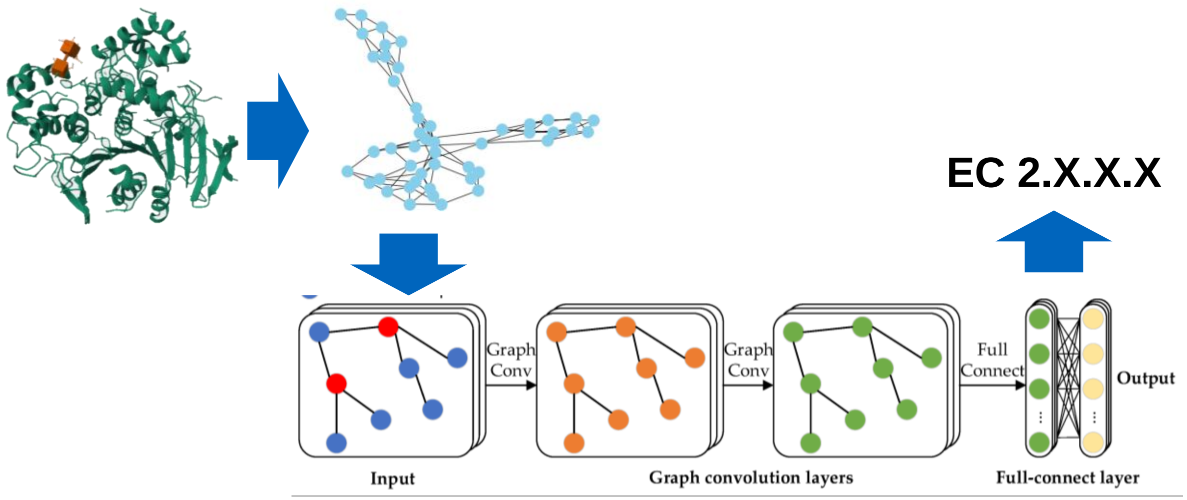
Final Project:
GNN for Multiclass Enzyme Classification



