Yandex
SCHOOL OF DATA ANALYSIS

Neural Networks for Network-like structures

2018-08-09, Oxford

Andrey Ustyuzhanin

NRU HSE

YSDA

ICL

Common examples so far

- Plain features (tables)
- Image
- Text
- Time series

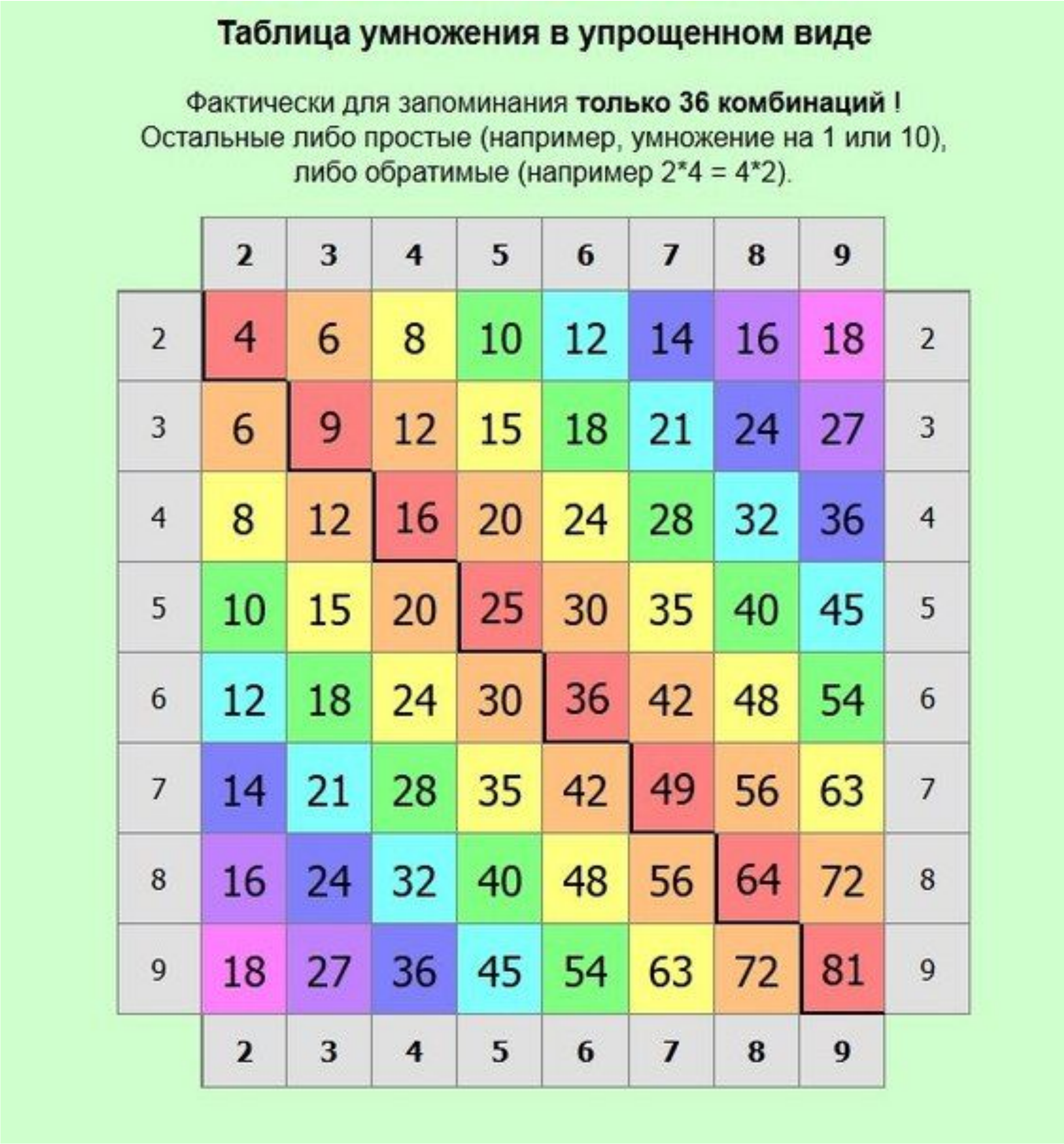


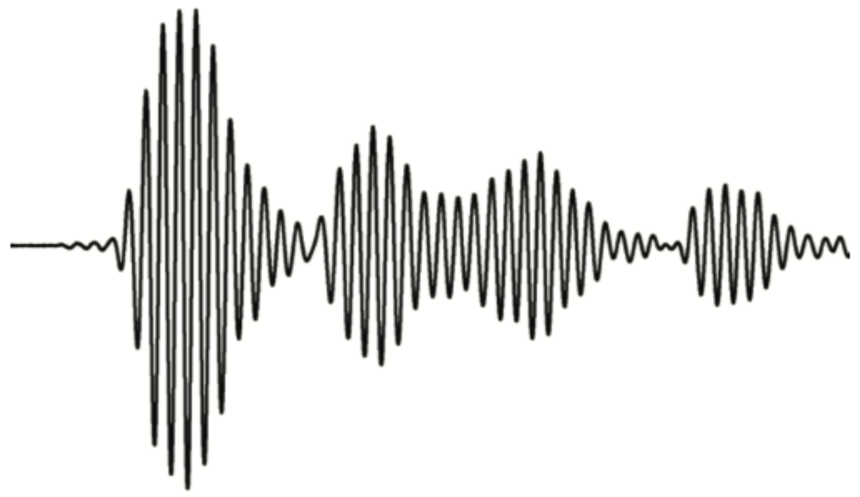
Таблица умножения в упрощенном виде

Фактически для запоминания **только 36 комбинаций** !
Остальные либо простые (например, умножение на 1 или 10),
либо обратимые (например 2*4 = 4*2).

