Graph Constructions using Non-Negative Kernel (NNK) Regression

Ansh Khandelwal

Project Division

There were two main tasks and experimentations done for this project:

1. To get a better understanding of the NNK approach as compared to the KNN algorithm, both the algorithms have been implemented and experimented with 5 different synthetically generated datasets. Experiments are done to check how both the algorithms perform in terms of making a sparse graph for different datasets.

2. Implemented the NNK algorithm for image representation and compared with the bilateral filter output. Experiments have been done to check the sparsity, time taken for graph construction, and energy compaction of the graph-image representations.

Introduction: What is NNK?

NNk refers to as Non-negative kernels: Help to obtain neighborhoods that lead to better sparse representation. It is an approach for graph construction.

Algorithms used before for graph learning of mainly two types:

- Similarity based approaches (KNN), try to sparsify the graph based on similarity of nearby nodes (or pixels for images)
- Locality inducing approaches (local linear embedding), this method minimises: $\underset{\boldsymbol{\theta}:\ \boldsymbol{\theta}\geq 0}{\text{minimize}}\ ||\boldsymbol{x}_i - \boldsymbol{X}_S\boldsymbol{\theta}||_{l_2}^2$

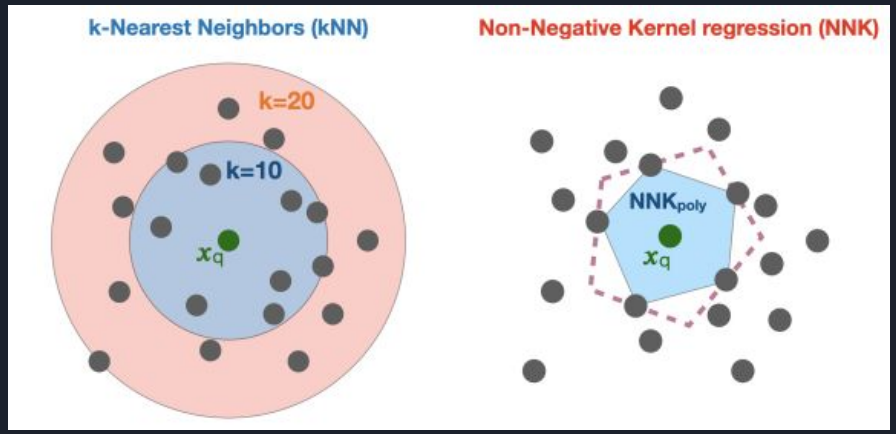
The NNK approach is different: optimization at each node→ $\min_{\boldsymbol{\theta}:\ \boldsymbol{\theta}\geq 0}\ ||\boldsymbol{\phi}_i - \boldsymbol{\Phi}_S\boldsymbol{\theta}||_{l_2}^2$

- Data points are transformed into a Reproducing Kernel Hilbert Space (RKHS) and the optimisation is done with the formula mentioned above.
- NNk problem can be written like:
- Where K(i,j) refers to a kernel trick = k(xi,xj)

$$\boldsymbol{\theta}_S = \underset{\boldsymbol{\theta}:\ \boldsymbol{\theta}\geq 0}{\arg\min}\ \frac{1}{2}\boldsymbol{\theta}^\top \boldsymbol{K}_{S,S}\boldsymbol{\theta} - \boldsymbol{K}_{S,i}^\top\boldsymbol{\theta}$$

The geometric difference with the new NNK method

Geometric difference in neighborhood definitions using k-nearest neighbors (kNN) and the proposed NNK approach with a Gaussian kernel. kNN selects neighbors based on the choice of k and can lead to the selection of neighbors that are geometrically redundant, i.e., the vector from the query to neighbors is almost collinear with other vectors joining the query and neighbors. In contrast, the proposed NNK definition selects only neighbors that correspond to non-overlapping directions.



Now look at it from image representation perspective! The polytope made by NNK will have a selection criteria which remove points along similar direction. Thus making a sparse graph.

