

Protein-Protein Complexes Scoring Predictions using Graph Neural Networks

Protein complexes

- Group of 2 or more associated polypeptide chain.
 - Characterized by protein-protein interactions (PPI) and essential in all cellular processes of living organisms.
 - Structural information in PPI helps develop new therapeutics and understand diseases.
- Need to understand PPI interfaces and learn a meaningful compressed embedding that can encode interaction patterns in those interfaces.

Computation-based structure prediction methods

2 main methods that need to decide on:

Molecular dynamics:

1. **Sampling** by allowing all atoms to move using Newton mechanics
2. **Scoring** by using energy based on the force-field. Precise but computationally expensive

Docking:

1. **Sampling** with fixed atoms within molecules and movement controlled by **Rosetta**
2. **Scoring** to identify structures of complexes close to global minimum energy conformations.

Importance of scoring methods for docking

- Docking is computationally cheaper than molecular dynamics
- Generation of very large number of possible conformations of complexes (10s of 1000s per complexes)
- Develop scoring methods that reduce the space of the generated decoys by identifying best candidates
- Use the scoring to drastically decrease complexes generation time

Assessment of decoys

- Several scoring methods:
 - Geometric-based
 - Knowledge-based
 - Scoring based
 - Combination of above
- **I-RMSD**: common measure evaluating the differences between each conformations and the target complex as a 2-step computation:
 - Interface residues of bound complex mapped to the docked conformation using sequence alignments
 - 3D structure of bound interface of bound complex is superposed on the mapped interface of the conformation to calculate the bone RMSD
- I-RMSD (by providing a continuous value) addresses both
 - relative scoring i.e., **ranking**
 - absolute scoring i.e., **quality assessment**

Machine learning for scoring function prediction

Earlier attempts using Traditional ML:

- Random Forest
- Support Vector Machine
- Regression Methods

But:

- Rely heavily on hand-engineered features from physic knowledge
- Do not capture local contextual feature and can not learn interaction patterns in PPI.
- Did not account for the graph structure of the decoys.

Machine learning for scoring function prediction

More recently: **GRAPH NEURAL NETWORKS**

□ Need to develop new models **that take into account structure of the decoys: GRAPHS**

□ Can learn **complex interaction patterns** and **new feature representation** from raw data: **DEEP NEURAL NETS**

Energy-based Graph Convolutional Networks for
Scoring Protein Docking Models
(Running Title: EGCN for Scoring in Protein Docking)

Yue Cao¹ and Yang Shen^{1,2*}

¹Department of Electrical and Computer Engineering, ²TEES-AgrLife Center for
Bioinformatics and Genomic Systems Engineering, Texas A&M University, College Station,
Texas 77843, USA

* Correspondence: yshen@tamu.edu

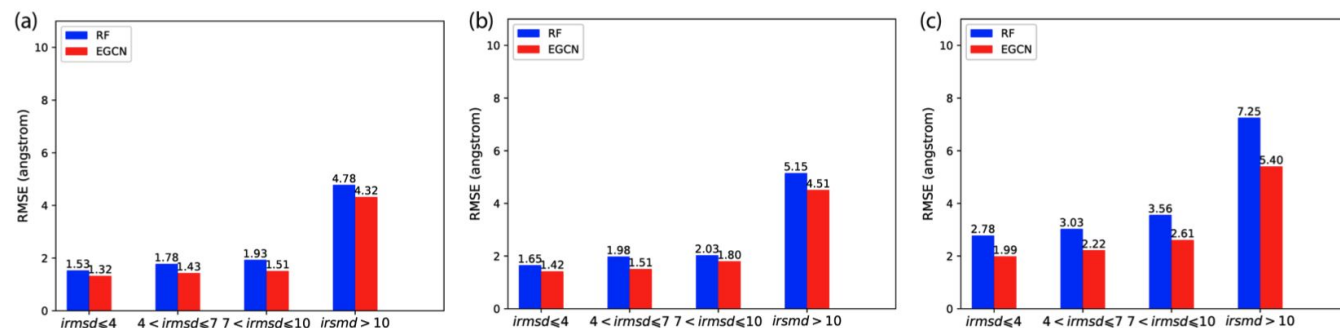
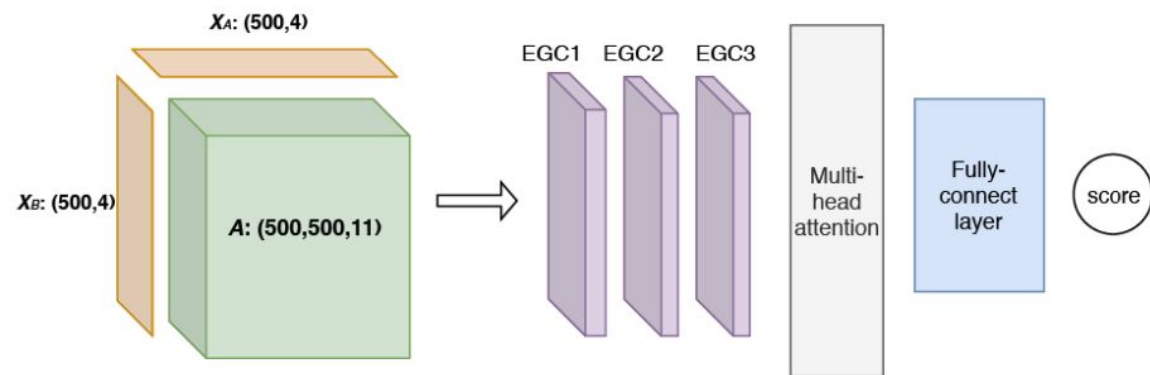
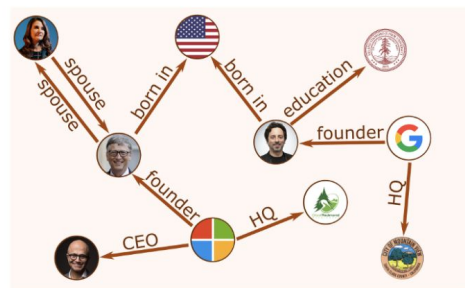


Figure 3. Comparing absolute scoring (quality estimation) performances among RF and EGCN. Reported are the RMSE of iRMSD predictions for (a) benchmark test set, (b) CAPRI test set, and (c) Score_set, a CAPRI benchmark for scoring.

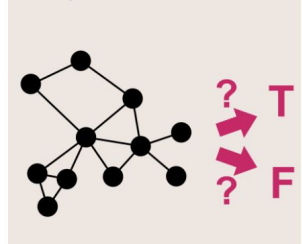
Graph Neural Networks



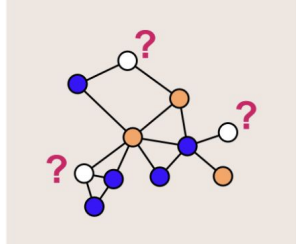
Applications



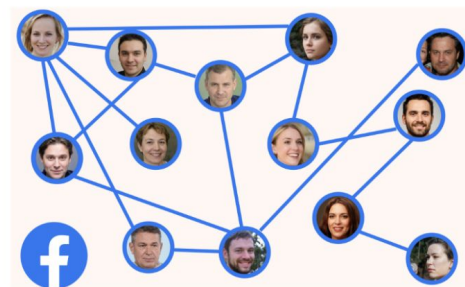
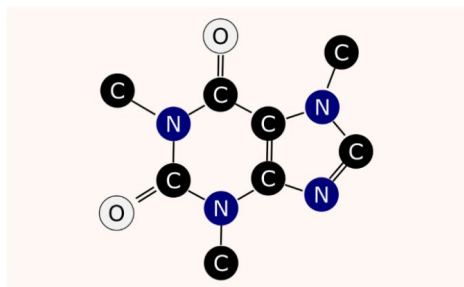
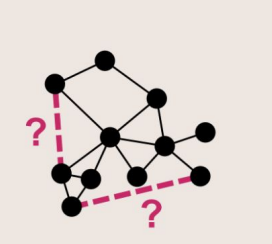
Graph Classification



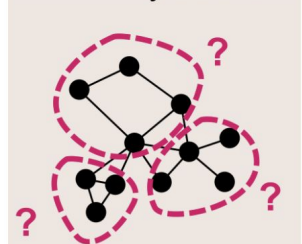
Node Classification



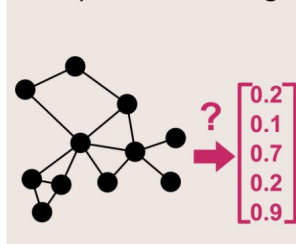
Link Prediction



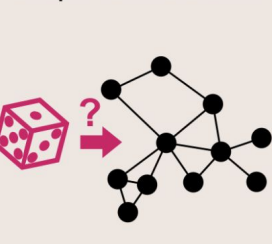
Community Detection



Graph Embedding



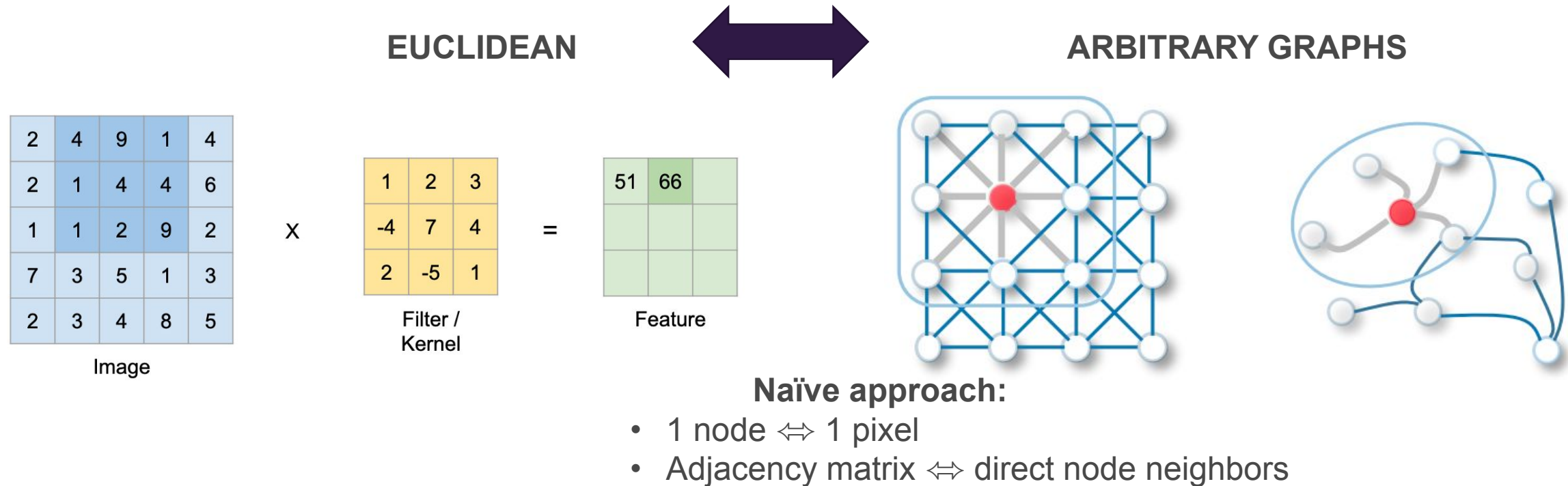
Graph Generation



Convolutional layer: from Images to Graph

Goal: Convolution operation considering node's local neighborhood to update node's features

Convolution from:

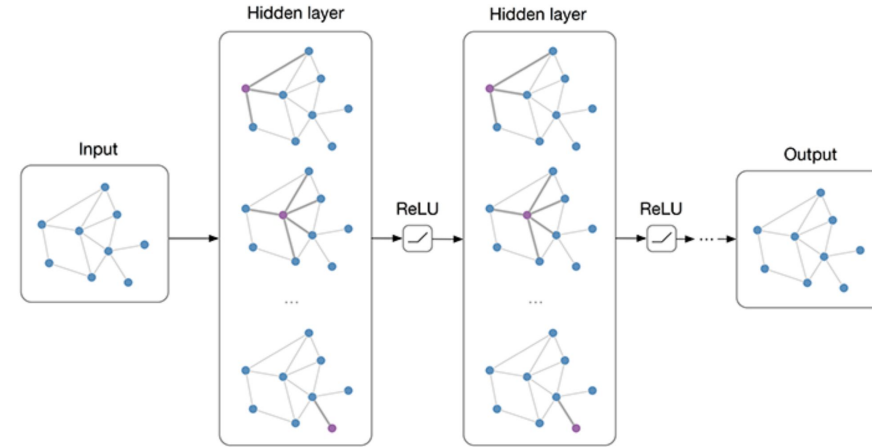


Convolution extracts spatial patterns and regularities in images/graphs

Graph Convolution network

Objective: design convolution operators that do not rely:

1. on regular structure
2. order of the node neighbors.



$$h'_i = \sigma_W(h_i, \{h_1, \dots, h_N\}) = \sigma\left(W^c h_i + \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} W_j^N h_j + \frac{1}{|\mathcal{N}_i|} \sum_{j \in \mathcal{N}_i} W_j^E A_{ij} + b\right)$$

Non-linear weighted combination of the node, neighboring nodes, edges features

Stack multiple layers of convolution:

- to increase the abstraction representation level learnt.
- to allow more information to propagate (region size increases).

