Generalized Neural Sorting Networks with Error-Free Differentiable Swap Functions

Jungtaek Kim[1] Jeongbeen Yoon[2] Minsu Cho[2]

[1]University of Pittsburgh [2]POSTECH

Introduction

- Traditional sorting algorithms are a well-established approach to arranging given instances in computer science.
- Sorting networks are structurally designed as an abstract device with a fixed number of wires.
- Sorting networks have been used to perform a sorting algorithm on computing hardware.

Sorting

- Standard sorting formulation:

Given an unordered sequence of n elements $\mathbf{s} = [s_1, \ldots, s_n] \in \mathbb{R}^n$, the problem of sorting is defined to find a permutation matrix $\mathbf{P} \in \{0,1\}^{n \times n}$:

$$\mathbf{s}_{\text{o}} = \mathbf{P}^\top \mathbf{s}, \tag{1}$$

where a sorting algorithm is a function f of $\mathbf{s}$:

$$\mathbf{P} = f(\mathbf{s}). \tag{2}$$

- More generalized sorting formulation:

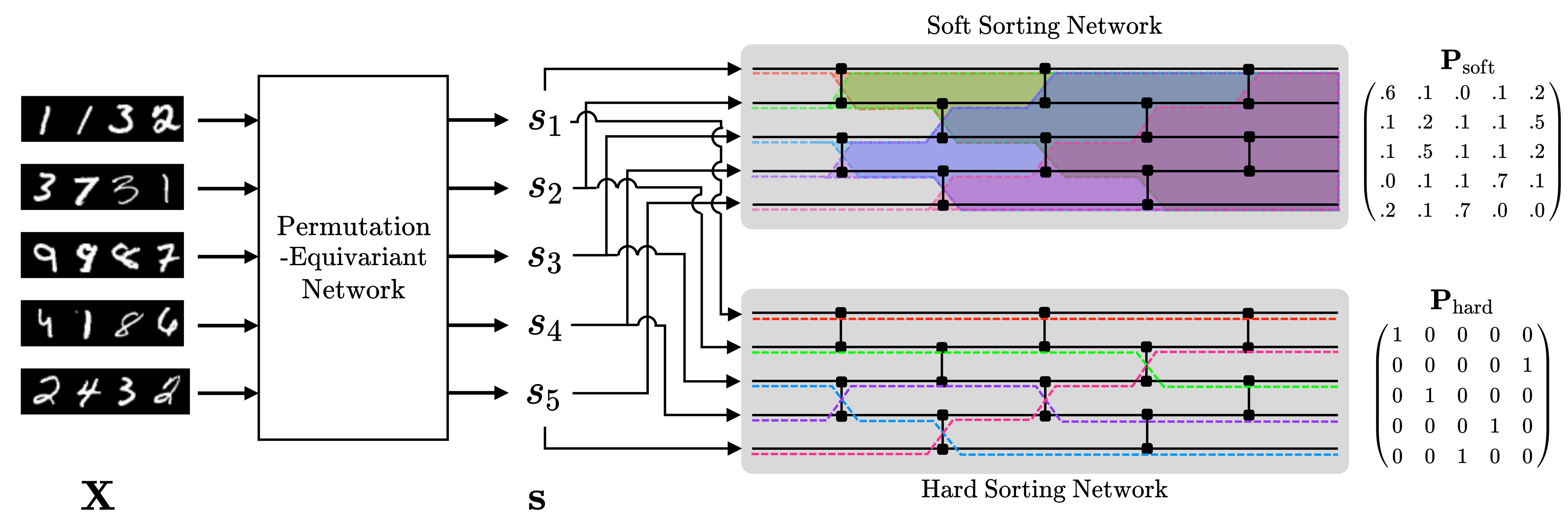
Instead of $\mathbf{s}$, it is to sort $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n]^\top \in \mathbb{R}^{n \times d}$:

$$\mathbf{X}_{\text{o}} = \mathbf{P}^\top \mathbf{X}. \tag{3}$$

- This generalized sorting problem can be reduced to (1) if we are given a proper mapping g from an input $\mathbf{x} \in \mathbb{R}^d$ to an ordinal value $s \in \mathbb{R}$.
- Without such a mapping g, predicting $\mathbf{P}$ in (3) remains more challenging than in (1) because $\mathbf{x}$ is often a highly implicative high-dimensional input.

Contributions

- Define a softening error, which is a difference between original and smoothed values.
- Propose an error-free DSF that resolves the error accumulation problem of conventional DSFs and is still differentiable.
- Adopt a permutation-equivariant network with multi-head attention as a mapping from inputs to ordinal variables $g(\mathbf{X})$, unlike $g(\mathbf{x})$.