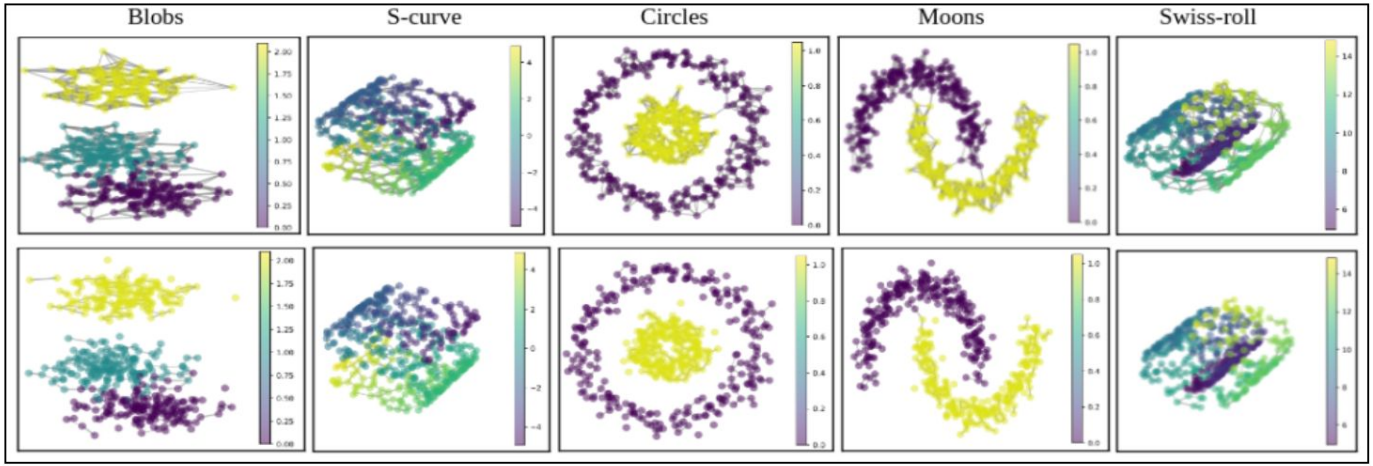
Experiment 1: NNK vs KNN

I have implemented the NNK and KNN algorithm for graph construction. The idea is to test both these algorithms for different types of synthetically generated datasets and determine whether the NNK algorithm generates a sparser graph. I will compare the number of edges these algorithms make during graph learning. The below comparison has been done for k=5 neighbours. The plots are made from 500 nodes for different synthetic datasets. Also, the experiments have been done for varying nodes from 50 to 1000, and the results have been tabulated. The datasets used are synthetically generated using the popular sklearn library of Python.

The datasets used are:

- Make_moons
- Make_blobs
- Make_circles
- Make_s_curve
- Make_swiss_roll.

Results



Number of nodes	Datasets and the corresponding number of edges formed									
	make_blobs		make_s_curve		make_circles		make_moons		make_swiss_roll	
	NNK	KNN	NNK	KNN	NNK	KNN	NNK	KNN	NNK	KNN
50	50	163	62	149	56	159	55	146	75	148
100	112	325	120	306	123	290	116	292	122	312
500	610	1563	631	1499	606	1558	571	1591	624	1531
1000	1253	3067	1238	2998	1203	3121	1202	3127	1271	2997

Observations from Experiment-1

- We see from the plots that the number of edges made by NNK is significantly lesser. It can be observed that the NNK algorithm is superior in terms of geometric representation (sparsity) from the KNN algorithm for all types of datasets.
- We can observe that edge to node ratio for KNN is approximately three times for the make_blobs dataset. The edge-to-node ratio is always higher for the KNN algorithm.
- We can also observe that edge to node ratio mostly remains consistent with an increase in the number of nodes for the KNN and NNK algorithms.
- Since we have been looking at the graph construction from a sparsity angle, the closer the edge-to-node ratio is to one, the better graph.
- Therefore, these experiments show that NNK is superior to the KNN algorithm for making sparser graphs.