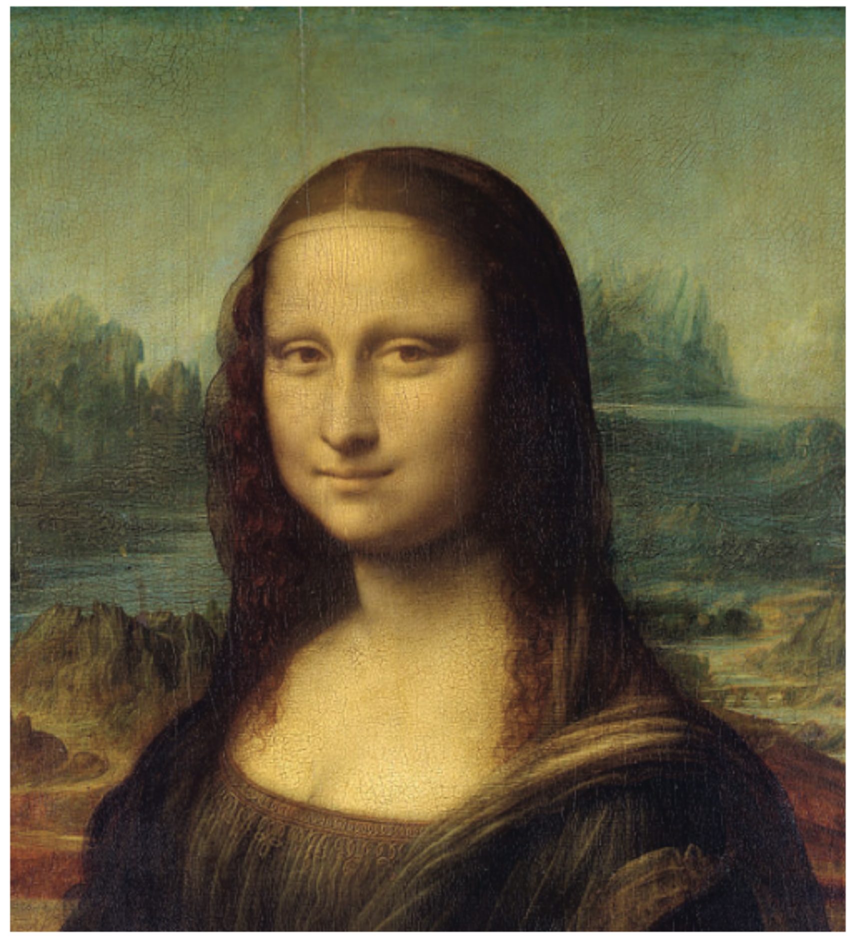
	2	3	4	5	6	7	8	9	
2	4	6	8	10	12	14	16	18	2
3	6	9	12	15	18	21	24	27	3
4	8	12	16	20	24	28	32	36	4
5	10	15	20	25	30	35	40	45	5
6	12	18	24	30	36	42	48	54	6
7	14	21	28	35	42	49	56	63	7
8	16	24	32	40	48	56	64	72	8
9	18	27	36	45	54	63	72	81	9
	2	3	4	5	6	7	8	9	

Doubt thou the stars are fire;
Doubt that the sun doth move;
Doubt truth to be a liar;
But never doubt I love...

