

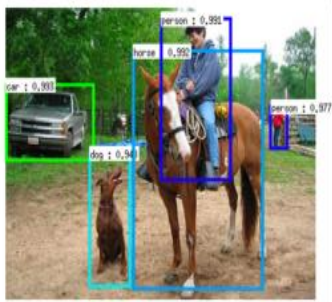
Tutorial on Graph Neural Networks for Computer Vision

AAAI 2020

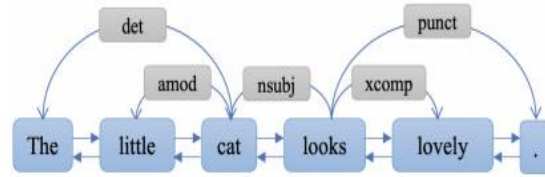
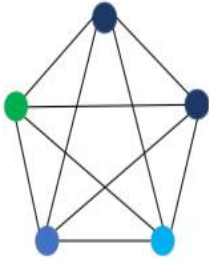
TRÌNH BÀY: LŨ MẠNH HÙNG

Contents

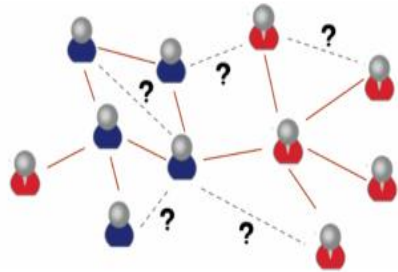
1. Why are graphs useful?
2. Why is it difficult to define convolution on graphs?
3. Convolution on images in terms of graphs
4. Conclusion



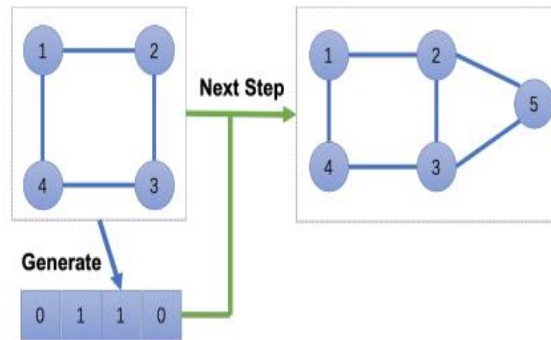
(c) image



(d) text



(e) social network

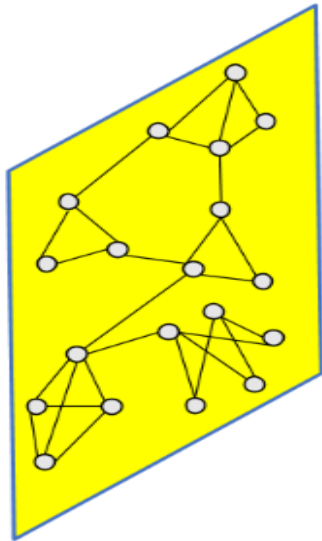
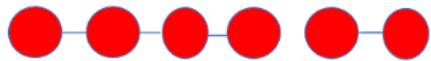
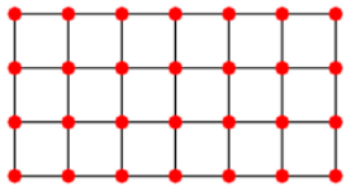


(f) generation

What is graph?

- Set of nodes (vertices)
- Connected by directed/undirected edges
- In many practical cases, it is actually **you** who gets to decide what are the nodes and edges in a graph.

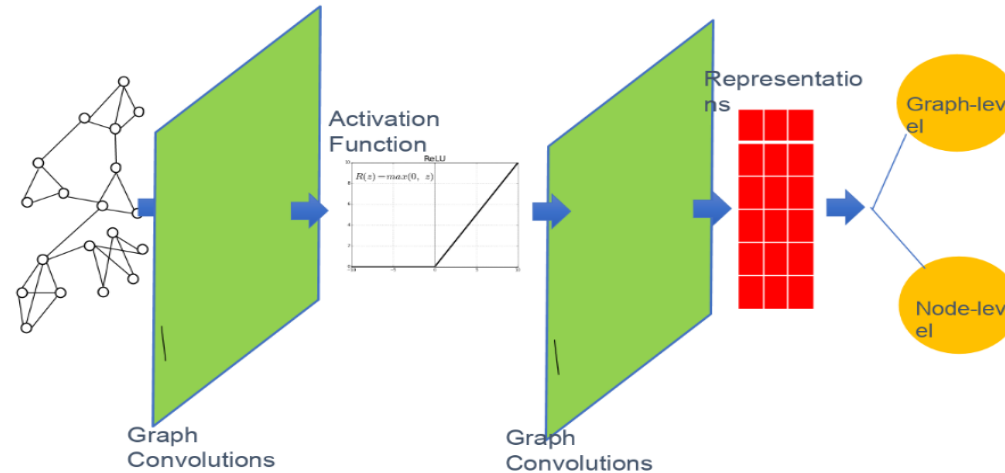
Why graphs can be useful?



