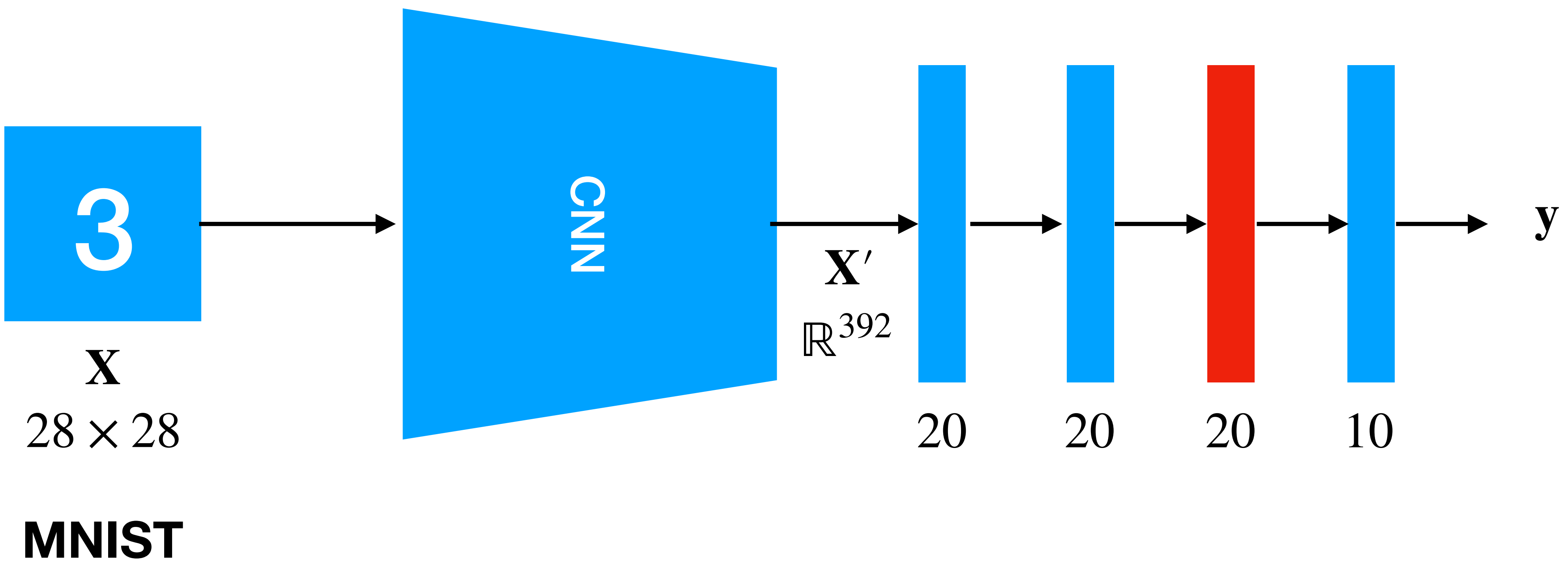
KDN Extended to CNNs (#4)
&
Weighted KDNs

KD-CNN

base architecture



KD-CNN

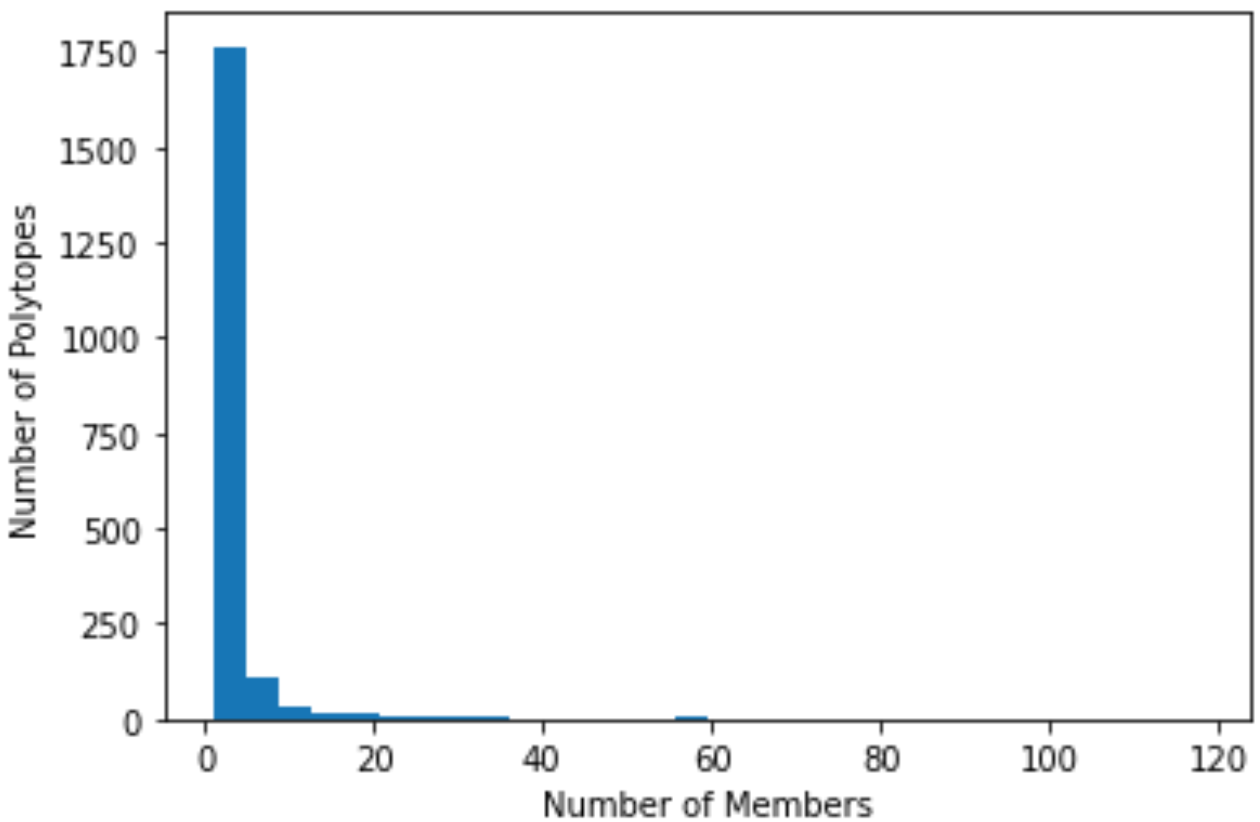
MNIST (latent dims = 392, 4 FC layers)

Model	Accuracy
Vanilla CNN	0.9671
KD-CNN (all the FC layers) $T = 1$	0.2574
KD-CNN (all the FC layers) $T = 10$	0.2574
KD-CNN (all the FC layers) $T = 100$	0.2574
KD-CNN (Penultimate FC layer) $T = 1$	0.9143
KD-CNN (Penultimate FC layer) $T = 10$	0.9143
KD-CNN (Penultimate FC layer) $T = 100$	0.9143