Iswara Jay Junior

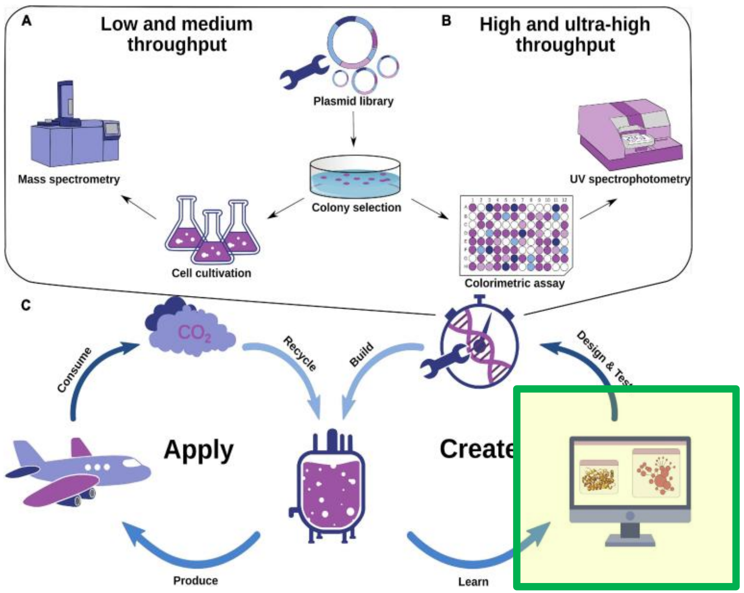
23.05.2025

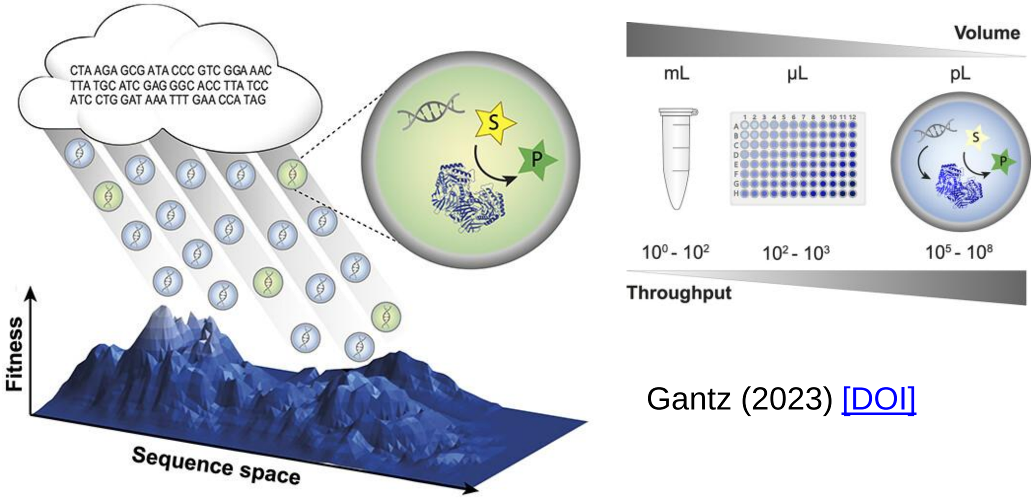
Outline

1. Motivation
2. Objectives
3. Methods
4. Conclusion & Recommendations

Motivation

Enzyme engineering





Gantz (2023) [DOI]

In general screening 1 AA modification in 1 position
= screening ~ 260.000 gene variants

Modify enzyme structures (primary) via **mutagenesis** to get the **best fitness** in its sequence space.

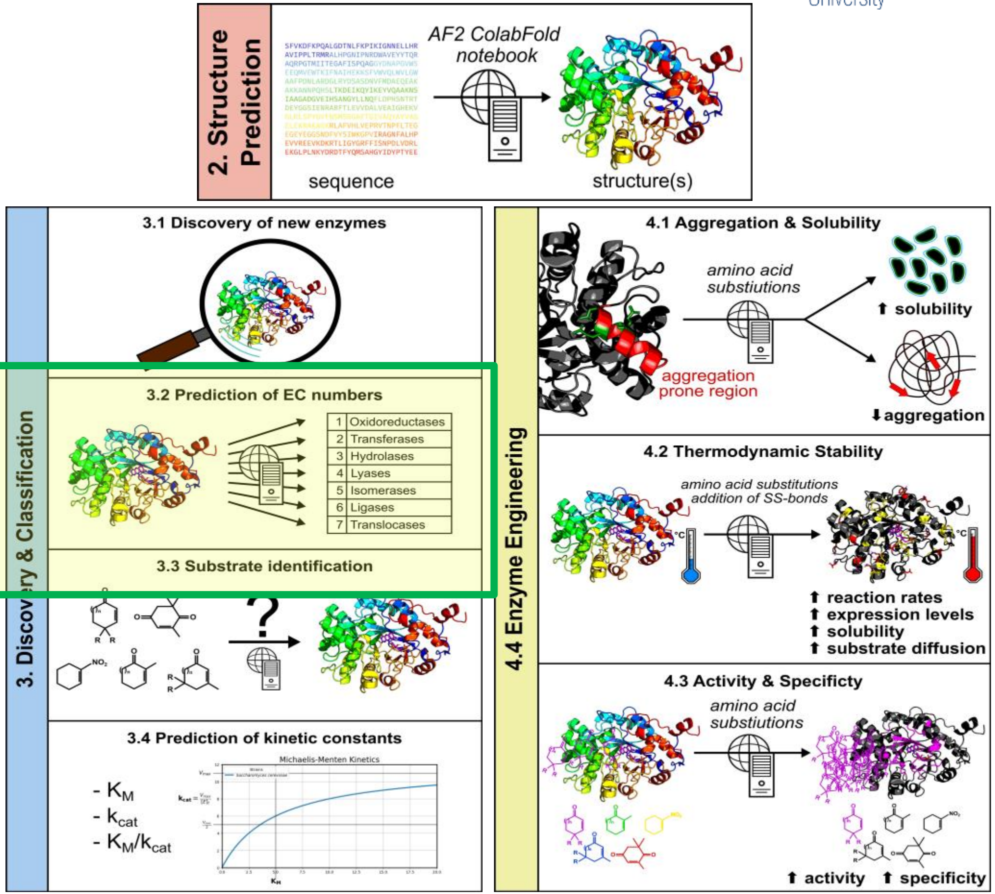
Scherer (2021) [DOI]

Motivation

What tasks do Neural Networks can achieve in computational enzyme design?

