How Powerful Are Graph Neural Networks??

- Team-22 (Aa Naluguru)
- 1. Eswara Rohan 2020102002
- 2. Dheeraj Murugan Sai 2020102020
- 3. K. Sesha Sarath 2020102028
- 4. S. Deepak 2020102063

Problem Statement

- We have many GNNs that classify nodes and graphs with state of accuracies, but we need to develop a GNN which understands the structure properly. So we propose Graph Isomorphism Network(GIN) which is a type of GNN and serves the required purpose.
- Assumption — Node features form a countable set.

Importance Of GNN

- In real life for storing data of molecules, social, biological and financial networks etc. we use graph neural networks for effective representation learning of the graphs and to classify nodes and graphs.
- GNNs can do what CNNs fail to do such as providing tools for analyzing complicated relationships in a network.

Terminology In Graphs

- Let $G = (V, E)$ denote a graph with node feature vectors X_v for $v \in V$.
- There are two tasks of interest:

(1) Node classification, where each node $v \in V$ has an associated label y_v and the goal is to learn a representation vector hv of v such that v's label can be predicted as $y_v = f(h_v)$;

(2) Graph classification, where, given a set of graphs $\{G_1, \ldots, G_N\} \subseteq G$ and their labels $\{y_1, \ldots, y_N\} \subseteq Y$, we aim to learn a representation vector h_G that helps predict the label of an entire graph, $y_G = g(h_G)$.

General Approach In GNN

- As we know in CNN, In a image the pixel value gets updated with a function which involves its neighboring pixels in the convolution layer.
- HERE in GNN we follow a similar approach where we use AGGREGATE and COMBINE operations on each node for K iterations (GNN with K layers).

$$a_v^k = AGGREGATE^{(k)}\left(\left\{h_u^{(k-1)} : u \epsilon N(v)\right\}\right), \quad h_v^k = COMBINE^{(k)}\left(h_v^{k-1}, a_v^k\right)$$

where h_v^k is the feature vector of node v at the kth iteration/layer. We initialize $h_v^{(0)} = X_v$, and $N(v)$ is a set of nodes adjacent to v. The choice of $AGGREGATE^{(k)}(.)$ and $COMBINE^{(k)}(.)$ is very crucial as it may affect the accuracy of the model.

- For node classification, the node representation h_v^k of the final iteration is used for prediction.
- For graph classification, the READOUT function aggregates node features from the final iteration to obtain the entire graph's representation h_G.

$$h_G = READOUT\left(\left\{h_v^{(k)} | v \epsilon G\right\}\right)$$

GCN and GraphSAGE

- In this project, we compare our results with GCN and GraphSAGE in which the aggregation functions are MEAN and MAX pooling respectively.

GraphSAGE -

AGGREGATE - $a_v^{(k)} = MAX(ReLU(W.h_u^{(k-1)}), \forall\, u \epsilon\, N(u))$

COMBINE - Concatenation followed by a linear mapping W.

GCN-

The AGGREGATE and COMBINE steps in GCN are integrated as follows

$h_v^{(k)} = ReLU(W.MEAN\{h_u^{(k-1)}, \forall\, u\, \epsilon\, N(v)\, \cup\, v\})$

Weisfeiler-Lehman test (WL Test):

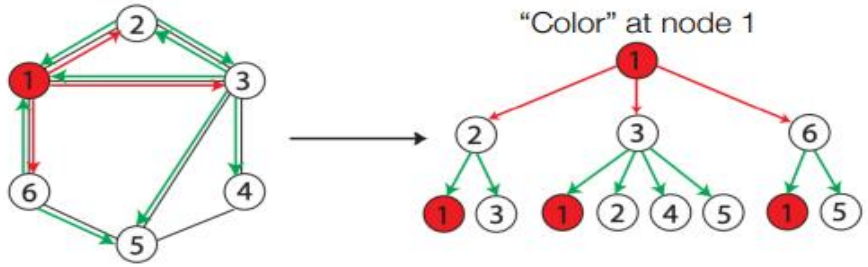
- WL graph isomorphism test is a powerful test known to distinguish a broad class of graphs.
- WL test iteratively **aggregates** the labels of nodes and their neighborhoods and **hashes** the aggregated labels into unique new labels.
- The algorithm decides that two graphs are non-isomorphic if at some iteration the labels of the nodes between the two graphs differ.
- What makes the WL test so powerful is its injective aggregation update that maps different node neighborhoods to different feature vectors.

