Hcore-Init: Neural Network Initialization based on Graph Degeneracy

Stratis Limnios[14]
George Dasoulas[1 3]
Dimitrios Thilikos[2]
Michalis Vazirgiannis [1]

[1]Data Science and Mining (DaSciM) team, Laboratoire d'Informatique (LIX)
École Polytechnique, Palaiseau, France.

[2]AlGCo project team, CNRS, LIRMM,
Université de Montpellier, Montpellier, France.

[3]Noah's Ark Lab, Huawei Technologies, Paris, France.

[4]Alan Turing Institute, London, UK

Outline and contributions

- We provide a general framework to transform neural networks to collections of bipartite graphs,
- We define a degeneracy framework on hypergraphs, the **k-hypercore**, and apply it to their incidence graphs, i.e. bipartite graphs,
- Finally we use this decomposition to provide a new initialization method for CNN and MLPs.

Neural Networks and Graphs

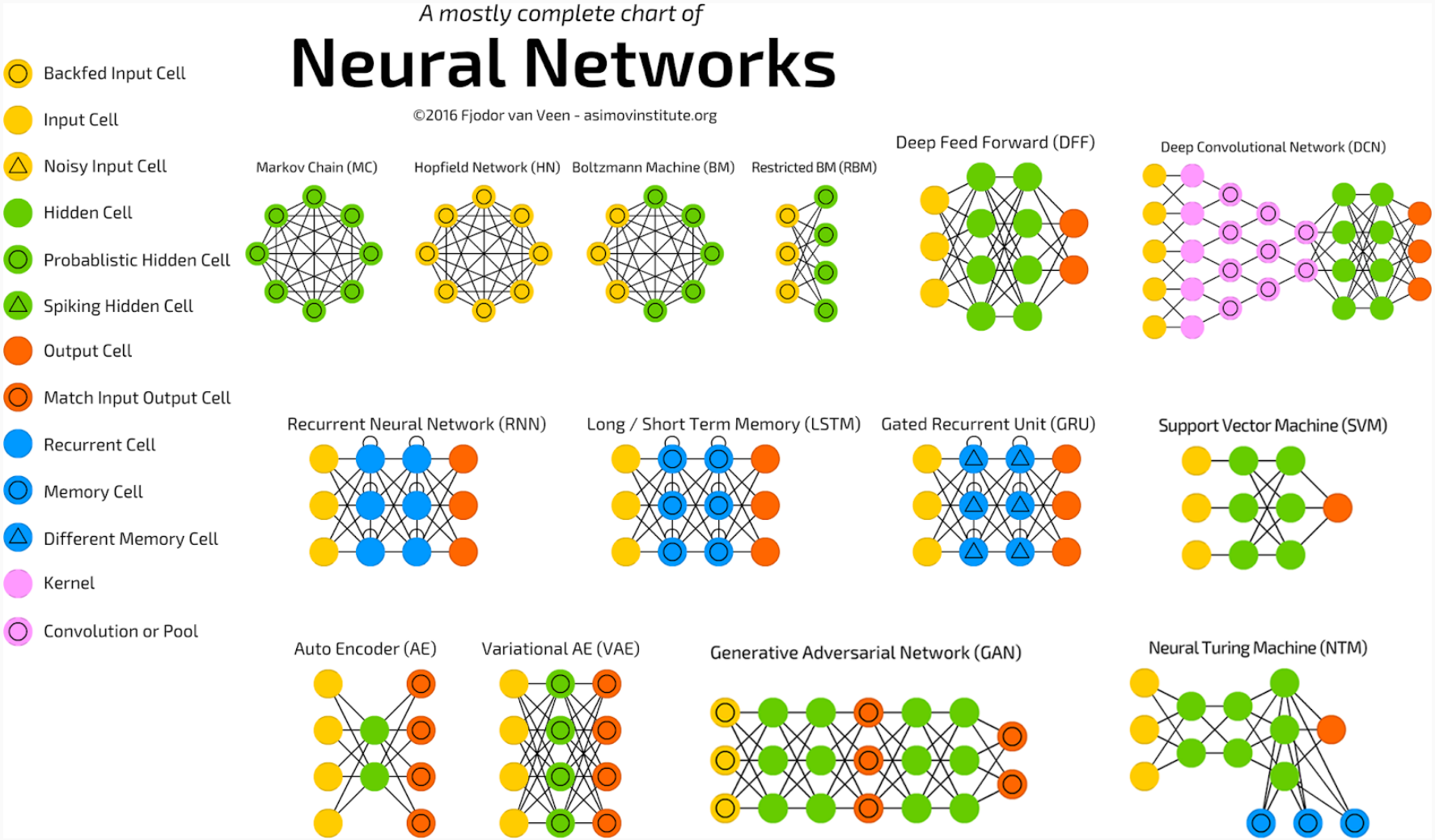

Figure 1: Graph representations of neural network models

What happens in details

Transforming neural networks to graphs seems fairly straigth forward. Unfortunatly it is not necessarily the case :

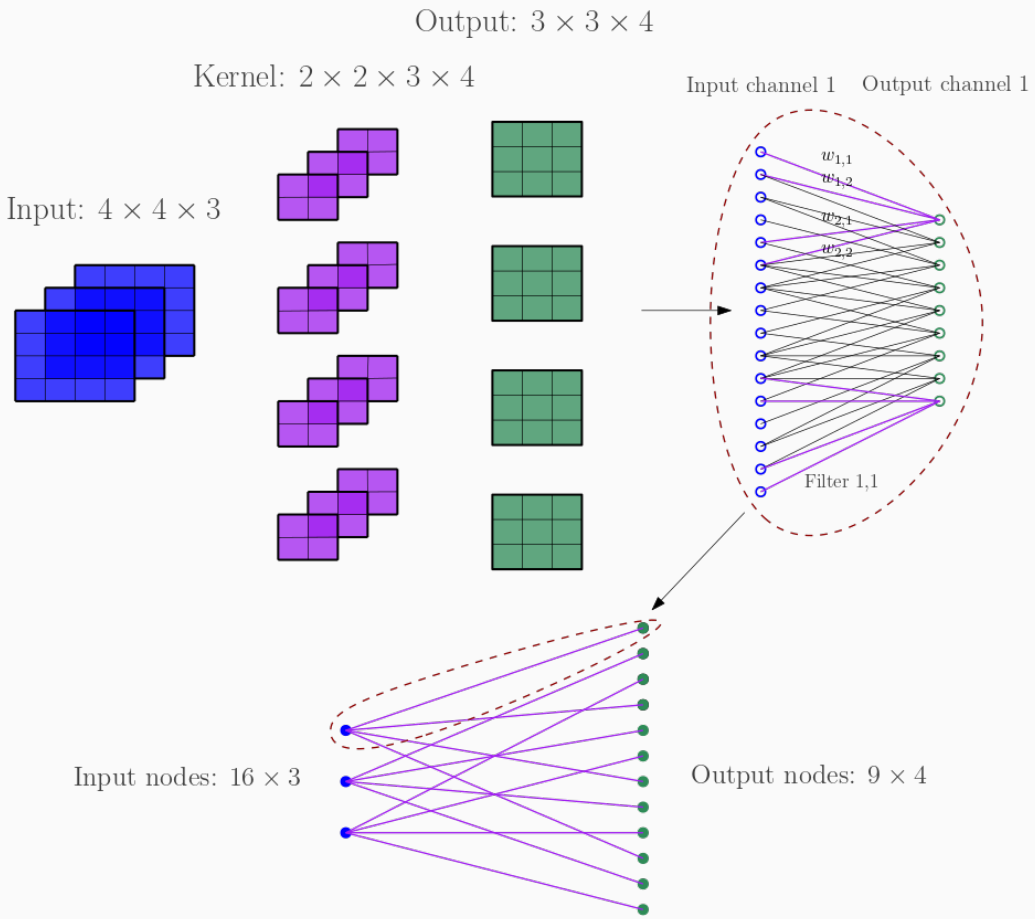

Figure 2: CNN to graph, with only one convolution layer

Why Hypergraphs?

