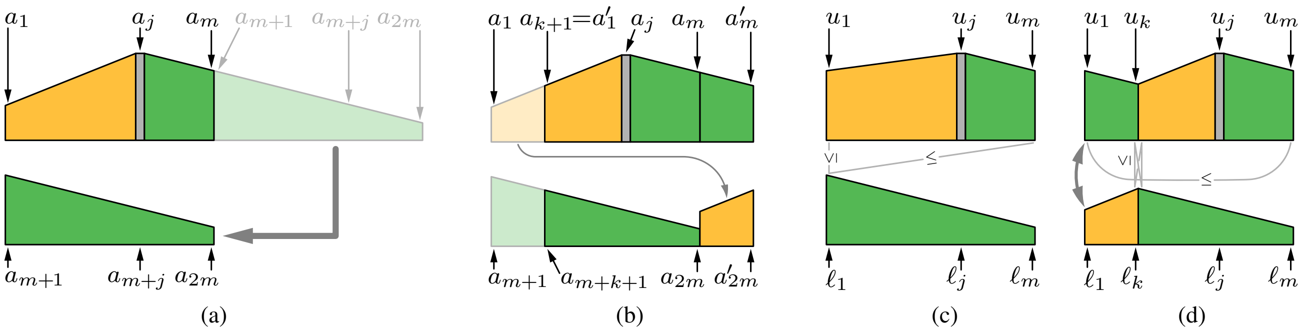

Figure 6: Bitonic merge turns a bitonic input sequence into two bitonic output sequences, with all elements in the one (upper, u_i) sequence larger than all elements in the other (lower, ℓ_i) sequence. The diagrams show the vertical alignment of elements to compare (a) and the invariance to cyclic permutations (b). Depending on the values in (a), no exchanges (c) or exchanges (d) are executed.

A. The Bitonic Sorting Network

In the following, we detail the bitonic sorting network and sketch a proof of why the bitonic sorting networks sorts:

A bitonic sequence is sorted by several bitonic merge blocks, shown in orange and green in Figure 2. Each block takes a bitonic input sequence $a_1 a_2 \dots a_{2m}$ of length $2m$ and turns it into two bitonic output sequences $\ell_1 \ell_2 \dots \ell_m$ and $u_1 u_2 \dots u_m$ of length m that satisfy $\max_{i=1}^m l_i \leq \min_{i=1}^m u_i$. These subsequences are recursively processed by bitonic merge blocks, until the output sequences are of length 1. At this point, the initial bitonic sequence has been turned into a monotonic sequence due to the minimum/maximum conditions that hold between the output sequences (and thus elements).

A bitonic merge block computes its output as $\ell_i = \min(a_i, a_{m+i})$ and $u_i = \max(a_i, a_{m+i})$. This is depicted in Figure 2 by the arrows pointing from the minimum to the maximum. To demonstrate that bitonic merge works, we show that this operation indeed produces two bitonic output sequences for which the relationship $\max_{i=1}^m l_i \leq \min_{i=1}^m u_i$ holds.

Note that neither a cyclic permutation of the sequence ($a'_i = a_{(i+k-1 \bmod 2m)+1}$ for some k, Figure 6b), nor a reversal, change the bitonic character of the sequence. As can be seen in Figure 6b, even under cyclic permutation still the same pairs of elements are considered for a potential swap. Thus, as a cyclic permutation or a reversal only causes the output sequences to be analogously cyclically permuted or reversed, this changes neither the bitonic character of these sequences nor the relationship between them. Therefore, it suffices to consider the special case shown in Figure 6a, with a monotonically increasing sequence (orange) followed by a monotonically decreasing sequence (green) and the maximum element a_j (gray) in the first half. Note that in this case $\forall i; j \leq i \leq m : a_i \geq a_{m+i} \wedge u_i = a_i \wedge \ell_i = a_{m+i}$. For this case, we have to distinguish two sub-cases: $a_1 \geq a_{m+1}$ and $a_1 < a_{m+1}$.

If, on one hand, $a_1 \geq a_{m+1}$, we have the situation shown in Figure 6c: the output sequence $u_1 u_2 \dots u_m$ is simply the first half of the sequence, the output sequence $\ell_1 \ell_2 \dots \ell_m$ is the second half. Thus, both output sequences are bitonic (since they are subsequences of a bitonic input sequence) and $\min_{i=1}^m u_i = \min(u_1, u_m) \geq \ell_1 = \max_{i=1}^m \ell_i$.