Published model:

- TopEC (~800 EC)
- CLEAN-CONTACT (~5200 EC)
- DeepECTransformer (~5300 EC)



Tripp (2024) [DOI]

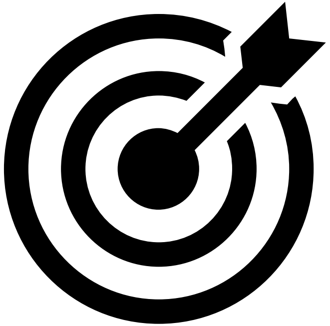
Objectives

Model Objective

Formulate a GNN-based architecture for a **multiclass graph-level classification** of enzyme tertiary structures dataset (ENZYMES from TUDataset)

Success Criteria

1. Overall accuracy (test) ≥ 0.75
2. Class wise F1-score > 0.7
3. Model max. training memory ≤ 100 MB



Methods

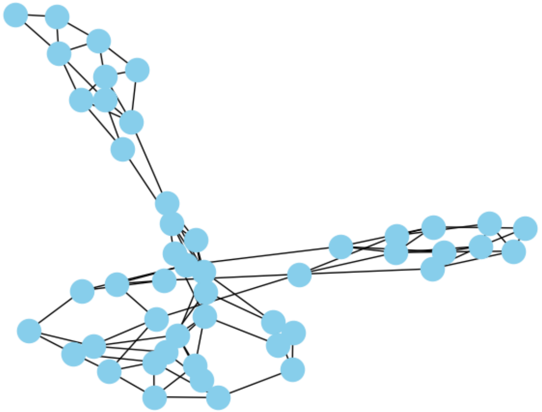
Dataset ENZYMES from TUDataset

Description

Public graph dataset of **tertiary enzyme structures. Labels encode** the first number to one of the 6 EC top-level classes (EC 1.X.X.X - 6.X.X.X)

Properties

- Static graph $\mathcal{G} = (\mathcal{V},\ \mathcal{E})$, **n = 600**
- **21 Node features**
  - 3 one-hot encoding for secondary structures
  - 18 physico-chemical properties of secondary structures
- Edge = neighbors along the AA sequence or one of three nearest neighbors in space
- **No edge and graph attributes**



EC 2.X.X.X Enzyme (Transferase)

Data split: Stratified 80% Train, 10% Validation, 10% Test

Methods

Model Screening

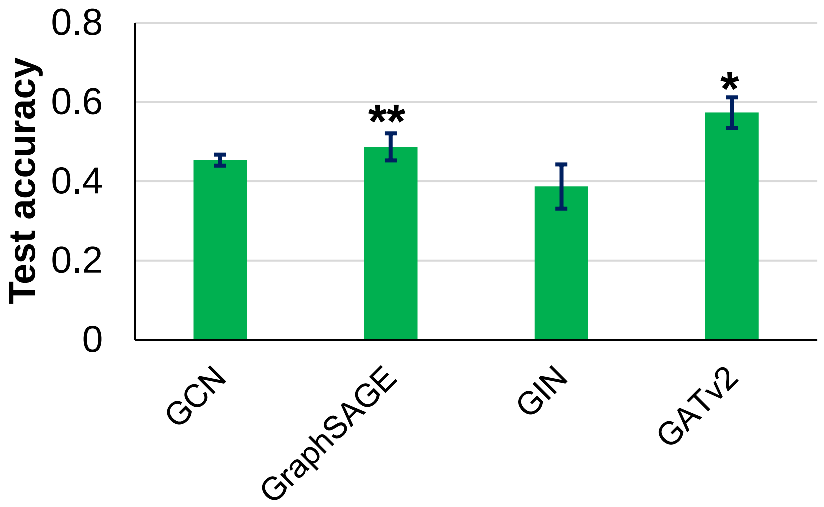
Objective : Get the most optimal algorithm for a base model layout

Method : Try out different algorithm for static graph (5 runs)

1. (spectral) GCN
2. (spatial) GraphSAGE
3. (spatial) GATv2
4. (spatial) GIN

Base layout

GNN Layer (3, 128) → Global mean pool → MLP layer (3, 64)