Sorting Networks with Differentiable Swap Functions

- A swap function is a key ingredient of sorting algorithms and sorting networks:

$$(x', y') = \operatorname{swap}(x, y), \tag{4}$$

where $x' = \min(x, y)$ and $y' = \max(x, y)$.

- We can express $\min(\cdot,\cdot)$ and $\max(\cdot,\cdot)$:

$$\min(x, y) = x\lfloor\sigma(y - x)\rceil + y\lfloor\sigma(x - y)\rceil, \tag{5}$$

$$\max(x, y) = x\lfloor\sigma(x - y)\rceil + y\lfloor\sigma(y - x)\rceil, \tag{6}$$

where $\lfloor\cdot\rceil$ rounds to the nearest integer and $\sigma(\cdot)$ transforms an input to a bounded value.

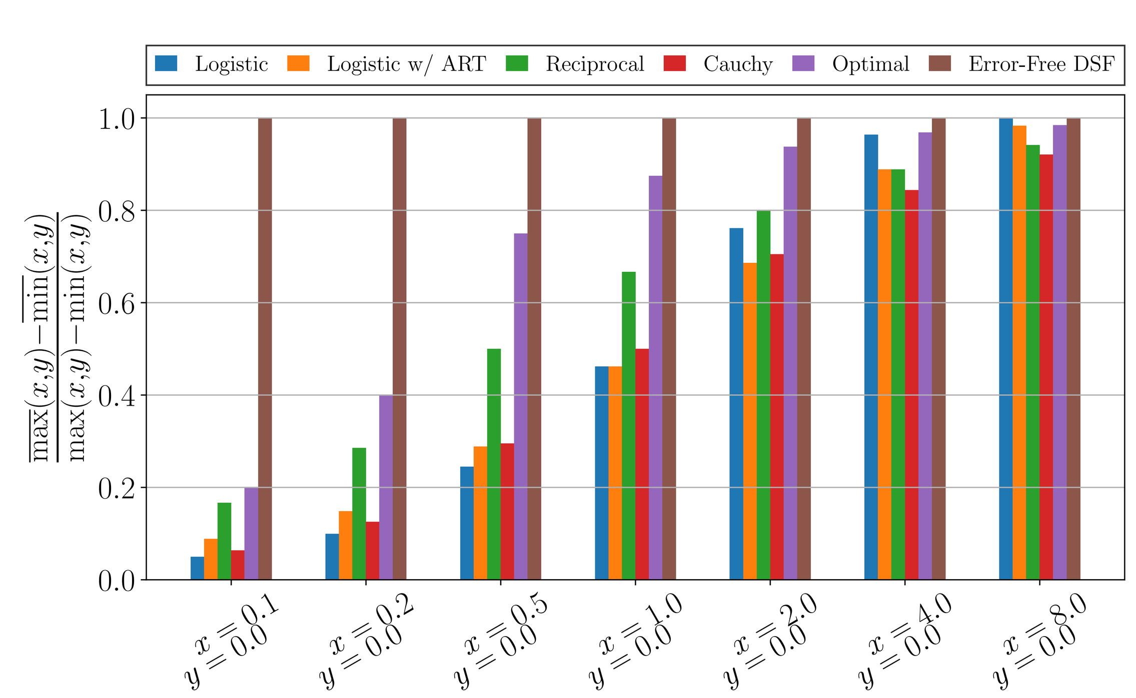
- To enable us to differentiate a swap function, the soft versions of min and max can be defined:

$$\overline{\min}(x, y) = x\sigma(y - x) + y\sigma(x - y), \tag{7}$$

$$\overline{\max}(x, y) = x\sigma(x - y) + y\sigma(y - x), \tag{8}$$

where $\sigma(\cdot)$ is differentiable.

- A sigmoid function $\sigma(x)$ satisfies the following properties that (i) $\sigma(x)$ is non-decreasing, (ii) $\sigma(x) = 1$ if $x \to \infty$, (iii) $\sigma(x) = 0$ if $x \to -\infty$, (iv) $\sigma(0) = 0.5$, and (v) $\sigma(x) = 1 - \sigma(-x)$.



Error-Free Differentiable Swap Functions

- We propose an error-free differentiable swap function:

$$(x', y') = \operatorname{swap}_{\text{error-free}}(x, y), \tag{9}$$

where

$$x' = (\min(x, y) - \overline{\min}(x, y))_{\text{sg}} + \overline{\min}(x, y), \tag{10}$$

$$y' = (\max(x, y) - \overline{\max}(x, y))_{\text{sg}} + \overline{\max}(x, y). \tag{11}$$

Note that sg indicates that gradients are stopped amid backward propagation, inspired by a straight-through estimator.