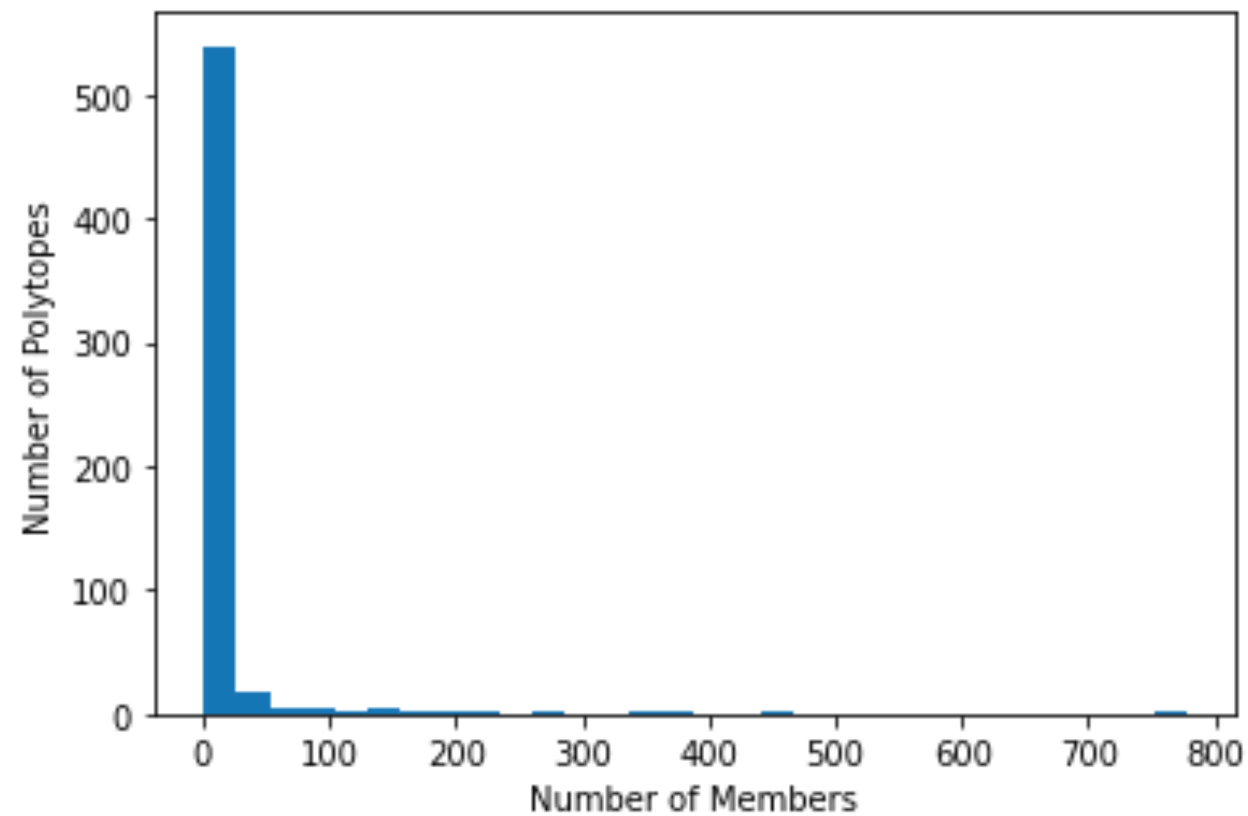
KD-CNN

When all the FC layers are used to compute polytope memberships

MNIST class 0



MNIST class 1

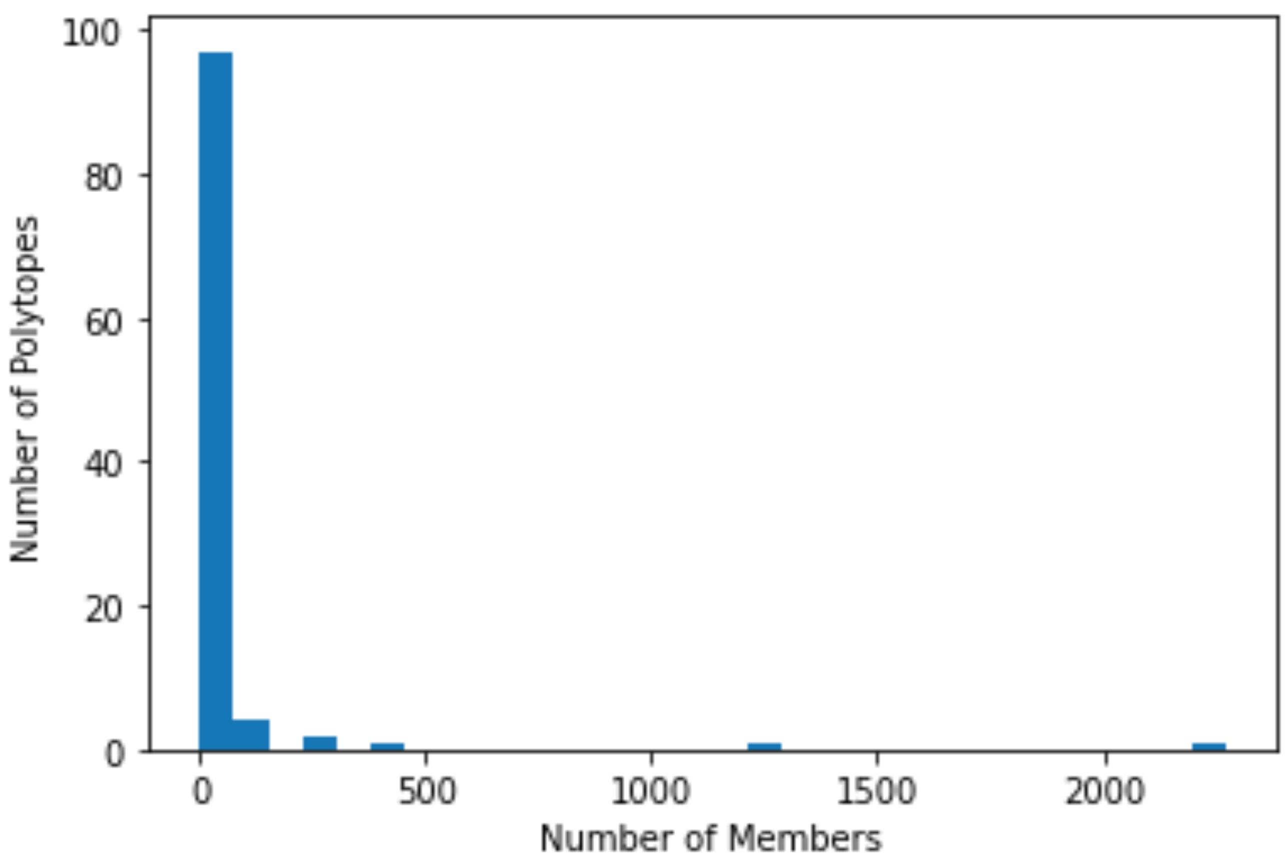


number of unique polytopes is high
members per polytope is generally low

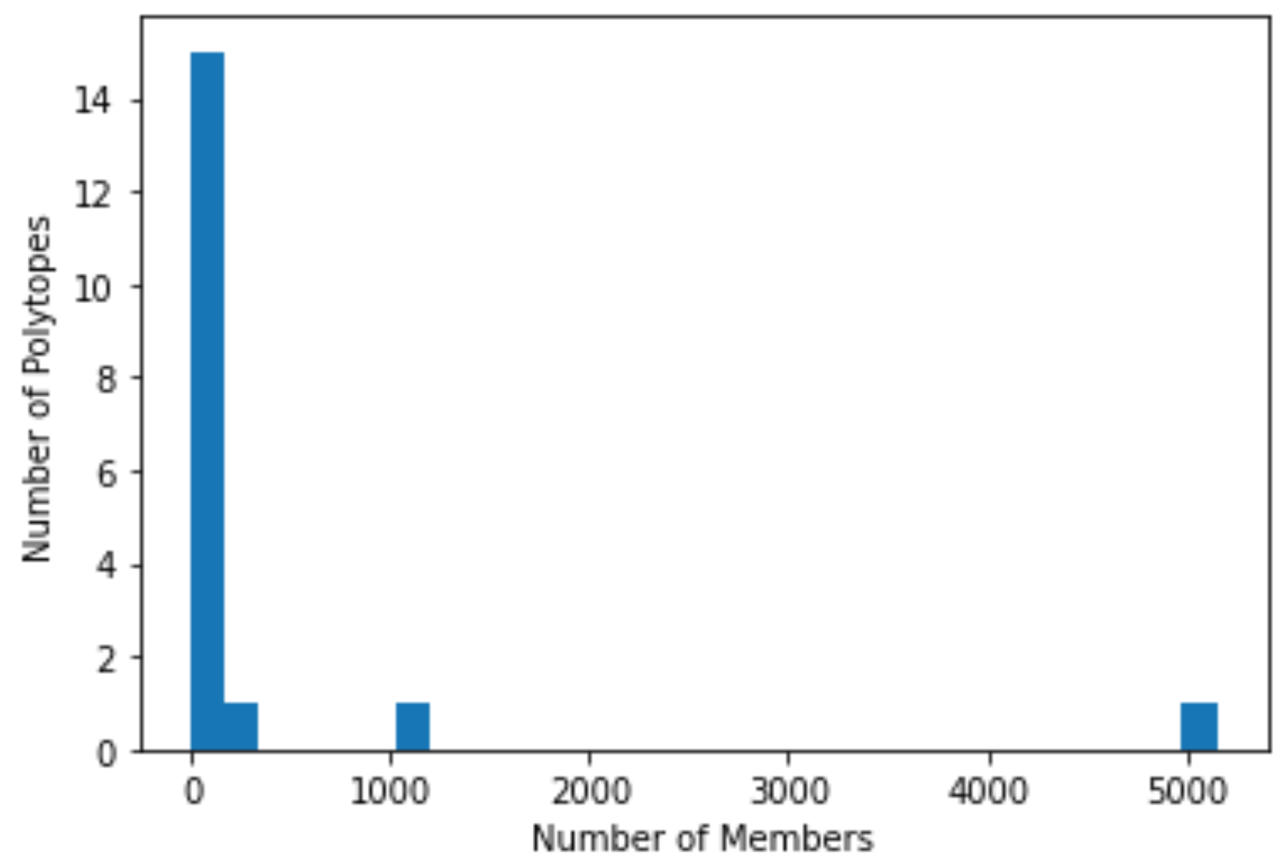
KD-CNN

When only the penultimate layer is used to compute polytope memberships

MNIST class 0



MNIST class 1



number of unique polytopes is low
members per polytope is generally high

KD-CNN

CIFAR-10 (latent dims = 512)

Model	Accuracy
Vanilla CNN	0.6276
KD-CNN (all the FC layers) $T = 1$	-
KD-CNN (Penultimate FC layer) $T = 100$	0.4242

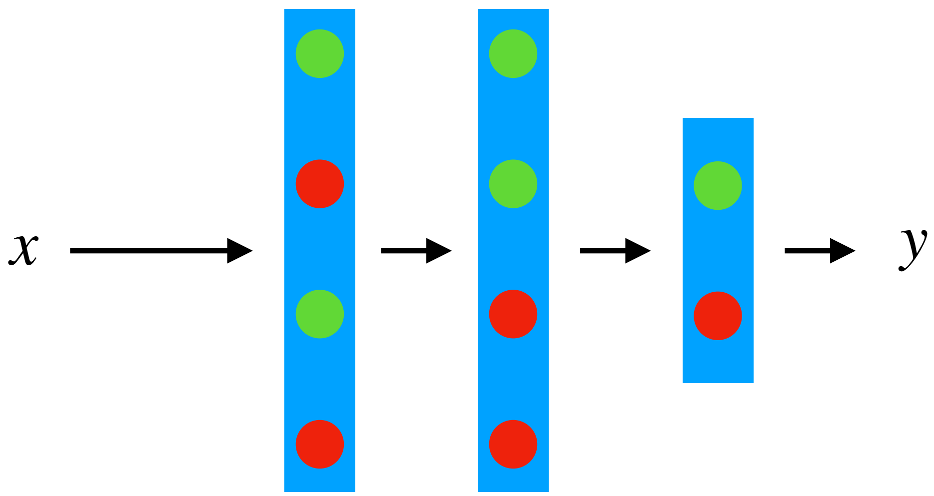
KD-CNN

CIFAR-10 (latent dims = 1024)

Model	Accuracy
Vanilla CNN	>>0.1
KD-CNN (all the FC layers) $T = 1$	-
KD-CNN (Penultimate FC layer) $T = 100$	0.1

When the dimensionality of the latent representation increases KDCNN does not perform well

Weighting Schemes



1010 1100 10

activation pattern

Weighting Schemes

native polytope activation : 1010 1100 10

foreign polytope activation : 1010 0101 10

Total number of matches based weighting (TM) = $\dfrac{\textbf{number of matches}}{\textbf{sequence length}}$

First mismatch based weighting (FM) = $\dfrac{\textbf{sequence length up to the first mismatch}}{\textbf{sequence length}}$

KD-CNN

MNIST (latent dims = 392, 4 FC layers)

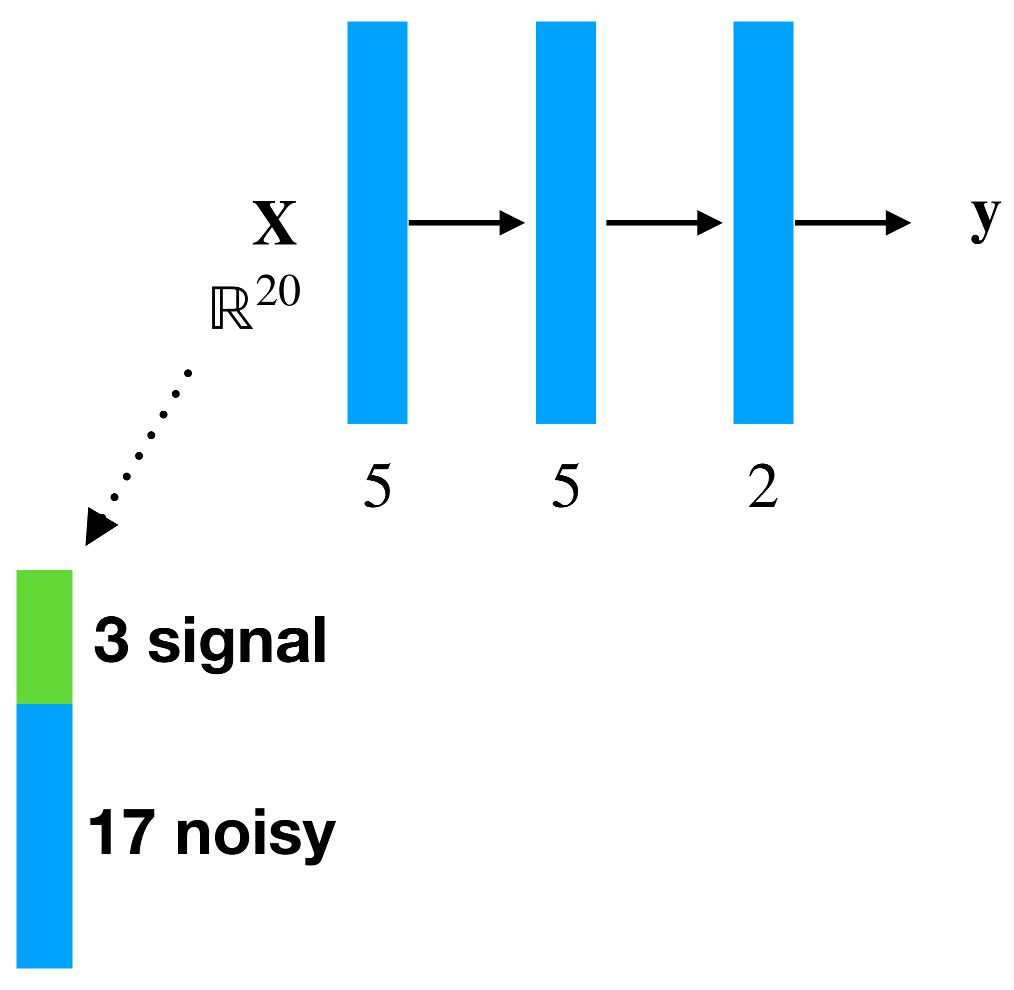
Model	Accuracy
Vanilla CNN	0.9671
KD-CNN (all the FC layers)	**0.2574**
KD-CNN (Penultimate FC layer)	0.9143
KD-CNN (all the FC layers + TM weighting)	**0.9624**
KD-CNN (all the FC layers + FM weighting)	**0.9571**