Every hypergraph can be represented as its incidence graph, which is a bipartite graph. As k-core is not effective on bipartite graphs, especially very dense ones, we needed to design an alternative on bipartite graphs. So we did it on hypergraphs.

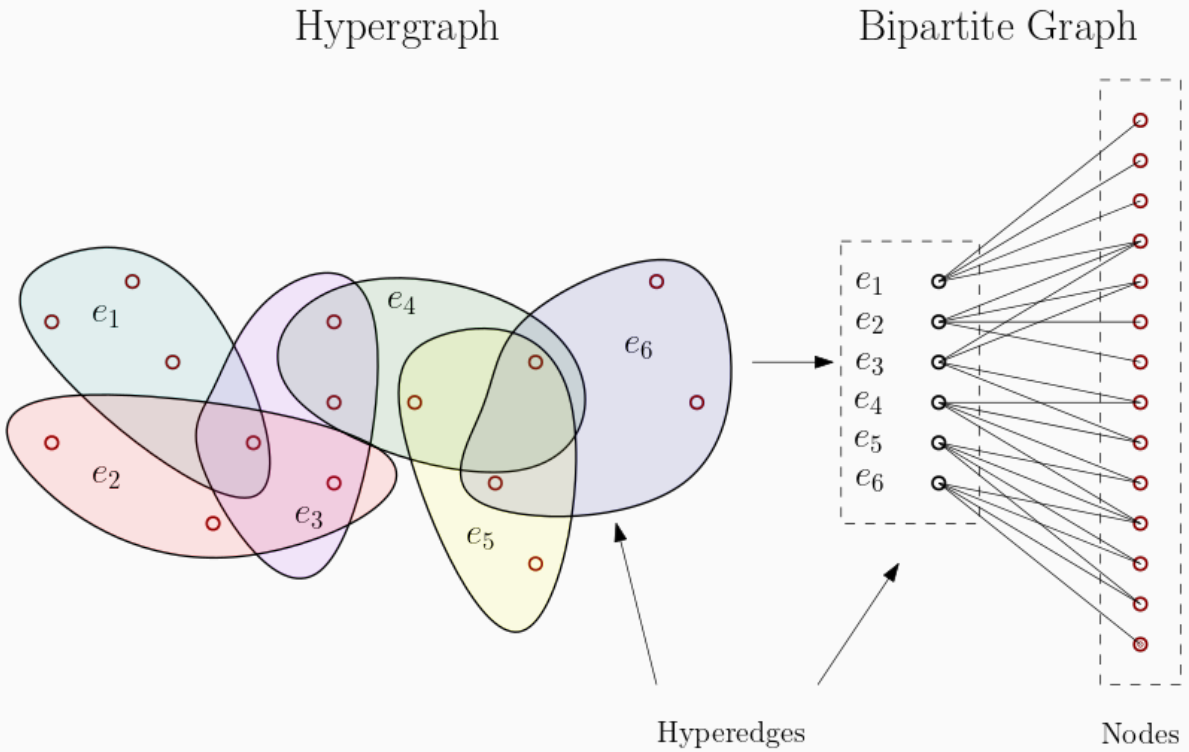

Figure 3: Hypercore decomposition example

Hypercore Definition

Definition (Hypercore)
Given a hypergraph $\mathcal{H} = (V, E_{\mathcal{H}})$ We define the (k, l)-**hypercore** as a maximal connected subgraph of $\mathcal{H}$ in which all vertices have hyperdegree at least k and all hyperedges have at least l incident nodes.