Attention mechanism “Attention is all you need ☺”

Residual Attention Network for Image Classification

Fei Wang¹, Mengqing Jiang², Chen Qian¹, Shuo Yang³, Cheng Li¹,
Honggang Zhang⁴, Xiaogang Wang³, Xiaoou Tang³

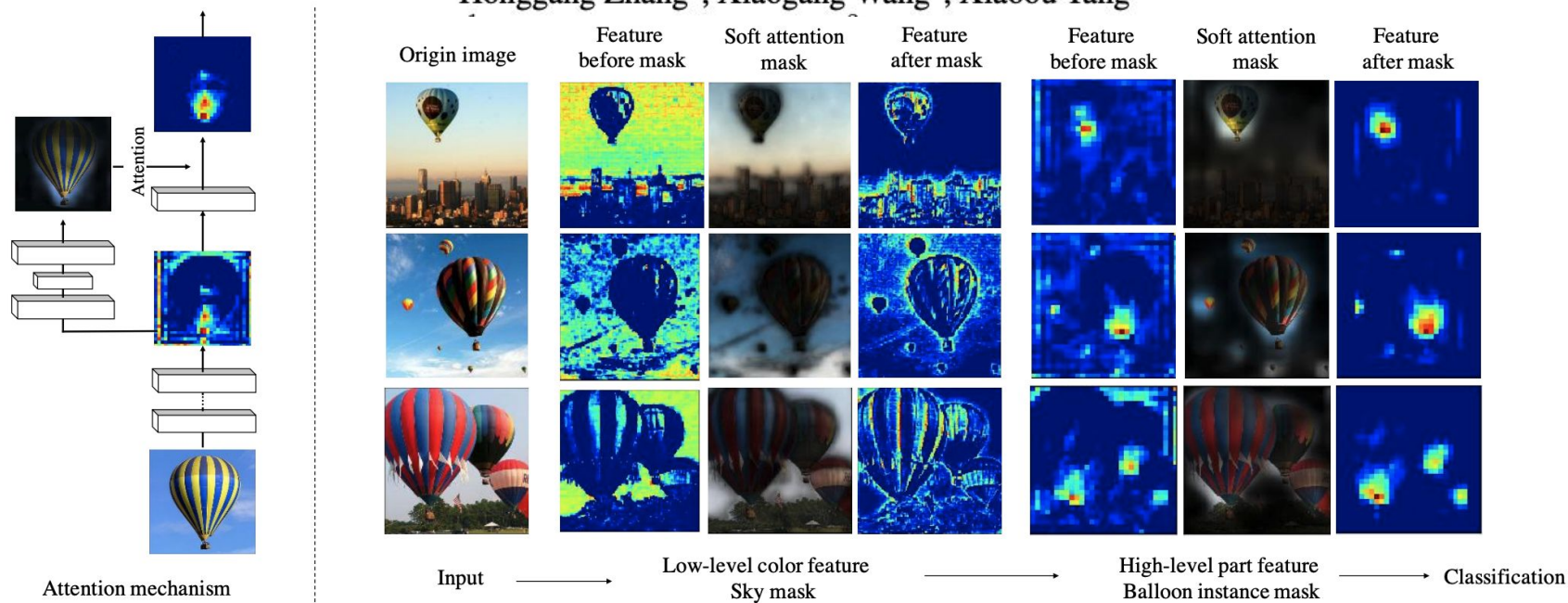
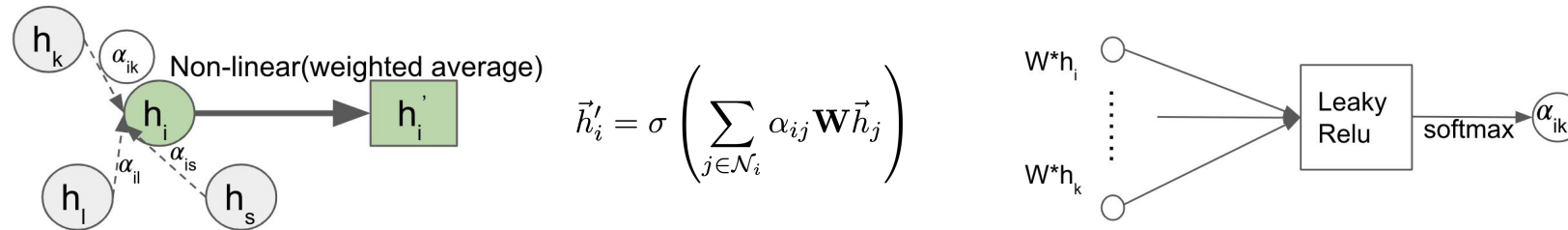


Figure 1: **Left:** an example shows the interaction between features and attention masks. **Right:** example images illustrating that different features have different corresponding attention masks in our network. The sky mask diminishes low-level background blue color features. The balloon instance mask highlights high-level balloon bottom part features.

Graph Node Attention Network

Compute hidden representations of graph node using self-attention strategy.

$$\mathbf{h} = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_N\}, \vec{h}_i \in \mathbb{R}^F \longrightarrow \mathbf{h}' = \{\vec{h}'_1, \vec{h}'_2, \dots, \vec{h}'_N\}, \vec{h}'_i \in \mathbb{R}^{F'}$$



Compute a linear combination of the features using the attention weights α_{ij} of node i 's neighbors and finally apply a non-linearity.

Node + Edge Attention Network

- Add a term that computes the aggregation at the edge level for the attention mechanism

$$\vec{h}'_i = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ij} \mathbf{W} \vec{h}_j \right) \longrightarrow \hat{h}_i = \sigma \left(\sum_{j \in \mathcal{N}_i} \alpha_{ji} W^\nu h_j + \sum_{j \in \mathcal{N}_i} \alpha_{ji} W^\varepsilon \xi_{ji} \right)$$

Graph Attention Network

Pros:

- **computationally efficient**: self-attention can be parallelized
- **interpretability** is possible through attention weight analysis
- attention mechanism is **independent** of graph structure

Cons:

- GPUs might not be efficient in sparse graphs
- Parallelization possible but likely to create **redundant computations** because of neighborhood overlapping.

SUMMARY GRAPH ATTENTION:

- operates on graph structured data
- does not depend on a fixed graph structure
- assigns different importance weights to different nodes within a neighborhood
- allows flexible sized neighborhoods

Fully automated training/testing pipeline: ML engineering and best practice

Search or jump to...

Pull requests

Issues

Marketplace

Explore

yanistazi / Graph_Neural_Net_Protein-Protein-Complexes

<> Code

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

Settings

main 1 branch 0 tags

Go to file

Add file

Code

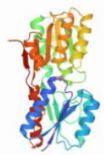
Yanis Tazi and Yanis Tazi DeepRank-GNN modified version a0ae58 30 minutes ago 4 commits

| | | |
|------------------------------------|--|----------------|
| DeepRank-GNN | DeepRank-GNN modified version | 30 minutes ago |
| Graph_Project | Protein-protein complexes quality assessment | 1 hour ago |
| hdf5_data_template/hdf5_pdb_gra... | Protein-protein complexes quality assessment | 1 hour ago |
| images | Protein-protein complexes quality assessment | 1 hour ago |
| .gitignore | Protein-protein complexes quality assessment | 1 hour ago |
| README.md | Protein-protein complexes quality assessment | 1 hour ago |
| environment_graph_predictions.yml | Protein-protein complexes quality assessment | 1 hour ago |

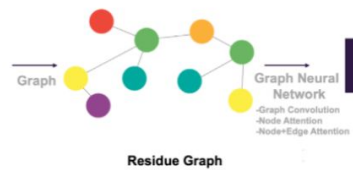
README.md

Graph Neural Network + Attention mechanism to predict scoring functions (i-RMSD) for protein complexes and decoys.


Tutorial for the data preparation, gridsearch training , testing and inference are available in this repository



PDB Protein



Residue Graph



Graph Neural Network

Prediction

1. Fully Automated data preparation pipeline that creates balanced graph datasets from PDB protein complexes and decoys files

About

Graph Neural Network (Convolution , Node Attention , Node+Edge Attention) to assess quality of protein-protein complexes.

Readme

Releases

No releases published
[Create a new release](#)

Packages

No packages published
[Publish your first package](#)