Weighted KDNs

Evaluating the Robustness to Noise

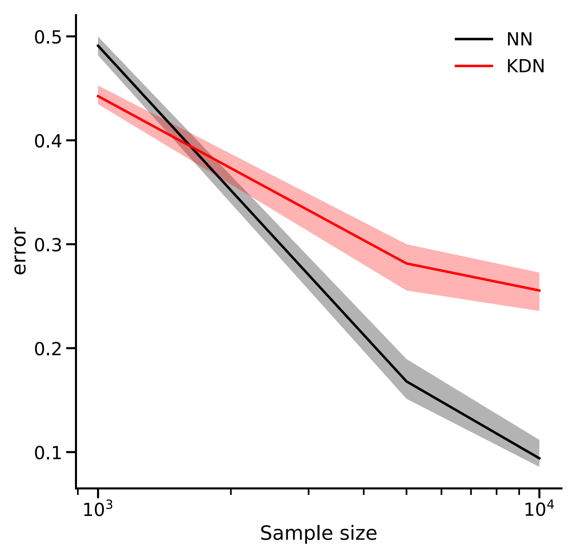
Weighted KDNs

base architecture

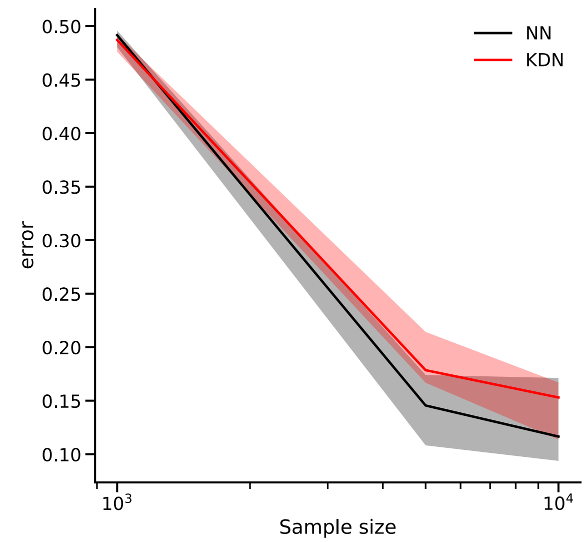


Weighted KDNs

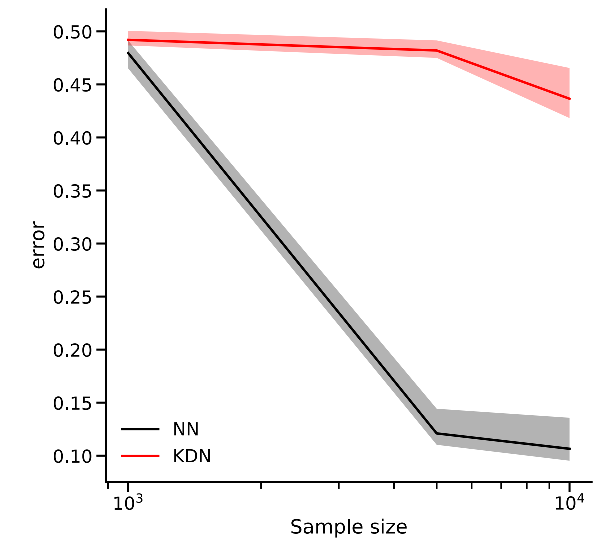
Evaluated on the Gaussian Sparse Parity Dataset (S3, N17)



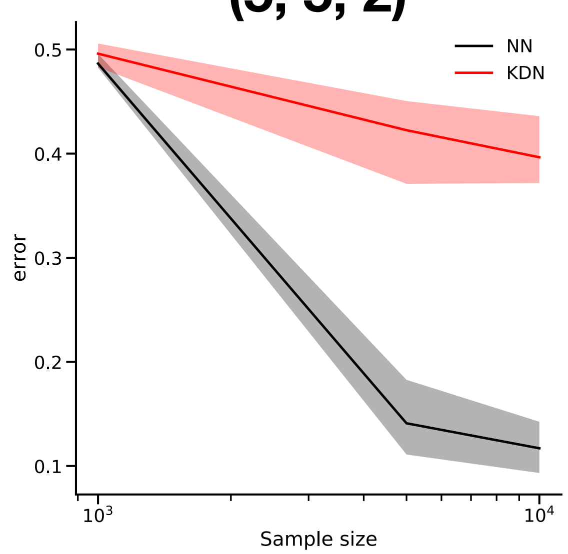
KDN + allFC + No Weighting (5, 5, 2)



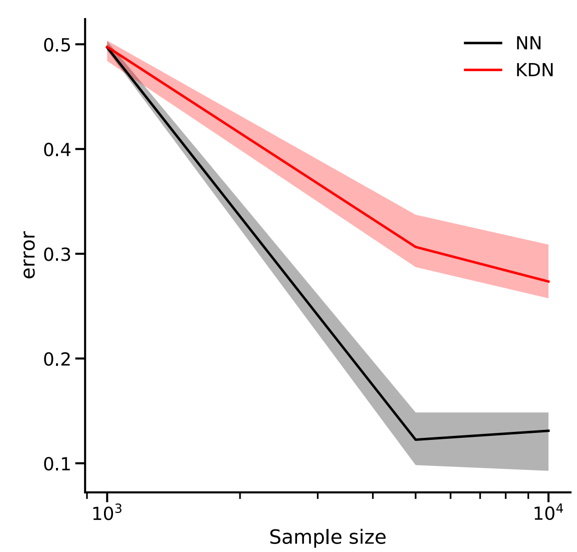
KDN + allFC + FM Weighting (5, 5, 2)



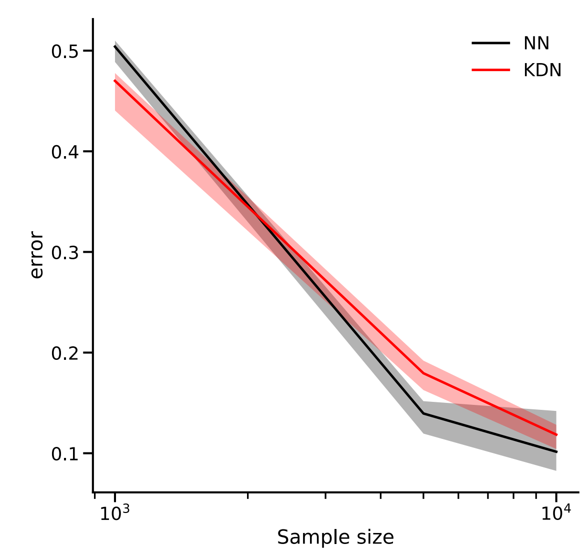
KDN + allFC + TM Weighting (5, 5, 2)



KDN + allFC + PLTM Weighting (5, 5, 2)



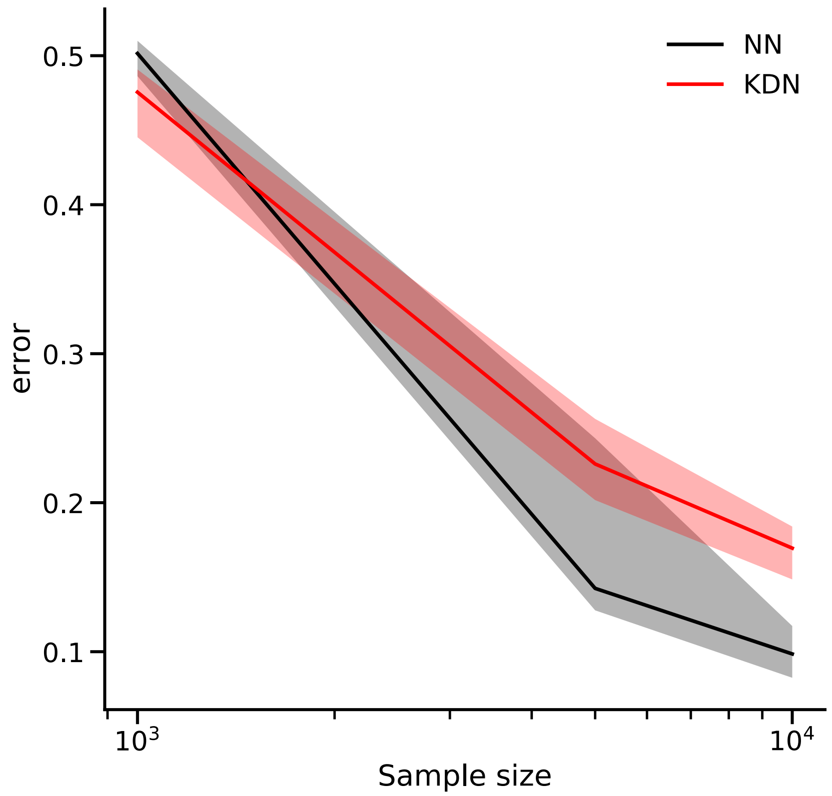
KDN + allFC + AP Weighting (5, 5, 2)



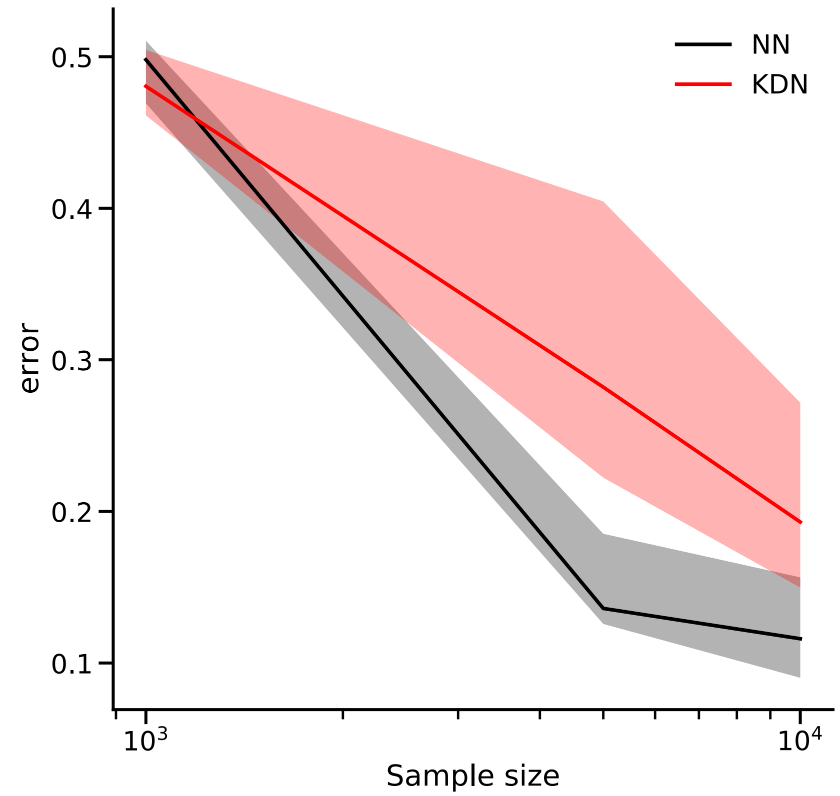
KDN + allFC + EFM Weighting (5, 5, 2)

Weighted KDNs

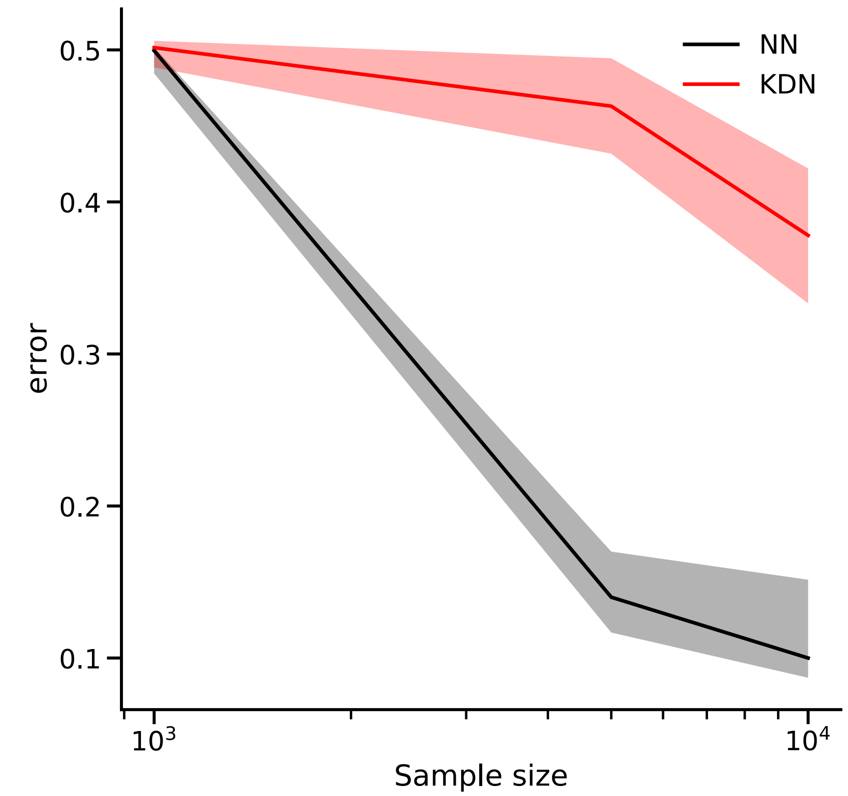
Evaluated on the Gaussian Sparse Parity Dataset (S3, N17)



