# Supplementary Material

**Aleksandar Terzić**[1,2]*, **Michael Hersche**[1,2], **Geethan Karunaratne**[1]
**Luca Benini**[2], **Abu Sebastian**[1], **Abbas Rahimi**[1]†
[1]IBM Research – Zurich, [2]ETH Zurich

## A Related work

Long sequence modeling is a rich area of research within the deep learning community. A non-exhaustive list of work relevant to ours is outlined below.

### Linear recurrent models

A particularly prominent family of linear recurrent models comes in the form of linear state-space models (LSSMs). In their original formulation, linear state-space models are based on the HiPPO framework [1], in which an optimal mechanism for incrementally updating a fixed-size state while processing online streams of information by projecting onto an orthogonal polynomial basis is derived. Based on the HiPPO framework, the seminal work S4 [2] introduces a new addition into the neural sequence processing family of operators, linear state-space models. S4 is, to the best of our knowledge, the first method to significantly advance the state-of-the-art on LRA classification [3]. Many other linear state-space models follow; S4D [4] diagonalizes the linear recurrence, GSS [5] introduces a gating mechanism for improving LSSMs' performance on language modeling, S5 [6] introduces MIMO LSSMs.

We had mentioned that linear state-space models, in their typical formulation, are not suitable for language modeling. H3 [7], motivated by contemporary work on mechanistic interpretability [8], proposes a multiplicative interconnection of linear state-space models which they find significantly improves their performance on this task.

One of the newer additions to the family of linear recurrent models is the *LRU* model [9], which steps away from the state-space model framework which is based on discretizing an implicit continuous state-space model while still achieving near-state-of-the-art accuracies on LRA.

### Long convolutional models

We denote models which apply convolutions whose kernels are of the same length as the input sequence as *long convolutional models*. SGConv [10] constructs a sparsely parametrized long kernel with an exponentially decaying structure and finds that this achieves strong performance on LRA. Hyena hierarchy [11] achieves impressive scores on language modeling with a long convolutional kernel, without attention. It is to the best of our knowledge the highest performing attention-free language model. FlashButterfly [12] explores simple long convolution kernel constructions that are effective on LRA classification and furthermore develops a hardware-aware algorithm for efficient computation of FFT-based long convolutions.

### Linear complexity attention

A rich area of research addresses the quadratic scaling of the attention operation by deriving more scalable approximations thereof. A survey of such approaches can be found in [13].

### TCNs for sequence modeling

*Research conducted at IBM Research – Zurich.

†Corresponding author: abr@zurich.ibm.com

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

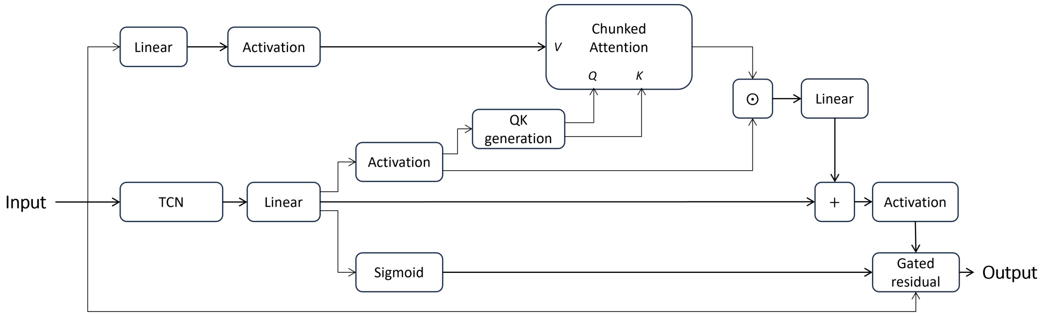

Figure A1: The module interconnection used in TCNCA is inherited from MEGA and is significantly more complex than the simplified sketch we have shown in the main text. It involves several multiplicative and additive interactions between outputs of different operators. QK generation applies a diagonal linear transformation on the input data and adds a trainable bias to it, with separate parameters for queries and keys. The gated residual is equivalent to the construction defined in Highway Networks [20].

CDIL-CNN employs circular dilated convolutions on several datasets including Long-Range-Arena [14]. WaveNet [15] is a generative audio model based on the TCN architecture. ByteNet [16] employs the TCN for machine translation.