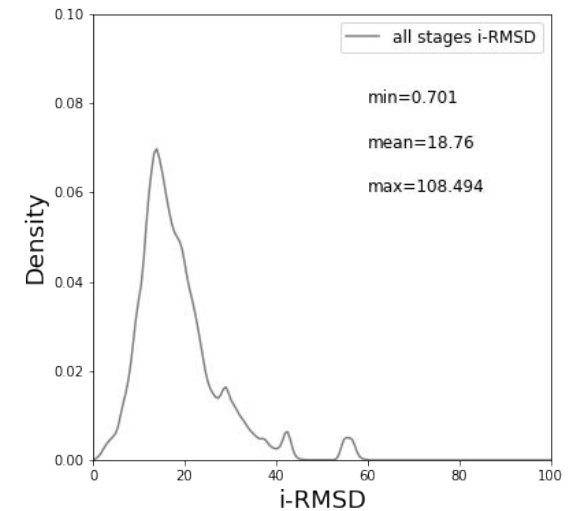
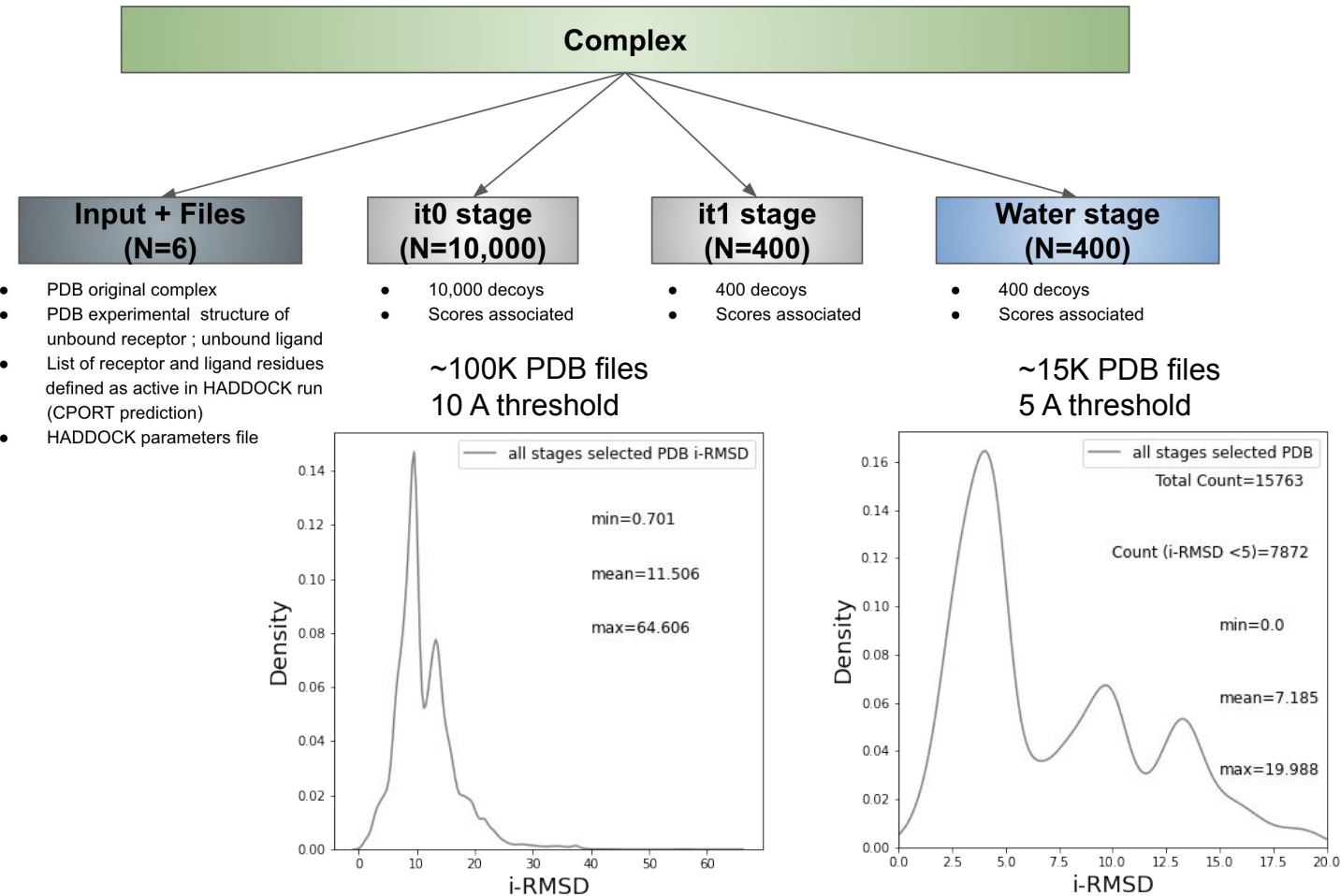
Languages

Jupyter Notebook 99.0%

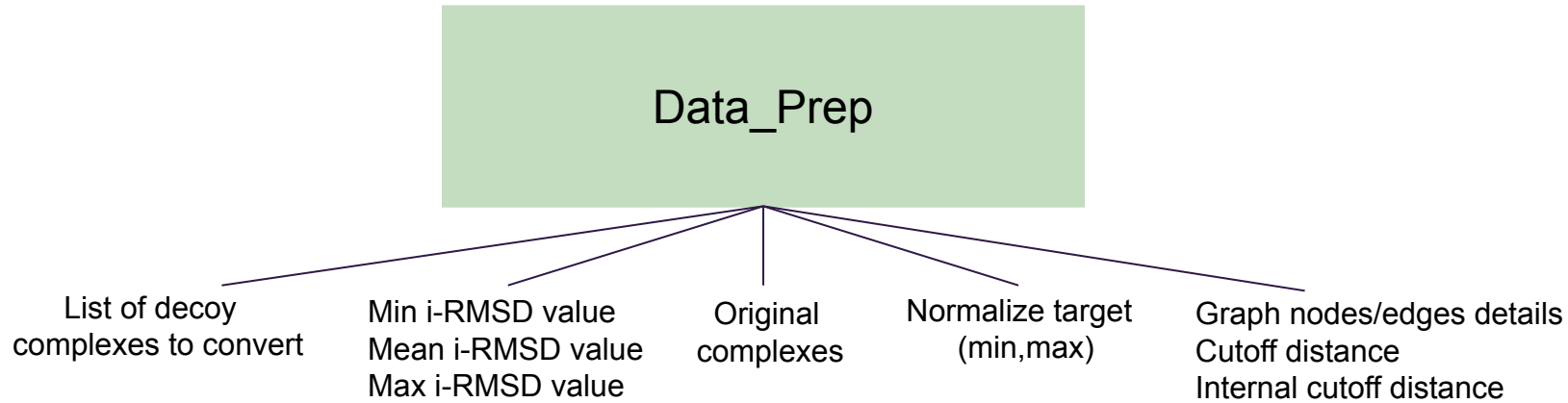
Python 1.0%

Original Data

- 600K decoys PDB files from 54 complexes in DB5 ; 300 G
- More data easily generated with simulations and docking



Data Preparation automation : creation of a balanced dataset



3 steps process:

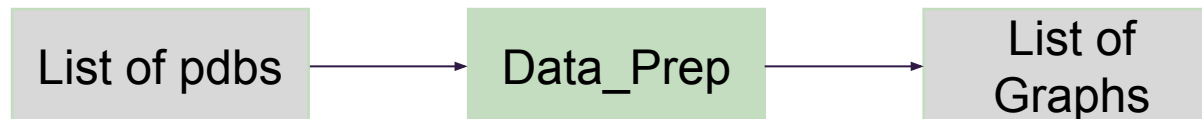
1.create_dict_selected_pdbs():

create a dictionary with the selected pdb files that follow the above criteria and their corresponding i-RMSD .

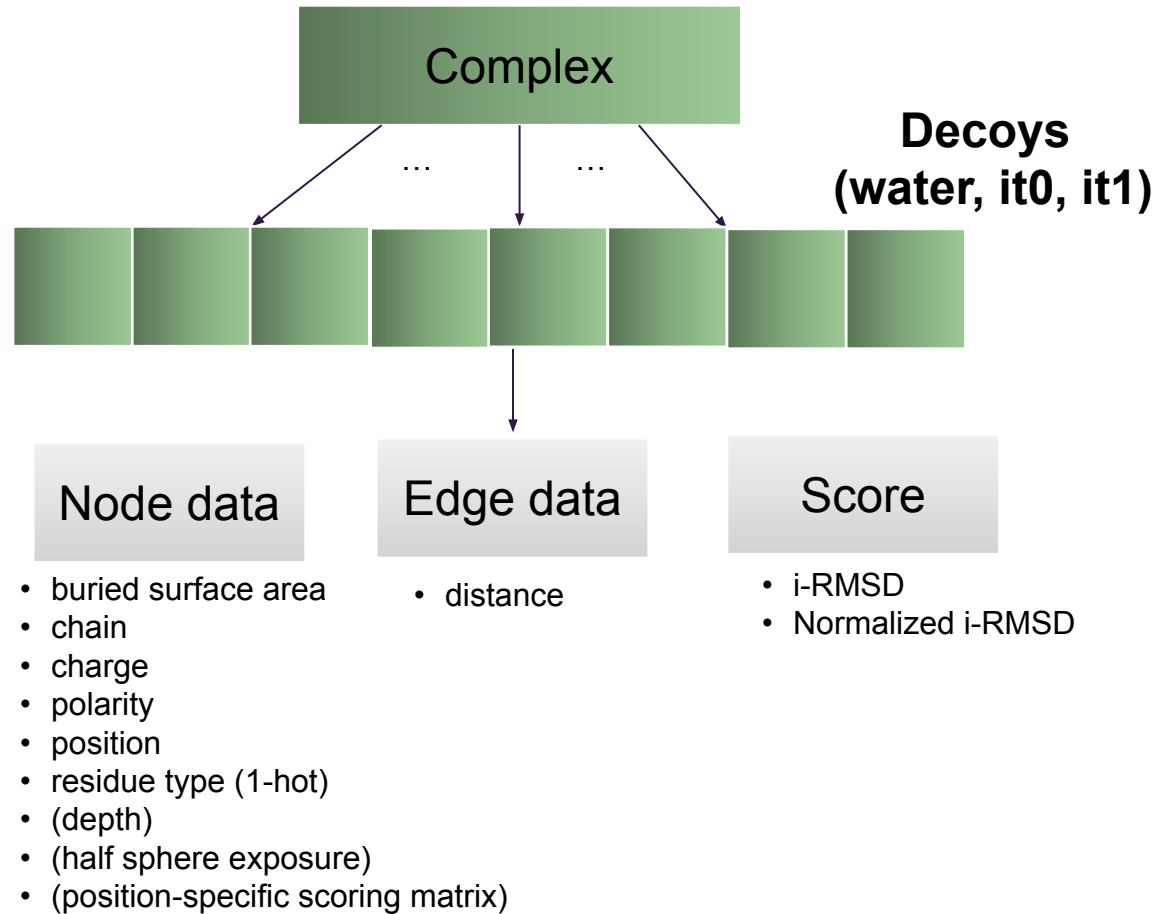
Example: {'1JTD_it1_complex_329.pdb': 14.98, '1JTD_it1_complex_359.pdb': 11.385}

2.prepare_pdb_selected_data(): Prepare pdb files and corresponding output (normalized)

3.prepare_hdf5_graphs(): Create the hdf5 graphs corresponding to the inputs

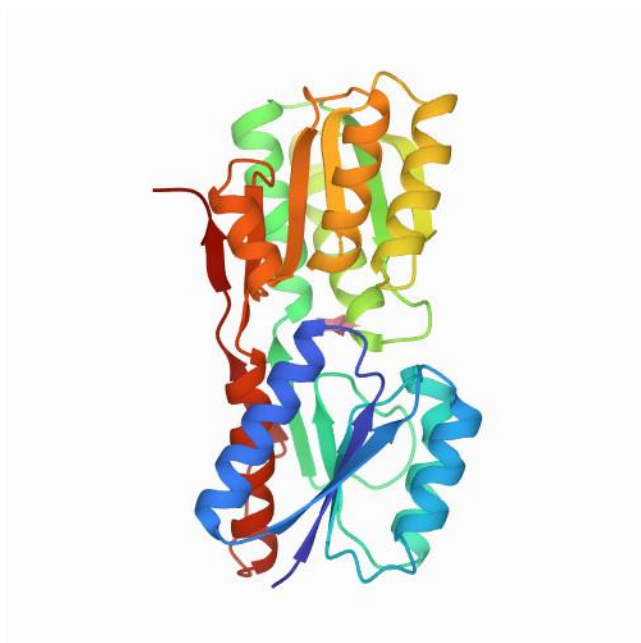
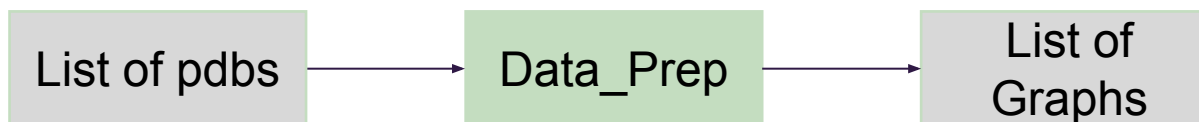