- Traditional DL is designed for simple grids or sequences:
 - CNNs for fixed-size images/grids
 - RNNs for text/sequences
- But nodes on graphs have different connections:
 - Arbitrary neighbor size
 - Complex topological structure
 - No fixed node ordering

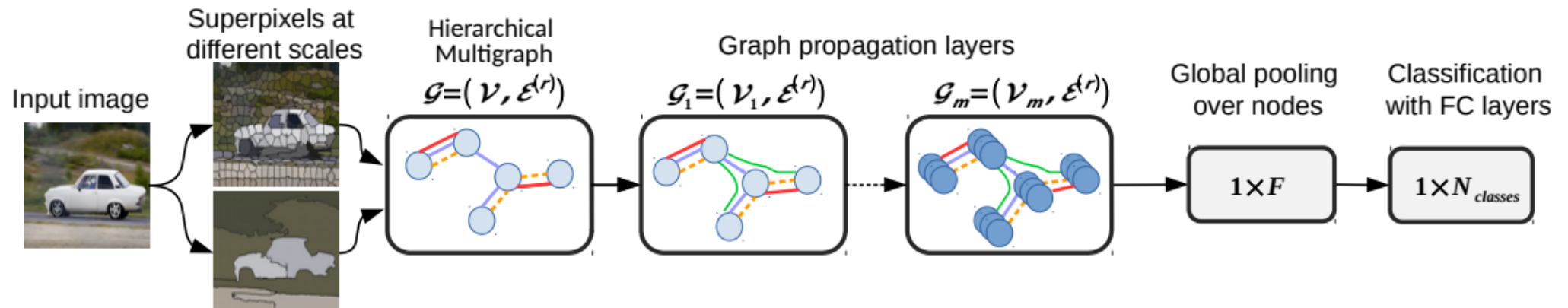
Why graphs can be useful?

- Neural network itself can be viewed as a graph
 - Nodes are neurons and edges are weights
 - Nodes are layers and edges denote flow of forward/backward pass
- Representing your data as graph(s) gives you a lot of flexibility
- Give you a very different and interesting perspective on your problem



Why graphs can be useful?

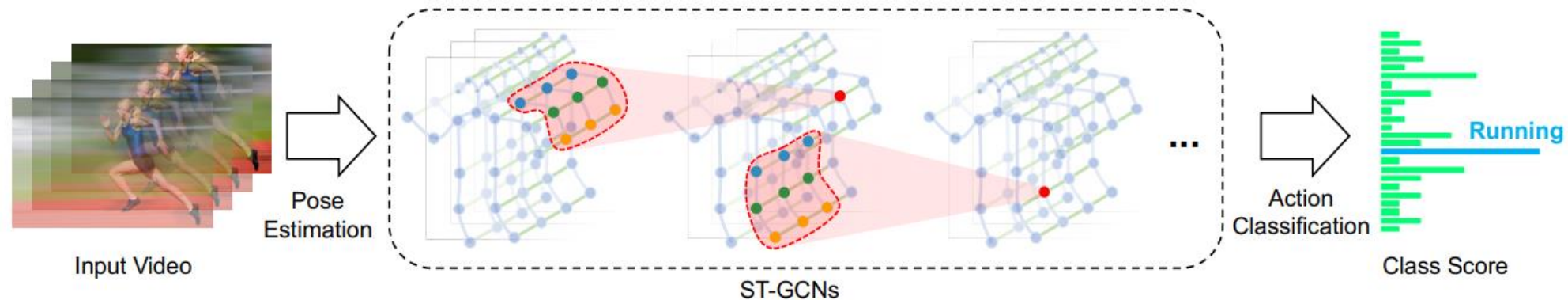
➤ Data can be viewed as graphs



Instead of learning from image pixels you can learn from “[superpixels](#)”

Why graphs can be useful?