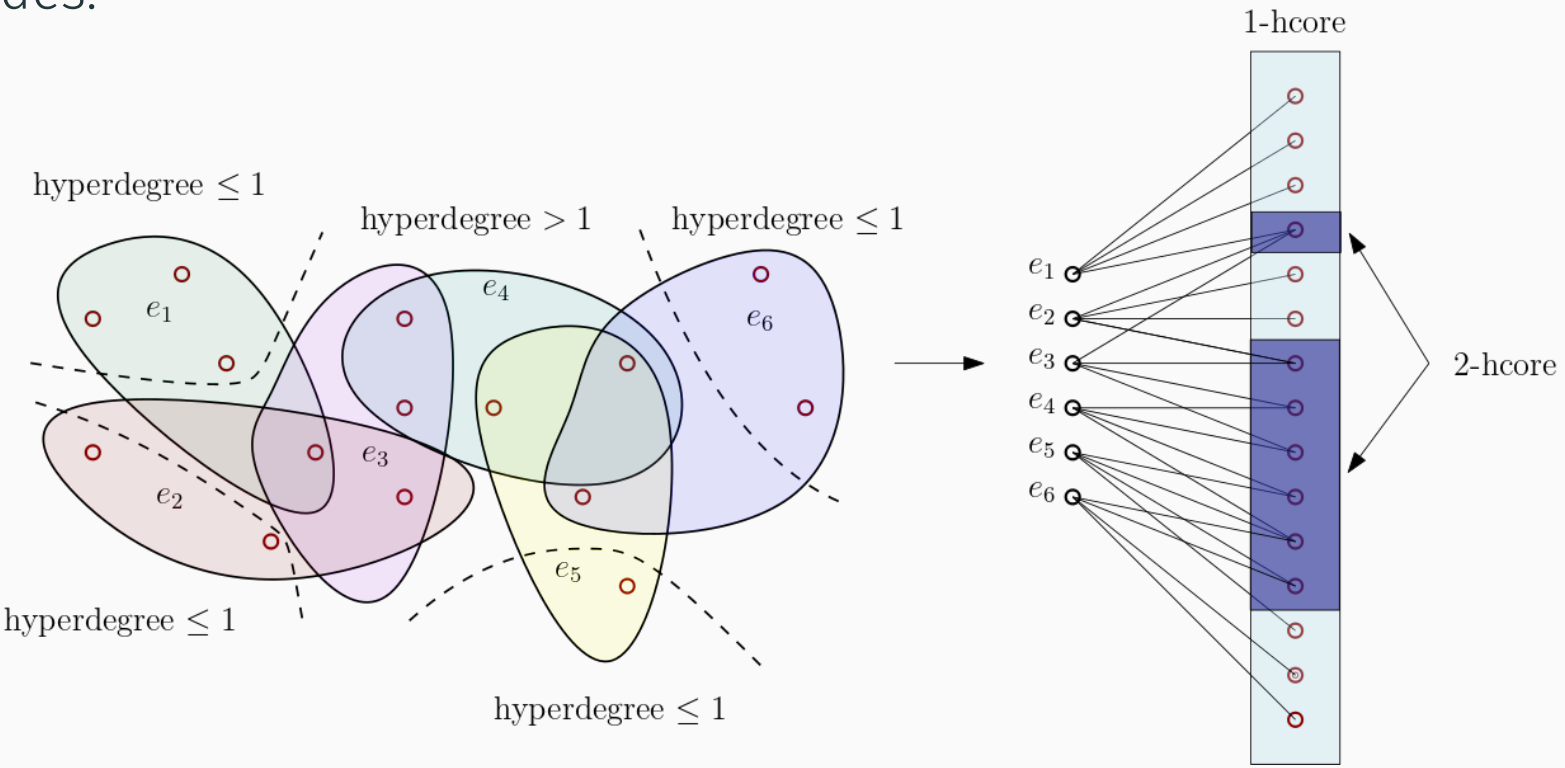

Figure 4: Hypergraph and Bipartite graph representation, We now will refer to the $(k, 2)$-hypercore as the k-hcore

Initialization Method

Neural Network Initialization

Initialization methods

- Standard initialization methods on neural networks focus on the definition of the standard deviation of the normal distribution.
  - Xavier Glorot init
  - Kaiming He init
- What if we could encourage ”important” nodes/weights according to graph information?