Protein as a graph



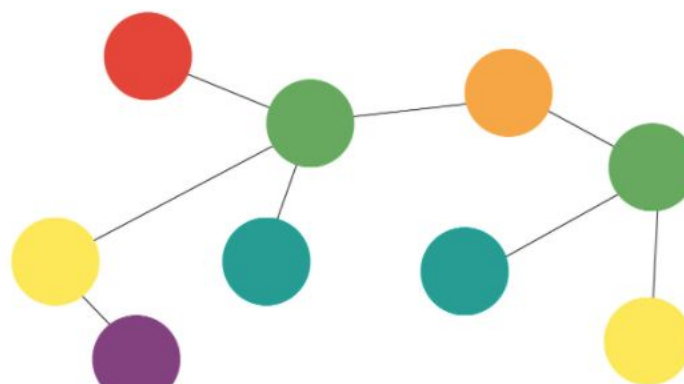
- Amino acid residues correspond to nodes with different properties
- Spatial relationships between different residues are characterized by the edges

PDB Graphs



PDB Protein

→
Graph



Residue Graph

Details about graph preparation process

1. Get dictionary with all contact residues pairs using cutoff contact distance(A,B)= 8.5A.

Example: {('A', 66, 'ASP'): [('B', 1071, 'ARG')],

 ('A', 67, 'GLU'): [('B', 1070, 'MET'), ('B', 1071, 'ARG')]}
□ ('A', 66, 'ASP') is in contact with ('B', 1071, 'ARG')

□ ('A', 67, 'GLU') is in contact with ('B', 1070, 'MET') and ('B', 1071, 'ARG')

2. Filter valid residue nodes using predefined set of names:

['ALA', 'ARG', 'ASN', 'ASP', 'CYS', 'GLU', 'GLN', 'GLY', 'HIS', 'ILE', 'LEU', 'LYS', 'MET', 'PHE', 'PRO', 'SER', 'THR', 'TRP', 'TYR', 'VAL', 'ASX', 'SEC', 'GLX']

3. Create interface edges between chain A and chain B

(compute distance for all residue contact pairs). Residue type = interface (between chain no threshold). Returns a distance for edge

4. Create internal edges for all residues per chain and add them to the edge if distance < internal_cutoff=3

Residue type = internal (within chain with threshold). Returns a distance for the edge.

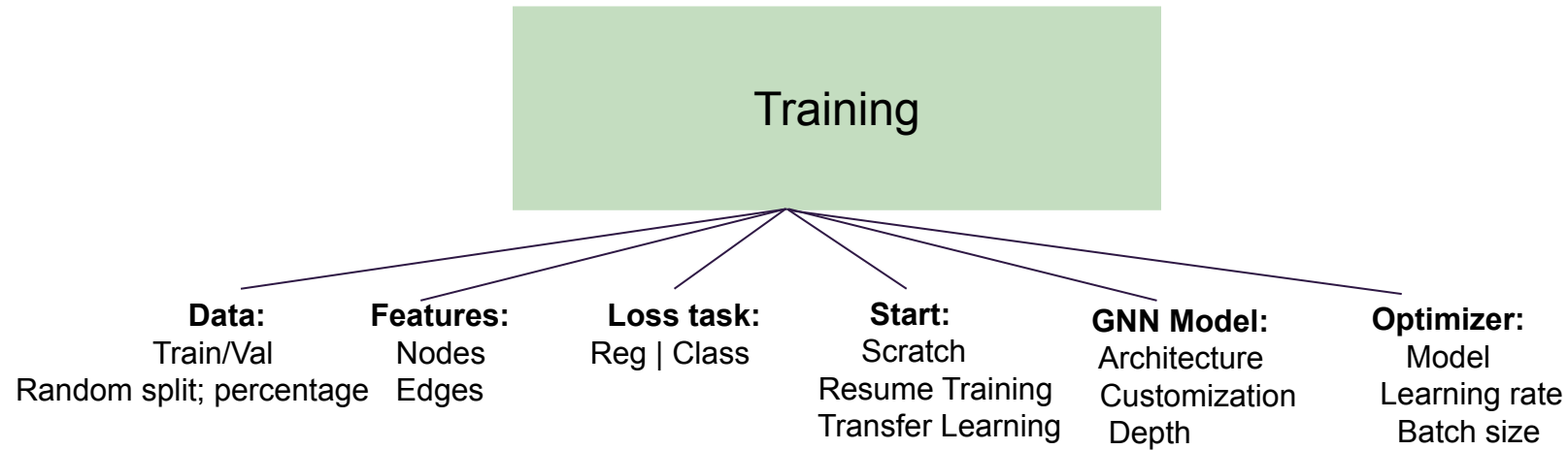
5. Prepare node features: chain, pos, residue type, charge, polarity, bsa

Optional (depth, hse ; if pssm data (pssm, cons, ic))

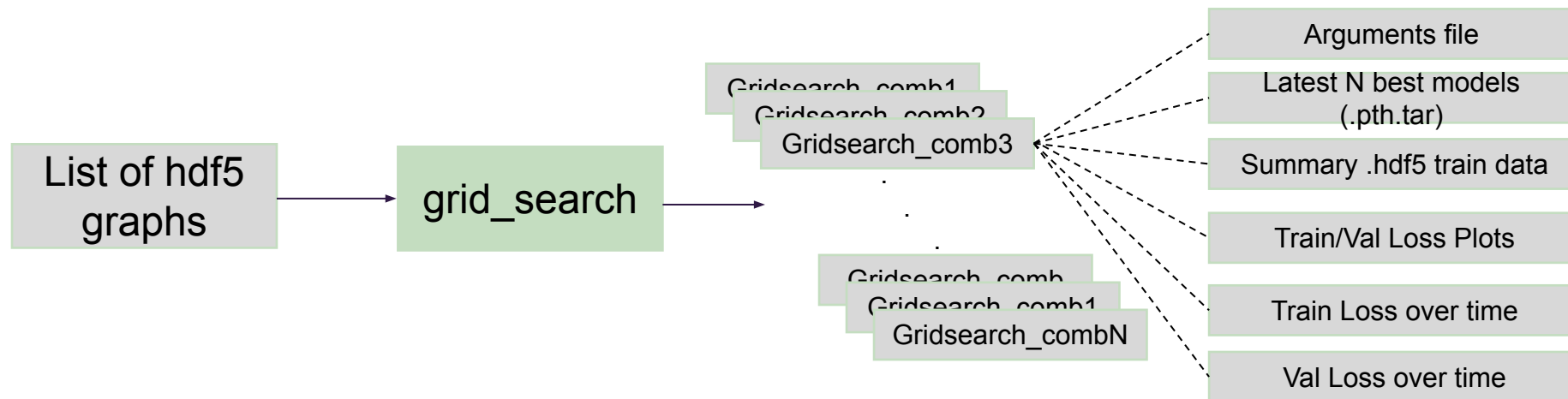
6. Prepare edge features: distance and type

6.(Optional): Add target (i-RMSD values in this case)

Training and gridsearch automation



grid_search(): performs training for all possible parameter combinations. It creates one folder per combinations and help in the optimal model selection by evaluating validation loss.



Main hyperparameters

Graph Neural Network

Convolution

```
(conv1): FoutLayer(30, 16)
(conv2): FoutLayer(16, 32)
(fc1): Linear(in_features=32, out_features=64, bias=True)
(fc2): Linear(in_features=64, out_features=1, bias=True)
```

4.2K params

Node Attention

```
(conv1): GraphAttention(30, 16)
(conv2): GraphAttention(16, 32)
(fc1): Linear(in_features=32, out_features=64, bias=True)
(fc2): Linear(in_features=64, out_features=1, bias=True)
```

4.2K params

Edge + Node Attention

```
(conv1): EGAT(30, 16)
(conv2): EGAT(16, 32)
(conv1_ext): EGAT(30, 16)
(conv2_ext): EGAT(16, 32)
(fc1): Linear(in_features=64, out_features=128, bias=True)
(fc2): Linear(in_features=128, out_features=1, bias=True)
```

10.6K params

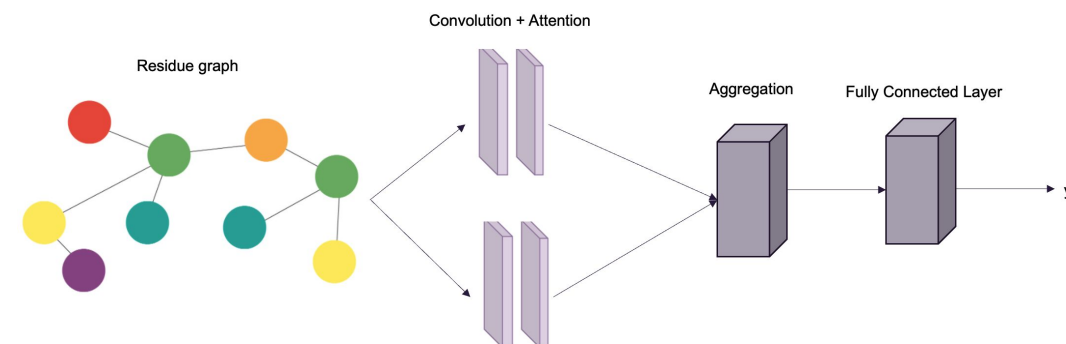
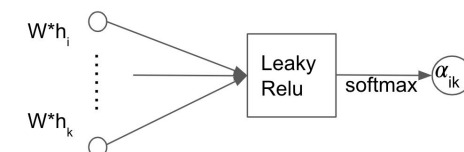
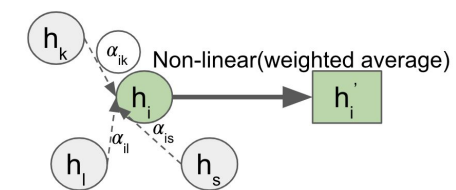
Customizable architecture

```
(conv1): GraphAttention(30, 16)
(conv2): GraphAttention(16, 32)
(fc1): Linear(in_features=32, out_features=64, bias=True)
(fc2): Linear(in_features=64, out_features=128, bias=True)
(fc3): Linear(in_features=128, out_features=1, bias=True)
```

