improves from 30.2% to 2.5% word error rate (WER) on LL Test Clean, establishing it as the most robust model for LF audio. However, this comes at the cost of increased WER on SF data. Given that EBranch-FA (NbP) followed by SummaryMixing maintain the strongest performance when fine-tuned on LL 360, we select these models for fine-tuning on both SF and LF data. To ensure consistency,

³<https://github.com/openai/whisper>

Table 2: WERs [%] for Librispeech and LibriLong test sets across different training and positional encoding configurations. Models have around $\sim 100M$, except for Whisper and OWSM v4 Small, which have around 244M parameters.

Model	Positional Encoding	Training	LS Test Clean	LS Test Other	LL Test Clean	LL Test Other
EBranch-FA	Abs.	LS 960	2.4	5.5	29.2	23.8
EBranch-FA	RoPE	LS 960	2.4	5.5	31.7	27.1
EBranch-FA	NbP	LS 960	2.4	5.4	30.2	24.9
Fastformer	Abs.	LS 960	2.3	5.6	25.2	16.6
BiMamba 2	-	LS 960	3.2	7.5	18.8	13.8
SummaryMixing	Abs.	LS 960	2.5	5.7	17.4	12.0
EBranch-FA	Abs.	LS 960 \rightarrow LL 360	3.2	9.0	2.7	5.7
EBranch-FA	NbP	LS 960 \rightarrow LL 360	3.1	8.3	2.5	5.2
BiMamba 2	-	LS 960 \rightarrow LL 360	3.9	9.8	3.4	7.6
SummaryMixing	Abs.	LS 960 \rightarrow LL 360	7.4	13.8	2.7	6.0
EBranch-FA	NbP	LS 960 \rightarrow {LF+SF}	2.7	7.2	2.3	4.6
SummaryMixing	Abs.	LS 960 \rightarrow {LF+SF}	2.8	7.3	2.5	5.6
Whisper Small	Abs.	680K hours	3.3	7.7	2.9	5.5
OWSM v4 Small	Abs.	320K hours	2.5	5.9	3.0	5.0

Table 3: WERs [%] without and with the KNN-MAN.

Model	LL Test Clean	LL Test Other
EBranch-FA (NbP)	29.2	23.8
+KNN-MAN	28.5	23.7
Fastformer	25.2	16.6
+KNN-MAN	25.0	16.5
BiMamba 2	18.8	13.8
+KNN-MAN	17.5	13.7
SummaryMixing	17.4	12.0
+KNN-MAN	16.8	11.5
Whisper Small	2.9	5.5
+KNN-MAN	2.9	5.4
SummaryMixing (LF+SF)	2.5	5.6
+KNN-MAN	2.5	5.5
EBranch-FA (LF+SF)	2.3	4.6
+KNN-MAN	2.3	4.6

we use LS 360 (Table 1) for SF data, matching the number of hours in LL 360. This approach, not only further enhances LF performance, but also maintains comparable SF results to models trained exclusively on SF data.

Finally, we compare our EBranch-FA (NbP) model, fine-tuned on both SF and LF data, with Whisper Small and OWSM v4 Small, that process audio in 30-second windows, and show that it outperforms both foundation models for LF data. Notably, OWSM v4 Small is trained on LibriSpeech, while Whisper Small is not.

Overall, we conclude that evaluating LF performance solely based on models trained on SF data or fine-tuned on LF data can lead to inconsistent interpretations. However, when both SF and LF data are used in a balanced manner, this inconsistency is mitigated.

KNN-MAN. We implement our KNN-MAN approach on the linear transformer variant models listed in Table 2, which were trained on LS 960 and fine-tuned on LS+SF, as well as on Whisper Small. For all models, we use a memory size E of 1000, $k=8$ and $\rho=0.1$. As depicted in Table 3, our approach either enhances or maintains performance, without requiring additional training and can be easily integrated into any encoder-decoder ASR system.

LLH Benchmark. For the LLH benchmark, we evaluate

LLH-Test set with EBranch-FA, trained on both LF and SF, from Table 2, and with strong foundation models. Additionally, we present results for variants of the same set categorized by different average durations (Section 4). From Figure 1, we observe that for foundation models, the WER remains constant across lengths, as these models use a sliding-window with an input of 30 seconds. However, compared to Whisper Small, our EBranch-FA model shows performance improvements, except for 7 minutes. Also, we show that our KNN-MAN improves performance on Whisper Small for LF audio.

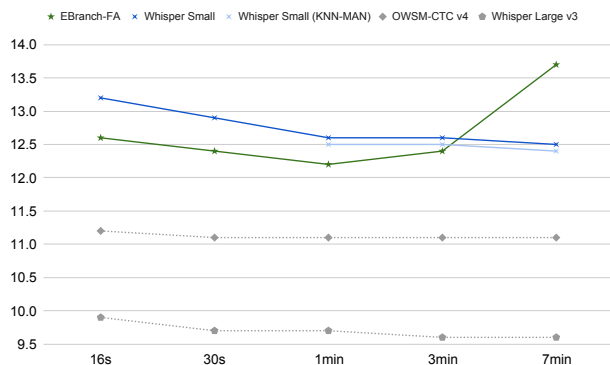


Figure 1: WERs [%] for models evaluated on LLH-Test set with varying average durations.