Text



Audio signals

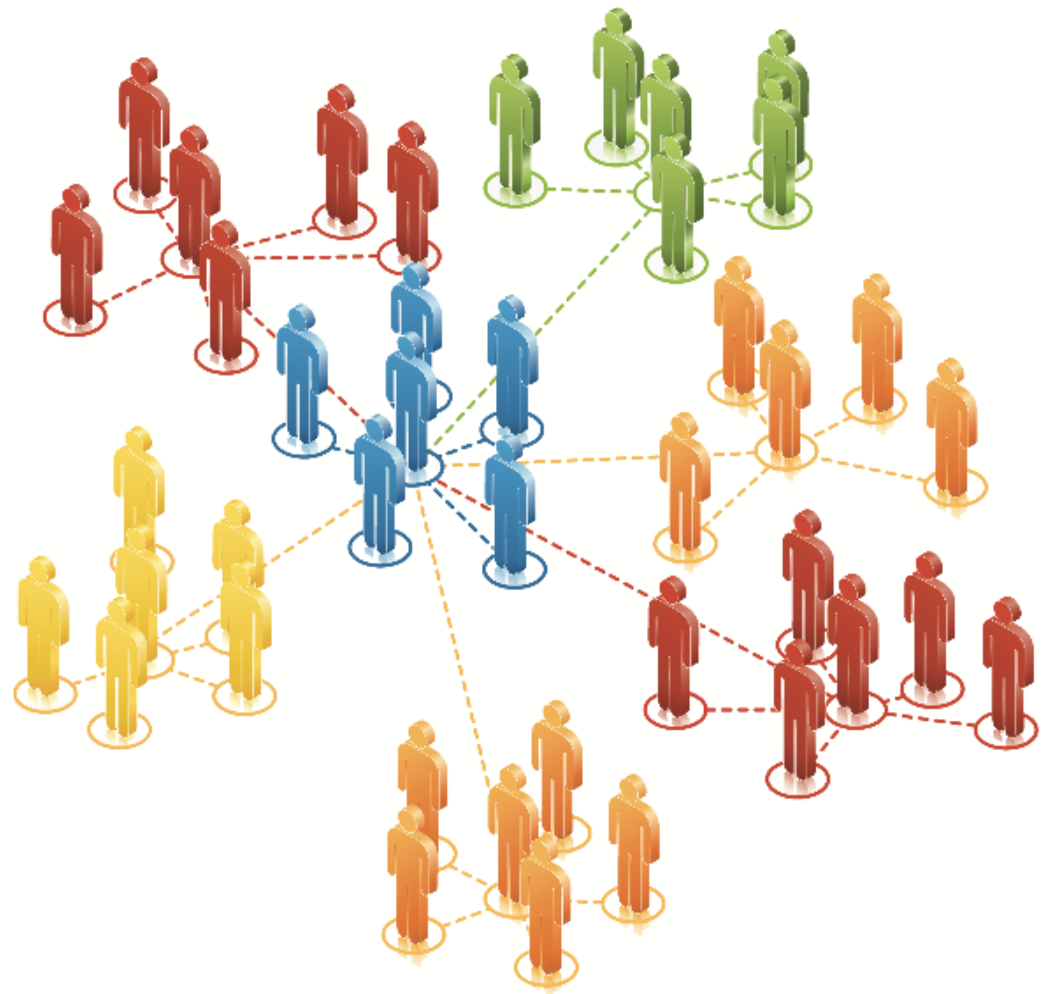


Images

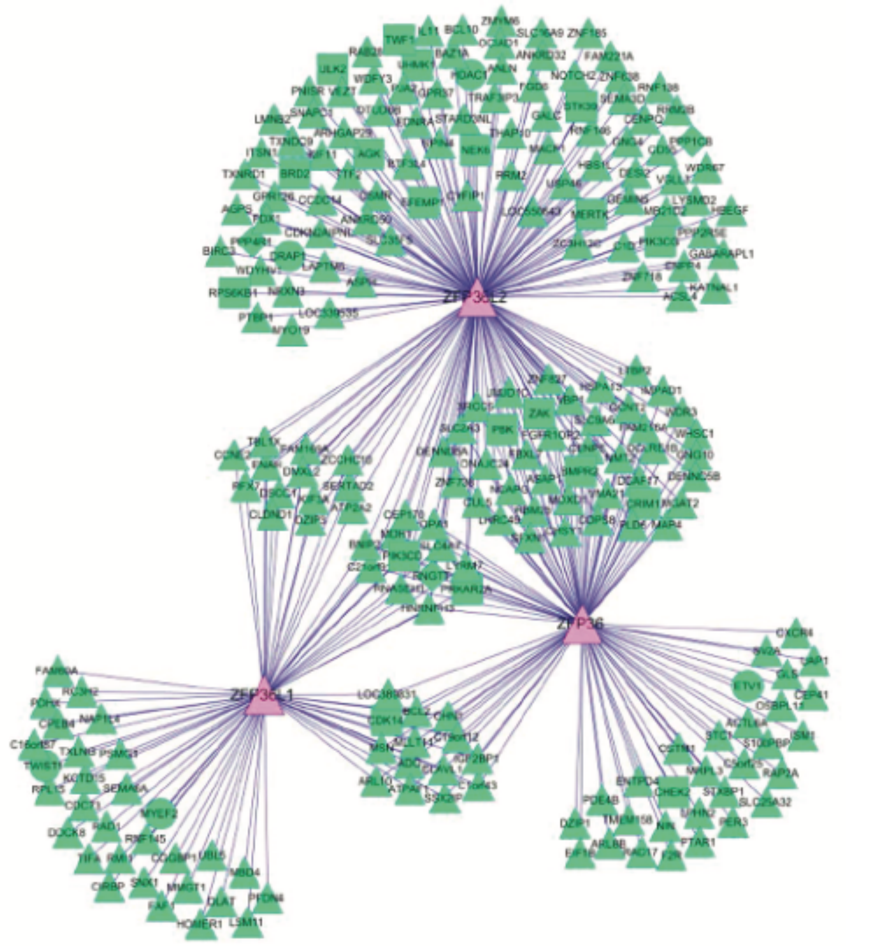
What about those?

- Geometry
- Manifolds
- Graphs

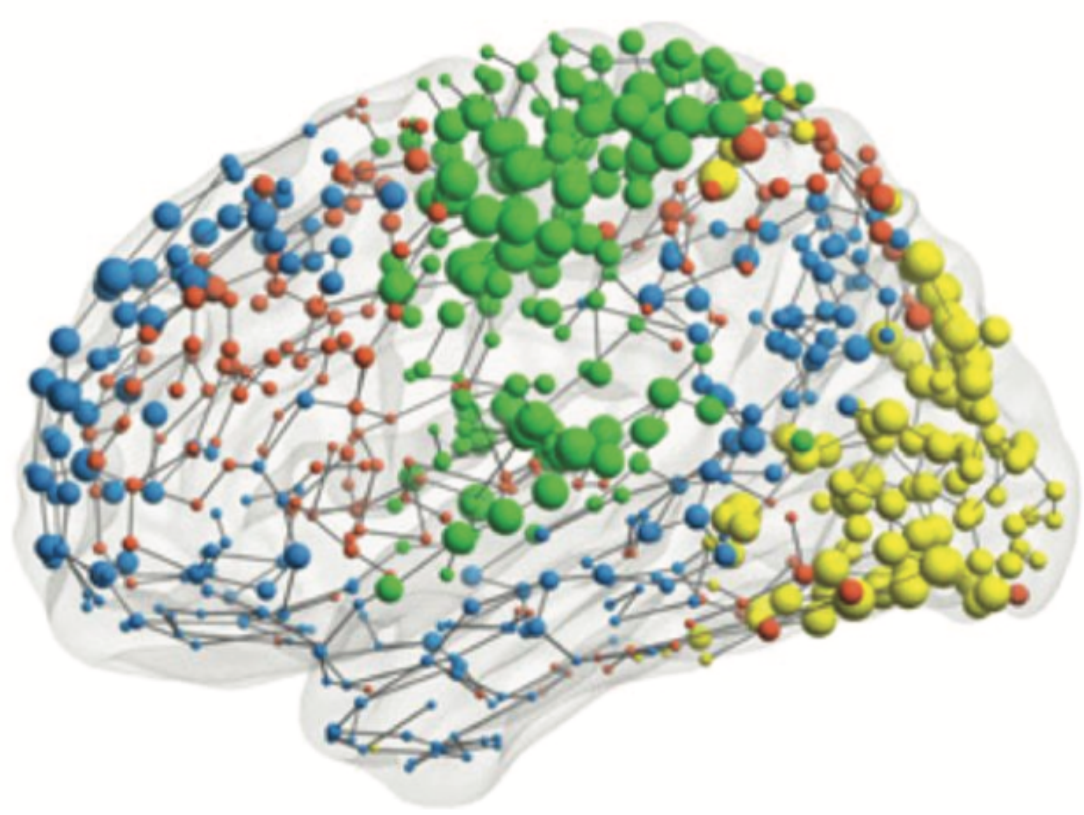
Non-euclidean distance,
twisted connectivity