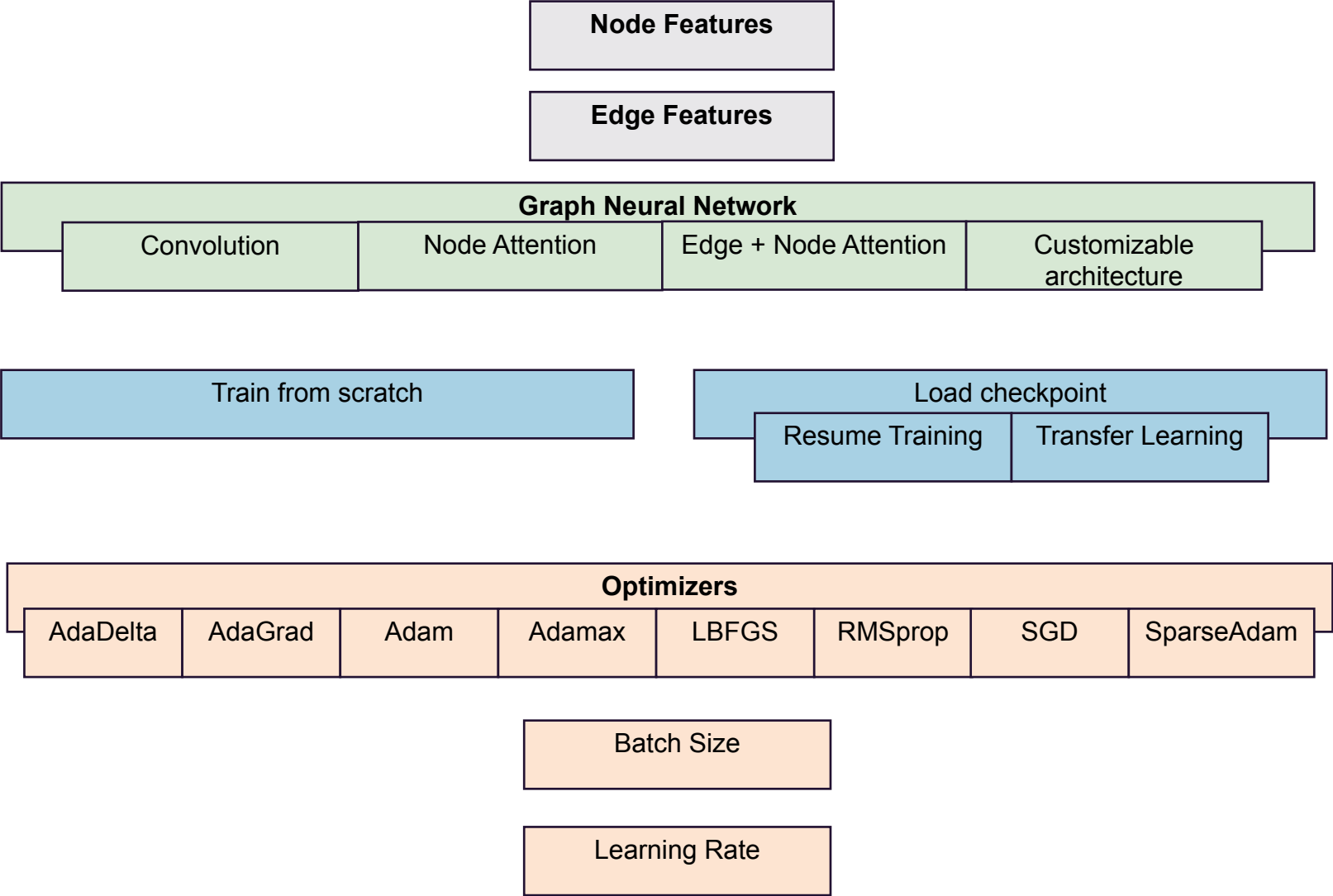
12.6 K params

```
(conv1): GraphAttention(30, 16)
(conv2): GraphAttention(16, 32)
(conv3): GraphAttention(32, 64)
(fc1): Linear(in_features=64, out_features=128, bias=True)
(fc2): Linear(in_features=128, out_features=256, bias=True)
(fc3): Linear(in_features=256, out_features=1, bias=True)
```

48 K params



Main hyperparameters



Why?

Table 2. A comparison of the predictive performance of our proposed EGRET and GAT-PPI with other state-of-the-art methods on the benchmark dataset. The best and the second best results for each metric are shown in bold and italic, respectively. Values which were not reported by the corresponding source are indicated by “-”. PPI site predictions

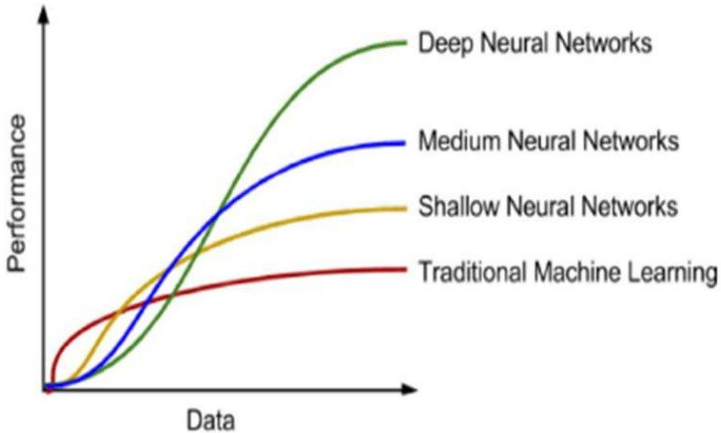
| Method | ACC | Precision | Recall | F1 | AUROC | AUPRC | MCC |
|--------------------------|--------------|--------------|--------------|--------------|--------------|--------------|--------------|
| SPPIDER ^{1,2} | 0.622 | 0.209 | 0.459 | 0.287 | - | 0.23 | 0.089 |
| ISIS ² | <i>0.694</i> | 0.211 | 0.362 | 0.267 | - | 0.24 | 0.097 |
| PSIVER ² | 0.653 | 0.253 | 0.468 | 0.328 | - | 0.25 | 0.138 |
| SPRINGS ² | 0.631 | 0.248 | 0.598 | 0.35 | - | 0.28 | 0.181 |
| RF_PPI ² | 0.598 | 0.173 | 0.512 | 0.258 | - | 0.21 | 0.118 |
| IntPred ^{1,2} | 0.672 | 0.247 | 0.508 | 0.332 | - | - | 0.165 |
| SCRIBER | 0.616 | 0.274 | 0.569 | 0.37 | 0.635 | 0.307 | 0.159 |
| DeepPPISP ^{1,2} | 0.655 | 0.303 | 0.577 | 0.397 | 0.671 | 0.32 | 0.206 |
| DELPHI | 0.667 | <i>0.32</i> | <i>0.604</i> | 0.418 | 0.69 | 0.36 | 0.236 |
| GAT-PPI ¹ | 0.653 | 0.318 | 0.659 | <i>0.429</i> | <i>0.714</i> | <i>0.398</i> | <i>0.252</i> |
| EGRET ¹ | 0.715 | 0.358 | 0.561 | 0.438 | 0.719 | 0.405 | 0.27 |

¹ Uses structural information.
² Results reported by DeepPPISP Zeng *et al.* (2020).

Model

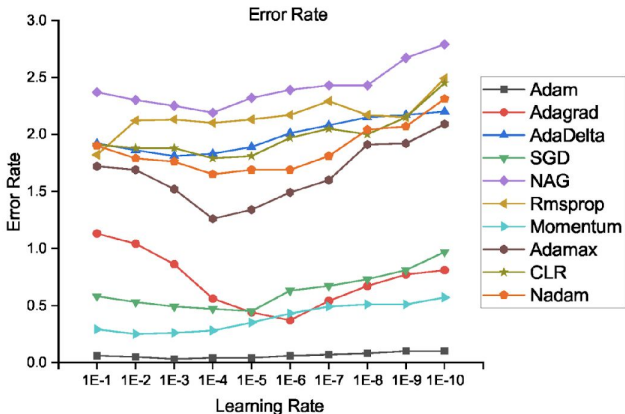
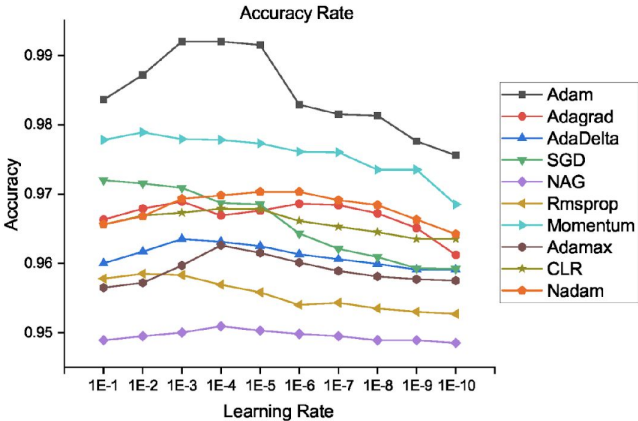
Node attention

Edge + Node attention



Architecture

Optimizers

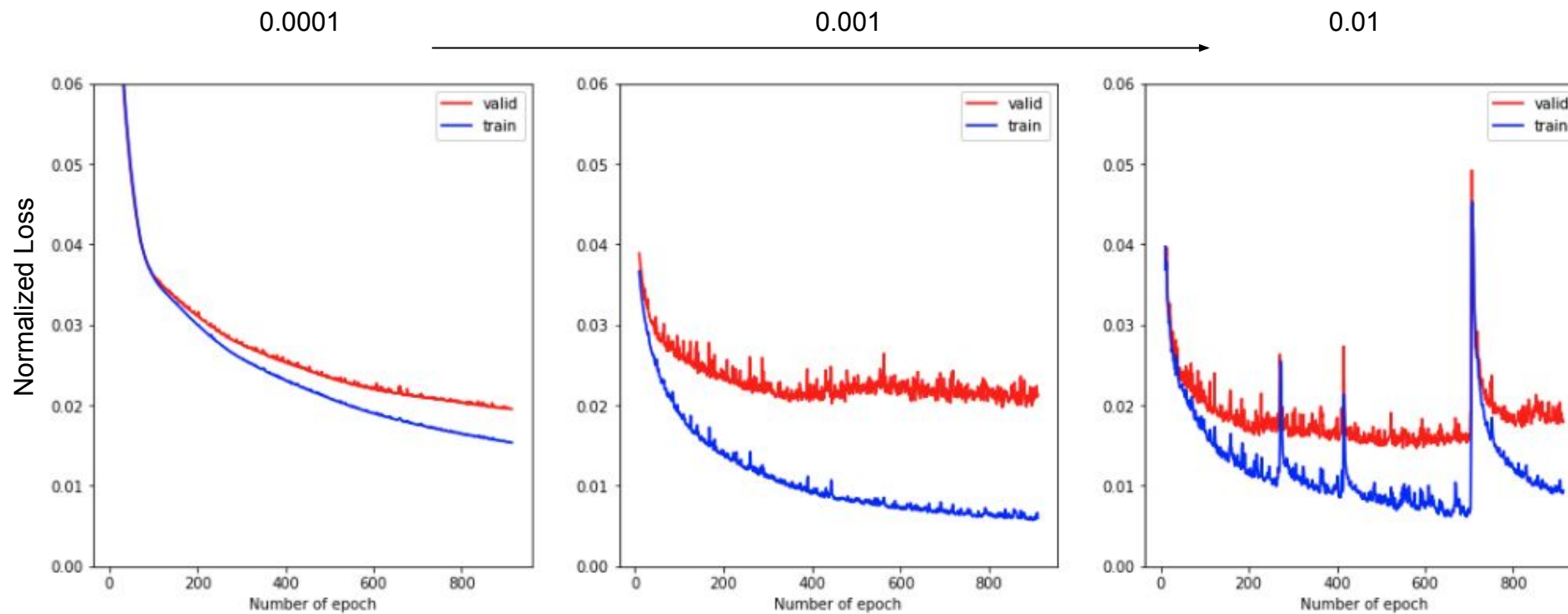


Other hyperparameters

- Graph Preparation (distance to create nodes/edges)
- Loss function
- complexes to train/validate on