Experiment 2: Image representation using NNK graphs

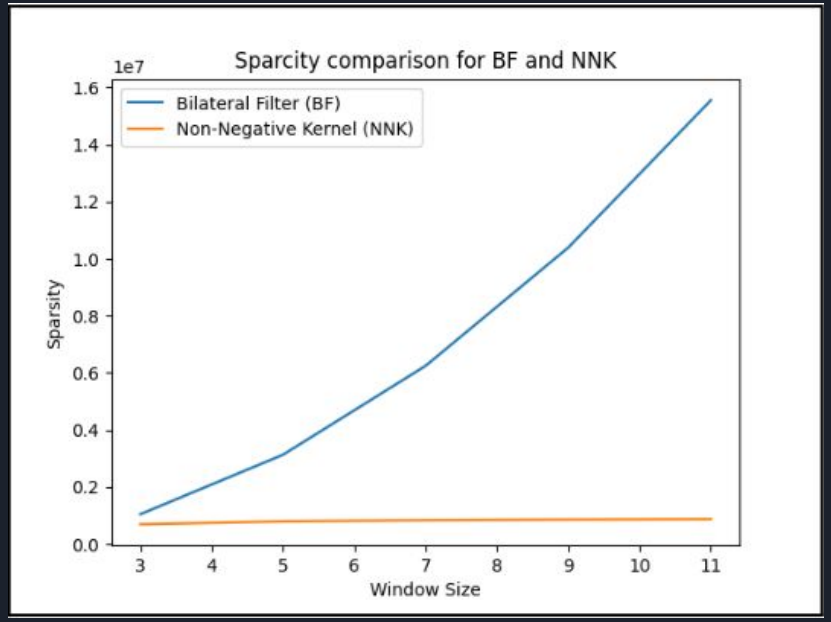
One can represent images as graphs and pixel intensity as graph signal. Each pixel is a node and edge weight is referred to as the similarity between pixel intensities. One approach to do this is to use window-based filters like bilateral filters, where each centre pixel is connected to its neighbouring pixels. The bilateral filter kernel is as follows:

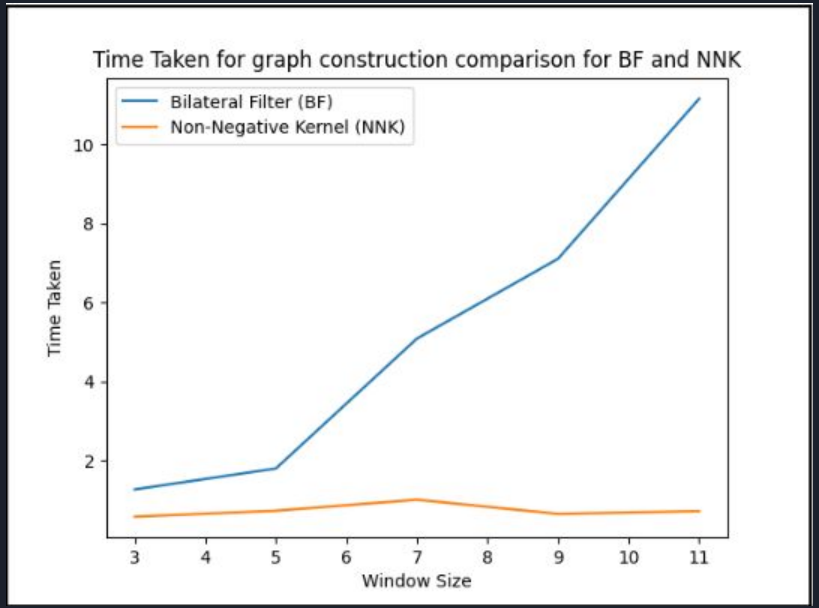
$$K_{i,j} = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\sigma_d^2}\right) \exp\left(-\frac{\|\boldsymbol{f}_i - \boldsymbol{f}_j\|^2}{2\sigma_f^2}\right) \qquad \begin{array}{ll} \boldsymbol{x}_i: & \text{pixel position} \\ \boldsymbol{f}_i: & \text{pixel intensity} \end{array}$$

However, the issue with this method is that the sparsity of the image graph is only dependent on window size w. We can use NNK approach as it is robust to hyperparameters, and edges are defined by the local geometry of the data. Thus, in contrast to KNN graphs, even when window size w increases, the number of connected nodes does not necessarily grow, so NNK graphs tend to reflect the actual data topology better. To check how the change in hyperparameter w (window size) changes the graph construction via bilateral filters and NNK graphs, I tested both algorithms on an image by varying the filter size. The three different evaluation parameters are as follows:

1. Sparsity comparison
2. Time taken for graph construction
3. Energy compaction of images (variance of the wavelet signals act as an indicator of information content corresponding to the frequency band of the wavelet)

Results-1

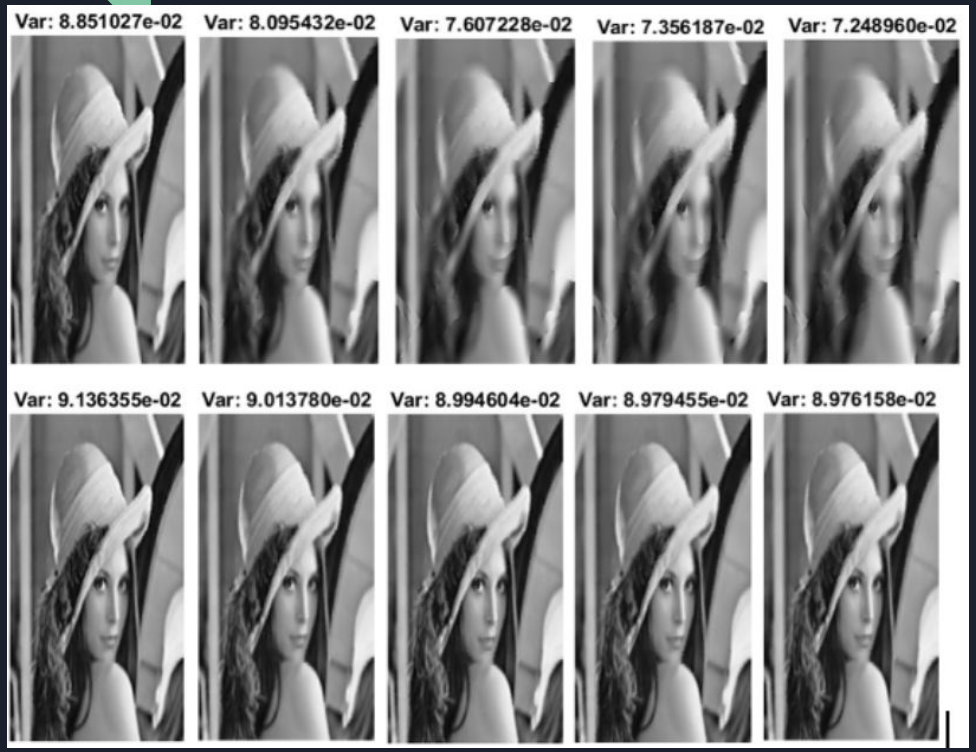

Sparcity comparison for BF and NNK
1e7
Bilateral Filter (BF)
Non-Negative Kernel (NNK)
Sparsity
Window Size
0.0
0.2
0.4
0.6
0.8
1.0
1.2
1.4
1.6
3
4
5
6
7
8
9
10
11


Time Taken for graph construction comparison for BF and NNK
Bilateral Filter (BF)
Non-Negative Kernel (NNK)
Time Taken
Window Size
2
4
6
8
10
3
4
5
6
7
8
9
10
11

Observations-1

- We can clearly observe that the sparsity value increases (almost linearly) with an increase in window size for bilateral filters.
- For the NNK algorithm, sparsity is mostly consistent. The low value of sparsity helps with several downstream tasks, as it helps in efficiently representing the images in space.
- We also see an increase in time for graph construction with an increase in window size for bilateral filters.
- But the time taken remains the same for the NNK algorithm for graph (sgwt) construction due to its dependence on pixel intensity and geometric variability of pixels. NNK approach doesn’t depend on window size.

Result and Observation-2



(Top: Bilateral Filter Graph vs Bottom: Proposed NNK Graph Construction)

The experiment: the window size is increased from left to right (from 3X3 to 11X11) for both approaches. The idea is to evaluate image representation by calculating the variance of the wavelet signals as an indicator of information content corresponding to the frequency band of the wavelet. The more the variance, the more the energy of the image captured by the approaches will be.

Observations:

- We can observe that the NNK approach captures more energy for all the window sizes than the BF method.
- We can also see the variance decreases with an increase in window size. This is intuitive as more neighbouring nodes connected will affect the image representation and make it more distorted.

Thanks!