

# GraphNNs Practice

# Installation Instructions

FOR ALL:

```
source /opt/anaconda/bin/activate root
conda deactivate
```

CPU/GPU (for cuda 10; adapt the command to your own config):

```
conda create -n graphnn_gpu_env python=3.6 anaconda -y && \
conda activate graphnn_gpu_env && \
conda install pytorch=1.4.0 cudatoolkit=10.1 -c pytorch -y && \
conda install -c dglteam dgl-cuda10.1 -y
```

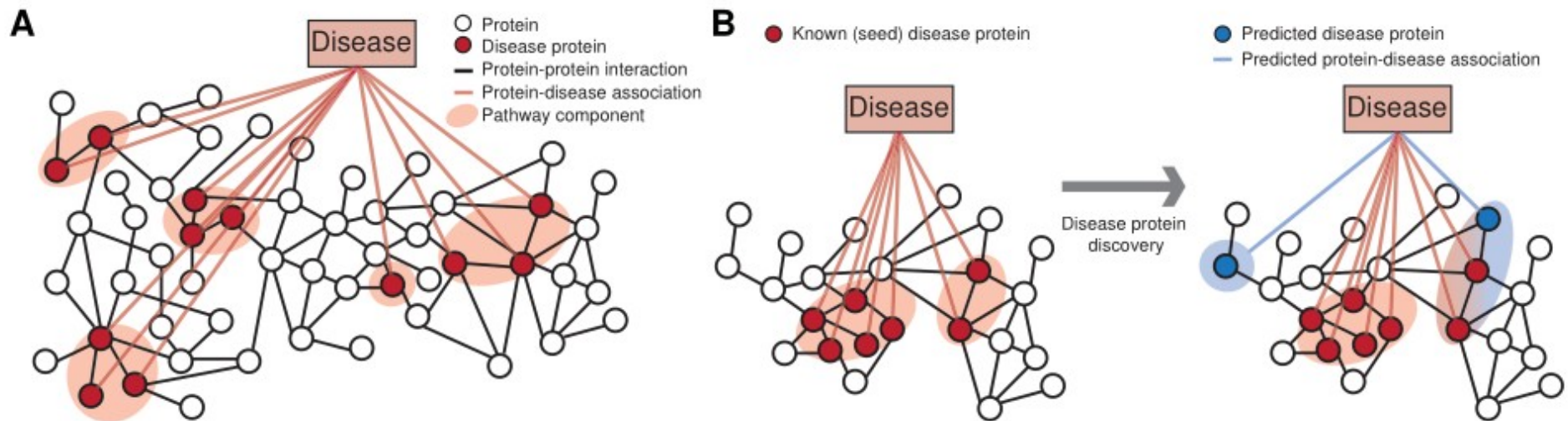
CPU ONLY (should work in all cases):

```
conda create -n graphnn_cpu_env python=3.6 anaconda -y && \
conda activate graphnn_cpu_env && \
conda install -c pytorch pytorch-cpu -y && \
conda install -c dglteam dgl -y
```

# Description of the dataset

20 graphs for training, 2 graphs for test  
2372 nodes on average per graph  
Each node has 50 features and 121 labels

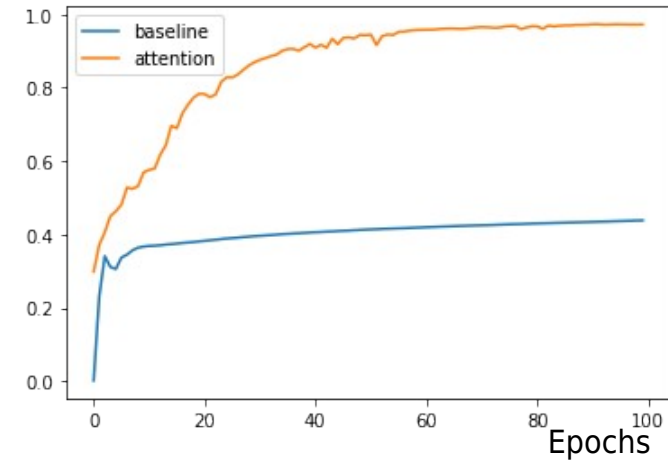
<https://cs.stanford.edu/~jure/pubs/pathways-psb18.pdf>