Learning Rate Effect on Graph Node attention Model

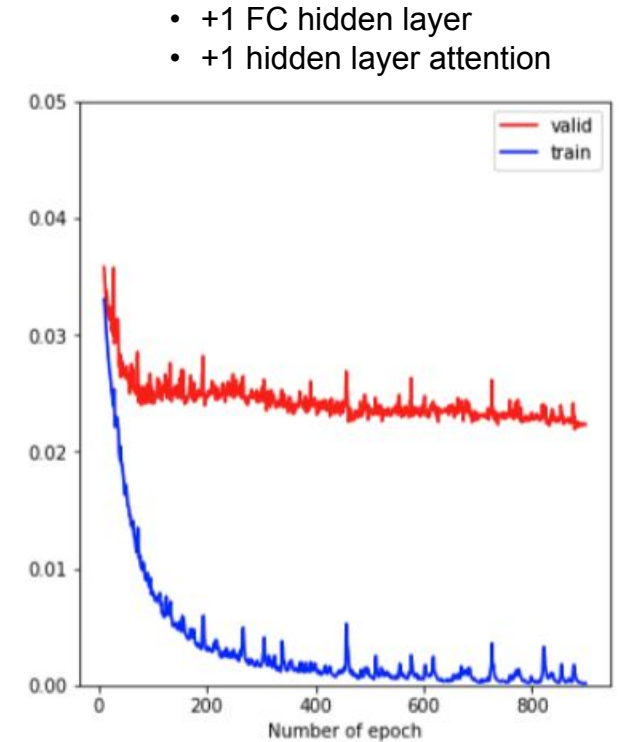
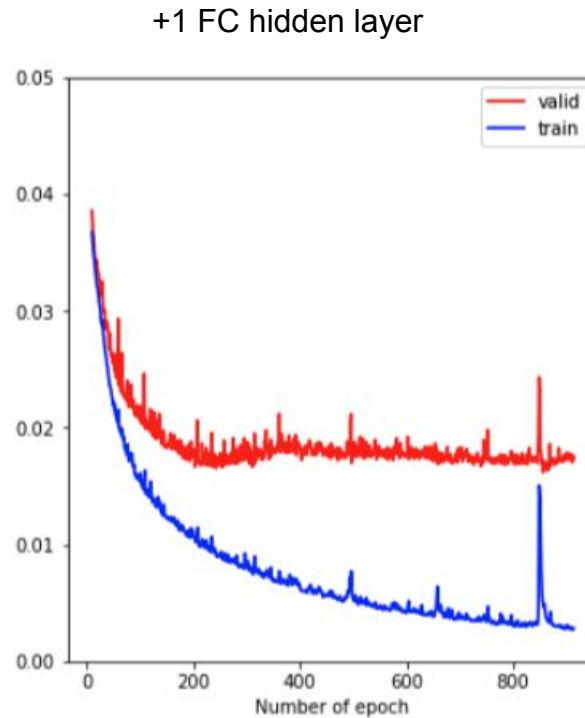
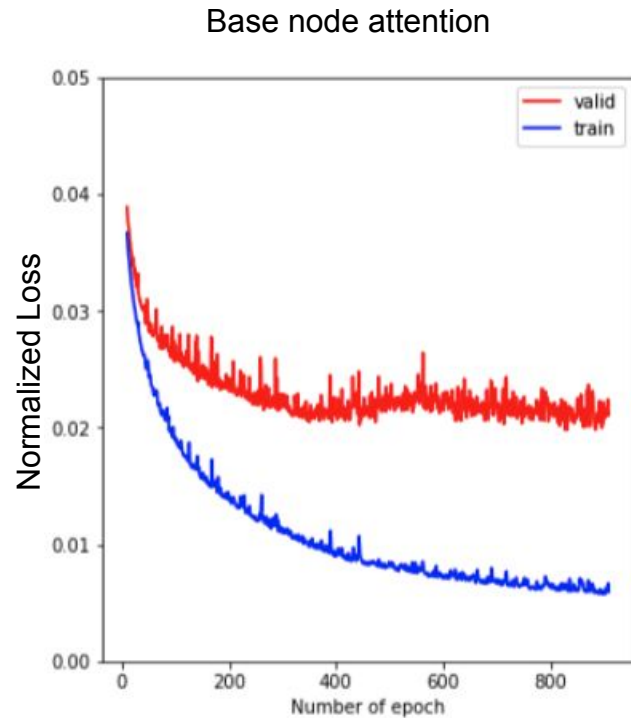
All other parameters equal



Causes drastic updates;
Divergent behaviors

Model Depth Effect for Node attention architecture

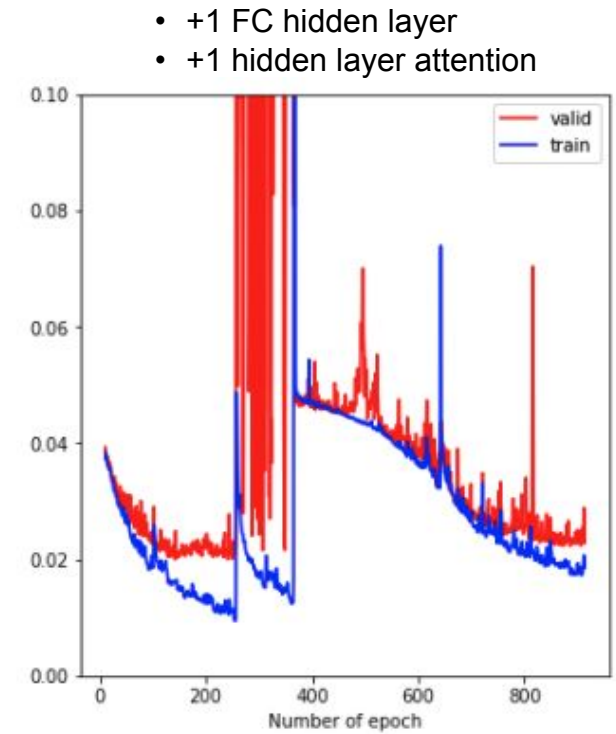
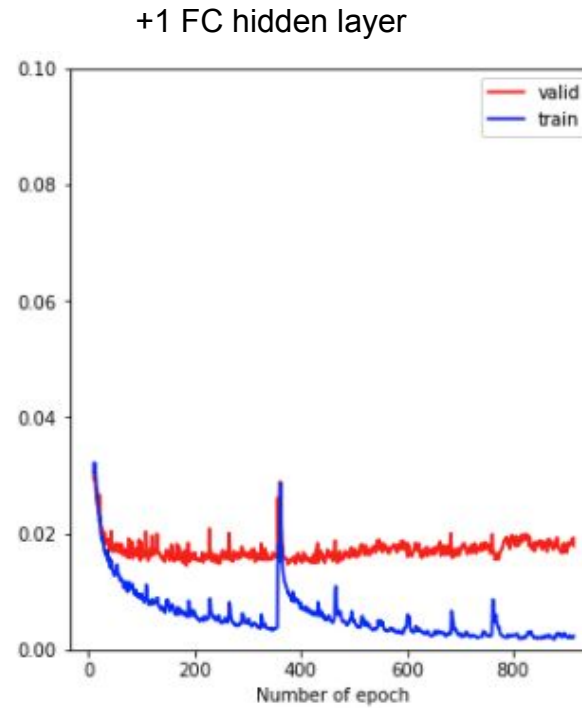
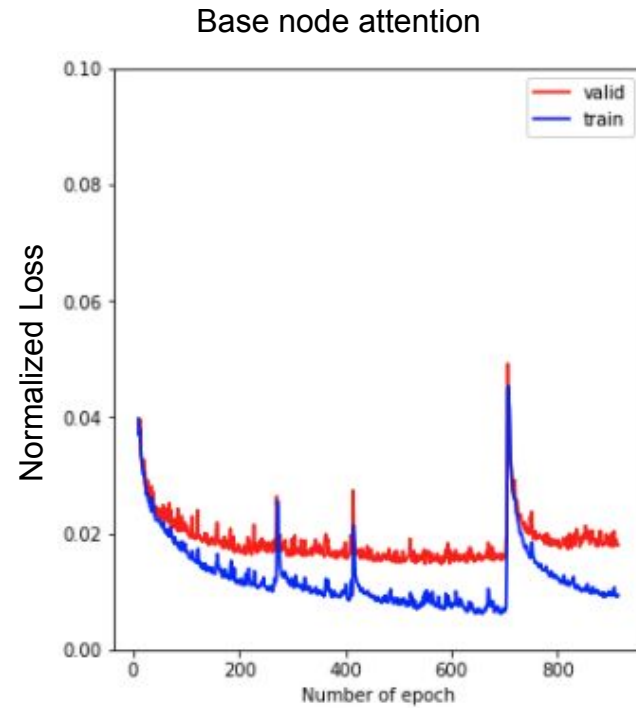
All other parameters equal