KDN + PL + No Weighting
(5, 5, 2)



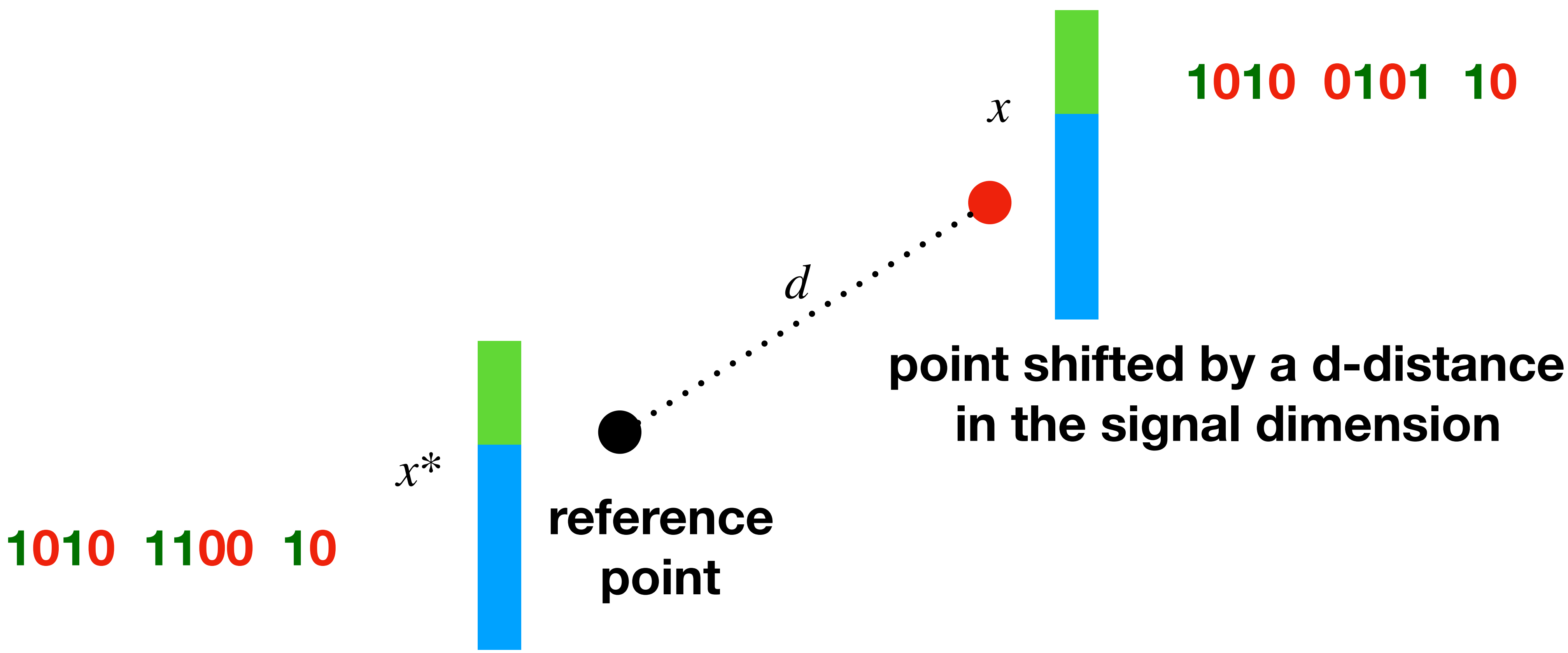
KDN + PL + FM Weighting
(5, 5, 2)



KDN + PL + TM Weighting
(5, 5, 2)

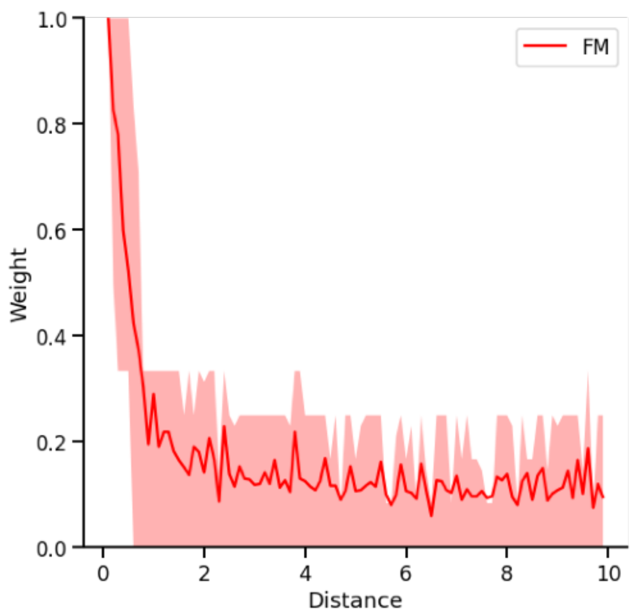
Distance vs. Weight

Evaluated on the Gaussian Sparse Parity Dataset (S3, N17)

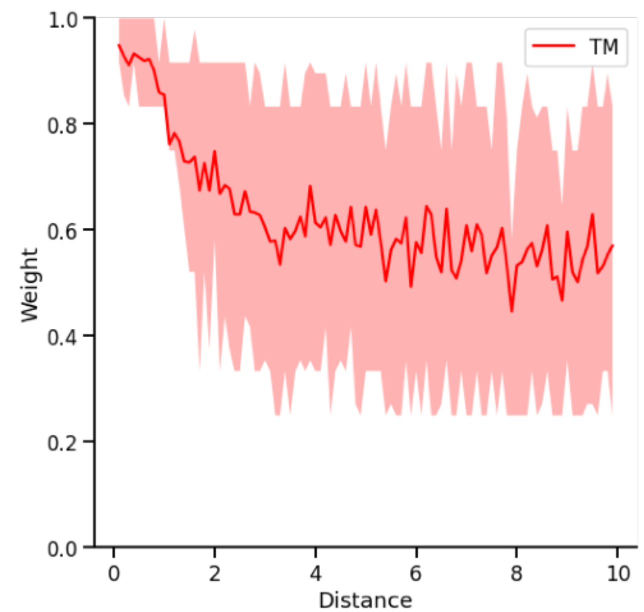


Distance vs. Weight

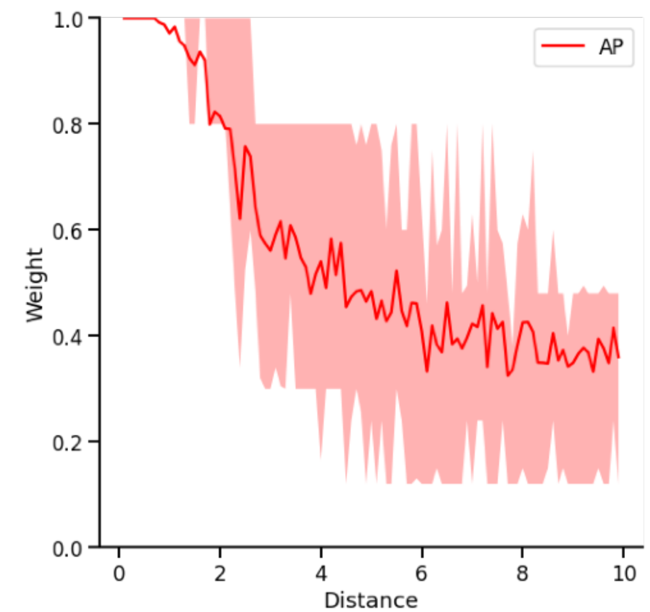
Evaluated on the Gaussian Sparse Parity Dataset (S3, N17)