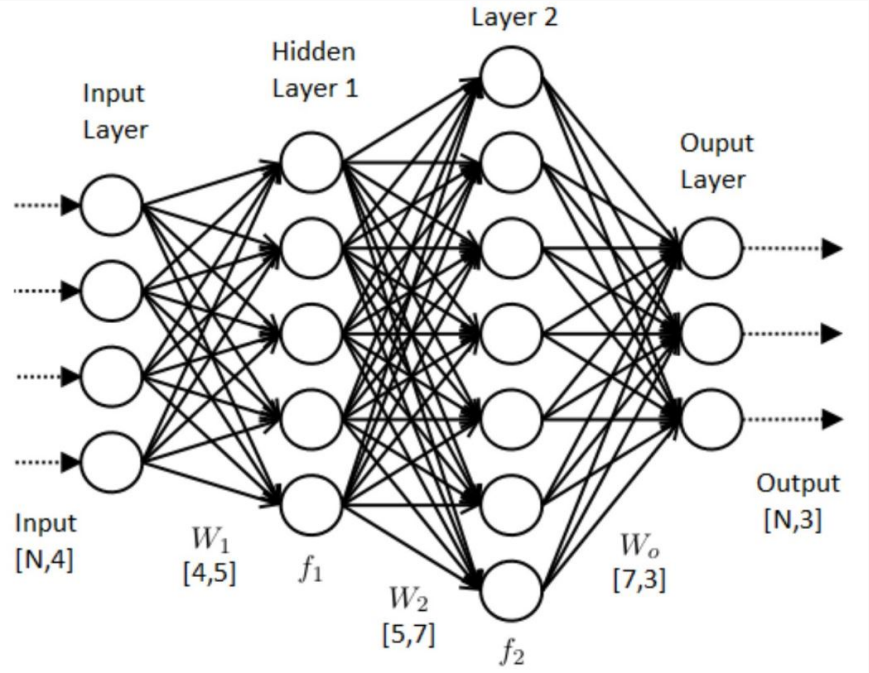

Figure 5: *Neural Network architecture (Source:VIASAT).*

Overview of our Method

Method: The graph-based initialization method consists of:

1. Pretraining of NN for x epochs.
2. Construction of weighted graph structure of NN architecture.
3. Hypercore decomposition of the contstructed graph.
4. Weight initialization of the NN based on the output hypercore values.

Initialization of the MLP

The given pair of layers weights $w_{i,j}$, are initialized, depending on their sign, with a normal law with expectancy:

- for all i if $w_{i,j} \geq 0$, $M = \frac{c_j^+}{\sum_{1 \leq k \leq n_2} c_k^+}$,
- else $M = \frac{c_j^-}{\sum_{1 \leq k \leq n_2} c_k^-}$

Hence $w_{i,j}$ follow a $\mathcal{N}(M, \frac{2}{n_2^2})$ which variance is from the Kaiming initilization method.