Effect of high learning rate on Node attention architecture

All other parameters equal

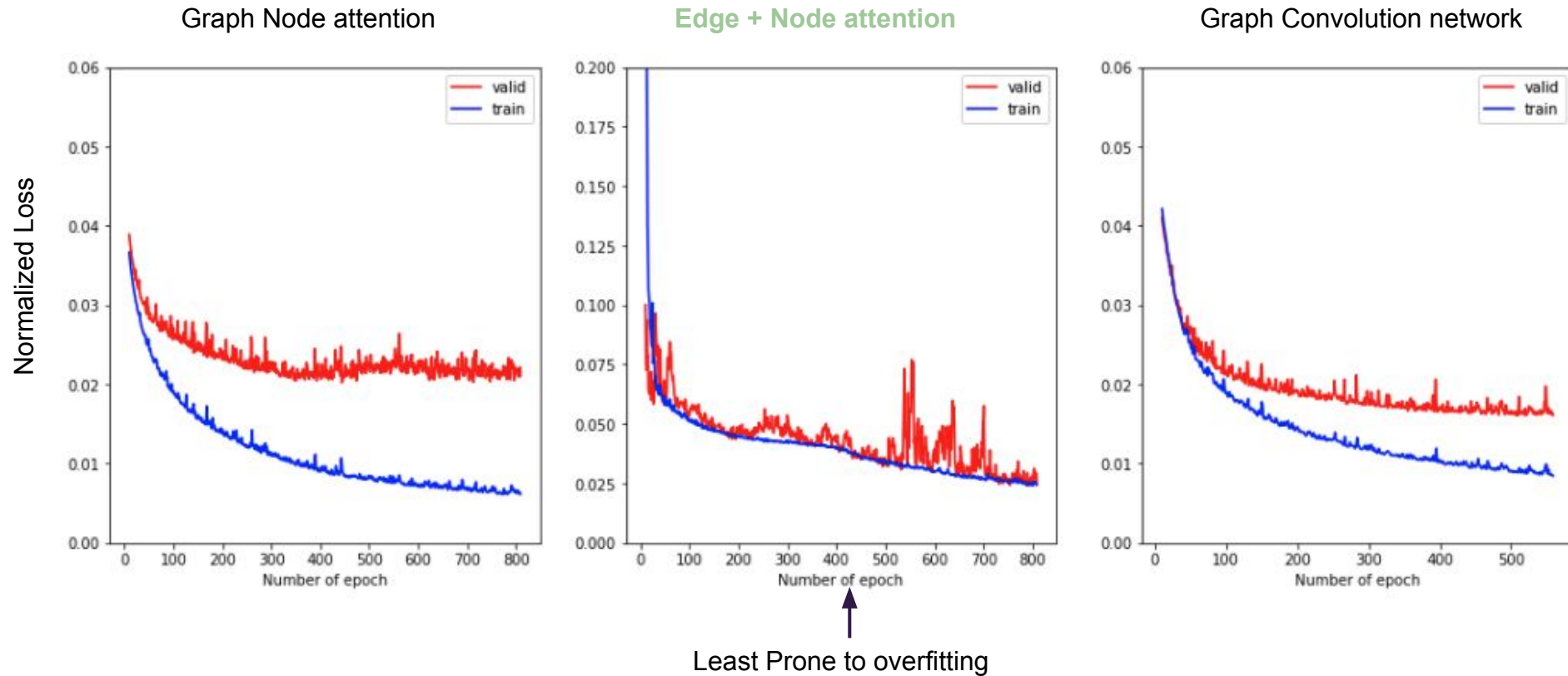
$lr=0.01$



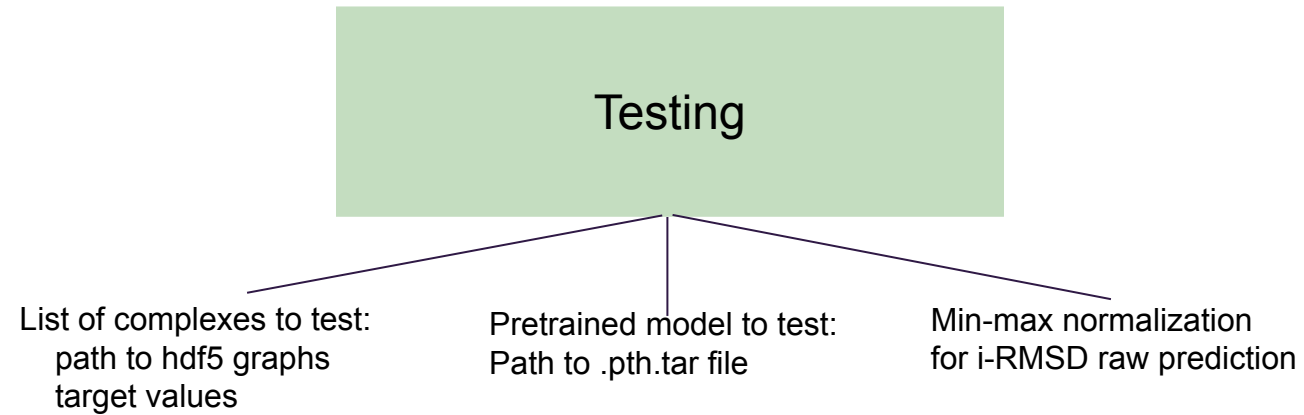
Effect of Model Types

All other parameters equal

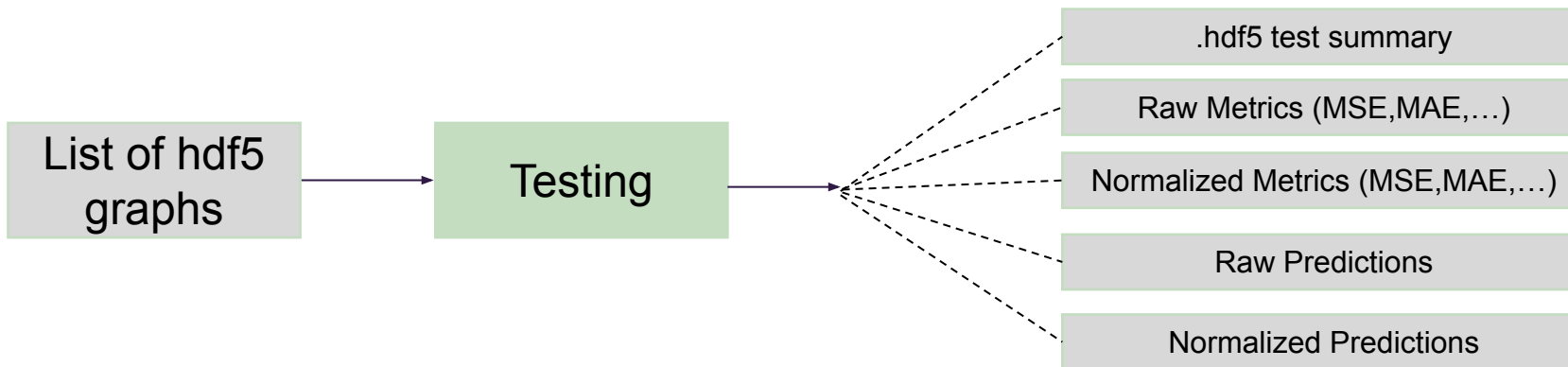
Comparison of three architectures



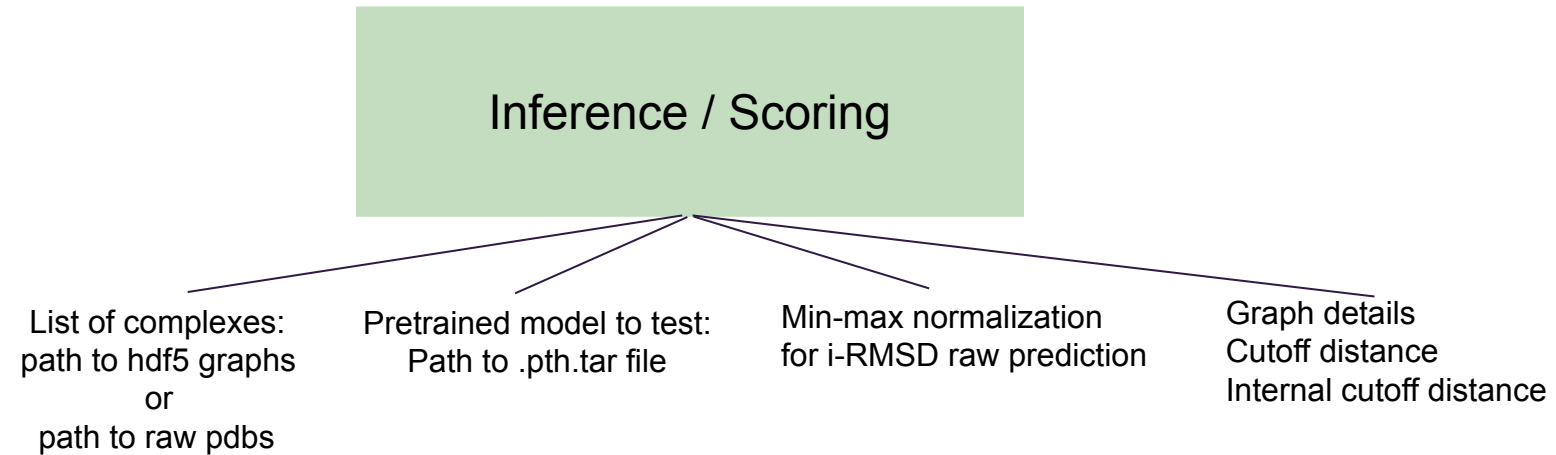
Testing automation



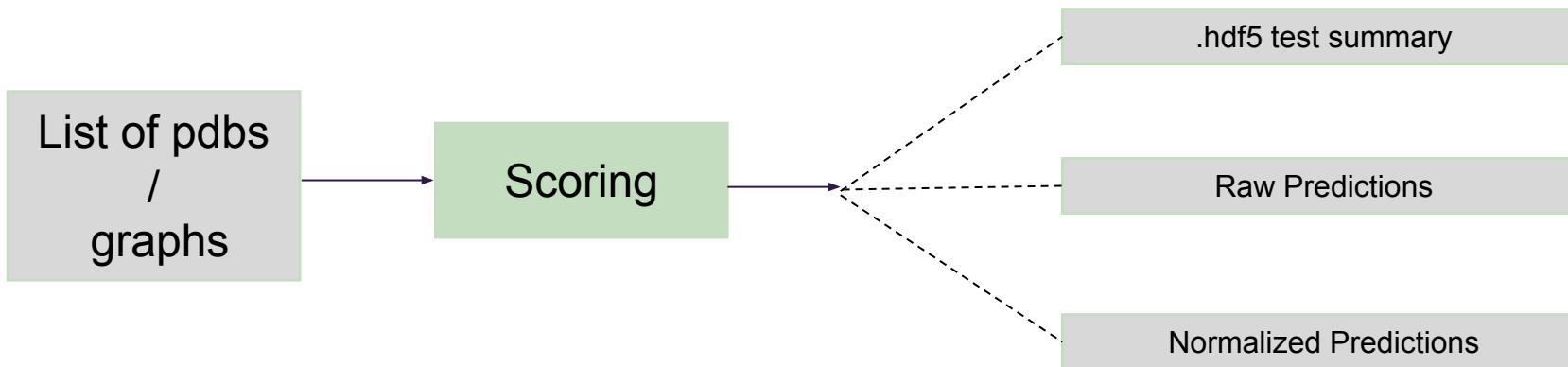
model_testing(): test any trained model and return summary , predictions and metrics



Inference / Scoring automation



model_new_predictions(): it returns normalized and raw predicted i-RMSD values from the raw pdbs or the .hdf5 graphs



Results

- Trained/validated on **15K pdb files** from DB5 from **52 complexes**
- Tested on **11K pdb files** from **3 new complexes**: 3PC8, 3K75 from DB5 and 3HMX

X-ray crystal structure of the heterodimeric complex of XRCC1 and DNA ligase III-alpha BRCT domains.

DOI: [10.2210/pdb3PC8/pdb](https://doi.org/10.2210/pdb3PC8/pdb)

Classification: **DNA BINDING PROTEIN/LIGASE**

Organism(s): *Mus musculus*, *Homo sapiens*

Expression System: *Escherichia coli* BL21(DE3)

Mutation(s): Yes ⓘ

Deposited: 2010-10-21 Released: 2011-06-15

Deposition Author(s): Cuneo, M.J., Krahn, J.M., London, R.E.



3PC8

X-ray crystal structure of reduced XRCC1 bound to DNA pol beta catalytic domain

DOI: [10.2210/pdb3K75/pdb](https://doi.org/10.2210/pdb3K75/pdb)

Classification: **DNA BINDING PROTEIN**

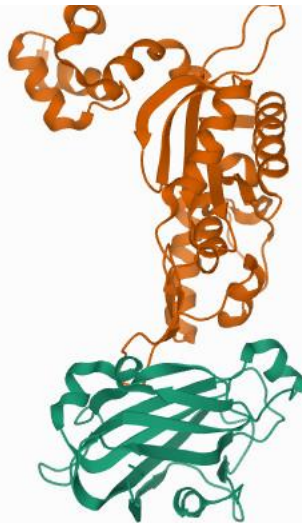
Organism(s): *Homo sapiens*, *Rattus norvegicus*

Expression System: *Escherichia coli*

Mutation(s): No ⓘ

Deposited: 2009-10-12 Released: 2010-04-28

Deposition Author(s): Cuneo, M.J., London, R.E.



3K75

Crystal structure of ustekinumab FAB/IL-12 complex

DOI: [10.2210/pdb3HMX/pdb](https://doi.org/10.2210/pdb3HMX/pdb)