If, on the other hand, $a_1 < a_{m+1}$, we can infer $\exists k; 1 \leq k < j : a_k > a_{m+k} \wedge a_{k+1} \leq a_{m+k+1}$. This situation is depicted in Figure 6d. Thus, $\forall i; 1 \leq i \leq k : u_i = a_{m+i} \wedge \ell_i = a_i$ and $\forall i; k < i \leq m : u_i = a_i \wedge \ell_i = a_{m+i}$. Since $u_k = a_{m+k} > a_k = \ell_k$, $u_k = a_{m+k} \geq a_{m+k+1} = \ell_{k+1}$, $u_{k+1} = a_{k+1} \geq a_{m+k+1} = \ell_{k+1}$, $u_{k+1} = a_{k+1} \geq a_k = \ell_k$, we obtain $\max_{i=1}^m l_i \leq \min_{i=1}^m u_i$. Figure 6d shows that the two output sequences are bitonic and that all elements of the upper output sequence are greater than or equal to all elements of the lower output sequence.

B. Implementation Details

B.1. MNIST

For the MNIST based task, we use the same convolutional neural network architecture as in previous works (Grover et al., 2019; Cuturi et al., 2019). That is, two convolutional layers with a kernel size of 5×5, 32 and 64 channels respectively, each followed by a ReLU and MaxPool layer; after flattening, this is followed by a fully connected layer with a size of 64, a ReLU layer, and a fully connected output layer mapping to a scalar.

B.2. SVHN

For the SVHN task, we use a network with four convolutional layers with a kernel size of 5×5 and $(32, 64, 128, 256)$ filters, each followed by a ReLU and a max-pooling layer with stride 2×2; followed by a fully connected layer with size 64, a ReLU, and a layer with output size 1.

B.3. Fast Sort & Rank

To evaluate the fast sorting and ranking method by Blondel et al. (2020), we used the mean-squared-error loss between predicted and ground truth ranks as this method does not produce differentiable permutation matrices.

B.4. Top-k Supervision

For top-k supervision, we use ResNet18 as well as a Vanilla CNN with 4 convolutional and 2 fully connected layers. The vanilla CNN is has the following architecture: `C16-BN-R-C32-BN-R-Max2-C64-BN-R-C128-BN-R-Max2-F256-Fc` where `Ck` denotes a convolutional layer with `k` output channels, a 3×3 kernel, and padding of 1, `BN` denotes BatchNorm (Ioffe & Szegedy, 2015), `R` denotes ReLU, `Max2` denotes MaxPool with a 2×2 kernel, and `Fk` denotes a fully connected layer with `k` outputs. This vanilla CNN is inspired from Blondel et al. (2020). We train each model using Adam (Kingma & Ba, 2015) for 500 epochs at a learning rate of 10^{-3}.

C. Standard Deviations of the Results

Tables 7, 8, 9, 10, and 11 display the standard deviations for the results in this work.

Table 7: Same as Table 1 but with additional standard deviations.

MNIST	$n=3$	$n=5$	$n=7$	$n=8$	$n=15$
Fast Sort & Rank	90.6 | 93.5 | 73.5 ±0.4 | ± 0.3 | ± 0.8	71.5 | 87.2 | 71.5 ±0.9 | ± 0.4 | ± 0.9	49.7 | 81.3 | 70.5 ±0.6 | ± 0.3 | ± 0.4	29.0 | 75.2 | 69.2 ±1.1 | ± 0.6 | ± 0.7	2.8 | 60.9 | 67.4 ±0.2 | ± 0.4 | ± 0.6
Odd-Even	95.2 | 96.7 | 86.1 ±0.3 | ± 0.2 | ± 0.6	86.3 | 93.8 | 86.3 ±0.9 | ± 0.4 | ± 0.9	75.4 | 91.2 | 86.4 ±1.8 | ± 0.6 | ± 0.9	64.3 | 89.0 | 86.7 ±1.8 | ± 0.6 | ± 1.1	35.4 | 83.7 | 87.6 ±1.8 | ± 0.5 | ± 0.5
MNIST	$n=2$	$n=4$	$n=8$	$n=16$	$n=32$
Odd-Even	98.1 | 98.1 | 84.3 ±0.3 | ± 0.3 | ± 0.9	90.5 | 94.9 | 85.5 ±1.2 | ± 0.6 | ± 1.5	63.6 | 87.9 | 83.6 ±11.6 | ± 4.2 | ± 6.1	31.7 | 82.8 | 87.3 ±1.5 | ± 0.5 | ± 0.5	1.7 | 69.1 | 86.7 ±0.5 | ± 1.5 | ± 1.0
Bitonic	98.1 | 98.1 | 84.0 ±0.2 | ± 0.2 | ± 1.2	91.4 | 95.3 | 86.7 ±0.6 | ± 0.3 | ± 0.4	70.6 | 90.3 | 86.9 ±4.4 | ± 1.3 | ± 1.8	30.5 | 81.7 | 86.6 ±1.8 | ± 1.2 | ± 0.9	2.7 | 67.3 | 85.4 ±1.3 | ± 2.7 | ± 1.7