Max. training memory

- GraphSAGE : 38.5 MB
- **GATv2 : 144.6 MB**

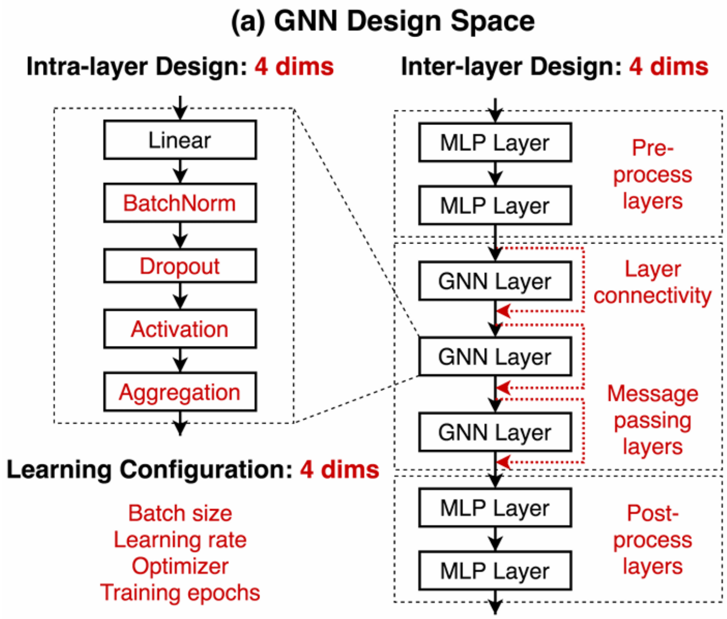
Methods

Design Space Optimization (GraphSAGE)

Objective : Get the most optimal model design by modifying design space.

Method : Sequential tuning, 5-run experiment, 100 epochs

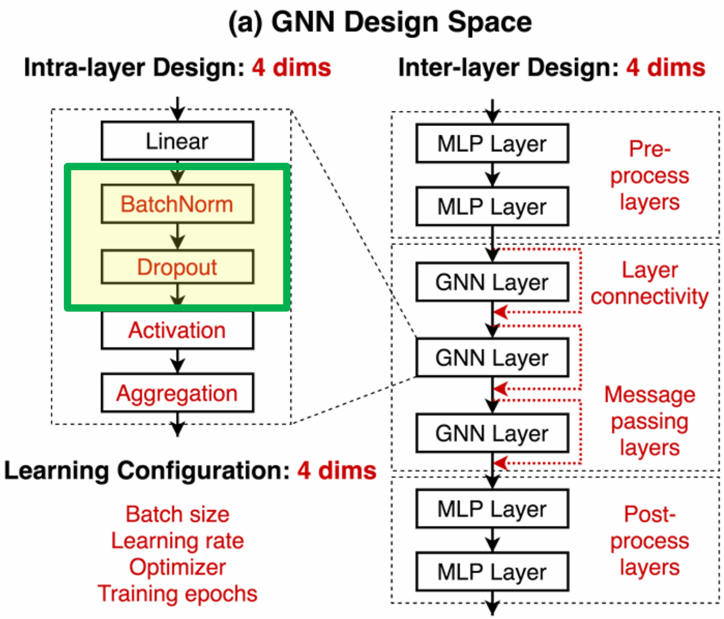
(1st) Dropout ∈ {0.0, 0.2, 0.3, 0.5}
(2nd) Normalization ∈ {None, Batch, Layer, Graph}
(3rd) Jumping Knowledge ∈ {None, cat, max}



You, 2021 [DOI]