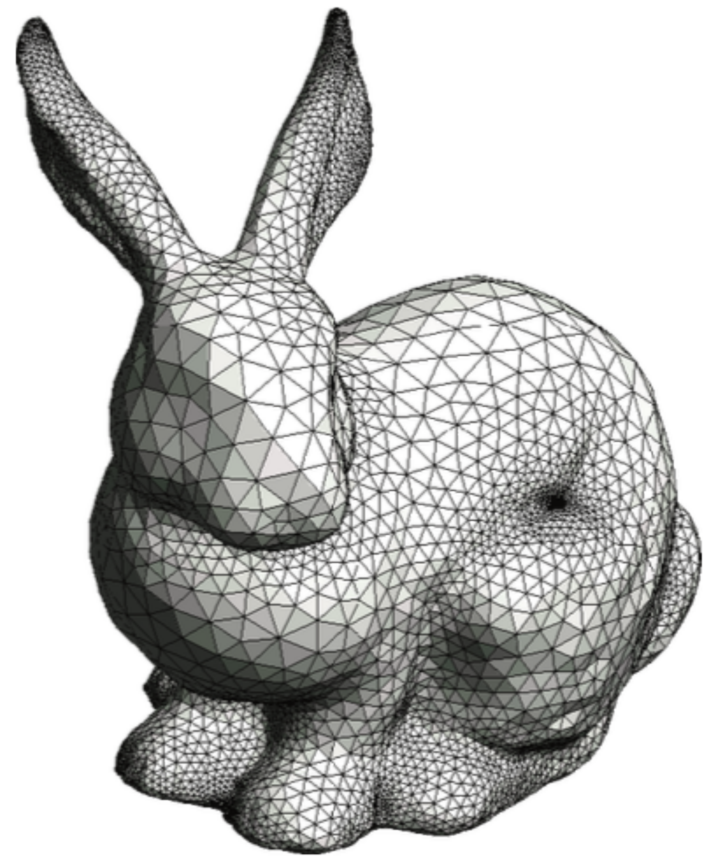
Social networks



Regulatory networks



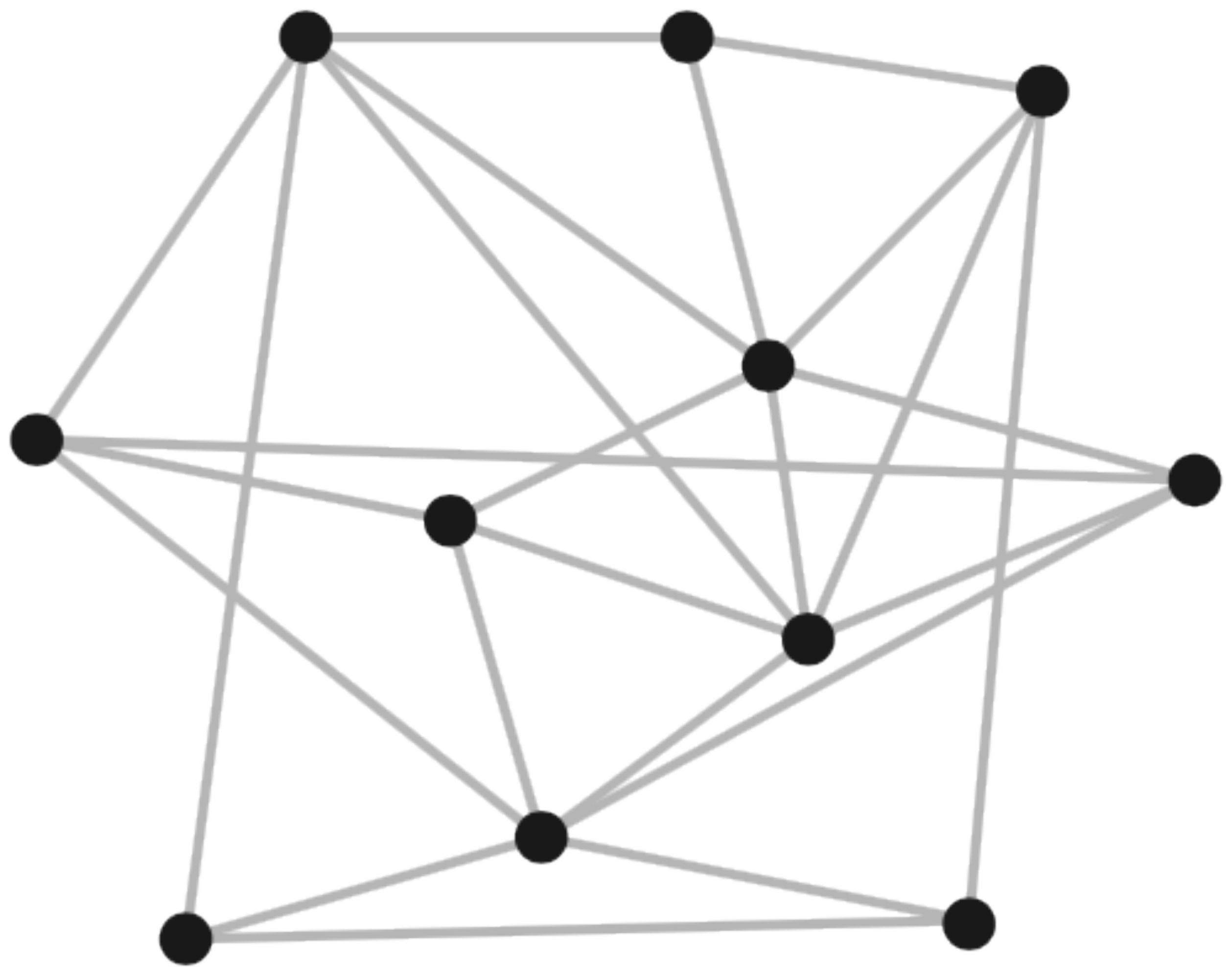
Functional networks



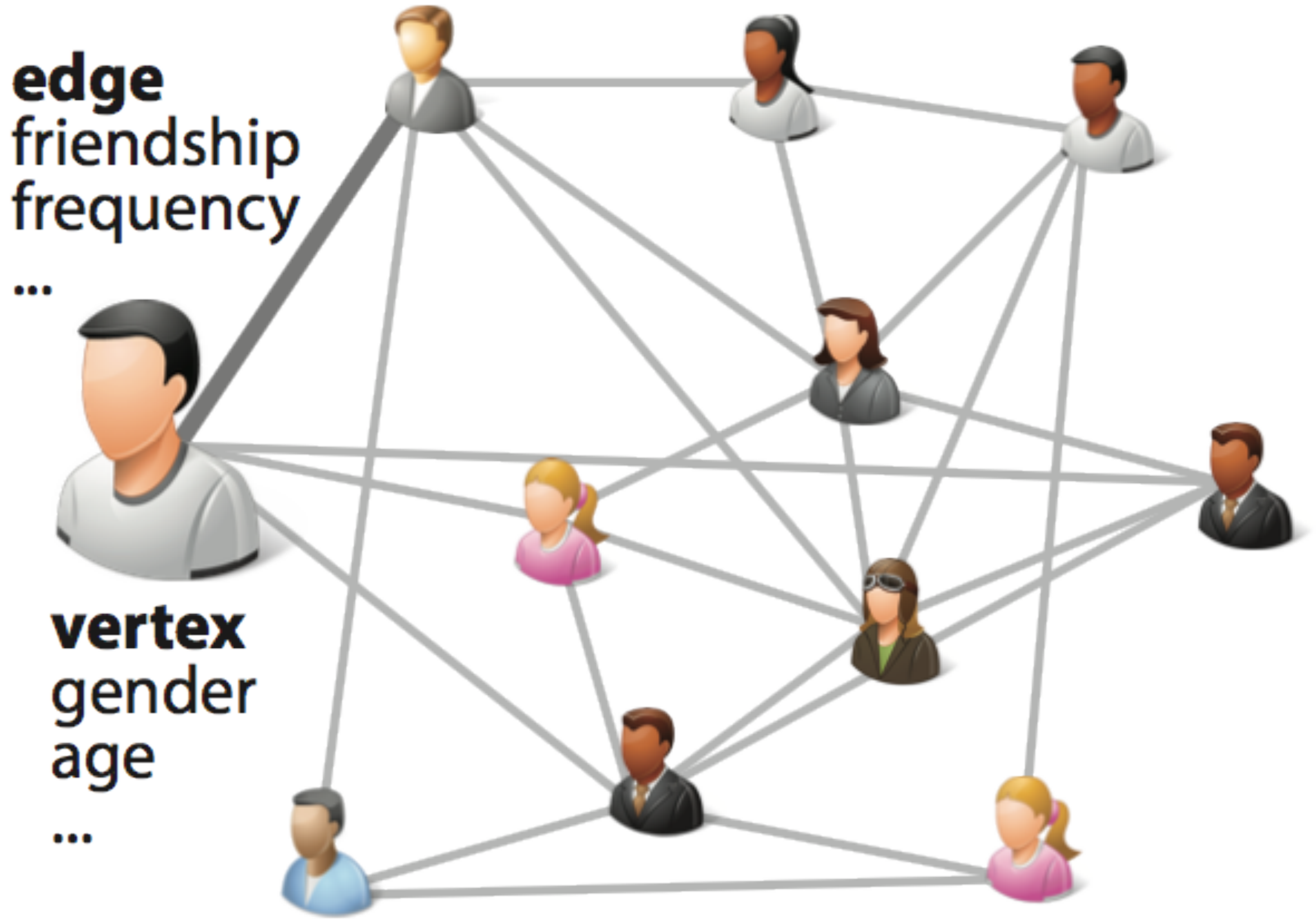
3D shapes

Formalization

- **Graph (structure):**
  - $G(V, E)$, A – adjacency matrix
- **Features (data):**
  - Vertex (node) features
  - Edge features