6. Conclusion

This work evaluates linear transformer variants like BiMamba, SummaryMixing, Fastformer and EBranch-FA for LF ASR and observe that EBranch-FA is one of the strongest models, especially when fine-tuned on both SF and LF data. We also introduce a new RoPE-based positional embedding for EBranch-FA, NbP, which improves when compared to vanilla RoPE. EBranch-FA, fine-tuned on both SF and LF data is also evaluated on the new LF benchmark, LLH, and compared with strong foundation models. Additionally, we propose KNN-MAN, which can be inserted between the encoder and decoder of any ASR system, further improving LF performance at inference time, especially for zero-shot. For future work, we plan to explore efficient training strategies for the KNN-MAN system.

7. References

- [1] L. Dong, S. Xu, and B. Xu, "Speech-transformer: A no-recurrence sequence-to-sequence model for speech recognition," in *Proc. ICASSP*, 2018, pp. 5884–5888.
- [2] S. Karita, N. Chen, T. Hayashi, T. Hori, H. Inaguma, Z. Jiang, M. Someki, N. E. Y. Soplin, R. Yamamoto, X. Wang, S. Watanabe, T. Yoshimura, and W. Zhang, "A comparative study on transformer vs rnn in speech applications," in *Proc. ASRU*, 2019.
- [3] A. Gulati, J. Qin, C. Chiu, N. Parmar, Y. Zhang, J. Yu, W. Han, S. Wang, Z. Zhang, Y. Wu, and R. Pang, "Conformer: Convolution-augmented transformer for speech recognition," in *Proc. INTERSPEECH*, 2020.
- [4] Y. Peng, S. Dalmia, I. Lane, and S. Watanabe, "Branchformer: Parallel MLP-attention architectures to capture local and global context for speech recognition and understanding," in *Proc. ICML*, 2022.
- [5] K. Kim, F. Wu, Y. Peng, J. Pan, P. Sridhar, K. J. Han, and S. Watanabe, "E-Branchformer: Branchformer with enhanced merging for speech recognition," in *Proc. SLT*, 2023.
- [6] C.-C. Chiu, W. Han, Y. Zhang, R. Pang, S. Kishchenko, P. Nguyen, A. Narayanan, H. Liao, S. Zhang, A. Kannan, R. Prabhavalkar, Z. Chen, T. Sainath, and Y. Wu, "A comparison of end-to-end models for long-form speech recognition," in *Proc. ASRU*, 2019.
- [7] A. Narayanan, R. Prabhavalkar, C.-C. Chiu, D. Rybach, T. N. Sainath, and T. Strohmaier, "Recognizing long-form speech using streaming end-to-end models," in *Proc. ASRU*, 2019, pp. 920–927.
- [8] J. D. Fox, D. Raj, N. Delworth, Q. McNamara, C. Miller, and M. Jetté, "Updated corpora and benchmarks for long-form speech recognition," in *Proc. ICASSP*, 2024.
- [9] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever, "Robust speech recognition via large-scale weak supervision," in *Proc. ICML*, 2023.
- [10] Y. Peng, J. Tian, B. Yan, D. Berrebbi, X. Chang, X. Li, J. Shi, S. Arora, W. Chen, R. Sharma, W. Zhang, Y. Sudo, M. Shakeel, J.-W. Jung, S. Maiti, and S. Watanabe, "Reproducing whisper-style training using an open-source toolkit and publicly available data," in *Proc. ASRU*, 2023.
- [11] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," in *Proc. NIPS*, 2017.
- [12] W. R. Huang, S. yiin Chang, D. Rybach, R. Prabhavalkar, T. N. Sainath, C. Allauzen, C. Peyser, and Z. Lu, "E2e segmenter: Joint segmenting and decoding for long-form asr," in *Proc. INTERSPEECH*, 2022.
- [13] T. Hori, N. Moritz, C. Hori, and J. L. Roux, "Advanced long-context end-to-end speech recognition using context-expanded transformers," in *Proc. INTERSPEECH*, 2021.
- [14] W. Chen, T. Kano, A. Ogawa, M. Delcroix, and S. Watanabe, "Train long and test long: leveraging full document contexts in speech processing," in *Proc. ICASSP*, 2024.
- [15] A. Gu and T. Dao, "Mamba: Linear-time sequence modeling with selective state spaces," *arXiv preprint arXiv:2312.00752*, 2023.
- [16] T. Parcollet, R. van Dalen, S. Zhang, and S. Bhattacharya, "SummaryMixing: A linear-complexity alternative to self-attention for speech recognition and understanding," in *Proc. INTERSPEECH*, 2024.
- [17] J. Lee-Thorp, J. Ainslie, I. Eckstein, and S. Ontanon, "FNet: Mixing tokens with Fourier transforms," in *Proc. ACL*, 2022, pp. 4296–4313.