- A mapping $g : \mathbb{R}^d \to \mathbb{R}$ has to satisfy a permutation-equivariant property:

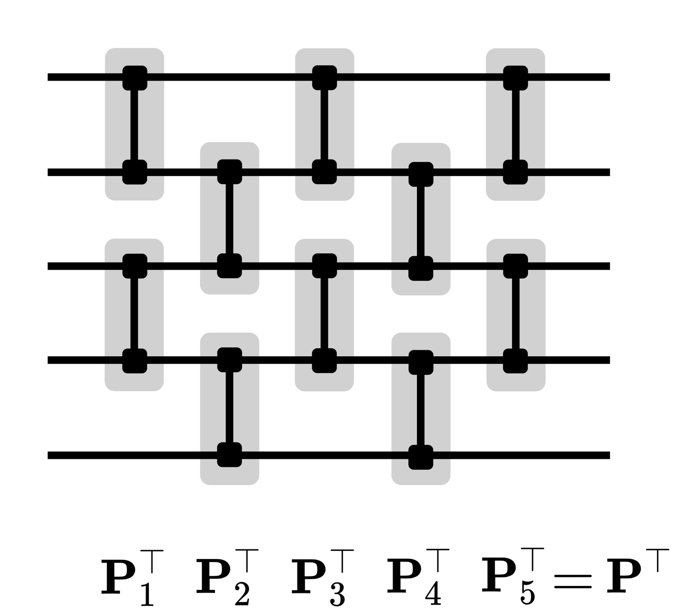
$$[g(\mathbf{x}_{\pi_1}), \ldots, g(\mathbf{x}_{\pi_n})] = \pi([g(\mathbf{x}_1), \ldots, g(\mathbf{x}_n)]), \tag{12}$$

where $\pi_i = [\pi([1, \ldots, n])]_i \; \forall i \in [n]$.

- Both objectives for $\mathbf{P}_{\text{soft}}$ and $\mathbf{P}_{\text{hard}}$ are defined:

$$\mathcal{L}_{\text{soft}} = -\sum_{i=1}^{n}\sum_{j=1}^{n}[\mathbf{P}_{\text{gt}} \log \mathbf{P}_{\text{soft}}]_{ij} - \sum_{i=1}^{n}\sum_{j=1}^{n}[(1 - \mathbf{P}_{\text{gt}}) \log(1 - \mathbf{P}_{\text{soft}})]_{ij}, \tag{13}$$

$$\mathcal{L}_{\text{hard}} = \|\mathbf{P}_{\text{hard}}^\top \mathbf{X} - \mathbf{P}_{\text{gt}}^\top \mathbf{X}\|_F^2. \tag{14}$$



Experiments

Table: Sorting the four-digit MNIST dataset.

Method		Model	Sequence Length 3	5	7	9	15	32
NeuralSort		CNN	91.9 (94.5)	77.7 (90.1)	61.0 (86.2)	43.4 (82.4)	9.7 (71.6)	0.0 (38.8)
Sinkhorn Sort		CNN	92.8 (95.0)	81.1 (91.7)	65.6 (88.2)	49.7 (84.7)	12.6 (74.2)	0.0 (41.2)
Fast Sort & Rank		CNN	90.6 (93.5)	71.5 (87.2)	49.7 (81.3)	29.0 (75.2)	2.8 (60.9)	–
Diffsort	Log.	CNN	92.0 (94.5)	77.2 (89.8)	54.8 (83.6)	37.2 (79.4)	4.7 (62.3)	0.0 (56.3)
Diffsort	Log. w/ ART	CNN	94.3 (96.1)	83.4 (92.6)	71.6 (90.0)	56.3 (86.7)	23.5 (79.4)	0.5 (64.9)
Diffsort	Reciprocal	CNN	94.4 (96.1)	85.0 (93.3)	73.4 (90.7)	60.8 (88.1)	30.2 (81.9)	1.0 (66.8)
Diffsort	Cauchy	CNN	94.2 (96.0)	84.9 (93.2)	73.3 (90.5)	63.8 (89.1)	31.1 (82.2)	0.8 (63.3)
Diffsort	Optimal	CNN	94.6 (96.3)	85.0 (93.3)	73.6 (90.7)	62.2 (88.5)	31.8 (82.3)	1.4 (67.9)
Ours	E.-F. DSFs	CNN	95.2 (96.7)	87.2 (94.2)	76.6 (91.6)	64.8 (89.2)	34.7 (83.3)	2.1 (69.2)
Ours	E.-F. DSFs	Tr-S	95.9 (97.1)	94.8 (97.5)	90.8 (96.5)	86.9 (95.7)	74.3 (93.6)	37.8 (87.7)
Ours	E.-F. DSFs	Tr-L	96.5 (97.5)	95.4 (97.7)	92.9 (97.2)	90.1 (96.5)	82.5 (95.0)	46.2 (88.9)

Table: Sorting the SVHN dataset.