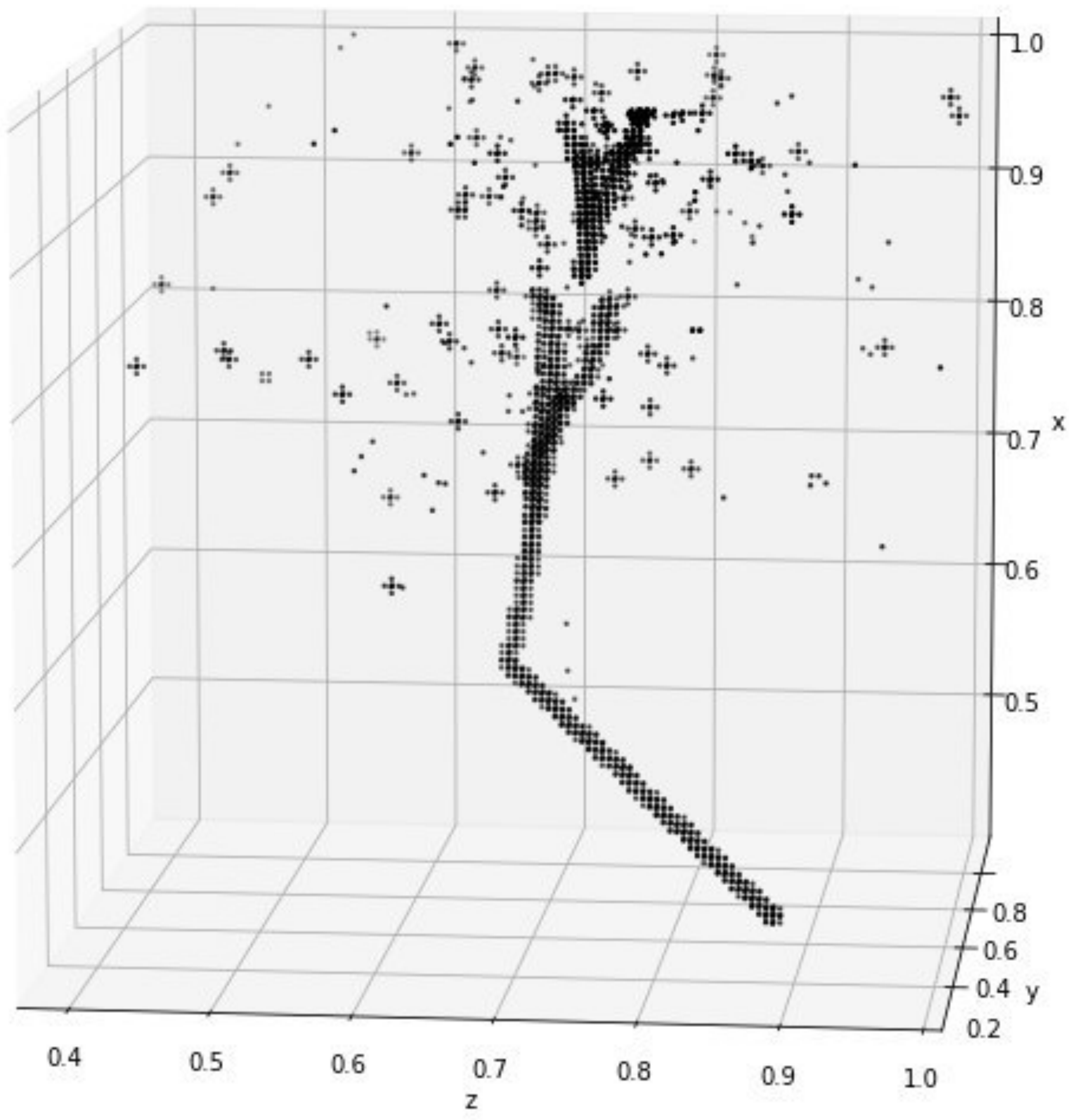
Domain structure



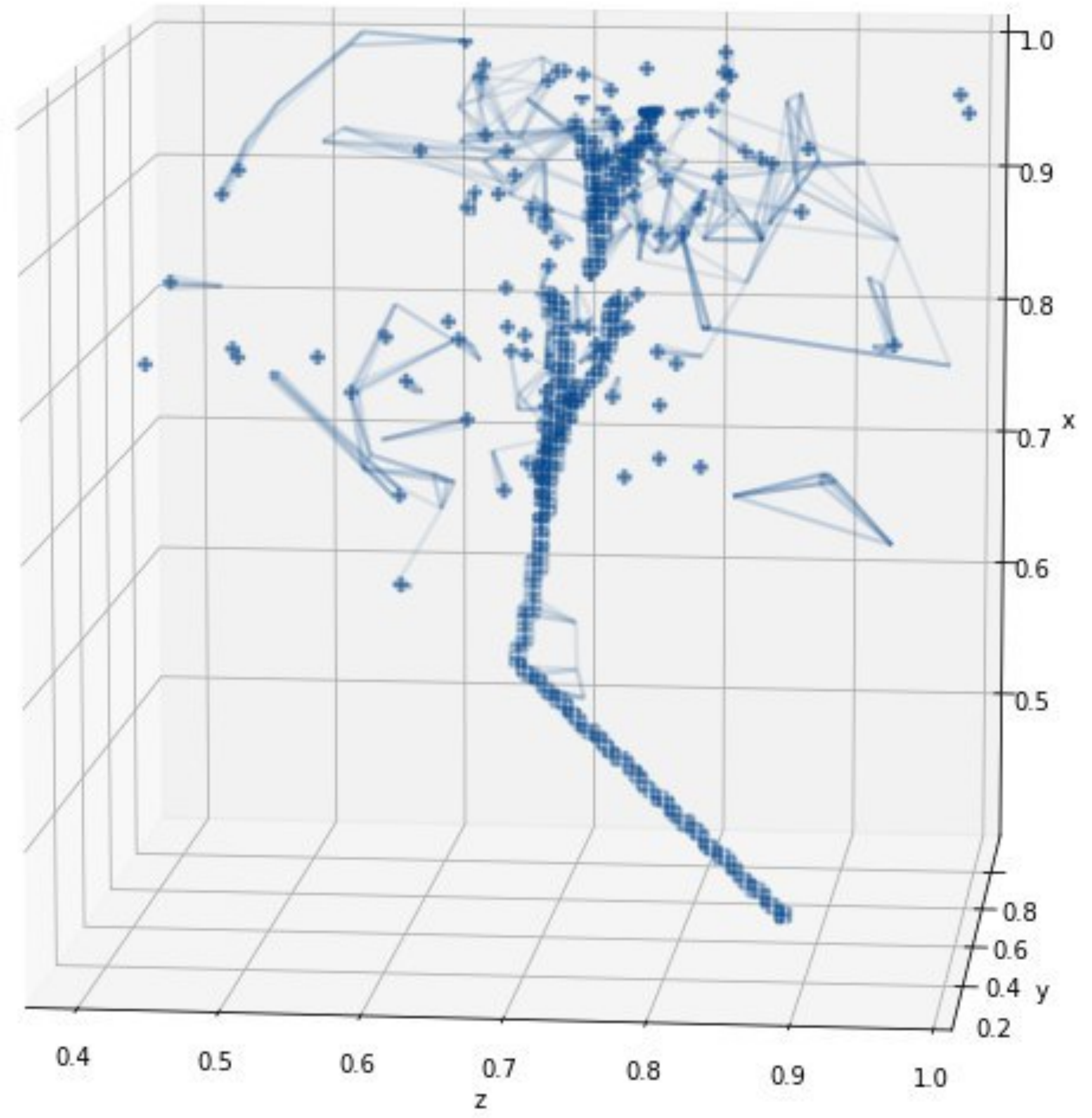
Data on a domain

From hits to graph

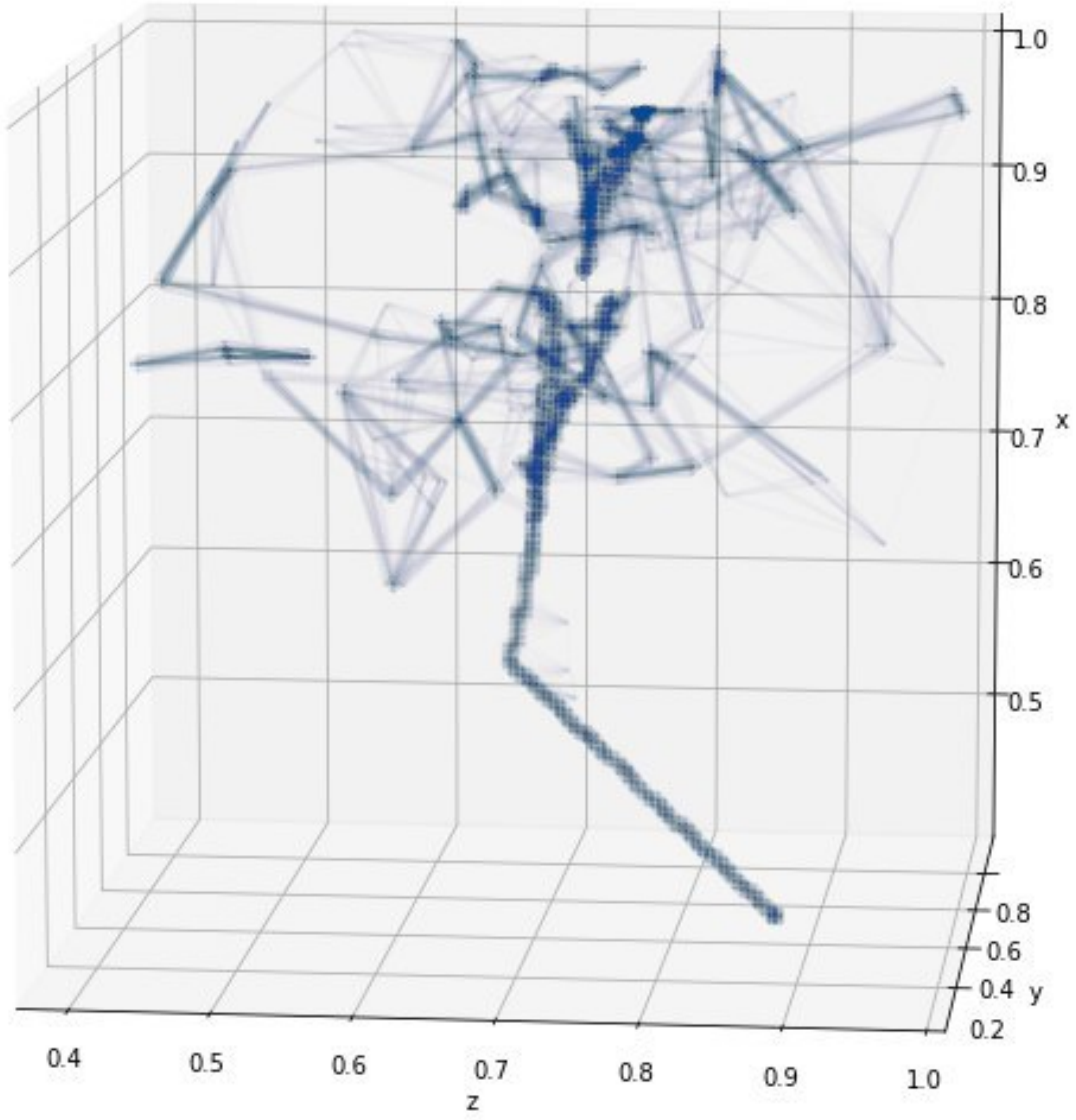
- **K-nearest neighbors, euclidean distance**
  - Iterate by hits
  1. Find K closest hits for each hit
  2. Connect with edges
- **Radius graph, euclidean distance**
  - Iterate by hits
  1. Find nodes within given radius and connect with an edge