Classification: **Cytokine/IMMUNE SYSTEM**

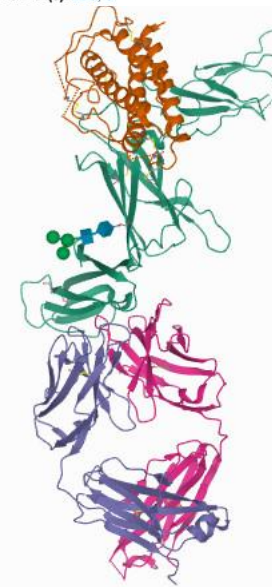
Organism(s): *Homo sapiens*

Expression System: *Homo sapiens*

Mutation(s): No ⓘ

Deposited: 2009-05-29 Released: 2010-06-09

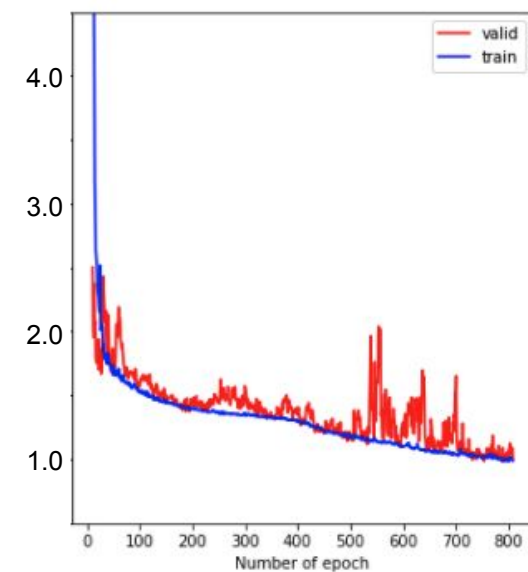
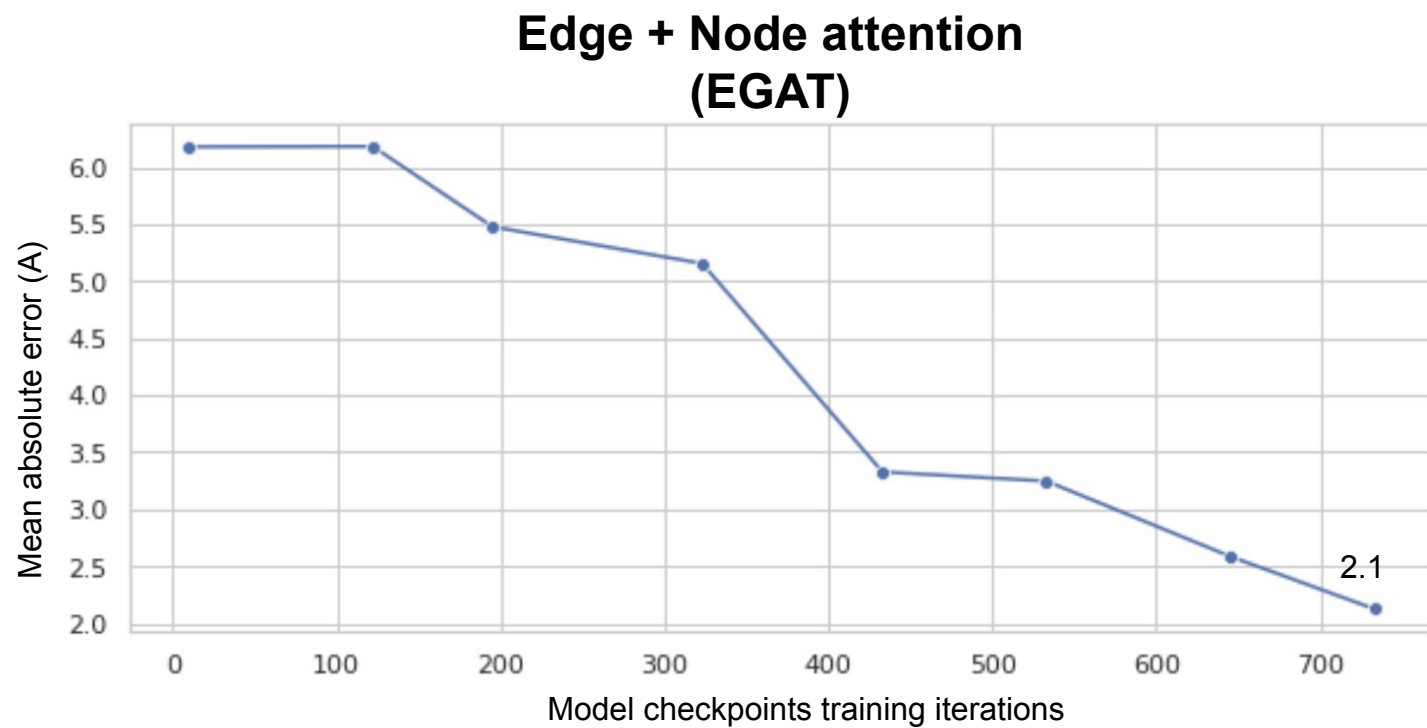
Deposition Author(s): Luo, J.



3HMX

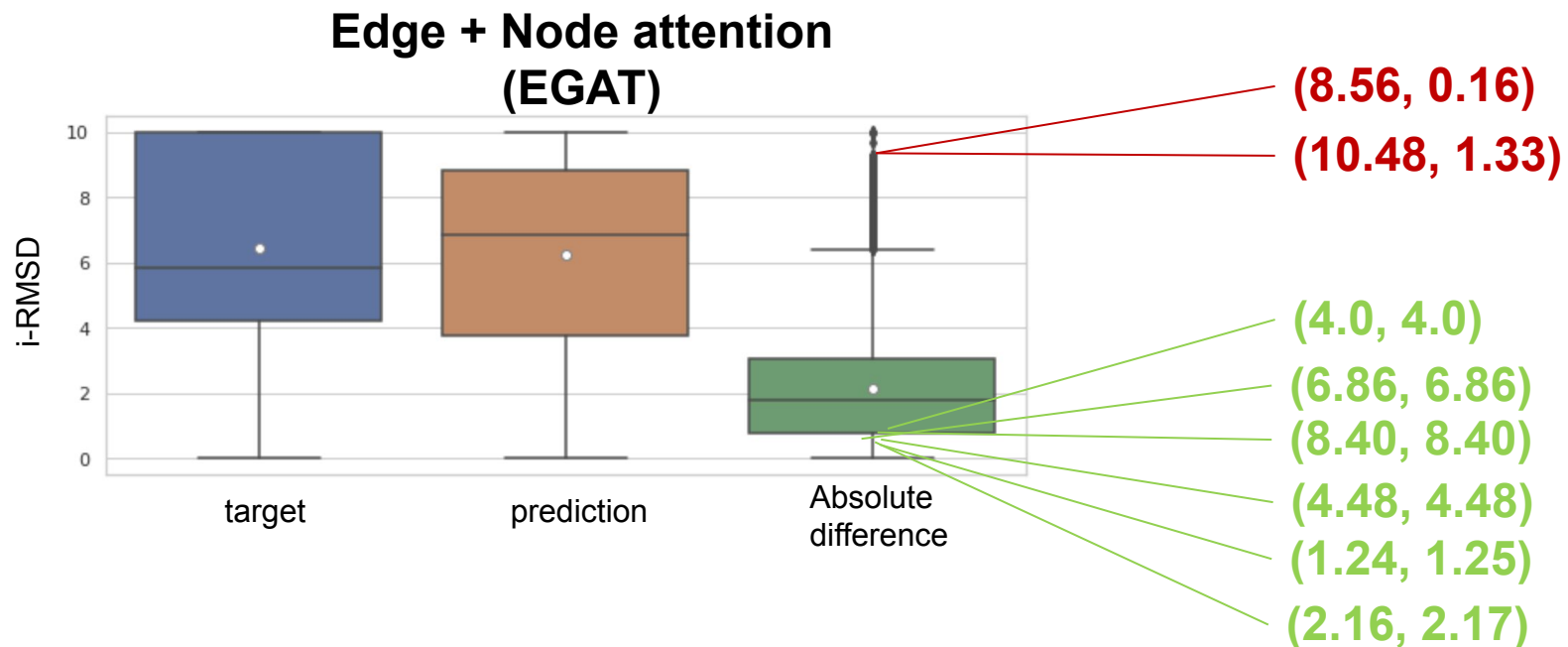
Results

- Trained/validated on **15K pdb files** from DB5 from **52 complexes**
- Tested on **11K pdb files** from **3 new complexes**: 3PC8, 3K75 from DB5 and 3HMX



Zoom on best/worst performing decoys

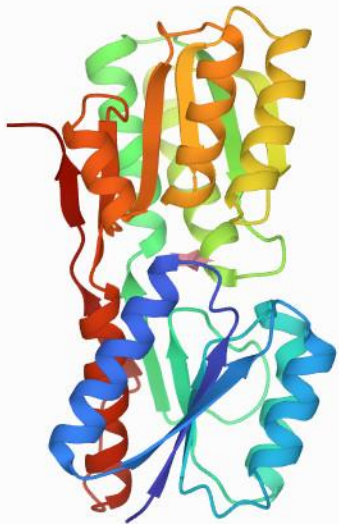
- Trained/validated on **15K pdb files** from DB5 from **52 complexes**
- Tested on **11K pdb files** from **3 new complexes**: 3PC8, 3K75 from DB5 and 3HMX



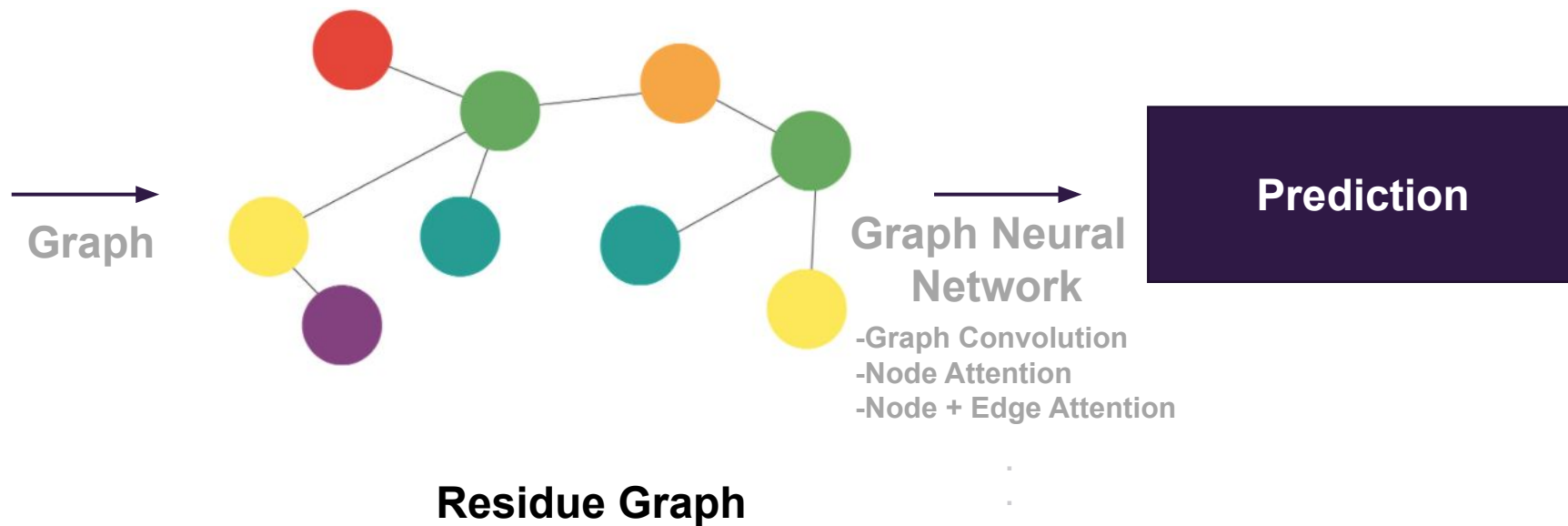
Challenges

- Training can be very long to reach optimal performances (not reached yet).
(several weeks)
- Interruption due to memory issues . Advsim nodes have max 96GB.
 - □ migration to alphafold nodes with 512 GB
- Needs to have efficient parallelization process

Summary



PDB Protein



- Very encouraging results on 11K pdb files from 3 new complexes within only a few weeks of experimentation
- Importance of user-friendly fully automated / customizable ML framework pipeline (tutorial available) that:
 - prepare the data
 - train and test the models
 - predict the scoring on new structures
 - allow reproducibility of the results

Extensions/Future work

1. Augmented Training dataset:

- with docking: lots of negative examples with high i-RMSD
- with molecular dynamics: closer to the complex of interest
- from existing datasets (like DB5)

2. Features:

- pssm score of residues
 - average atom depth of atoms in a residue
 - Half sphere exposure
- } 10* more computation time

3. Hyperparameter search space:

- Genetic algorithms for gridsearch
- Multi-GPU training for speed performance improvement

4. Make use of pretrained models:

- Find pretrained model from the literature to use as a feature embedding

5. Model Performances:

- Explore deeper version of the Edge+Node Attention Network (see customizable architecture)
- Ensemble methods

6. Scoring Functions:

- Δ_{Energy} (Complex,Decoy) as a scoring function