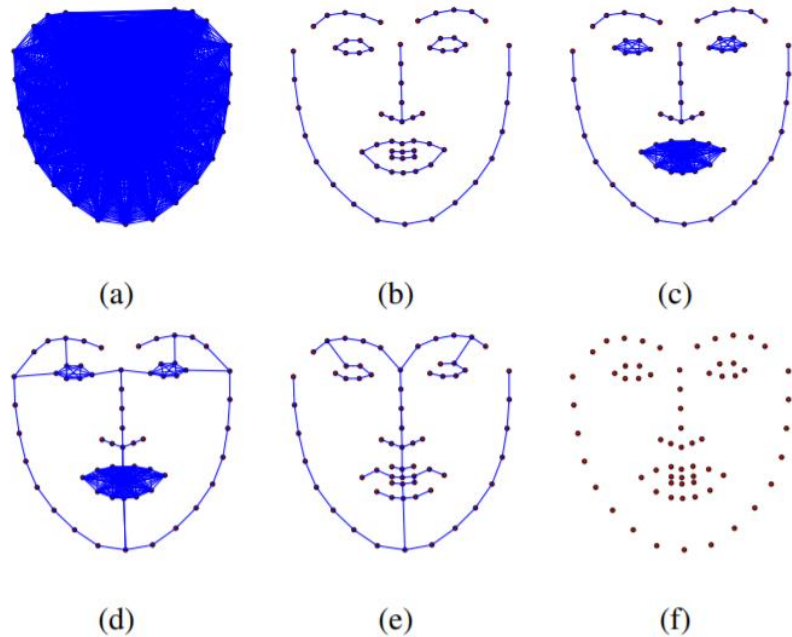
➤ Data can be viewed as graphs



About a human pose, your relational bias can be a graph of skeleton joints of a human body

Why graphs can be useful?

➡ Data can be viewed as graphs



Another example can be representing facial landmarks as a graph

Figure 2: Employed graph structures. (a) Complete graph. (b) Chain per area. (c) Chain and complete per area. (d) Chain and complete per area with connections between them. (e) Minimum spanning tree. (f) Empty graph.

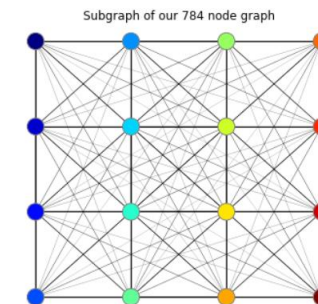
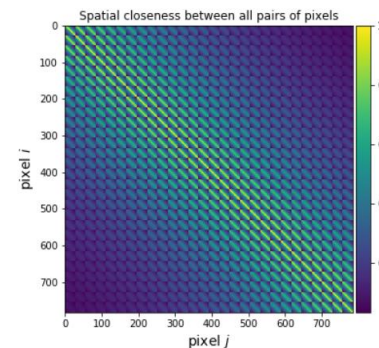
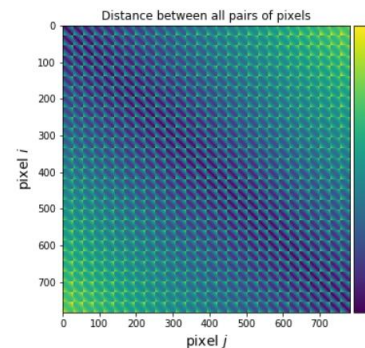
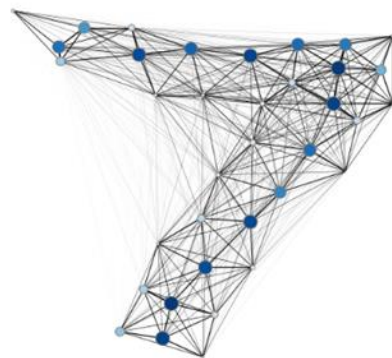
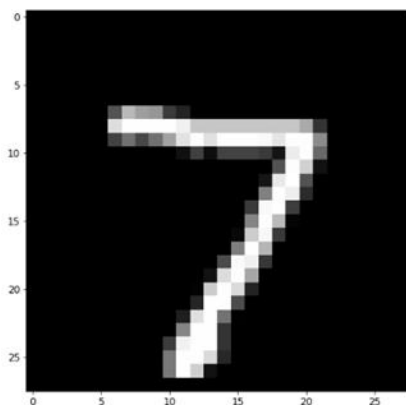
Why is it difficult to define convolution on graphs?