After graph-ication



Neighbors = 5



Neighbors = 20

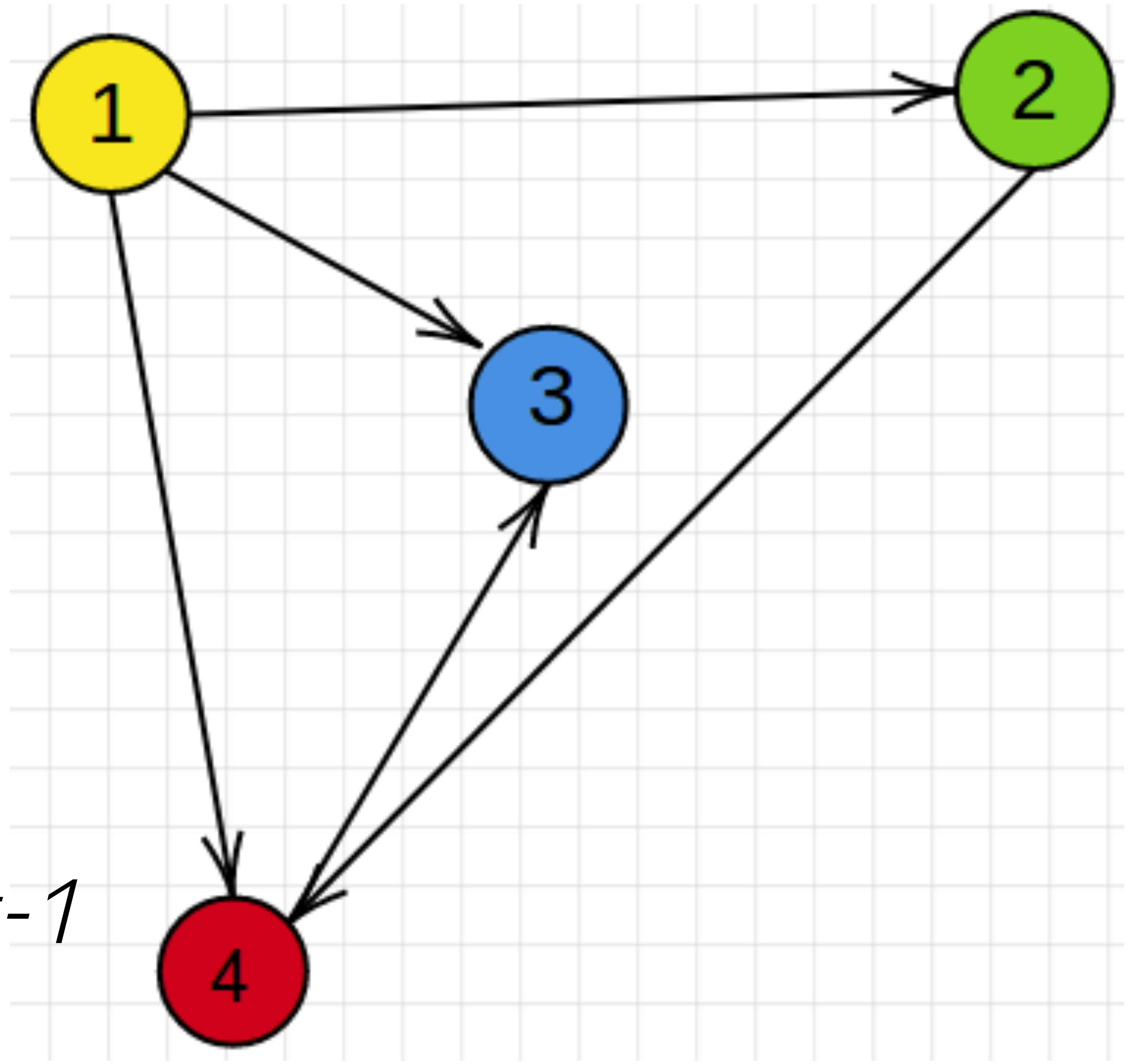
Neural Networks over graph

- We want to learn some structural patterns in nodes adjacency (connectivity), that correlate with labels we want to predict for every node.
- Connectivity patterns correspond to some structure in adjacency matrix. How can we reveal it using neural network?
- Let’s assume that labels we want to predict correlate with certain computable function over incoming/outgoing edges of each node

Message Passing Neural Networks

We want to predict label for each node
Let's assume the following :

- Each *node has a state* (node features) at time *t*
- Each *edge has a state* (edge features) at time *t*
- *Message* is a **function** of source node state and edge state at *t-1*
- Node state is a **function** of incoming messages at *t-1*
- Node *label* is a **function** of its state at *t* (readout)



Trivial Example

Step 0: pick random node states h_i^0
Step 1. Messages pass node state:

$$m_{13} = h_1^0,\ m_{43} = h_4^0$$

Step 2. Messages are aggregated:

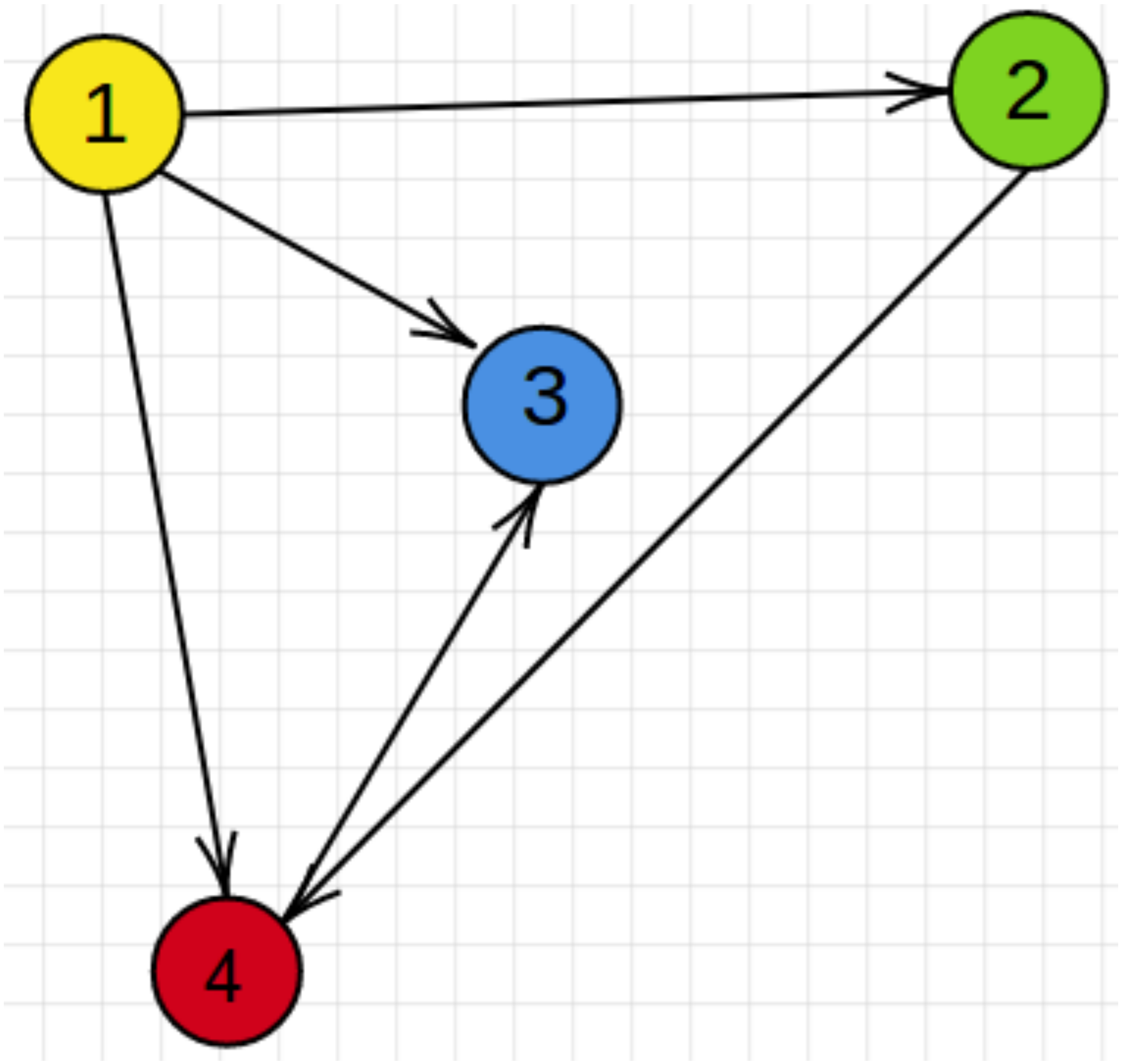
$$M_3 = (m_{13} + m_{43})/2$$

Step 3. Node updates its state:

$$h_3^1 = (h_3^0 + M_3)/2$$

Step 4. Estimate prediction (readout):

$$R_3 = ||h_3^1||$$



Starter Kit MPNN baseline

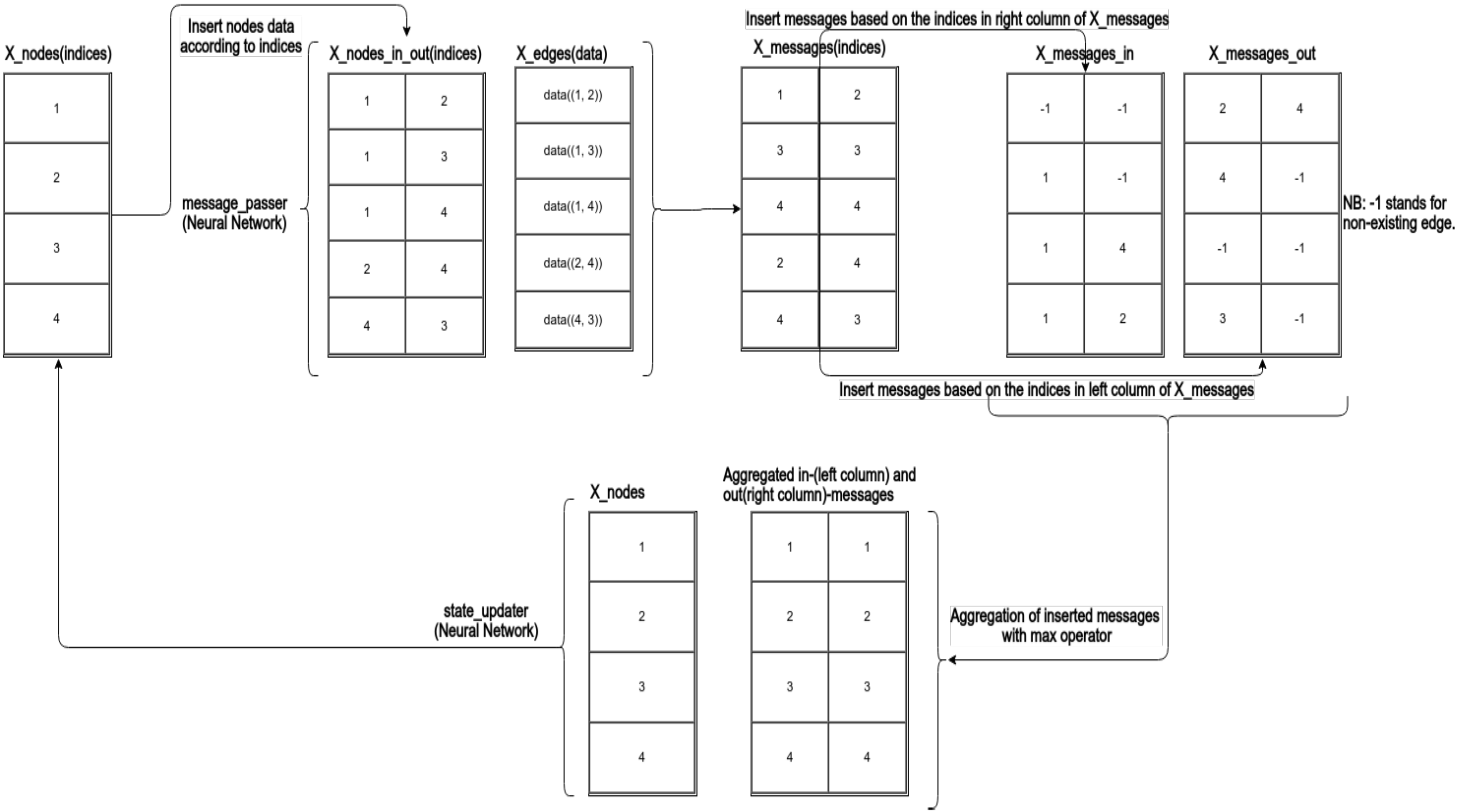
Ideas for improvement

Clustering algorithm type, parameters
Depth of messages (number of steps)
State updater and message passer can have longer memory GRU/LSTM
Use batches of events
Different optimizers / learning rates
Data augmentation

References

- Plot event as a graph: https://gist.github.com/SchattenGenie/28204a1135c3b7bca06162b7b2adf073
- Neural Message Passing for Quantum Chemistry, arXiv:1704.01212,
- Neural Message Passing for Jet Physics, https://orbi.uliege.be/bitstream/2268/226446/1/nips_dlps_2017_29.pdf
- http://geometricdeeplearning.com

Backup


X_nodes(indices)
Insert nodes data according to indices
message_passer (Neural Network)
X_nodes_in_out(indices)
X_edges(data)
data((1, 2))
data((1, 3))
data((1, 4))
data((2, 4))
data((4, 3))
Insert messages based on the indices in right column of X_messages
X_messages(indices)
X_messages_in
X_messages_out
NB: -1 stands for non-existing edge.
Insert messages based on the indices in left column of X_messages
X_nodes
Aggregated in-(left column) and out(right column)-messages
state_updater (Neural Network)
Aggregation of inserted messages with max operator