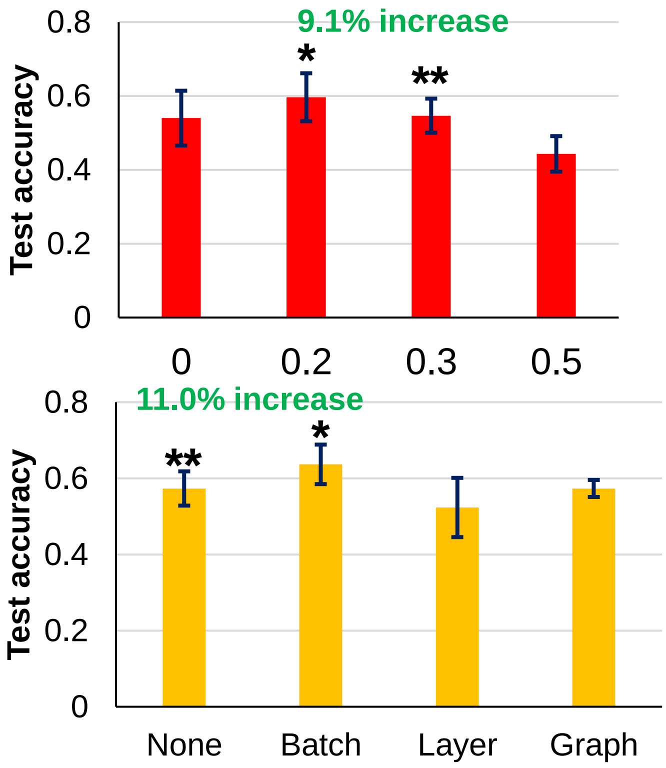
Methods

Design Space Optimization (GraphSAGE)



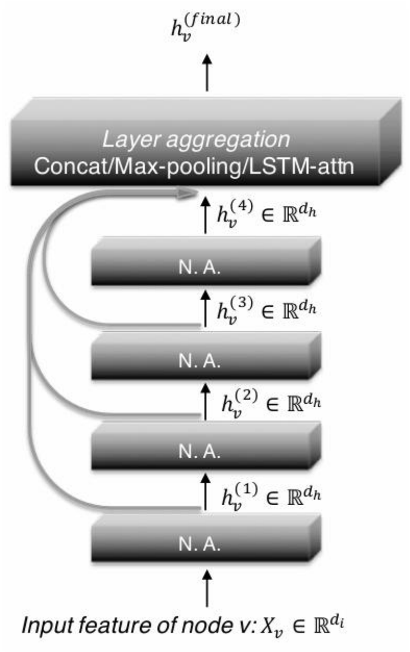
Dropout

Norm.



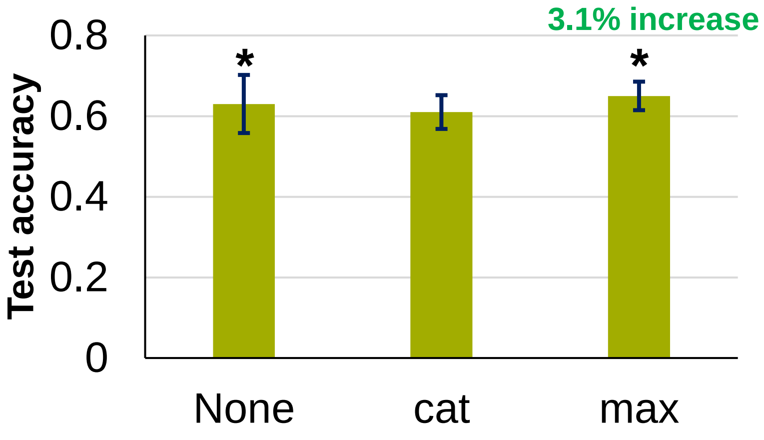
Methods

Design Space Optimization (GraphSAGE)



Xu, 2018 [DOI]