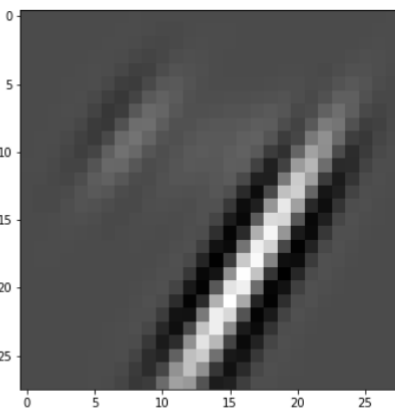
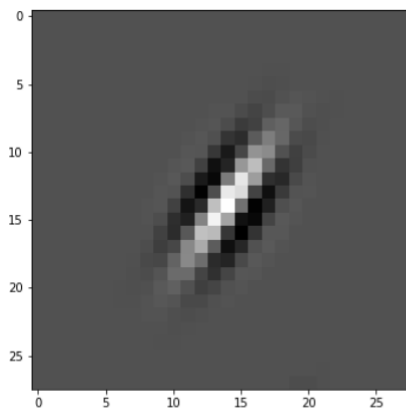
- ConvNets exploit a natural prior in images:
 - Shift-invariance
 - Locality
 - Compositionality (or hierarchy)
- ConvNets model is parametric
- Transfer all these properties to graph neural networks (GNNs)

Convolution on images in terms of graphs

➤ Image to graph



Adjacency matrix (N x N) in the form of distances (left) and closeness (middle) between all pairs of nodes. (right) A subgraph with 16 neighboring pixels corresponding to the adjacency matrix in the middle. Since it's a complete subgraph, it's also called a "clique".



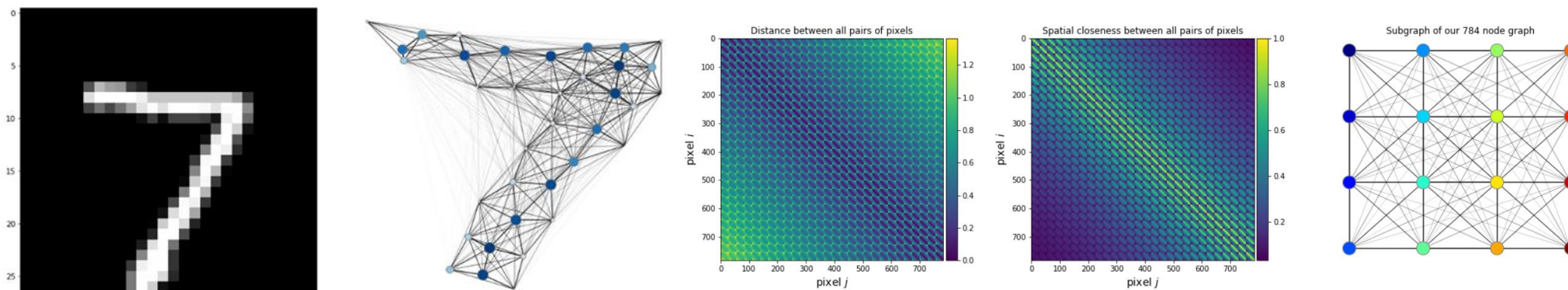
A 28x28 filter (left) and the result of 2D convolution of this filter with the image of digit 7 (right).

```
import numpy as np
from scipy.spatial.distance import cdist

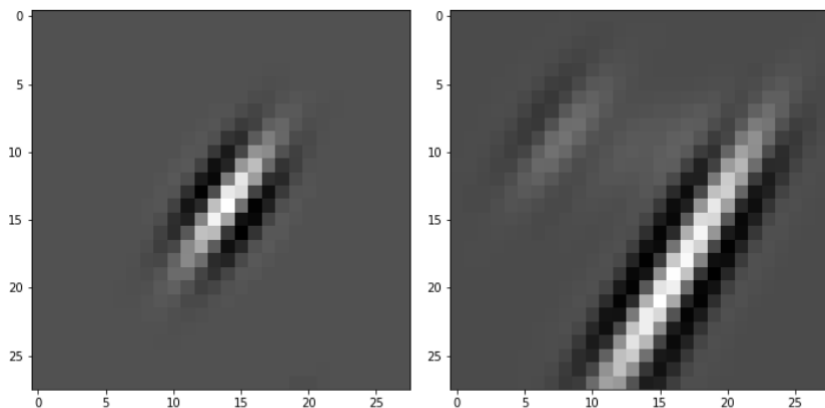
img_size = 28 # MNIST image width and height
col, row = np.meshgrid(np.arange(img_size), np.arange(img_size))
coord = np.stack((col, row), axis=2).reshape(-1, 2) / img_size
dist = cdist(coord, coord) # see figure below on the left
sigma = 0.2 * np.pi # width of a Gaussian
A = np.exp(- dist / sigma ** 2) # see figure below in the middle
```