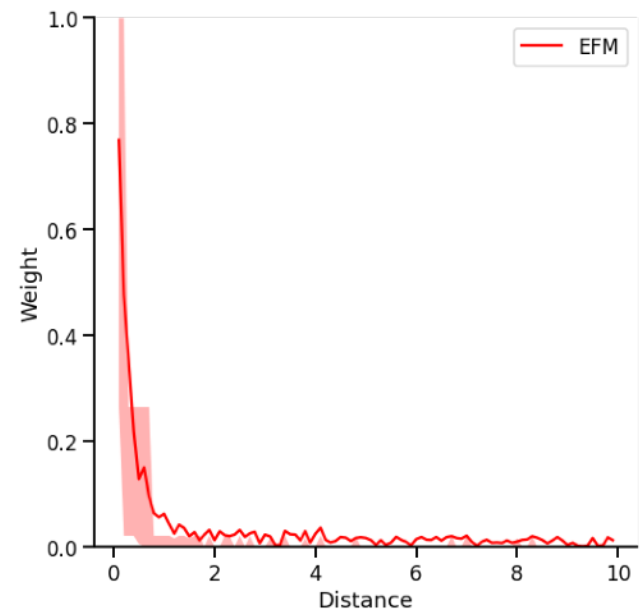
FM Weighting
(5, 5, 2)
allFC



TM Weighting
(5, 5, 2)
allFC



AP Weighting
(5, 5, 2)
allFC

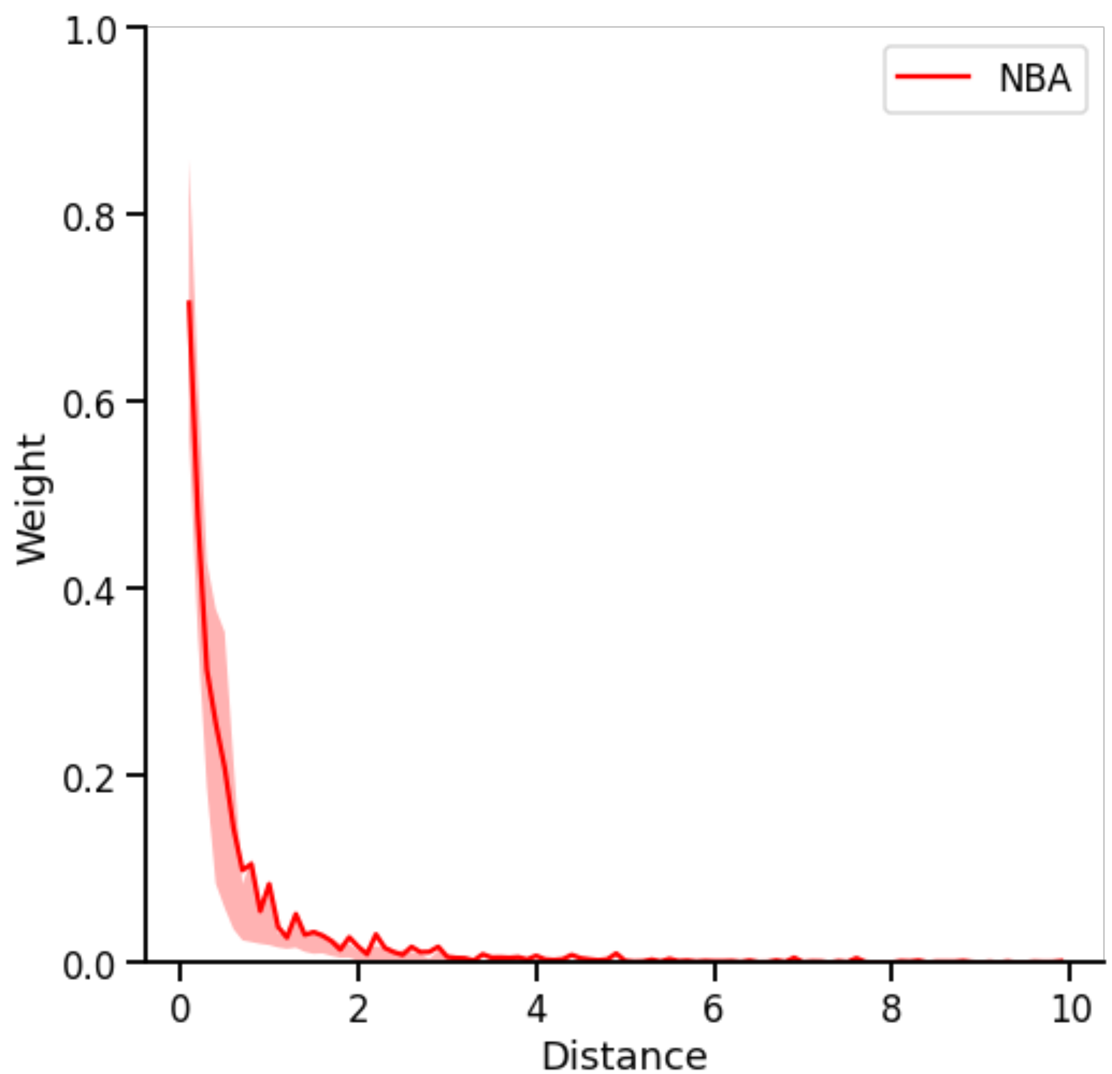


EFM Weighting
(5, 5, 2)
allFC

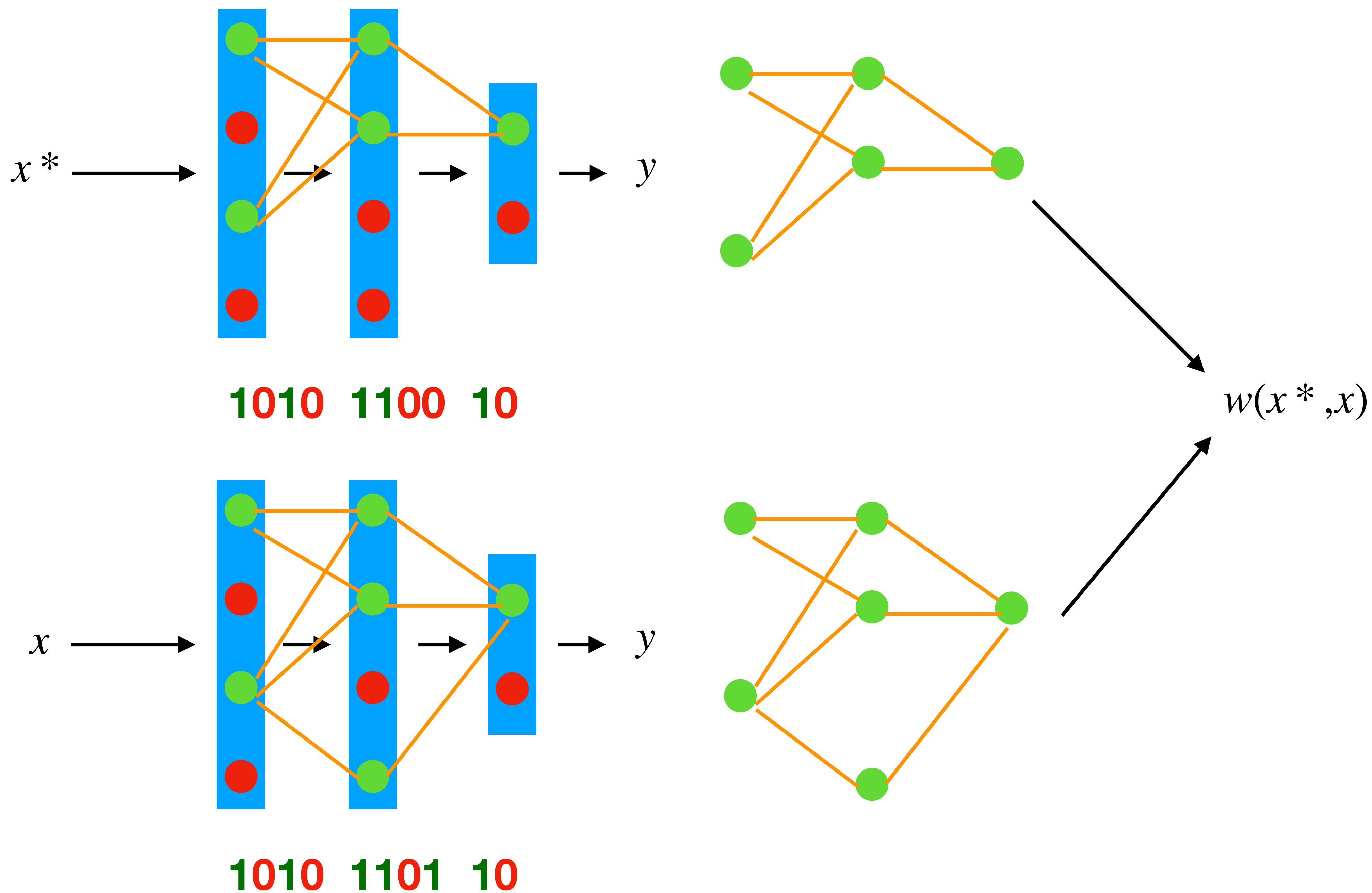
Distance vs. Weight

Evaluated on the Gaussian Sparse Parity Dataset (S3, N17)

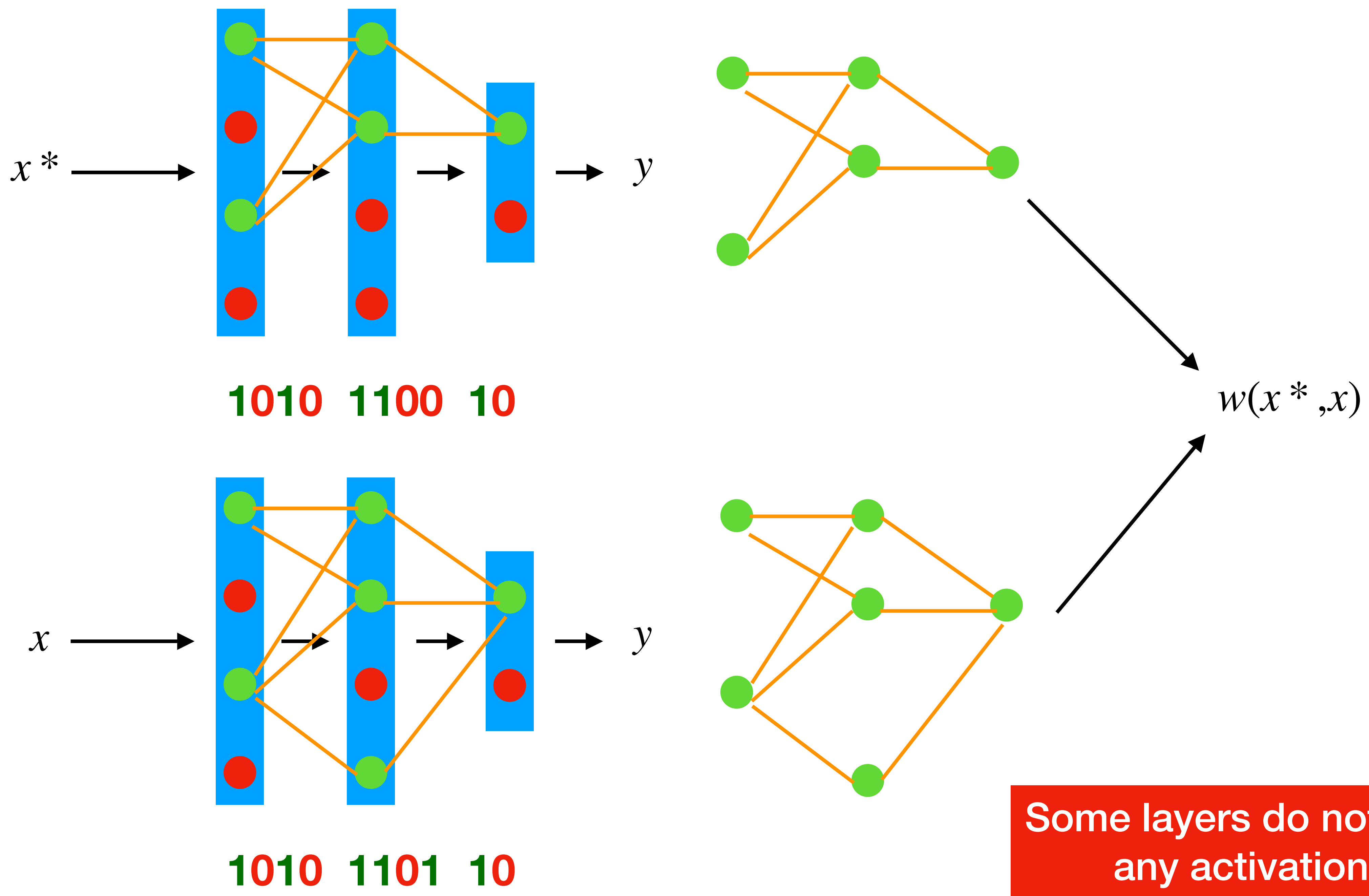
When the layer outputs (instead of the binary activation patterns) are used to weight



An Activation Path-based Weighting Scheme?



An Activation Path-based Weighting Scheme?

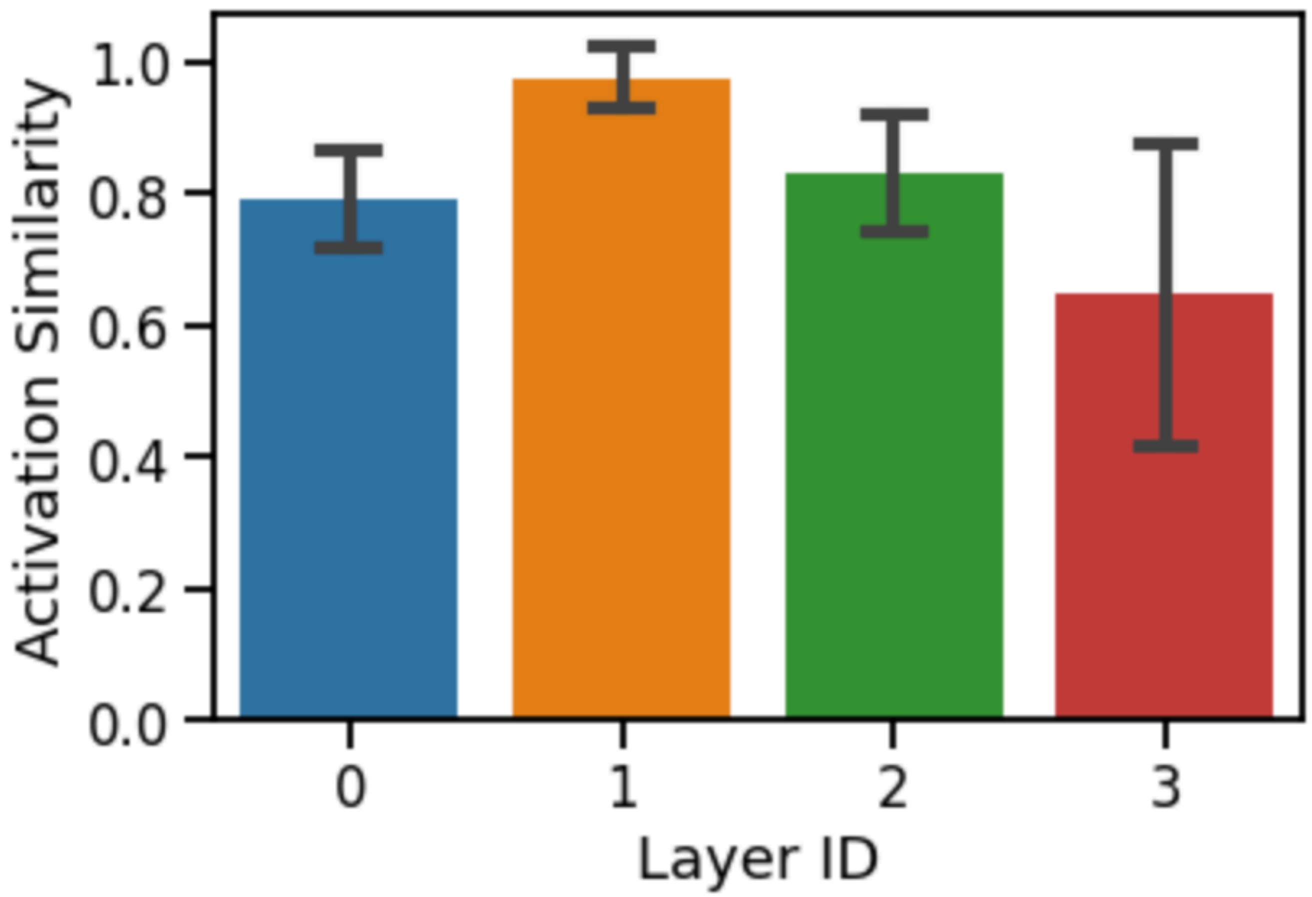


Some layers do not have any activations
APs get blocked

Noise Penetration

(25, 10, 5, 2) network on GSP(3,17)