**TCN-Attention hybrid models**

TCAN [17] employs a cascade of dilated convolutions with full self-attention. Their architecture scales quadratically with the sequence length, has one convolution per decoder stack, and the causal attention masking they implement, described in Section 2.3, is in fact not causal. TConvTransformer [18] describes a similar idea, a quadratic complexity concatenation of a TCN and multi-head self-attention, and is evaluated on 3D human pose and shape estimation.

## B Limitations

On generative tasks, an important example being language modeling, linear recurrent models offer the unique advantage of parallel training and recurrent inference. A temporal convolutional neural network is a naturally parallel operation, but it is not trivial to adapt it to efficiently operate on generative tasks. In a generative mode of operation, we would need to cache intermediate computations generated by the TCN and re-use them at later time steps. We have not implemented this mode of operation in our work, opting instead to demonstrate the feasibility and applicability of the idea. Implementing this mode of operation is left for future work.

Another advantage of recurrent models lies in the fact that their receptive field size is theoretically unlimited. The TCN's receptive field size is restricted through the construction that we impose by defining the kernel size, network depth, and dilation.

The TCN introduces three new hyperparameters, namely kernel size, dilation factor, and network depth. This significantly expands the hyperparameter search space.

There exists at least one alternative option for parallelizing linear recurrent systems, the *parallel scans* algorithm [19]. We have not compared the runtimes of the TCN with this algorithm.

As hinted at in the method description section in the main text, the chunked attention module is in fact significantly more intricately interconnected with the TCN than Figure 1 makes it seem like. The full construction is shown in Appendix C. Future work should investigate the necessity of each part of the module with the goal of simplifying the construction and maximizing its performance.

## C Detailed overview of the attention module

The attention module is more intricately interconnected with EMA/TCN than Figure 1 in the main text would make it seem. Details are given in Figure A1.

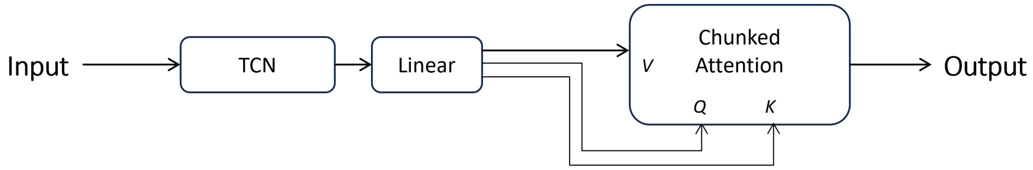

Figure A2: TCNCA-simple is a significantly simpler version of the full TCNCA model, and is used in the associative recall benchmark.

For the *associative recall* benchmark, we used a simpler construction, called TCNCA-simple, which is outlined in FigureA2

## D Hyperparameter selection

For LRA and EnWik8, the TCN structure, which is defined by the kernel size, dilation factor, and TCN depth, was selected by measuring inference run-times of a range of configurations, selecting a range of those which exhibit a speed-up, and finally training networks with those structures on the given dataset. It is a rather costly procedure, since there exist many different TCN structures which exhibit a speed-up, and they differ between benchmarks. A more principled way of selecting TCN configurations should be investigated in future work.

For experiments on the long-range-arena dataset, the same learning rate scheduler, learning rate, weight decay, dropout values, activation functions, and vector dimensionalities as those used in MEGA [21] were used.

For experiments on the EnWik8 dataset, the same learning rate scheduler, dropout values, activation functions and vector dimensionalities as those used in MEGA [21] were used. A learning rate of 0.005 and a weight decay of 0.2 were used to obtain the best score.

The only varied hyperparameters in the associative recall experiments are dropout and the learning rate. Dropout is selected from the small grid defined by two points, [0, 0.1]. Learning rate was swept over [1e-5, 1e-4, 1e-3, 1e-2, 1e-1].

## E TCN construction

In addition to the TCN hyperparameters introduced in the main text, these being kernel size $K$, dilation factor $f$, and depth $D$, we will in this section introduce a new hyperparameter, $\mathbf{B}$, which stands for the number of dilated convolutions within *a single residual block*. This is explained in Figure A3.

The receptive field size of the TCN can be calculated as $1+B*(K-1)*\frac{f^D-1}{f-1}$. It scales exponentially with $D$ and linearly with $B$. Keeping the total number of dilated convolution operations $D \times B$ equal, the largest receptive field size is achieved when $B$ is minimized and $D$ is maximized, which is the reason we opted for $B = 1$ in our experiments.