Jumping knowledge



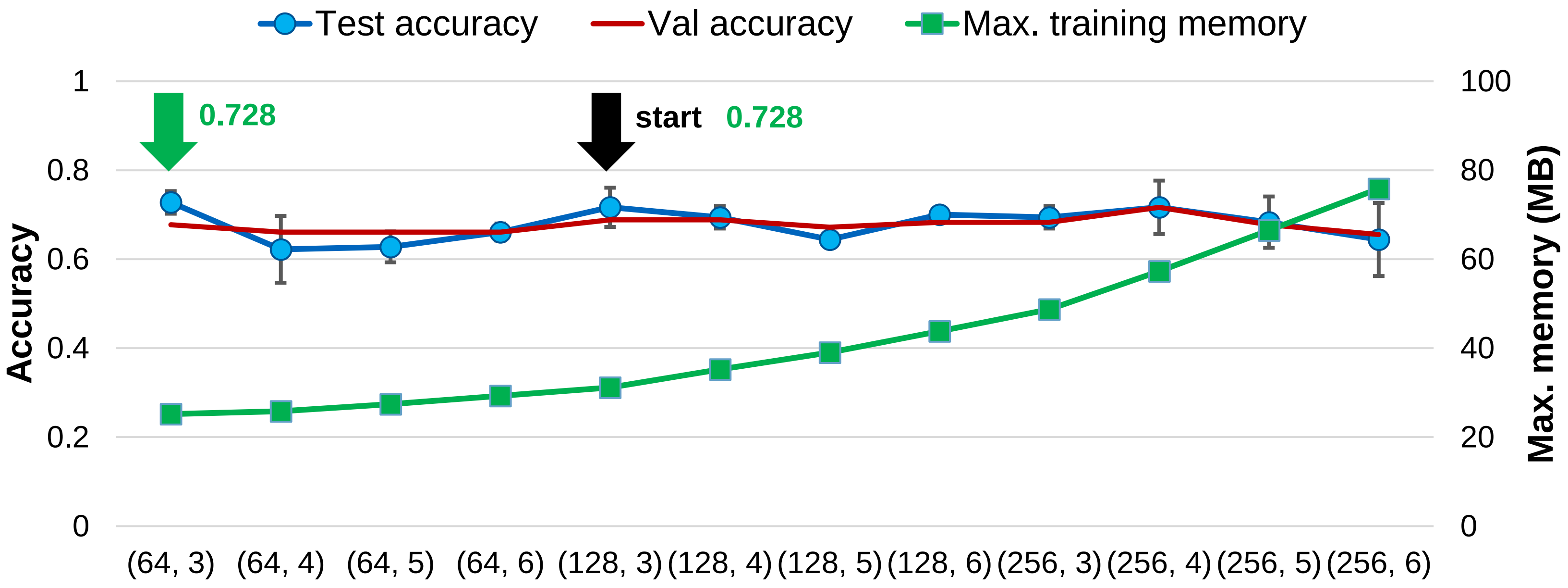
May be beneficial for:

- Oversmoothing for deep GNN
- Vanishing gradient for deep GNN
- Leveraging close neighbor and distant representation

Methods

Performance-Memory Tradeoff (GraphSAGE)

Objective : Get the most optimal model design by modifying the number of parameters. (GNN hidden channels, layers)



Conclusion

Best Model

GraphSAGE + MLP head

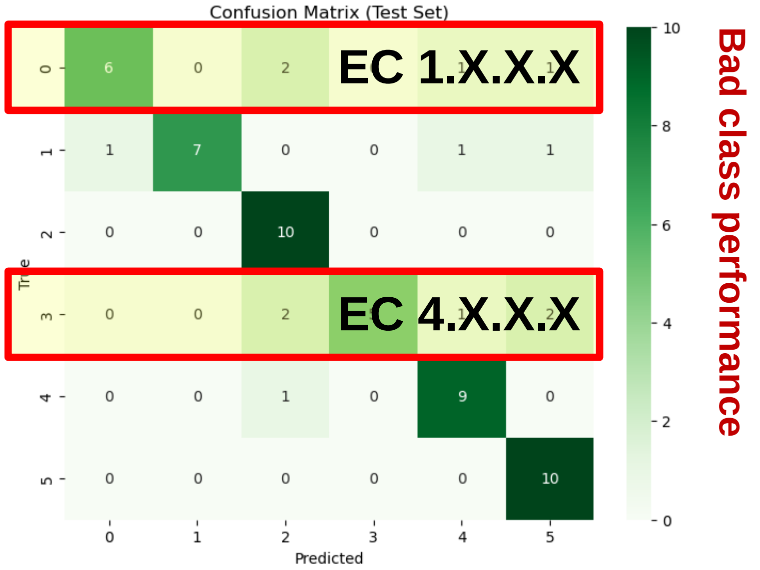
Utilizes dropout, normalization, and jumping knowledge.

Test set performance (200 epochs)

Class	f1-score
EC 1.X.X.X	0.706
EC 2.X.X.X	0.823
EC 3.X.X.X	0.800
EC 4.X.X.X	0.667
EC 5.X.X.X	0.818
EC 6.X.X.X	0.833
accuracy	**0.783**

Success Criteria

1. Overall accuracy (test) ≥ 0.75
2. Class wise F1-score > 0.7
3. Model max. memory allocation ≤ 100 MB



Recommendations

1. Enrich the dataset to potentially develop deeper classification task
   - Geometric graph
   - More samples
   - More complex graph

2. Feature engineering
3. Benchmark more algorithm/architecture
4. More complex learning methods (self-supervised, ensemble, …)