Multiset

- A multiset is a generalized concept of a set that allows multiple instances for its elements.
- More formally, a multiset is a 2-tuple $X = (S, m)$ where S is the underlying set of X that is formed from its distinct elements, and $m : S \rightarrow N_{\geq 1}$ gives the multiplicity of the elements.
- Our framework first represents the set of feature vectors of a given node's neighbors as a multiset, i.e., a set with possibly repeating elements.
- Then, the neighbor aggregation in GNNs can be thought of as an aggregation function over the multiset.
- A maximally powerful GNN maps two nodes to the same location only if they have identical subtree structures with identical features on the corresponding nodes.
- A maximally powerful GNN would never map two different neighborhoods, i.e., multisets of feature vectors, to the same representation.
- This means its aggregation scheme must be injective.

Discriminative Power of GNNs

- GNNs can be at most as powerful as Weisfelier-Lehman graph isomorphism test.
- $Power(GNNS) \leq Power(WL)$



- To distinguish 2 nodes, GNN needs to distinguish structure of their rooted subtrees.

Discriminative Power of GNNs

- For a GNN to be as powerful as WL test, it should map two nodes to the same location in the embedding space if and only if the nodes have same identical subtree structures (Multisets) I.e., aggregation scheme must be injective

- In WL Test we operate with node labels which cannot capture similarity between subtrees.

Theorem:

Theorem 1 -

Let $A : G \rightarrow R^d$ be a GNN With a sufficient number of GNN layers. A maps any graphs G1 and G2 that the WL Test of isomorphism decides as non-isomorphic, to different embeddings if the following conditions hold:

1. A aggregates and updates node features iteratively with:

   $$h_v^{(k)} = \phi(h_v^{(k-1)}, f(h_u^{(k-1)} : u \epsilon N(u))$$

   where the functions f, which operates on multisets, and ϕ are injective.

2. A's graph-level readout, which operates on the multiset of node features $\{h_v^{(k)}\}$ is injective.

Theorem:

- Any GNN that satisfying the above theorem can learn to map similar graph structures to similar embeddings and capture dependencies between graph structures which will be helpful when the co-occurrence of subtrees is sparse across different graphs or there are noisy edges and node features.

Graph Isomorphism Network(GIN):

- We will develop a simple architecture, Graph Isomorphism Network (GIN), that provably satisfies the conditions in Theorem 3.
- It is a generalized model of WL graph isomorphism test and achieves maximum discriminative power among GNNs.
- We use theory of deep multisets , i.e., parameterizing universal multiset functions with neural networks.

Lemma 3 -

Assume χ is countable. There exists a function $f : \chi \to R^n$ so that $h(X) = \Sigma_{x \epsilon X} f(x)$ is unique for each multiset $X \subset \chi$ of bounded size. Moreover, any multiset function g can be decomposed as $g(X) = \phi(\Sigma_{x \epsilon X} f(x))$ for some function ϕ .

Corollary 1 -

Assume χ is countable. There exists a function $f : \chi \to R^n$ so that for infinitely many choices of ϵ, including all irrational numbers, $h(c, X) = (1 + \epsilon).\, f(c) + \Sigma_{x \epsilon X} f(x)$ is unique for each pair (c, X), where $c \, \epsilon \, \chi \; and \; X \subset \chi$ is a multiset of bounded size. Moreover, any function g over such pairs can be decomposed as $g(c, X) = \psi((1 + \epsilon).\, f(c) + \Sigma_{x \epsilon X} f(x))$ for some function ψ.