Table 8: Same as Table 2 but with additional standard deviations.

SVHN	$n=2$	$n=4$	$n=8$	$n=16$	$n=32$
Det. NeuralSort	90.1 | 90.1 | 39.9 ±0.7 | ± 0.7 | ± 1.7	61.4 | 78.1 | 45.4 ±0.8 | ± 0.3 | ± 1.2	15.7 | 62.3 | 48.5 ±1.6 | ± 1.2 | ± 1.6	0.1 | 45.7 | 51.0 ±0.1 | ± 0.6 | ± 1.2	0.0 | 29.9 | 52.7 ±0.0 | ± 1.4 | ± 1.5
Optimal Transport	85.5 | 85.5 | 25.9 ±0.0 | ± 0.0 | ± 0.0	57.6 | 75.6 | 41.6 ±1.1 | ± 0.8 | ± 1.8	19.9 | 64.5 | 51.7 ±1.9 | ± 1.1 | ± 1.2	0.3 | 47.7 | 53.8 ±0.2 | ± 1.7 | ± 1.4	0.0 | 29.4 | 53.3 ±0.0 | ± 1.0 | ± 1.9
Fast Sort & Rank	93.4 | 93.4 | 57.6 ±0.7 | ± 0.7 | ± 3.7	58.0 | 75.8 | 41.5 ±1.1 | ± 0.7 | ± 1.0	8.6 | 52.7 | 34.4 ±1.0 | ± 0.6 | ± 0.3	0.3 | 36.5 | 41.6 ±0.2 | ± 1.4 | ± 1.8	0.0 | 14.0 | 27.5 ±0.0 | ± 3.1 | ± 9.1
Odd-Even	93.4 | 93.4 | 58.0 ±0.4 | ± 0.4 | ± 2.0	74.8 | 85.5 | 62.6 ±1.2 | ± 0.7 | ± 1.1	35.2 | 73.5 | 63.9 ±1.2 | ± 0.5 | ± 1.1	1.8 | 54.4 | 62.3 ±0.8 | ± 1.6 | ± 1.6	0.0 | 36.6 | 62.6 ±0.0 | ± 1.5 | ± 0.8
Bitonic	93.8 | 93.8 | 58.6 ±0.3 | ± 0.3 | ± 0.8	74.4 | 85.3 | 62.1 ±0.7 | ± 0.3 | ± 1.1	38.3 | 75.1 | 66.8 ±2.4 | ± 1.1 | ± 1.4	3.9 | 59.6 | 66.8 ±0.3 | ± 0.8 | ± 1.4	0.0 | 42.4 | 67.7 ±0.0 | ± 3.5 | ± 3.6

Table 9: Same as Table 3 but with additional standard deviations.