Same signal dimension with different noise (same class)

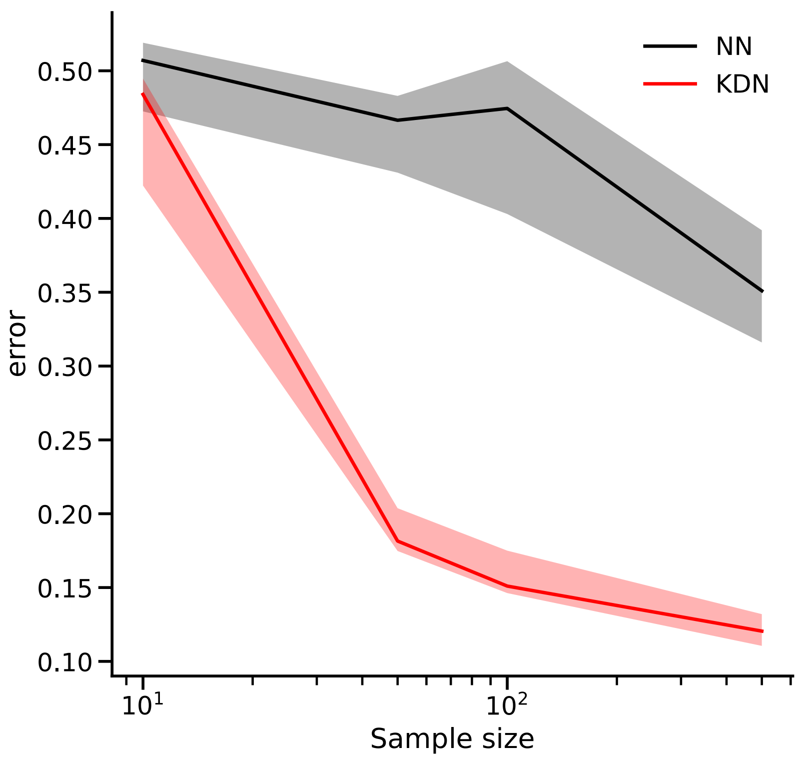


Activation patterns are more dissimilar in the prediction layer

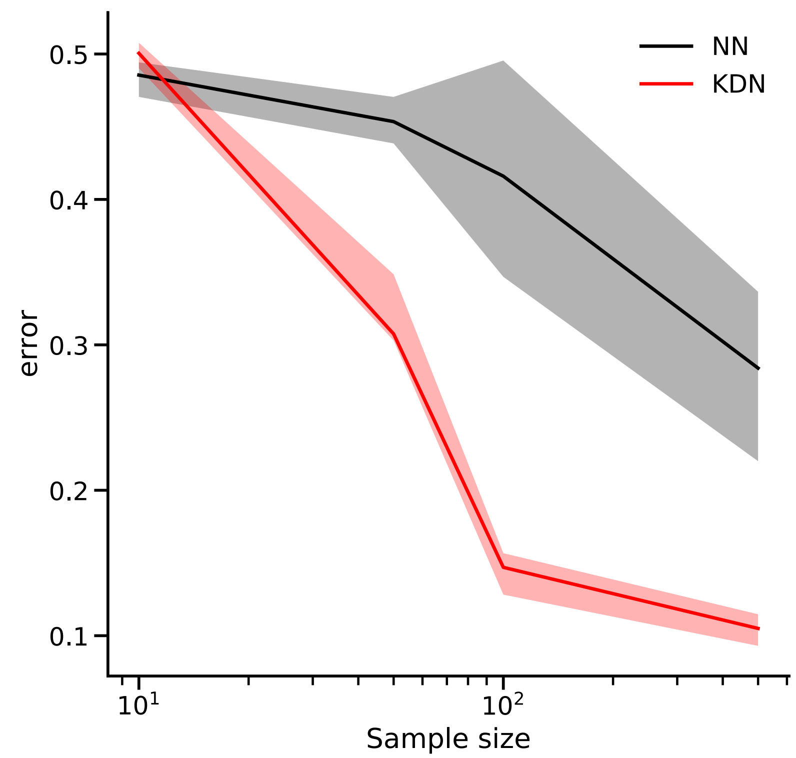
- Adapt the KDF code to KDN (98 and 137)
- Test on **Spiral** (check spiral_pdf) and Gaussian Parity
- RF, NN, KDN, KDF
- Weighting (increase the sample)
- (3,3,2) Without noise / with noise all the weighting schemes (FM and EFM)

Weighted KDNs

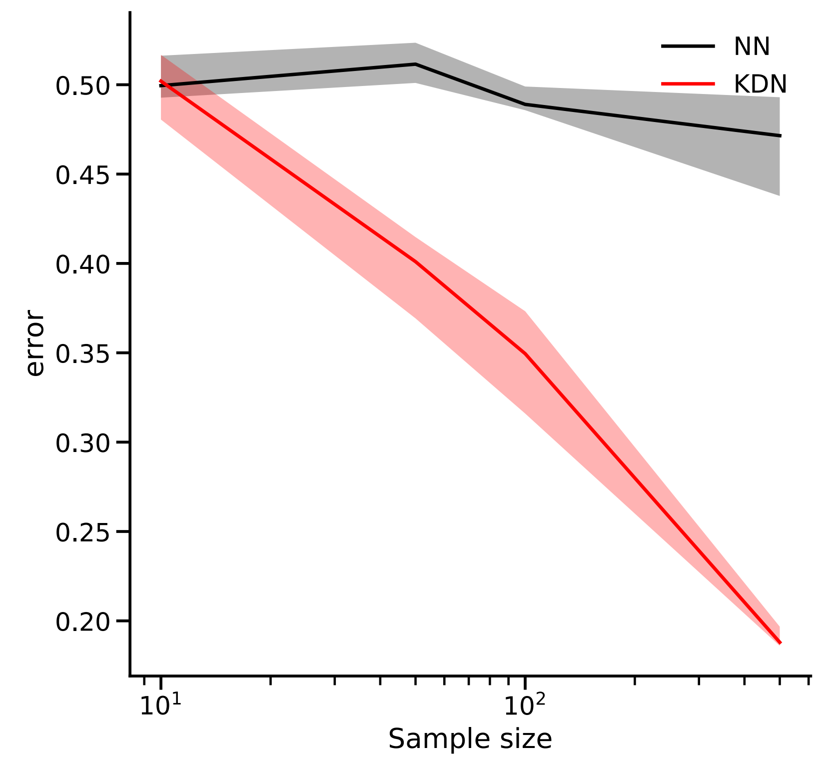
Evaluated on the Gaussian Sparse Parity Dataset (S3 No Noise)