Method		Model	Sequence Length 3	5	7	9	15
Diffsort	Log.	CNN	76.3 (83.2)	46.0 (72.7)	21.8 (63.9)	13.5 (61.7)	0.3 (45.9)
Diffsort	Log. w/ ART	CNN	83.2 (88.1)	64.1 (82.1)	43.8 (76.5)	24.2 (69.6)	2.4 (56.8)
Diffsort	Reciprocal	CNN	85.7 (89.8)	68.8 (84.2)	53.3 (80.0)	40.0 (76.3)	13.2 (66.0)
Diffsort	Cauchy	CNN	85.5 (89.6)	68.5 (84.1)	52.9 (79.8)	39.9 (75.8)	13.7 (66.0)
Diffsort	Optimal	CNN	86.0 (90.0)	67.5 (83.5)	53.1 (80.0)	39.1 (76.0)	13.2 (66.3)
Ours	E.-F. DSFs	CNN	86.8 (90.6)	68.9 (84.5)	53.4 (80.4)	40.0 (77.0)	12.0 (65.3)
Ours	E.-F. DSFs	Transformer-S	86.6 (90.2)	72.6 (85.7)	62.5 (83.5)	48.6 (79.3)	19.3 (69.6)
Ours	E.-F. DSFs	Transformer-L	88.0 (91.2)	74.0 (86.3)	63.9 (83.8)	50.2 (80.1)	21.7 (71.2)

Table: Sorting image fragments of MNIST and CIFAR-10.

Method		Model	MNIST 2×2 (14×14)	MNIST 3×3 (9×9)	CIFAR-10 2×2 (16×16)	CIFAR-10 3×3 (10×10)
Diffsort	Logistic	CNN	98.5 (99.0)	5.3 (42.9)	56.9 (73.6)	0.8 (27.7)
Diffsort	Logistic w/ ART	CNN	98.4 (99.1)	5.4 (42.9)	56.7 (73.4)	0.7 (27.7)
Diffsort	Reciprocal	CNN	98.4 (99.2)	5.3 (42.9)	56.7 (73.4)	0.7 (27.8)
Diffsort	Cauchy	CNN	98.4 (99.2)	5.3 (42.9)	56.9 (73.6)	0.9 (27.9)
Diffsort	Optimal	CNN	98.4 (99.1)	5.3 (43.0)	56.6 (73.4)	0.7 (27.7)
Ours	Error-Free DSFs	CNN	98.4 (99.2)	5.2 (42.6)	56.9 (73.6)	0.8 (28.0)
Ours	Error-Free DSFs	Transformer	98.6 (99.2)	5.6 (43.7)	58.1 (74.2)	0.9 (28.3)

Table: Comparisons of Diffsort with the optimal monotonic sigmoid function and our methods in the MNIST experiments.

Method	Model	Sequence Length 3	5	7	9	15	32
Diffsort	CNN	94.6 (96.3)	85.0 (93.3)	73.6 (90.7)	62.2 (88.5)	31.8 (82.3)	1.4 (67.9)
Ours	CNN	95.2 (96.7)	87.2 (94.2)	76.6 (91.6)	64.8 (89.2)	34.7 (83.3)	2.1 (69.2)
Diffsort	Transformer-S	95.9 (97.1)	90.2 (95.4)	83.9 (94.2)	77.2 (92.9)	57.3 (89.7)	16.3 (81.7)
Ours	Transformer-S	95.9 (97.1)	94.8 (97.5)	90.8 (96.5)	86.9 (95.7)	74.3 (93.6)	37.8 (87.7)
Diffsort	Transformer-L	96.5 (97.5)	92.6 (96.4)	87.6 (95.3)	82.6 (94.3)	67.8 (92.0)	32.1 (85.7)
Ours	Transformer-L	96.5 (97.5)	95.4 (97.7)	92.9 (97.2)	90.1 (96.5)	82.5 (95.0)	46.2 (88.9)

arXiv **GitHub**