## F EnWik8 train and test details

The data is split into consecutive chunks of size 2048, which is also what the attention chunk size is set to. At training time, we randomly load 2, 3, or 4 consecutive chunks of text to be processed by the model. During evaluation, the attention chunk size is set to 4096, and 3 consecutive chunks of text are loaded. We train for 250 epochs. Just as in MEGA [21], attention is augmented with relative positional bias, the particular one used in this task being RoPE [22].

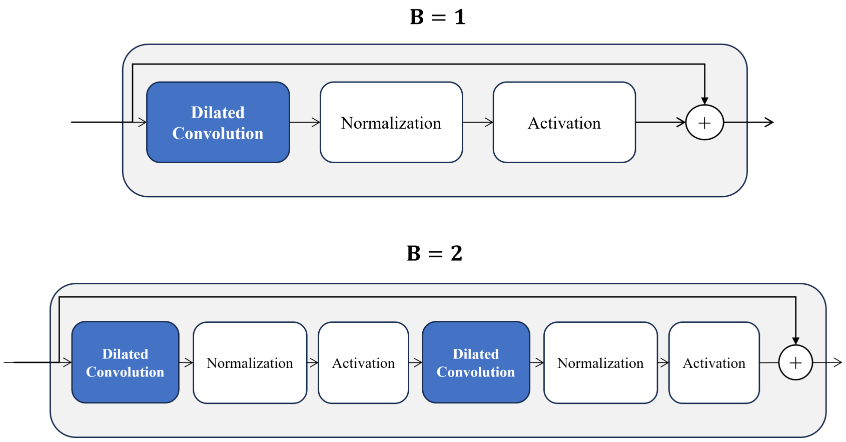

Figure A3: The hyperparameter $B$ controls the number of dilated convolutions with equal dilation within a single residual block. The full network still consists of $D$-many such blocks. On the top, we show a residual block with $B = 1$, which is what we used in our experiments. $B = 2$ is another popular option in published works on TCNs.

## G Associative recall setup

We must note that, while $N$ from Figure 1 (a) is 2, $D$ from Figure 1 (b) must be larger than 1 in order to obtain a large enough receptive field of the TCN. Hence, our model does consist of a depth-2 sequence decoding stack as was used in [7], but each of the decoder layers uses a deeper TCN, typically of depth 3 or 4.

For sequence length 64, we use embedding dimension 32 and an attention chunk size of 32, corresponding to the standard quadratic self-attention. For all other sequence lengths, an embedding dimension of 128 with a chunk size of 128 is used.

## The need for self-attention

In this section, we demonstrate the performance difference between attention-less models and those which hybridize EMA/TCN with attention on EnWik8 (Tables A1 and A2) and LRA (Table A3). On LRA, we are in fact able to achieve strong performance using a TCN-based attention-free model, outperforming the best TCN-based attention-free model known in the literature, CDIL-CNN [14].

Table A1: The role of attention in MEGA on EnWik8 after full training. The hyperparameters used in MEGA were re-used in the EMA-MLP experiment set. The result marked with a star (*) is taken from [21].

| | Enwik8 loss ↓ |
|---|---|
| EMA-MLP stack | 1.96 |
| MEGA | 1.02* |

Table A2: Introducing chunked attention after the TCN significantly reduces the BPC loss on enwik8. Both models went through a hyperparameter grid search but were trained for a limited number of epochs, hence the gap between the result reported here and in the main text.

| | Enwik loss after 250k training steps |
|---|---|
| TCN-MLP stack | 1.44 |
| TCNCA | 1.08 |

Table A3: Comparison of S5 [6], CDIL-CNN [14], and our implementation of a dilated convolutional neural network on several tasks from the long-range-arena dataset. Our implementation outperforms the best-known TCN-based result from literature, the CDIL-CNN work [14], on all four LRA benchmarks which we both evaluate on.

| | ListOps | Text | Retrieval | Image | Path | Path-X |
|---|---|---|---|---|---|---|
| S5 [6] | 62.1% | 89.3% | 91.4% | 88.0% | 95.3% | 98.6% |
| CDIL-CNN [14] | — | 87.6% | 84.3% | 64.5% | 91.0% | — |
| Our TCN-MLP | 56.9% | 89.6% | 85.9% | 91.4% | 97.2% | 91.6% |

## H Runtime benchmark methodology

The EMA hidden dimension (dimension expansion from [21], Section 3.1) is set to 8. Within the multi-dimensional damped EMA algorithm presented in MEGA [21], this results in 64 parameters. The TCN is always of depth $D = 4$ with 17 parameters in each layer, hence it consists of 68 parameters, slightly more than the amount present in EMA. The dilation factor is increased until the TCN's receptive field size is greater than or equal to the sequence length.

