Reparameterized Multi-Resolution Convolutions for Long Sequence Modelling

Harry Jake Cunningham[1], Giorgio Giannone[2], Mingtian Zhang[1] and Marc Peter Deisenroth[1]

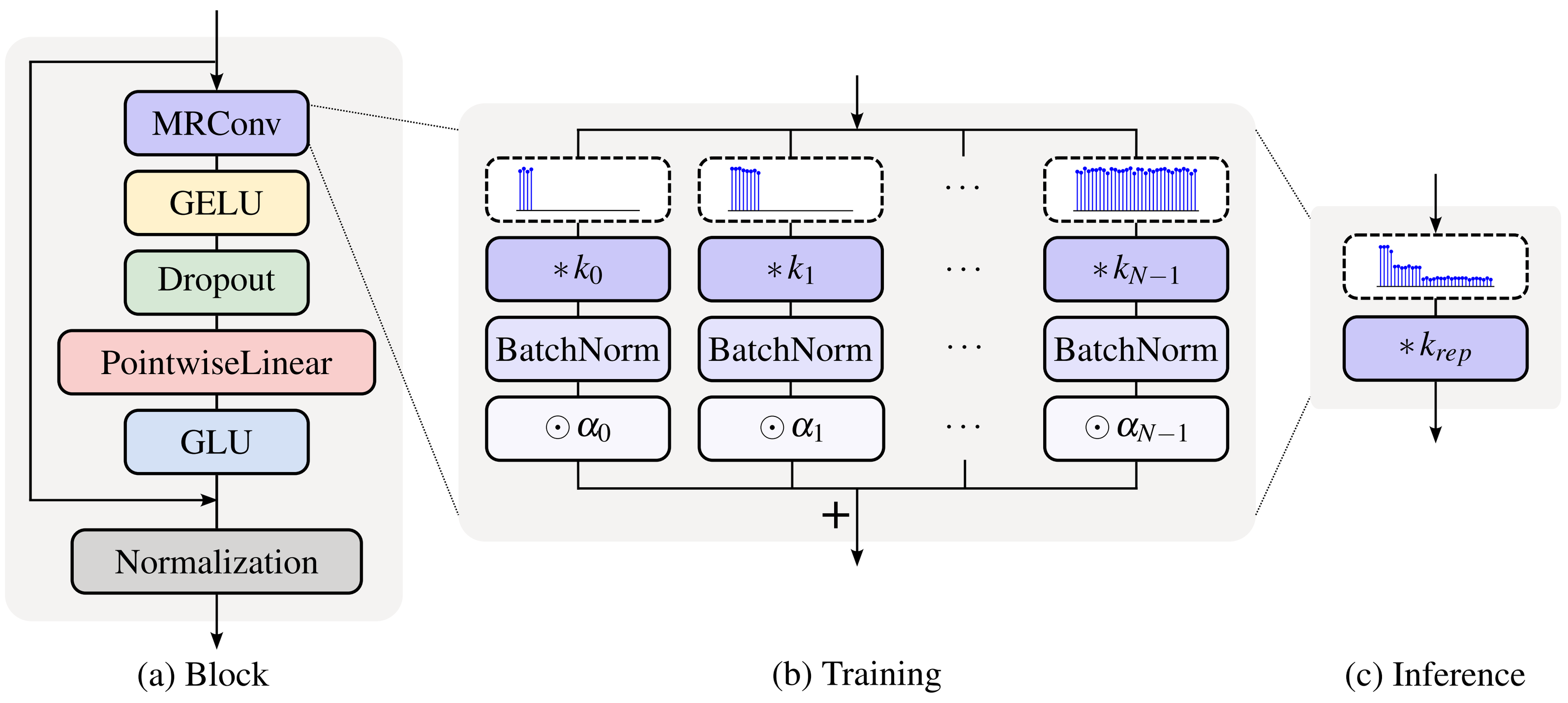
[1]University College London; [2] Amazon; * *Work completed whilst at UCL*

UCL

Introduction

- Global convolutions have shown increasing promise as efficient general-purpose sequence models.
- However, training long convolutions is hard, and kernel parameterizations must learn long-range dependencies without overfitting.
- This work introduces reparameterized multi-resolution convolutions (`MRConv`), which uses structural reparameterization to combine a set of low-rank sub-kernels of increasing length.