Experiments

Results on CIFAR-10 (1/3) Using CNN

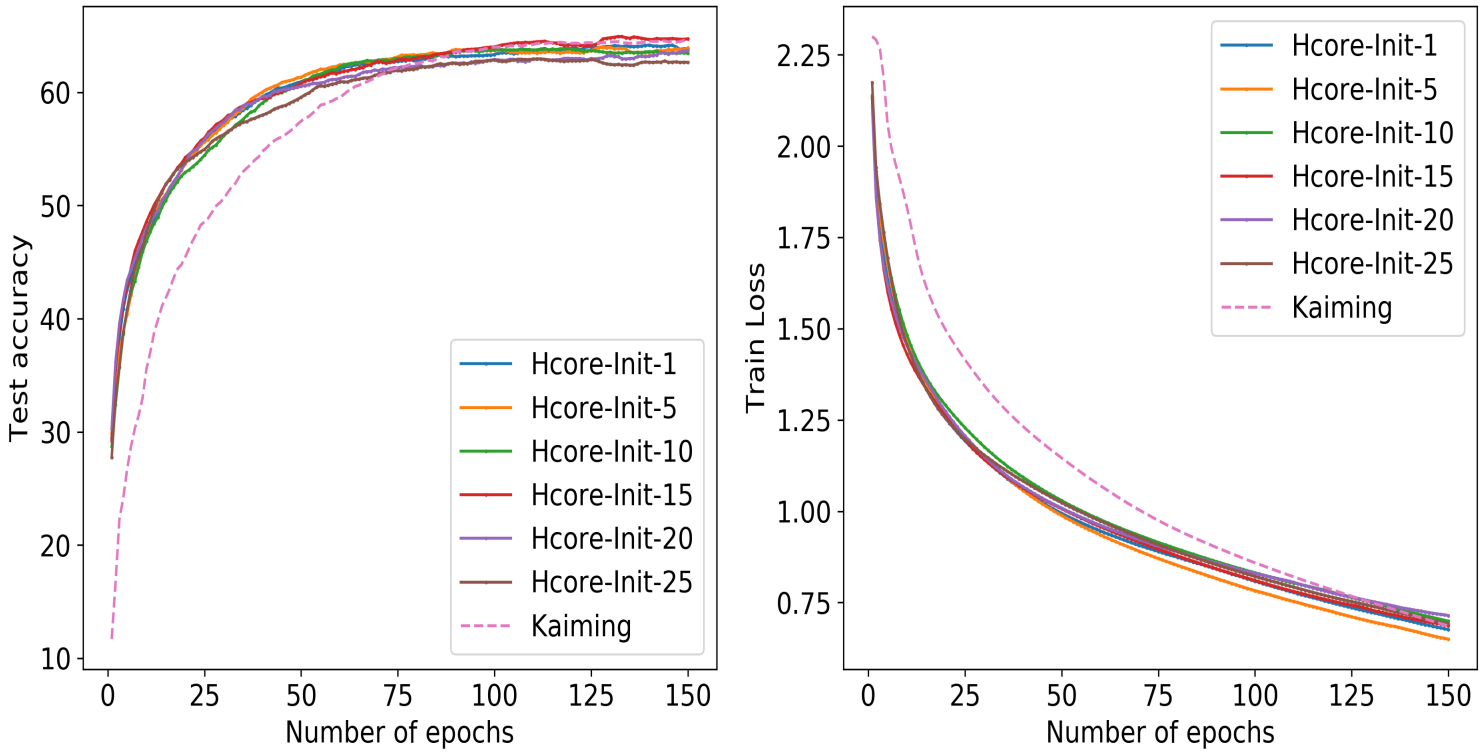

Figure 6: Test accuracy (left) and train loss (right) on CIFAR-10 for the combined application of the initialization on the linear and the convolutional layers. For the curves Hcore-init-x, x stands for the number of pretraining epochs.

Final Results

Table 1: Top Accuracy results over initializing the full model, only the CNN and only the FCNN for CIFAR-10, CIFAR-100, and MNIST. **Hcore-Init*** represent the top performance over all the pretraining epochs configurations up to 25

	CIFAR-10	CIFAR-100	MNIST
Kaiming	64.62	32.56	98, 71
Hcore-Init*	**65.22**	**33.48**	**98.91**
Hcore-Init-1	64.91	32.87	98.59
Hcore-Init-5	64.41	32.96	98.70
Hcore-Init-10	**65.22**	33.41	98.81
Hcore-Init-15	64.94	33.45	98.64
Hcore-Init-20	65.05	33.39	98.87
Hcore-Init-25	64.72	**33.48**	**98.91**

Conclusion

Conclusion and future research topics that we are interested in:

- We propose a new NN initialization method based on hypergraph degeneracy that outperforms existing state-of-the art initialization method,
- Hopefully this opens doors as to investigate the graph structure of neural networks and encourages our effort towards this domain.
- As our model is designed as ”plug-n-play” we hope to see results in more complex architectures using convolutional layer blocks and MLPs in a more intricate way.
- We also want to investigate the shadow graph of the weighted graph we build which is the one induced by the gradient descent.

THANK YOU