All operations were implemented in the PyTorch 2.0.1 framework. The run-time measurements were obtained using the PyTorch benchmark module [3]. LRA and EnWik8 run-times were measured on a 32 GB Nvidia V100 GPU. All other run-times were measured on a 16 GB Nvidia V100 GPU.

## I Effect of EMA kernel generation on inference runtime

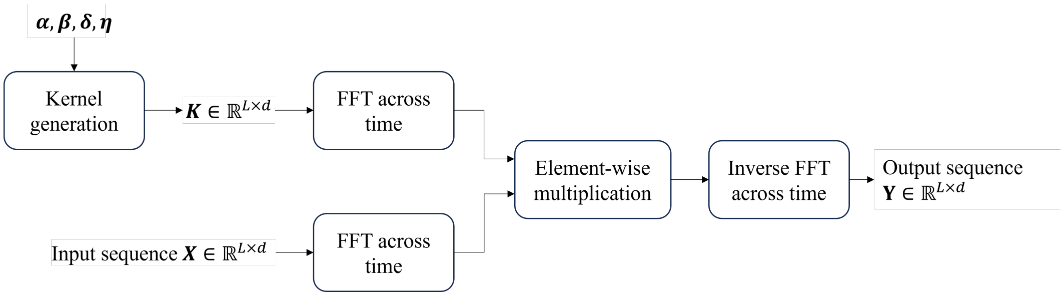
$\boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\delta}, \boldsymbol{\eta}$
Kernel generation
$\boldsymbol{K} \in \mathbb{R}^{L \times d}$
FFT across time
Element-wise multiplication
Inverse FFT across time
Output sequence $\mathbf{Y} \in \mathbb{R}^{L \times d}$
Input sequence $\boldsymbol{X} \in \mathbb{R}^{L \times d}$
FFT across time

Figure A4: Computing the EMA linear recurrence in parallel invokes the computational pipeline shown in this figure. The kernel computation is given in MEGA [21], Appendix A. There exists at least one alternative way of computing the long convolution, using the *parallel scans* algorithms [19], which we did not consider in this work.

The runtime measurements presented in the main text include the kernel generation cost. This is certainly necessary during training, but at inference, one might consider storing very long kernels and truncating them based on the input sequence length. This would reduce the inference runtimes of FFT-based EMA convolution. Runtime comparisons of FFT-based EMA convolutions with and without kernel generation are shown in Figure A5. Speed-ups of the version without kernel generation vs. the version that includes kernel generation are given in Table A4.

[3] https://pytorch.org/tutorials/recipes/recipes/benchmark.html

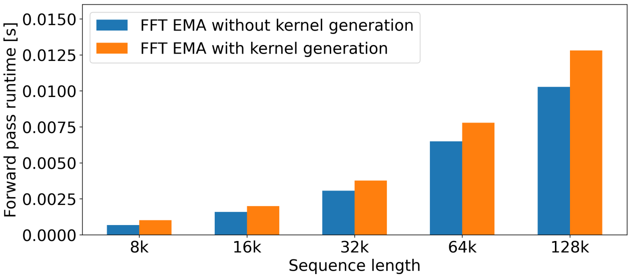

Figure A5: Comparing the forward-pass runtimes between the FFT-based parallel EMA with and without kernel generation (see Figure A4).

Table A4: Speedup of FFT-EMA without vs. with kernel generation.

| | 8k | 16k | 32k | 64k | 128k |
|---|---|---|---|---|---|
| Speedup | $1.51\times$ | $1.25\times$ | $1.23\times$ | $1.20\times$ | $1.25\times$ |

## References

[1] Albert Gu, Tri Dao, Stefano Ermon, Atri Rudra, and Christopher Ré. Hippo: Recurrent memory with optimal polynomial projections. *Advances in Neural Information Processing Systems (NeurIPS)*, 33:1474–1487, 2020.