(3, 3, 2)
No weighting
S3

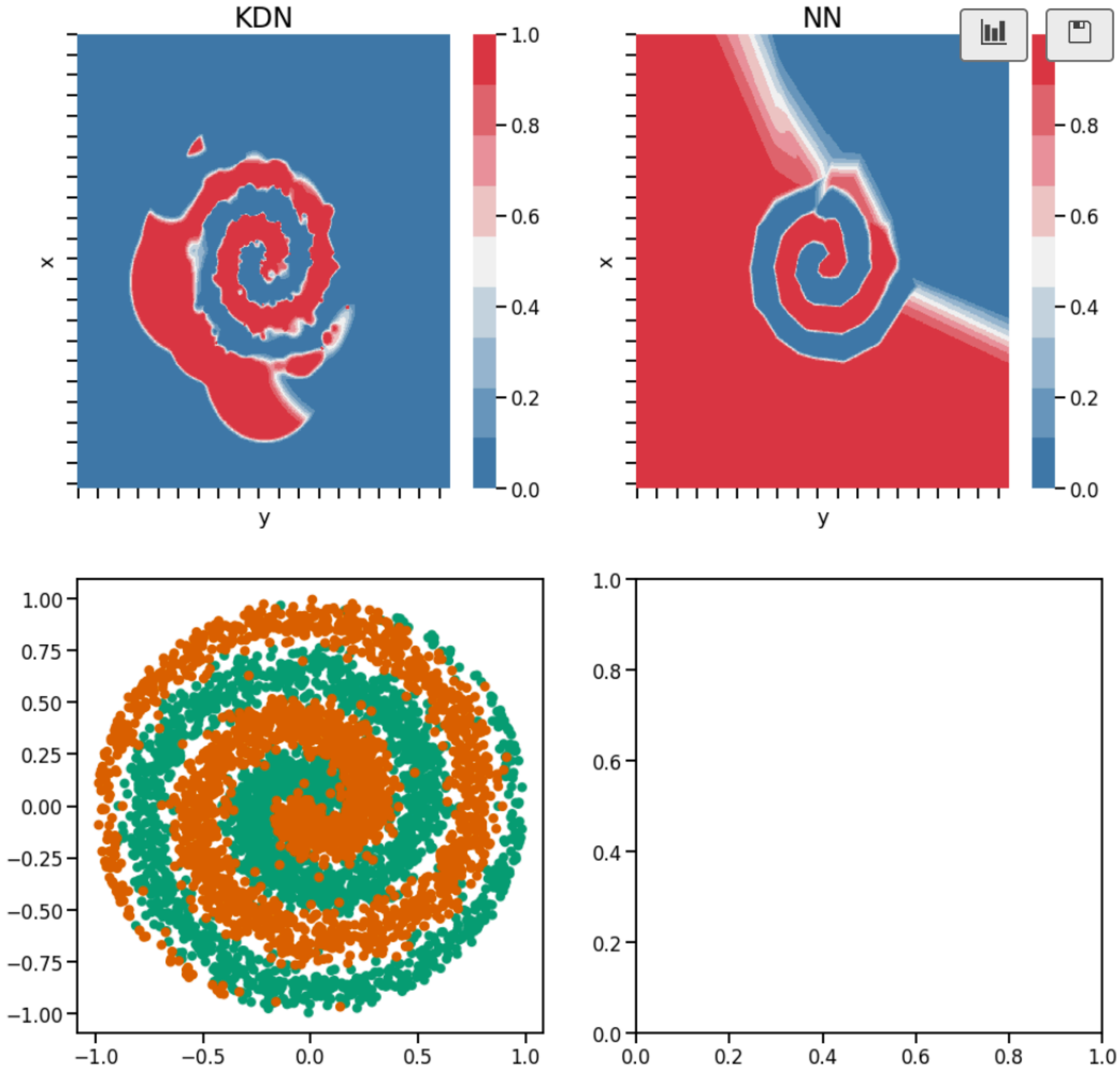


(5, 5, 2)
No weighting
S3

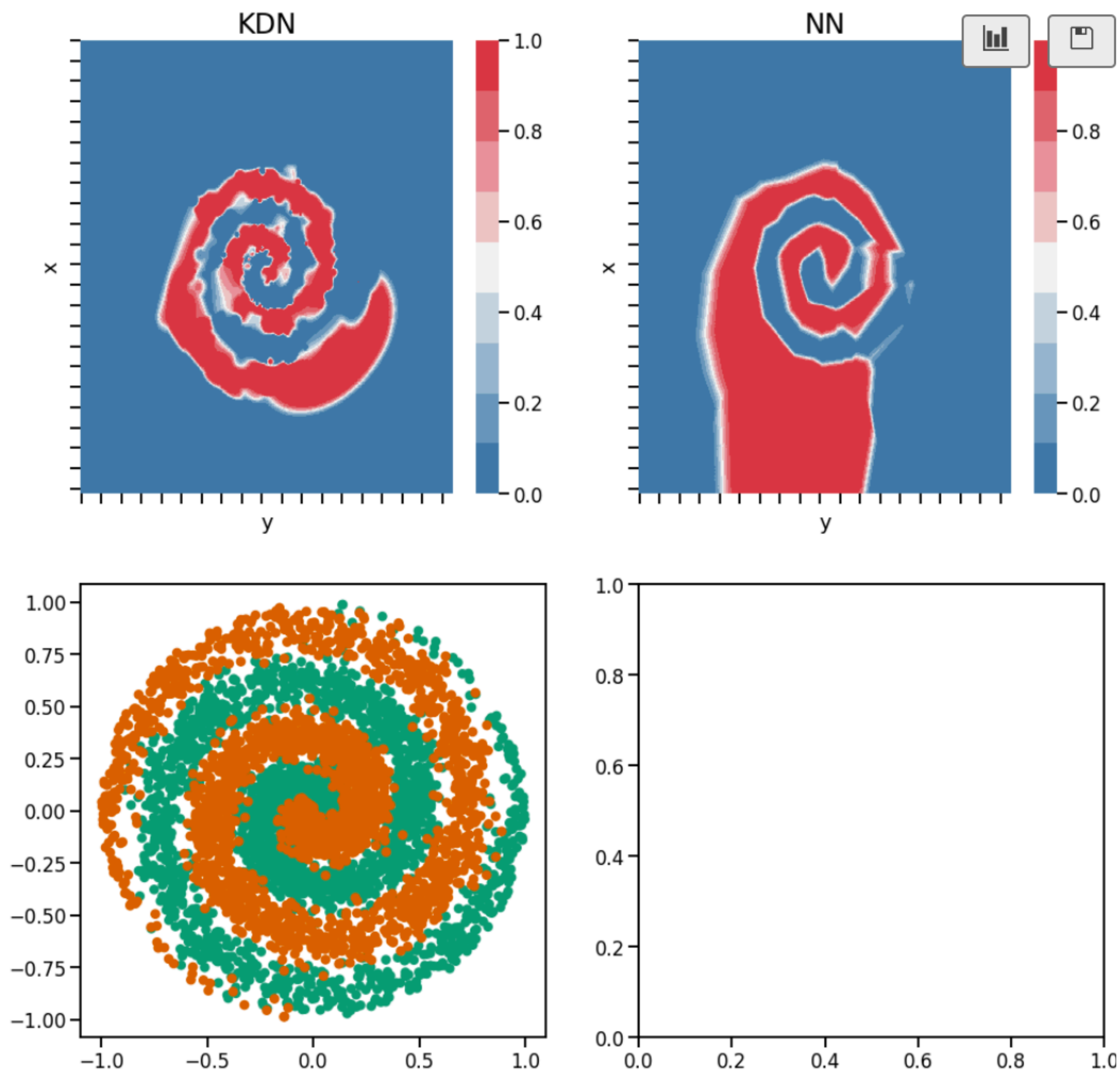


(3, 3, 2)
No weighting
S3 N3

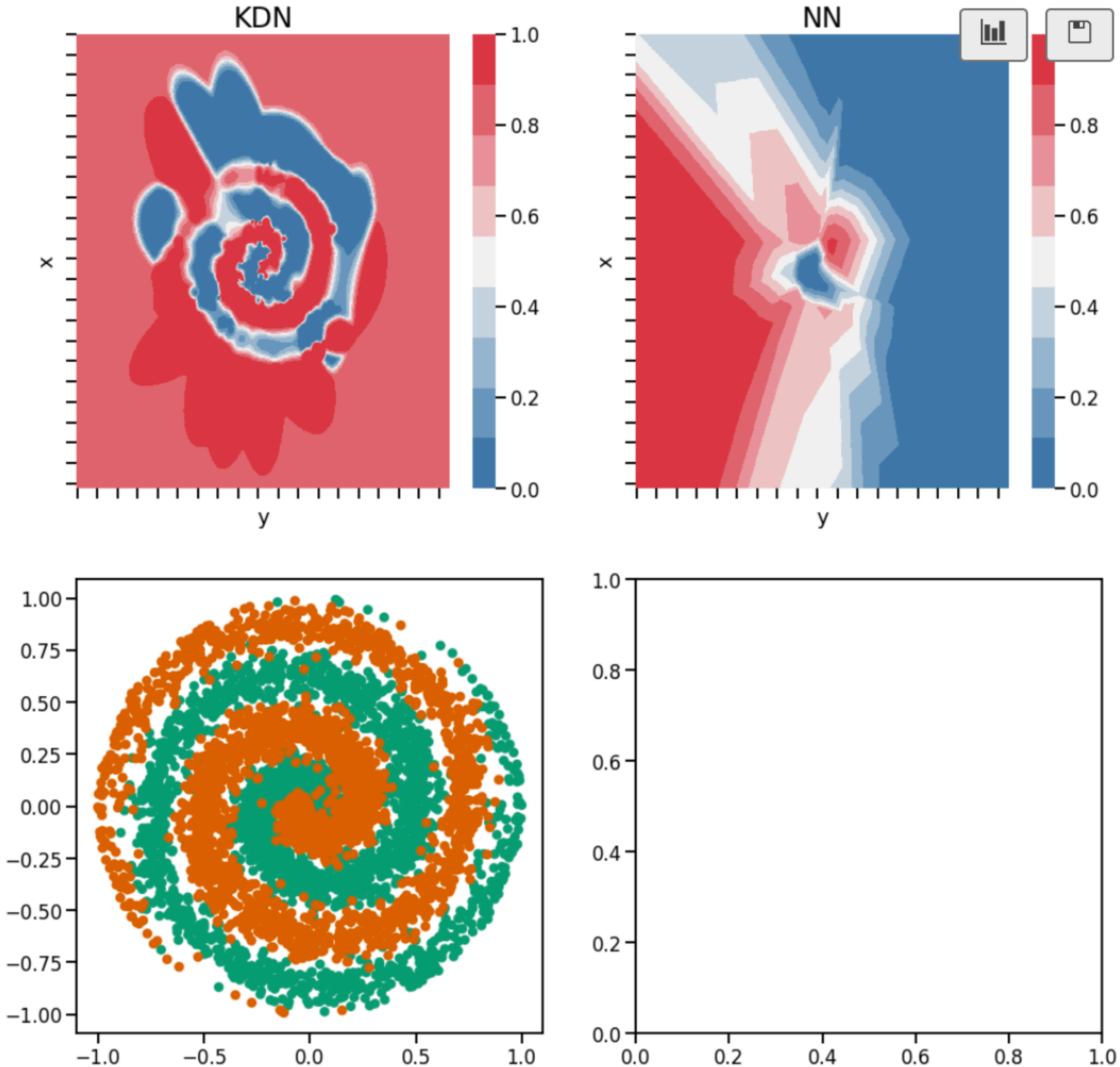
NN (10, 10, 10, 10, 2) + KDN
no weighting, no bias calc



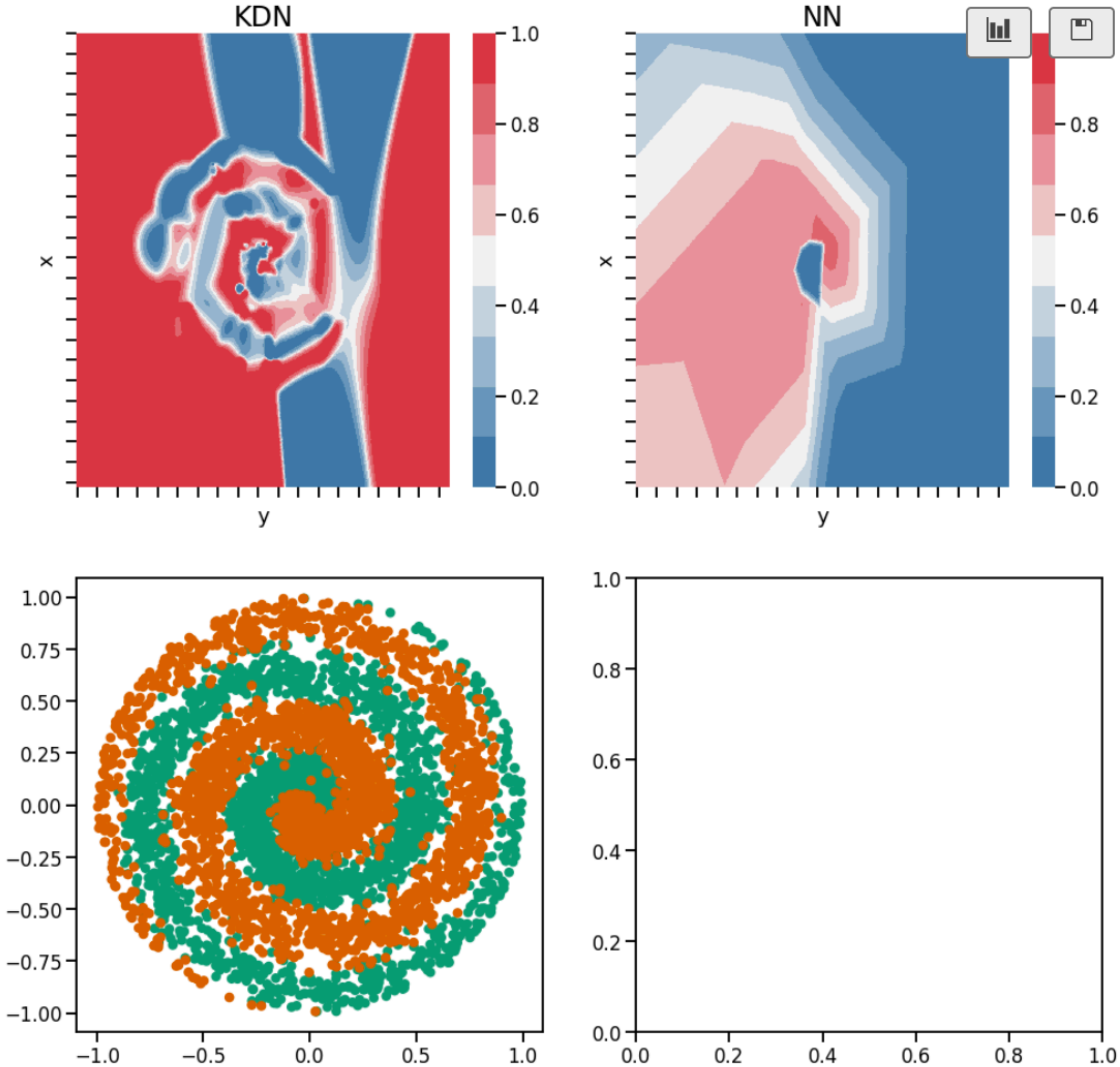
NN (10, 10, 10, 2) + KDN
no weighting, no bias calc