Convolution on images in terms of graphs

➤ Image to graph



Adjacency matrix ($N \times N$) in the form of distances (left) and closeness (middle) between all pairs of nodes. (right) A subgraph with 16 neighboring pixels corresponding to the adjacency matrix in the middle. Since it's a complete subgraph, it's also called a "clique".

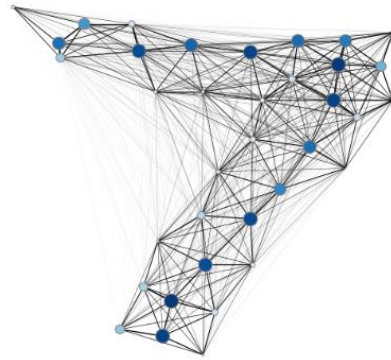
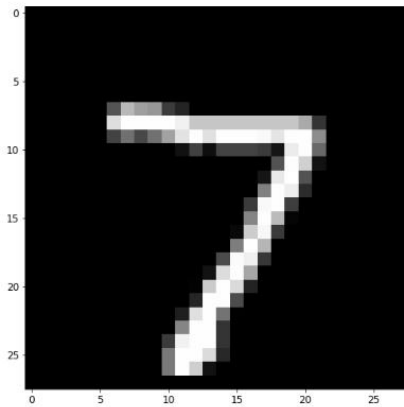


A 28x28 filter (left) and the result of 2D convolution of this filter with the image of digit 7 (right).

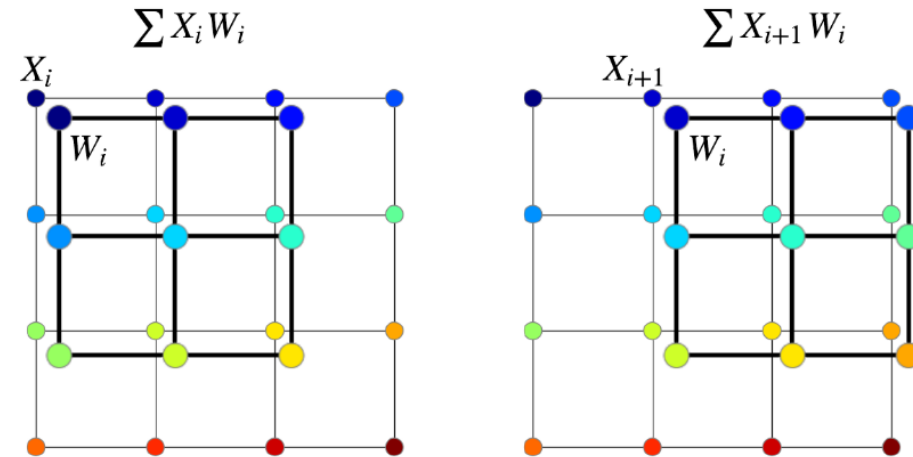
Instead of using a predefined filter, we can learn to predict an edge between any pair of pixels

Convolution on images in terms of graphs

➤ Aggregator operators



- Graph G is going to :
 - Have $N=784$ (28×28) nodes
 - Edges will have large values for closely located pixels
 - Small values for remote pixels.



