HadamRNN: Binary and Sparse Ternary Orthogonal RNNs

Armand Foucault[1] François Malgouyres[1] Franck Mamalet[2]

[1]Institut de Mathématiques de Toulouse ; UMR 5219
Université de Toulouse ; CNRS
UPS IMT F-31062 Toulouse Cedex 9, France

[2]Institut de Recherche Technologique Saint Exupéry, Toulouse, France

ICLR 2025

Introduction

ORNNs: RNNs with an orthogonal recurrent weight matrix:

$$h_t = \sigma\left(Wh_{t-1} + Ux_t + b_i\right), \quad \text{where } W \text{ is orthogonal}$$

Binarization / Ternarization: Constraining weights / activations to take 2 or 3 distinct values.

$\Longrightarrow$ **Goal:** Binarize / ternarize W

$\Longrightarrow$ **Problem:** Naïve binarization / ternarization moves away from orthogonality.

Notation:

$h_t \in \mathbb{R}^{d_h}$: *Hidden state* at time $t \in \mathbb{N}^*$

$x_t \in \mathbb{R}^{d_{in}}$: *Input data* at time $t \in \mathbb{N}^*$

$\sigma : \mathbb{R}^{d_h} \to \mathbb{R}^{d_h}$: *Activation function*

$W \in \mathbb{R}^{d_h \times d_h}$: *Recurrent weight matrix*

$U \in \mathbb{R}^{d_h \times d_{in}}$: *Input weight matrix*

$b_i \in \mathbb{R}^{d_h}$: *Bias*

Contributions

- *Lightweight, fully-quantized orthogonal recurrent neural networks with binary / ternary orthogonal recurrent weight matrix*
- *Long-term dependencies modeling:*

  $$\text{HadamRNN} \gg \text{LSTM / GRU (comparable size)}$$

- *Solves an array of benchmarks*
  - 1000-timesteps copy task
  - vanilla / permuted pixel-by-pixel MNIST
  - IMDB
  - two GLUE benchmarks
  - two IoT benchmarks

Hadamard and Block-Hadamard RNNs

- **Hadamard matrices:** Square matrices with values in $\{-1, 1\}$ with pairwise orthogonal rows

$$H \in \mathbb{R}^{d_h \times d_h},\ d_h \in \mathbb{N}^* \text{ Hadamard matrix} \implies HH' = d_h I_{d_h}$$

- **Hadamard RNNs (HadamRNN):** ORNN with linear activation (σ identity function), and $W \in \mathbb{R}^{d_h \times d_h}$ parametrized as

$$W(u) = \frac{1}{d_h} \operatorname{diag}(u)\, \mathbf{S}_{2^k}, \quad \text{orthogonal}$$

  - $I_{d_h} \in \mathbb{R}^{d_h \times d_h}$ identity matrix
  - $d_h = 2^k$
  - $u \in \{-1, 1\}^{d_h}$ trainable vector, $\operatorname{diag}(u) \in \mathbb{R}^{d_h \times d_h}$
  - $\mathbf{S}_{2^k}$ the Sylvester matrix of size 2^k (a Hadamard matrix)[1]

[1] KJ Horadam. Hadamard Matrices and Their Applications. Princeton University Press, 2007.

Hadamard and Block-Hadamard RNNs

- **Example:** $d_h = 4$, $u = (-1, 1, 1, -1)$

$$\implies W(u) = \frac{1}{4} \begin{pmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & -1 \end{pmatrix} \begin{pmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{pmatrix}$$

- **Training:**

$$W(u) = \frac{1}{d_h} \operatorname{diag}(u)\, \mathbf{S}_{2^k}, \tag{1}$$

$\mathbf{S}_{2^k}$ is fixed, u trained using the *Straight-Through Estimator* (STE)[2,3]

- **Block-Hadamard RNNs (Block-HadamRNN):** Extension of HadamRNN introducing sparsity in (1); set $d_h = q2^k$:

$$W(u) = \frac{1}{\sqrt{2^k}} \operatorname{diag}(u)\big(I_q \otimes \mathbf{S}_{2^k}\big) \in \mathbb{R}^{d_h \times d_h},$$

[2] G. Hinton. Neural networks for machine learning. Coursera, video lectures. Lecture 15b. 2012.

[3] Matthieu Courbariaux et al. "Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1". In: arXiv preprint arXiv:1602.02830 (2016).

Experimental results

- Results on **Copy task** and **Permuted MNIST**

Model	d_h	W bitwidth	U & V bitwidth	activation bitwidth	performance	size kBytes
Copy task ($T = 1000$, $d_{in} = 10$, $d_{out} = 9$, cross-ent, baseline = 0.021)						
LSTM (Jing et al. (2017))	80	FP	FP	FP	BL	112
ORNN (Kiani et al. (2022))	256	FP	FP	FP	1.1e-12	275
QORNN (Foucault et al. (2024))	"	8	8	12	1.7e-5	75.5
QORNN (Foucault et al. (2024))	"	5	5	12	2.5e-3	50.6
HadamRNN (ours)	128	1	2	FP	BL	1.44
			4	FP	1.6e-7	1.74
			4	12	2.3e-7	1.40
			6	FP	3.2e-8	2.33
Permuted MNIST ($T = 784$, $d_{in} = 1$, $d_{out} = 10$, accuracy)						
ORNN (Kiani et al. (2022))	512	FP	FP	FP	97.00	1046.0
ORNN (Kiani et al. (2022))	170	FP	FP	FP	94.30	120.2
LSTM (Kiani et al. (2022))	"	FP	FP	FP	92.00	456.9
QORNN (Foucault et al. (2024))	"	8	8	12	94.76	35.0
QORNN (Foucault et al. (2024))	"	6	6	12	93.94	27.9
HadamRNN (ours)	512	1	2	FP	91.13	3.48
			4	FP	94.88	4.85
			4	12	94.90	3.58
			6	FP	95.85	6.23

Conclusion and perspectives

- First Binary / Ternary learnable recurrent weights ORNN
- First fully quantized RNN able to solve 1000-timesteps **copy task**
- Performance competitive with full-precision RNNs on a large set of benchmarks
- Ability to fine-tune tradeoff performance / computational cost with our **sparse ternary extension**
- **Perspectives :**
  - Binarize / ternarize State Space Models (SSMs)[4]
  - Deploy HadamRNN / Block-HadamRNN on edge devices

[4] Albert Gu, Karan Goel, and Christopher Re. "Efficiently Modeling Long Sequences with Structured State Spaces". In: International Conference on Learning Representations. 2022.

Conclusion

Thank you !

- Courbariaux, Matthieu et al. “Binarized neural networks: Training deep neural networks with weights and activations constrained to +1 or -1”. In: arXiv preprint arXiv:1602.02830 (2016).
- Foucault, Armand, Franck Mamalet, and François Malgouyres. “HadamRNN: Binary and Sparse Ternary Orthogonal RNNs”. In: International Conference on Learning Representations. 2025.
- Gu, Albert, Karan Goel, and Christopher Re. “Efficiently Modeling Long Sequences with Structured State Spaces”. In: International Conference on Learning Representations. 2022.
- Hinton, G. Neural networks for machine learning. Coursera, video lectures. Lecture 15b. 2012.
- Horadam, KJ. Hadamard Matrices and Their Applications. Princeton University Press, 2007.