MRConv: Reparameterized Multi-Resolution Convolutions



Causal Structural Reparameterization

We can merge multiple causal convolutions into one as

$$y[t] = \sum_{n=0}^{N-1}(u * k_n)[t] = \left(u * \left(\sum_{n=0}^{N-1} k_n\right)\right)[t] = (u * k_{rep})[t], \quad (1)$$

where k_n is the convolution kernel of the nth branch

Causal Branch Addition with BatchNorm When merging kernels of different lengths, normalization becomes crucial due to the impact of kernel size on the output statistics,

$$k_{rep} = \overline{\text{BN}_0(k_0)} + \overline{\text{BN}_1(k_1)}, \quad (2)$$

Causal Branch Addition with Linear Rescaling When merging kernels of the same length, we use linear scaling allowing kernels to be reparameterized during training,

$$k_{rep} = \beta_0 \cdot k_0 + \beta_1 \cdot k_1. \quad (3)$$

Multi-Resolution Convolutions

At each resolution i, we define a kernel k_i of length $l_i = l_0 2^i$. We define the set of normalized multi-resolution convolutions $\tilde{\boldsymbol{c}} \in \mathbb{R}^{N\times D\times L}$ as,

$$\tilde{\boldsymbol{c}} = [\text{BN}_0(k_0 * u), \text{BN}_1(k_1 * u), \cdots, \text{BN}_{N-1}(k_{N-1} * u)]. \quad (4)$$

The output $y[t] \in \mathbb{R}^D$ at time step t is generated by computing a linear combination of the coefficients $\tilde{\boldsymbol{c}}[t]$ at time step t according to

$$y[t] = \boldsymbol{\alpha}^T \tilde{\boldsymbol{c}}[t], \quad (5)$$

where $\boldsymbol{\alpha} \in \mathbb{R}^{N\times D}$ is a learnable parameter. Applying $\boldsymbol{\alpha}$ across the sequence length we define the output $y \in \mathbb{R}^{D\times L}$ as the summation

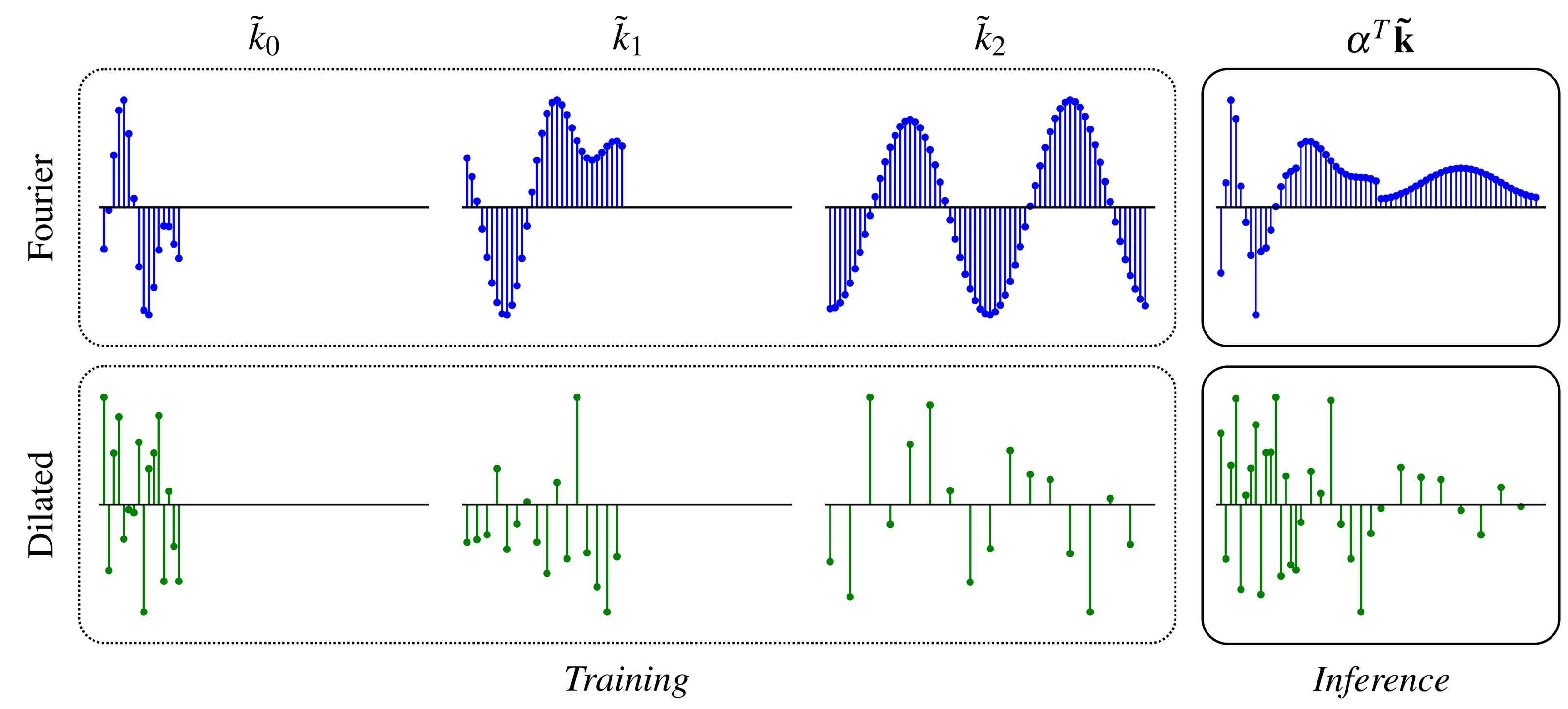
$$y = \alpha_0 \text{BN}_0(k_0 * u) + \alpha_1 \text{BN}_1(k_1 * u) + \cdots + \alpha_{N-1}\text{BN}_{N-1}(k_{N-1} * u). \quad (6)$$