- [18] N. Kitaev, L. Kaiser, and A. Levskaya, "Reformer: The efficient transformer," in *Proc. ICLR*, 2020.
- [19] R. Sharma, S. Palaskar, A. W. Black, and F. Metzger, "End-to-end speech summarization using restricted self-attention," in *Proc. ICASSP*, 2022, pp. 8072–8076.
- [20] C. Wu, F. Wu, T. Qi, Y. Huang, and X. Xie, "Fastformer: Additive attention can be all you need," *arXiv preprint arXiv:2108.09084*, 2021.
- [21] X. Zhang, Q. Zhang, H. Liu, T. Xiao, X. Qian, B. Ahmed, E. Am-bikairajah, H. Li, and J. Epps, "Mamba in speech: Towards an alternative to self-attention," *arXiv preprint arXiv:2405.12609*, 2024.
- [22] K. Miyazaki, Y. Masuyama, and M. Murata, "Exploring the capability of mamba in speech applications," in *Proc. INTERSPEECH*, 2024.
- [23] T. Dao, D. Y. Fu, S. Ermon, A. Rudra, and C. Ré, "FlashAttention: Fast and memory-efficient exact attention with IO-awareness," in *Proc. NEURIPS*, 2022.
- [24] J. Su, Y. Lu, S. Pan, B. Wen, and Y. Liu, "Roformer: Enhanced transformer with rotary position embedding," *arXiv preprint arXiv:2104.09864*, 2021.
- [25] T. Dao and A. Gu, "Transformers are SSMs: Generalized models and efficient algorithms through structured state space duality," in *Proc. ICML*, 2024.
- [26] B. Peng, J. Quesnelle, H. Fan, and E. Shippole, "YaRN: Efficient context window extension of large language models," in *Proc. ICLR*, 2024.
- [27] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: An asr corpus based on public domain audio books," in *Proc. ICASSP*, 2015.
- [28] C. Carvalho and A. Abad, "Memory-augmented conformer for improved end-to-end long-form asr," in *Proc. INTERSPEECH*, 2023.
- [29] W. Kang, X. Yang, Z. Yao, F. Kuang, Y. Yang, L. Guo, L. Lin, and D. Povey, "Libriheavy: a 50,000 hours asr corpus with punctuation casing and context," in *Proc. ICASSP*, 2024.
- [30] Y. Peng, Y. Sudo, M. Shakeel, and S. Watanabe, "OWSM-CTC: An open encoder-only speech foundation model for speech recognition, translation, and language identification," in *Proc. ACL*, 2024.
- [31] A. Graves, G. Wayne, and I. Danihelka, "Neural Turing machines," *arXiv preprint arXiv:1410.5401*, 2014.
- [32] Y. Wu, M. N. Rabe, D. Hutchins, and C. Szegedy, "Memorizing transformers," in *Proc. ICLR*, 2022.
- [33] W. Wang, L. Dong, H. Cheng, X. Liu, X. Yan, J. Gao, and F. Wei, "Augmenting language models with long-term memory," in *Proc. NEURIPS*, 2023.
- [34] A. Bertsch, U. Alon, G. Neubig, and M. R. Gormley, "Unlim-iformer: Long-range transformers with unlimited length input," *arXiv preprint arXiv:2305.01625*, 2023.
- [35] A. Behrouz, P. Zhong, and V. Mirrokni, "Titans: Learning to memorize at test time," *arXiv preprint arXiv:2501.00663*, 2024.
- [36] J. Kahn, M. Rivière, W. Zheng, E. Kharitonov, Q. Xu, P. Mazaré, J. Karadayi, V. Liptchinsky, R. Collobert, C. Fuegen, T. Likhomanenko, G. Synnaeve, A. Joulin, A. Mohamed, and E. Dupoux, "Libri-light: A benchmark for asr with limited or no supervision," in *Proc. ICASSP*, 2020.
- [37] S. Watanabe, T. Hori, S. Karita, T. Hayashi, J. Nishitoba, Y. Unno, N. Soplin, J. Heymann, M. Wiesner, N. Chen *et al.*, "Espnet: End-to-end speech processing toolkit," in *Proc. INTERSPEECH*, 2018.
- [38] M. Ravanelli, T. Parcollet, P. Plantinga, A. Rouhe, S. Cornell, L. Lugosch, C. Subakan, N. Dawalatabad, A. Heba, J. Zhong, J.-C. Chou, S.-L. Yeh, S.-W. Fu, C.-F. Liao, E. Rastorgueva, F. Grondin, W. Aris, H. Na, Y. Gao, R. D. Mori, and Y. Bengio, "SpeechBrain: A general-purpose speech toolkit," *arXiv preprint arXiv:2106.04624*, 2021.
- [39] Y. Masuyama, K. Miyazaki, and M. Murata, "Mamba-based decoder-only approach with bidirectional speech modeling for speech recognition," in *Proc. SLT*, 2024.
- [40] Y. Peng, S. Muhammad, Y. Sudo, W. Chen, J. Tian, C.-J. Lin, and S. Watanabe, "Owsm v4: Improving open whisper-style speech models via data scaling and cleaning," 2025.