[2] Albert Gu, Karan Goel, and Christopher Ré. Efficiently modeling long sequences with structured state spaces. In *International Conference on Learning Representations (ICLR)*, 2022.

[3] Yi Tay, Mostafa Dehghani, Samira Abnar, Yikang Shen, Dara Bahri, Philip Pham, Jinfeng Rao, Liu Yang, Sebastian Ruder, and Donald Metzler. Long range arena : A benchmark for efficient transformers. In *International Conference on Learning Representations (ICLR)*, 2021.

[4] Albert Gu, Karan Goel, Ankit Gupta, and Christopher Ré. On the parameterization and initialization of diagonal state space models. *Advances in Neural Information Processing Systems (NeurIPS)*, 35:35971–35983, 2022.

[5] Harsh Mehta, Ankit Gupta, Ashok Cutkosky, and Behnam Neyshabur. Long range language modeling via gated state spaces. *arXiv preprint arXiv:2206.13947*, 2022.

[6] Jimmy T.H. Smith, Andrew Warrington, and Scott Linderman. Simplified state space layers for sequence modeling. In *International Conference on Learning Representations (ICLR)*, 2023.

[7] Tri Dao, Daniel Y. Fu, Khaled K. Saab, Armin W. Thomas, Atri Rudra, and Christopher Ré. Hungry Hungry Hippos: Towards language modeling with state space models. In *International Conference on Learning Representations (ICLR)*, 2023.

[8] Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

[9] Antonio Orvieto, Samuel L Smith, Albert Gu, Anushan Fernando, Caglar Gulcehre, Razvan Pascanu, and Soham De. Resurrecting recurrent neural networks for long sequences. In *International Conference on Machine Learning (ICML)*, 2023.

[10] Yuhong Li, Tianle Cai, Yi Zhang, Deming Chen, and Debadeepta Dey. What makes convolutional models great on long sequence modeling? In *International Conference on Learning Representations (ICLR)*, 2023.

[11] Michael Poli, Stefano Massaroli, Eric Q. Nguyen, Daniel Y. Fu, Tri Dao, Stephen A. Baccus, Yoshua Bengio, Stefano Ermon, and Christopher Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning (ICML)*, 2023.

[12] Daniel Y Fu, Elliot L Epstein, Eric Nguyen, Armin W Thomas, Michael Zhang, Tri Dao, Atri Rudra, and Christopher Re. Simple hardware-efficient long convolutions for sequence modeling. In *ICLR 2023 Workshop on Mathematical and Empirical Understanding of Foundation Models*, 2023.

[13] Yi Tay, Mostafa Dehghani, Dara Bahri, and Donald Metzler. Efficient transformers: A survey. *ACM Computing Surveys*, 55:1 – 28, 2020.

[14] Lei Cheng, Ruslan Khalitov, Tong Yu, Jing Zhang, and Zhirong Yang. Classification of long sequential data using circular dilated convolutional neural networks. *Neurocomputing*, 2022.

[15] Aaron van den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, and Koray Kavukcuoglu. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 2016.

[16] Nal Kalchbrenner, Lasse Espeholt, Karen Simonyan, Aaron van den Oord, Alex Graves, and Koray Kavukcuoglu. Neural machine translation in linear time. *arXiv preprint arXiv:1610.10099*, 2016.

[17] Hongyan Hao, Yan Wang, Yudi Xia, Jian Zhao, and Furao Shen. Temporal convolutional attention-based network for sequence modeling. *arXiv preprint arXiv:2002.12530*, 2020.

[18] Xianjin Chao, Zhipeng Ge, and Howard Leung. Video2mesh: 3d human pose and shape recovery by a temporal convolutional transformer network. *IET Computer Vision*, 17, 02 2023.

[19] Eric Martin and Chris Cundy. Parallelizing linear recurrent neural nets over sequence length. *International Conference on Learning Representations (ICLR)*, 2018.

[20] Rupesh K Srivastava, Klaus Greff, and Jürgen Schmidhuber. Training very deep networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 28, 2015.

[21] Xuezhe Ma, Chunting Zhou, Xiang Kong, Junxian He, Liangke Gui, Graham Neubig, Jonathan May, and Zettlemoyer Luke. Mega: Moving average equipped gated attention. In *International Conference on Learning Representations (ICLR)*, 2023.

[22] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *arXiv preprint arXiv:2104.09864*, 2021.