Applying causal structural reparameterization at inference, we can rewrite the above process as a single convolution,

$$y = u * (\alpha_0 \overline{\text{BN}_0(k_0)} + \alpha_1 \overline{\text{BN}_1(k_1)} + \cdots + \alpha_{N-1}\overline{\text{BN}_0(k_{N-1})}) = u * k_{rep}, \quad (7)$$

eliminating the extra memory and computational cost of training with extra convolutions.

Low-Rank Kernel Parameterization



Dilated Kernels Variation on standard convolutional filters where p many zeros are padded between the elements of the kernel,

$$y[t] = (u * k_{dilated})[t] = \sum_{\tau=0}^{l-1} k[\tau]u[t - p\tau]. \quad (8)$$

Fourier Kernels Complex kernels $\hat{k} \in \mathbb{C}^{D\times L}$ parameterized in the Fourier domain as a small number m of low-frequency Fourier modes,

$$k_{fourier}[t] = \text{IFFT}[\text{ZeroPad}(\hat{k}, L - m)][t]. \quad (9)$$

Sparse Kernels We randomly sample kernel positions across the sequence length, where $\delta_t \in \mathcal{T}$ is the Kronecker delta as,

$$k_{sparse}[t] = \delta_{t\in\mathcal{T}} \cdot k_t, \quad (10)$$

Experiments

Long Range Arena. `MRConv` is competitive with other sub-quadratic complexity models, including SSMs and linear-time transformers.

Model (Input length)	`ListOps` (2,048)	`Text` (4,096)	`Retrieval` (4,000)	`Image` (1,024)	`Pathfinder` (1,024)	`Path-X` (16,384)	Avg.
Transformer	36.37	64.27	57.46	42.44	71.40	✗	53.66
Linear-Time Transformers:							
MEGA-Chunk	58.76	**90.19**	90.97	85.80	94.41	93.81	85.66
State Space Models:							
S4D-LegS	60.47	86.18	89.46	88.19	93.06	91.95	84.89
S4-LegS	59.60	86.82	90.90	88.65	94.20	96.35	86.09
Liquid-S4	**62.75**	89.02	91.20	<u>89.50</u>	94.8	96.66	87.32
S5	62.15	89.31	<u>91.40</u>	88.00	95.33	**98.58**	**87.46**
Convolutional Models:							
CCNN	43.60	84.08	-	88.90	91.51	✗	-
Long Conv	62.2	<u>89.6</u>	91.3	87.0	93.2	96.0	86.6
SGConv	61.45	89.20	91.11	87.97	<u>95.46</u>	<u>97.83</u>	87.17
MRConv	<u>62.40</u>	89.26	**91.44**	**90.37**	**95.55**	97.82	**87.81**

ImageNet Classification. Using optimized CUDA kernels for 1D FFT convolutions, we close the gap between theoretical and empirical throughput.