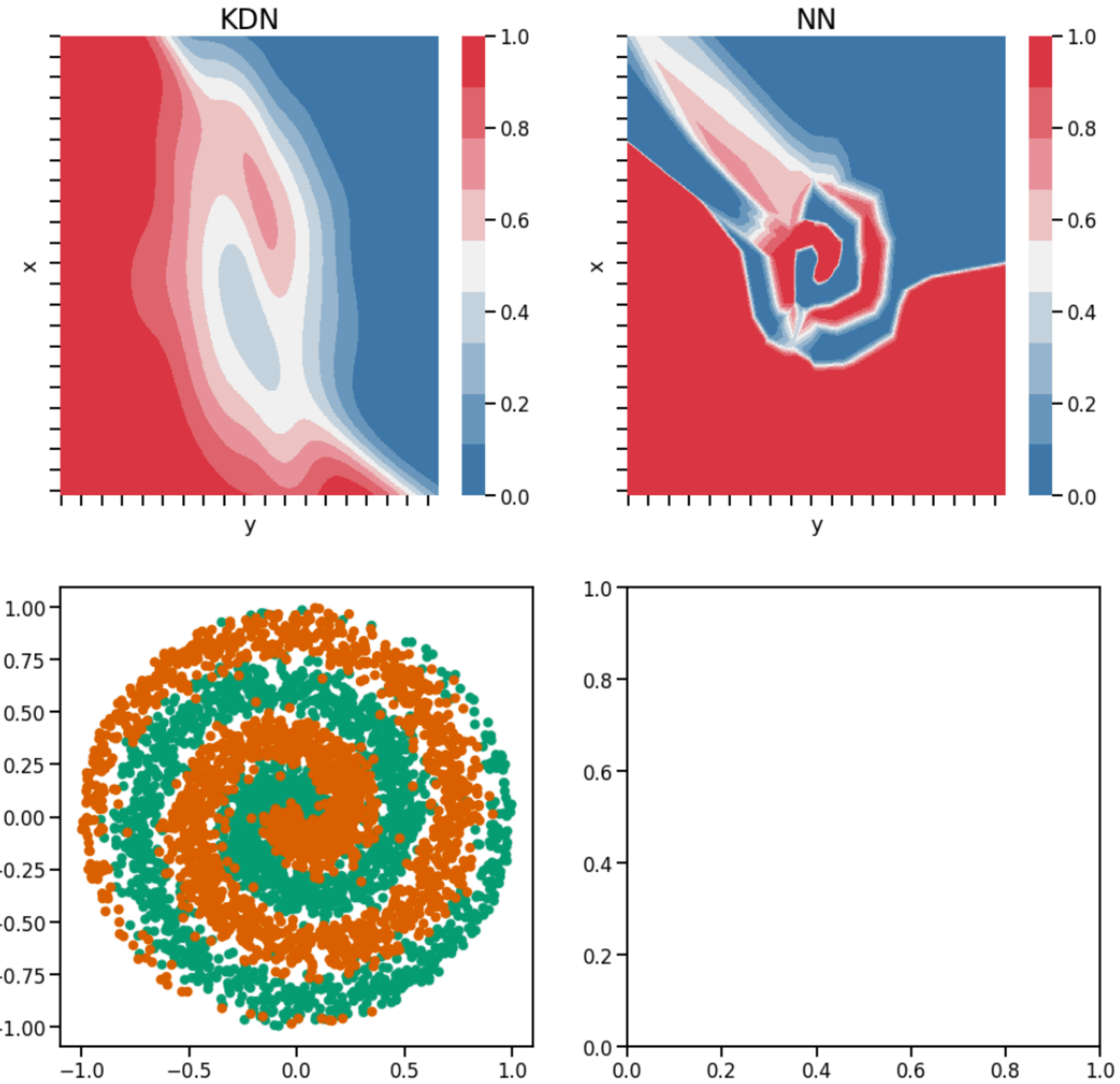
NN (10, 10, 2) + KDN
no weighting, no bias calc



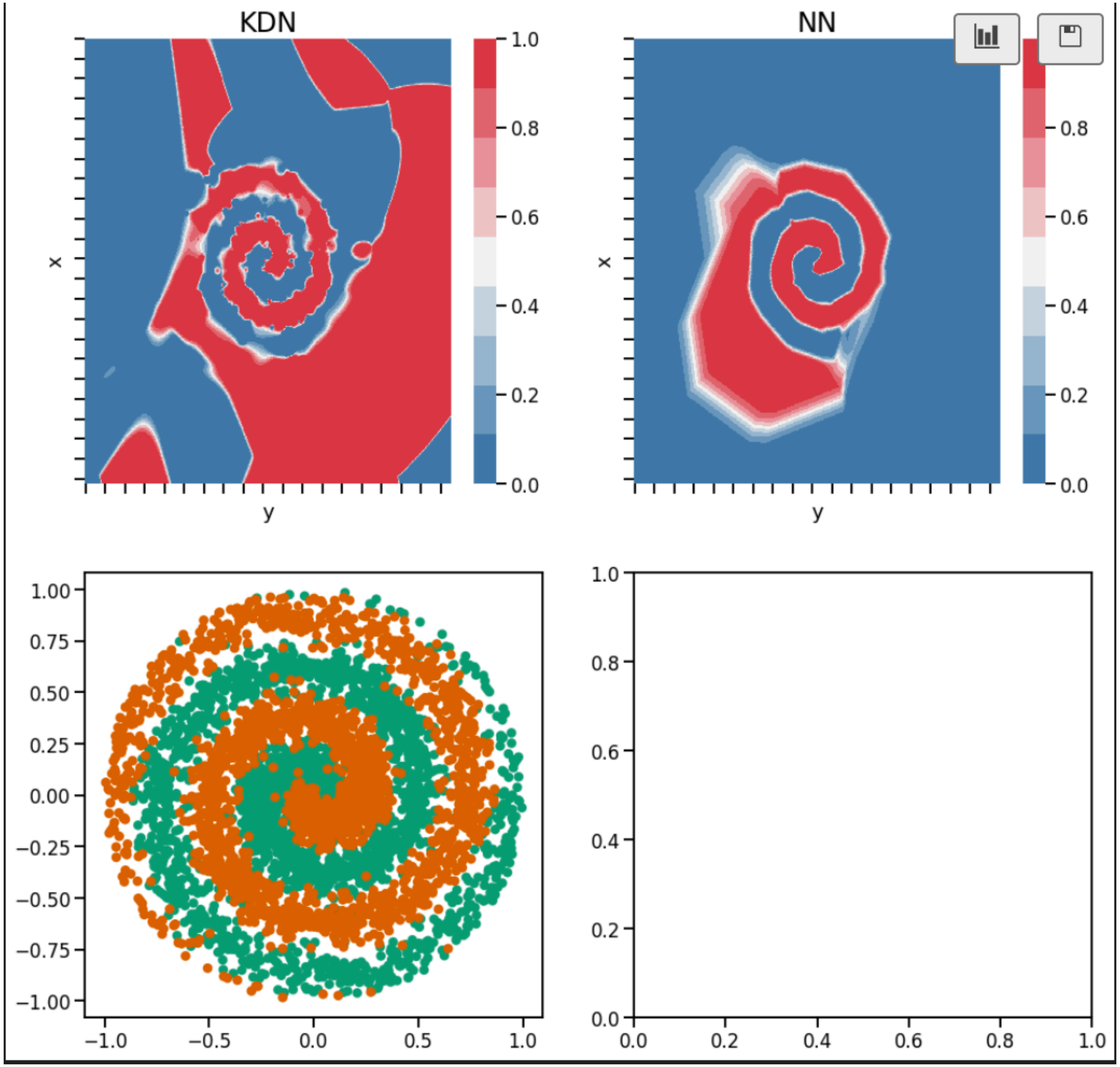
NN (5, 5, 2) + KDN
no weighting, no bias calc



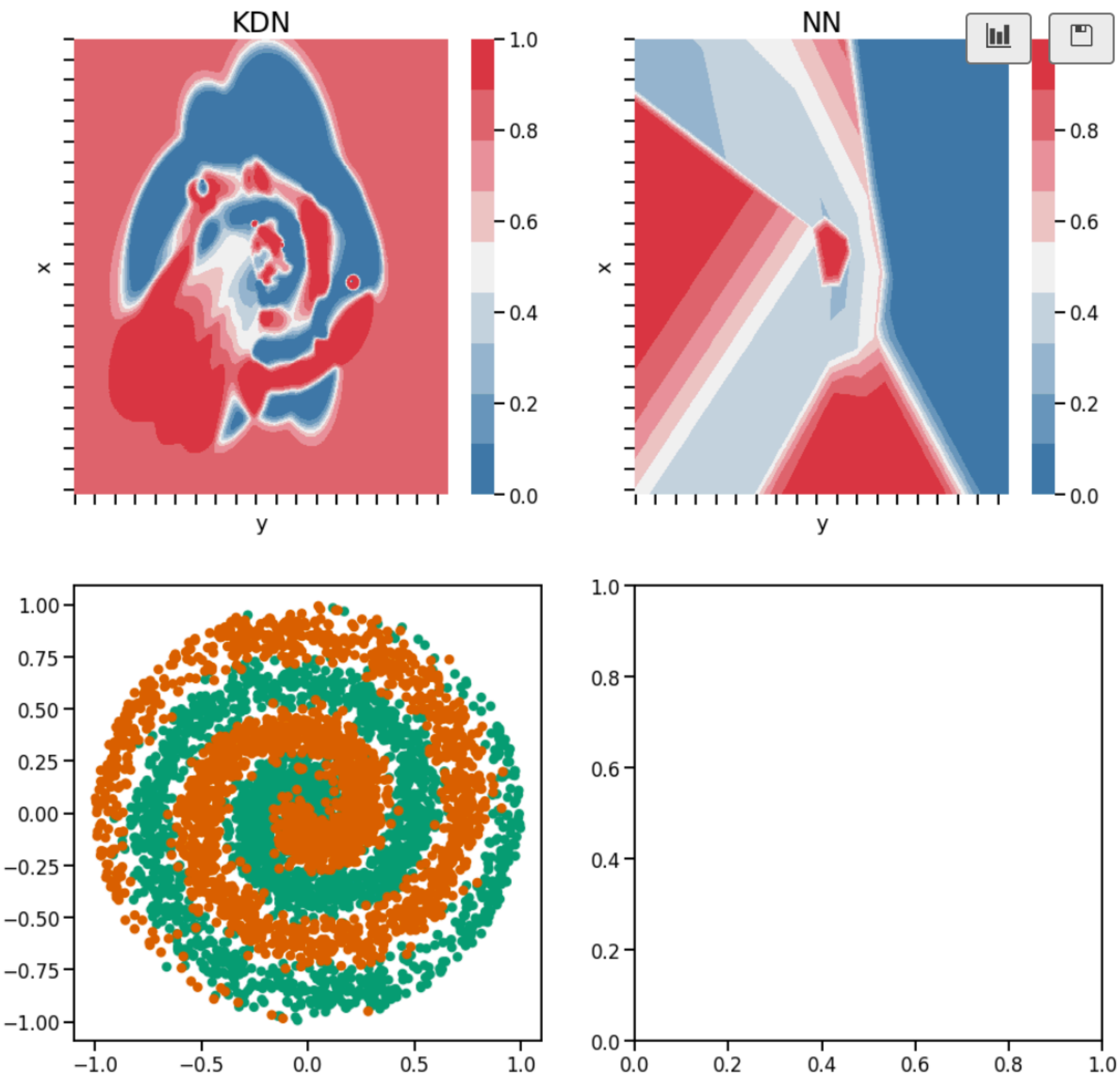
NN (10, 10, 10, 10, 2) + KDN
FM, no bias calc



NN (10, 10, 10, 10, 2) + KDN
No weighting, bias calc (k = 1/2.5)



NN (5, 5, 2) + KDN
No weighting, bias calc (k = 1/2.5)



NN (5, 5, 2) + KDN
FM, bias calc (k = 1/2.5)