λ	0.25	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4	0.4
n	32	32	64	128	256	512	1024	32	64	128	256	512	1024
batch size	128	128	64	32	16	8	4	4	4	4	4	4	4
$s = 30$	78.20 $\pm$2.35	79.89 $\pm$1.97	81.25 $\pm$1.93	82.50 $\pm$1.09	82.05 $\pm$2.62	82.50 $\pm$1.75	82.80 $\pm$2.27	71.08 $\pm$1.67	75.88 $\pm$2.30	79.43 $\pm$2.35	81.46 $\pm$1.47	82.98 $\pm$2.02	82.80 $\pm$2.27
$s = 32.5$	76.98 $\pm$0.86	79.62 $\pm$3.62	81.66 $\pm$2.42	80.15 $\pm$3.84	81.87 $\pm$2.19	82.64 $\pm$1.60	81.63 $\pm$6.22	72.31 $\pm$2.04	75.59 $\pm$2.05	79.71 $\pm$1.57	81.36 $\pm$1.98	82.99 $\pm$1.67	81.63 $\pm$6.22
$s = 35$	77.45 $\pm$1.64	80.93 $\pm$2.75	81.26 $\pm$2.41	80.72 $\pm$3.89	81.42 $\pm$2.09	81.51 $\pm$2.12	81.15 $\pm$3.12	71.15 $\pm$1.69	75.73 $\pm$2.46	78.81 $\pm$1.36	79.32 $\pm$4.85	82.30 $\pm$1.22	81.15 $\pm$3.12
$s = 37.5$	76.40 $\pm$3.90	80.02 $\pm$1.74	80.05 $\pm$1.93	81.50 $\pm$2.03	80.05 $\pm$3.94	82.67 $\pm$2.21	80.07 $\pm$3.67	70.69 $\pm$2.26	75.80 $\pm$1.22	79.11 $\pm$1.88	80.64 $\pm$2.18	82.70 $\pm$1.66	80.07 $\pm$3.67
$s = 40$	77.69 $\pm$1.54	80.97 $\pm$2.03	80.23 $\pm$3.51	81.55 $\pm$1.97	79.75 $\pm$5.41	81.89 $\pm$2.51	81.15 $\pm$3.31	70.20 $\pm$2.06	74.67 $\pm$2.45	78.14 $\pm$2.49	80.06 $\pm$1.93	81.39 $\pm$1.67	81.15 $\pm$3.31
mean	77.35 $\pm$2.06	80.29 $\pm$2.48	80.89 $\pm$2.48	81.28 $\pm$2.80	81.03 $\pm$3.48	82.24 $\pm$2.03	81.36 $\pm$3.97	71.09 $\pm$2.00	75.53 $\pm$2.10	79.04 $\pm$1.97	80.57 $\pm$2.77	82.47 $\pm$1.71	81.36 $\pm$3.97
best s	78.20 $\pm$3.90	80.97 $\pm$3.62	81.66 $\pm$3.51	82.50 $\pm$3.89	82.05 $\pm$5.41	82.67 $\pm$2.51	82.80 $\pm$6.22	72.31 $\pm$2.26	75.88 $\pm$2.46	79.71 $\pm$2.49	81.46 $\pm$4.85	82.99 $\pm$2.02	82.80 $\pm$6.22
worst s	76.40 $\pm$0.86	79.62 $\pm$1.74	80.05 $\pm$1.93	80.15 $\pm$1.09	79.75 $\pm$2.09	81.51 $\pm$1.60	80.07 $\pm$2.27	70.20 $\pm$1.67	74.67 $\pm$1.22	78.14 $\pm$1.36	79.32 $\pm$1.47	81.39 $\pm$1.22	80.07 $\pm$2.27

Table 10: Same as Table 5 but with additional standard deviations.

	$n = 4$		$n = 32$	
Setting / λ	0	0.25	0	0.25
Odd-Even (MNIST)	94.5 ± 0.3	94.9 ± 0.6	61.5 ± 1.9	69.1 ± 1.5
Bitonic (MNIST)	93.6 ± 1.4	95.3 ± 0.3	62.8 ± 15.5	67.3 ± 2.7
Odd-Even (SVHN)	77.3 ± 1.0	85.5 ± 0.7	28.5 ± 2.7	36.6 ± 1.5
Bitonic (SVHN)	78.1 ± 0.2	85.3 ± 0.3	35.0 ± 0.8	42.4 ± 3.5

Table 11: Same as Table 6 but with additional standard deviations.

Setting	Softmax CE	Diff. Top-k
CIFAR-10, Vanilla CNN	$87.2\% \pm 0.2\%$	$88.0\% \pm 0.4\%$
CIFAR-10, ResNet18	$91.0\% \pm 0.3\%$	$90.9\% \pm 0.2\%$
CIFAR-100, Vanilla CNN	$58.2\% \pm 0.3\%$	$56.3\% \pm 0.5\%$
CIFAR-100, ResNet18	$61.9\% \pm 0.4\%$	$63.3\% \pm 0.6\%$