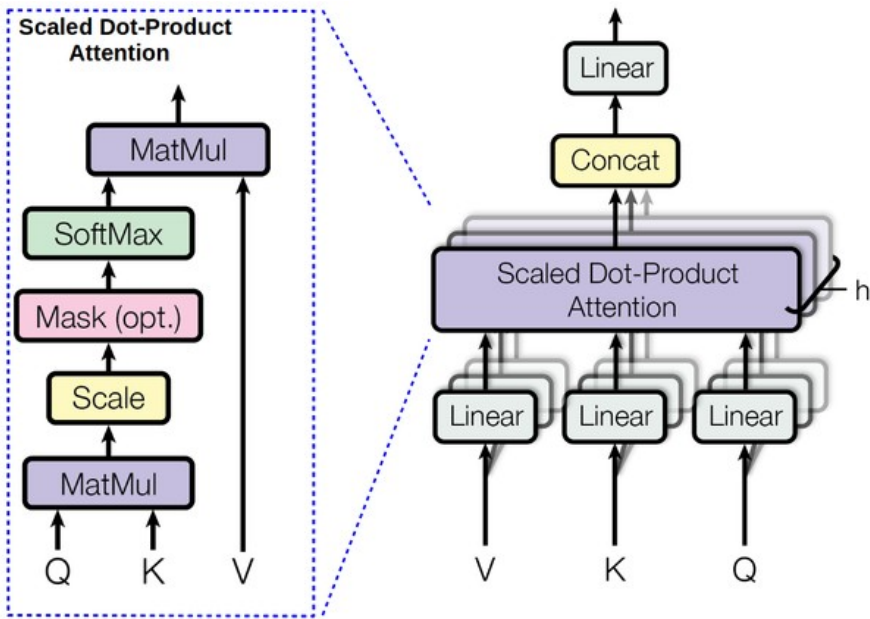
# The task

- Take the code that has been provided and improve it (most improvements will come from the architecture) (using a Graph Attention Network is highly recommended)  
→ This counts for half of the points
- Produce a diagram of the architecture that you're using (shape information must be included)  
→ This counts for the other half of the points
- In summary, you have 3 files to produce:
  - the modified code (`train_ppi.py`)
  - the weights of the model (`model_state.pth`)
  - the diagram of the model (an image)
- Do not change the signature the of the `train()` and `test()` functions!

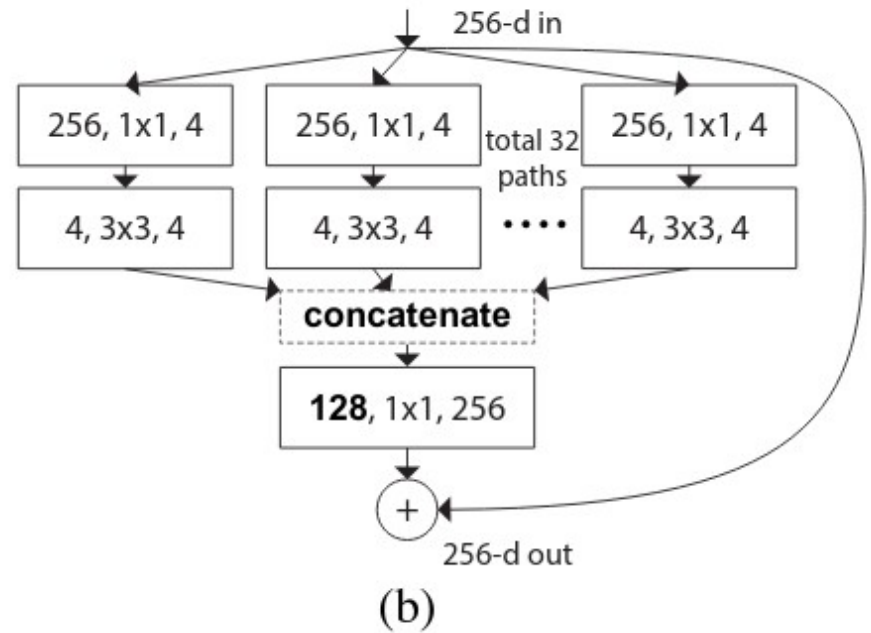
F1 Score



# Diagram examples



Multi-head attention  
(shape information is missing)



ResNeXt  
(a good legend is missing)

# Important Hints

Use a Graph Attention Network (Veličković et al.):

<https://arxiv.org/abs/1710.10903>

Use the examples from the DGL dependency:

<https://github.com/dmlc/dgl>

<https://www.dgl.ai/>

Use inspiration from pytorch geometric (but don't use it directly):

[https://github.com/rusty1s/pytorch\\_geometric](https://github.com/rusty1s/pytorch_geometric)