- Dot product used above to compute convolution at each step is sensitive to the order

Convolution on images in terms of graphs

➤ Aggregator operators

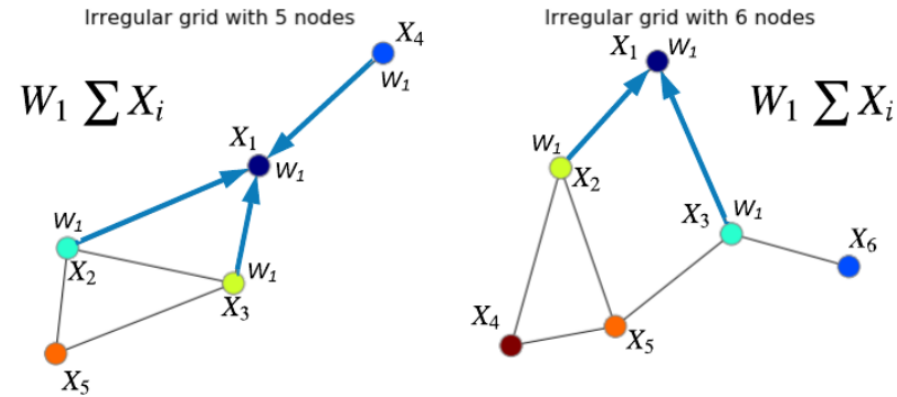
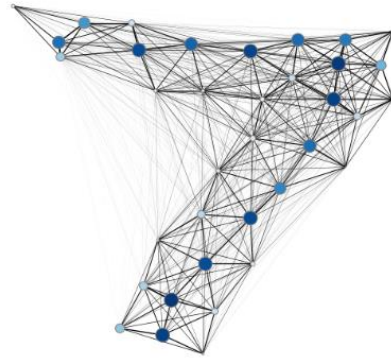
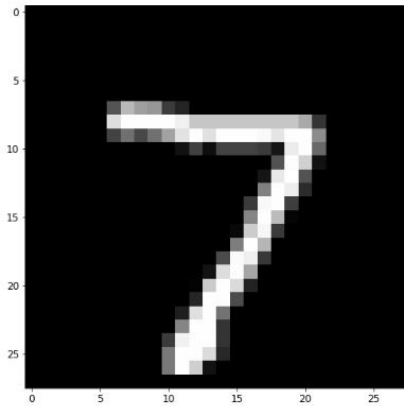
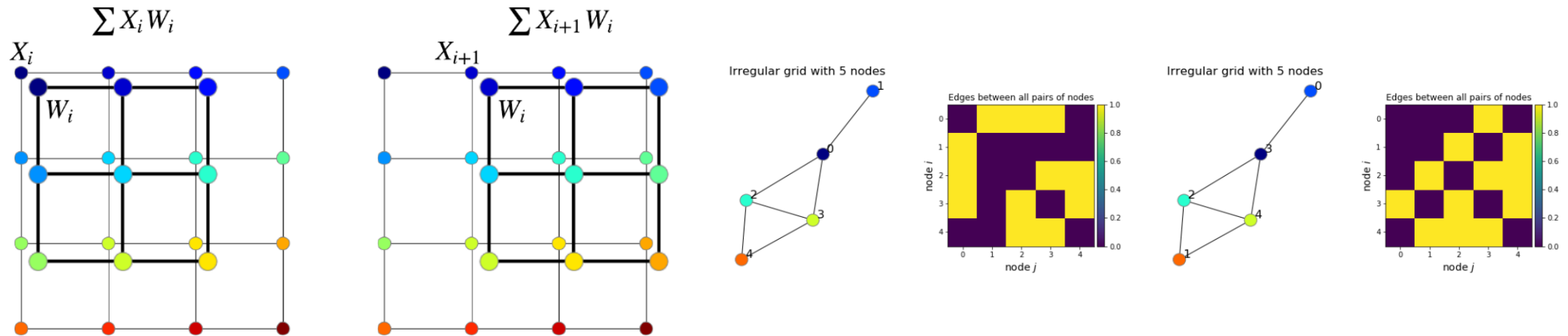


Illustration of “convolution on graphs” of node features X with filter W centered at node 1 (dark blue).

- Nodes are a set, and any permutation of this set does not change it
- Therefore, the aggregator operator should be permutation-invariant.
- The core idea behind GNNs is aggregation over “neighbors”

Convolution on images in terms of graphs

➤ Convolution



$$X^{(l+1)} = \mathcal{A} X^{(l)} W^{(l)}$$

Conclusion

- GNN are a very flexible and interesting that can be applied to complex data **at a certain cost**
- We should learn to predict an edge between any pair of pixels
- It is the difficulty of regularizing the model by defining such operators as convolution

The next week