Graph Isomorphism Network(GIN):

Graph Isomorphism Network(GIN):

Corollary (1) provides a simple and concrete formulation among many such aggregation schemes. We can use MLPs to model and learn $f\ and\ \psi$ in the corollary. In practice, we model $f^{(k+1)} \circ \psi^{(k)}$ with one MLP as MLPs can represent the composition of functions In the first iteration, we do not need MLPs before summation if input features are one-hot encodings as their summation alone is injective. We can make ϵ a learnable parameter or a fixed scalar. Then, GIN updates node representations as -

$$h_v^{(k)} = MLP^{(k)}((1+\epsilon^{(k)}).\, h_v^{(k-1)} + \Sigma_{u \epsilon N(v)} h_u^{(k-1)}).$$

Graph Level READOUT Function For GIN

$$h_G = \mathrm{CONCAT}\Big(\mathrm{READOUT}\Big(\Big\{h_v^{(k)} | v \in G\Big\}\Big) \mid k = 0, 1, \ldots, K\Big).$$

In GIN, the Readout function is sum of all node features in that iteration.

Popular GNN Variants

1. Single Layer Perceptron -

- The function f in Lemma 5 can be parameterized by an MLP but many GNNs use a 1-layer perceptron (like a linear classifier) for graph learning (which makes them not able to distinguish many multisets).

Lemma 4 -

There exist finite multisets $X_1 \neq X_2$, so that for any linear mapping W, $\Sigma_{x\epsilon\, X_1} ReLU(Wx) = \Sigma_{x\epsilon\, X_2} ReLU(Wx)$

Popular GNN Variants

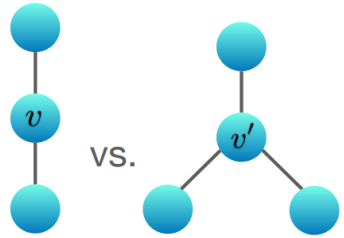
2. Mean Aggregation -

Corollary 2 -

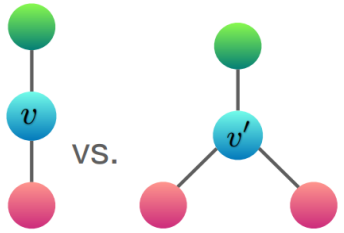
Assume ψ is countable. There exists a function $f : \psi \rightarrow R^n$ so that $h(X) = \frac{1}{|x|}\Sigma_{x\epsilon X} f(x)$, $h(X_1) = h(X_2)$ if and only if multisets X_1 and X_2 2 have the same distribution. That is, assuming $|X_2| \geq |X_1|$, we have $X_1 = (S, m)$ and $X_2 = (S, k.m)$ for some $k \epsilon N_{\geq 1}$.

Popular GNN Variants

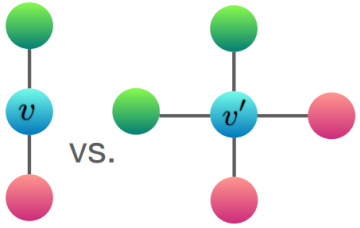
- If we consider two multisets where the unique elements are same but the multiplicity of one multiset is k times the multiplicity of the other multiset (if a node repeats twice in a multiset it repeats 2k times in the other multiset).
- The mean aggregator maps the both multisets to same embedding as we average over the individual element features i.e., we can say that mean captures the distribution of elements in the multiset but not the exact multiset.
- This explains why mean aggregator fails in a and c.



(a) Mean and Max both fail



(b) Max fails



(c) Mean and Max both fail

Popular GNN Variants

3. Max Pooling -

- Max-pooling captures neither the exact structure nor the distribution but it identifies the skeleton i.e., the unique nodes of the multiset.
- So, it is suitable in the cases where the "skeleton" is important rather than distribution like to find the skeleton of a 3D point cloud in which it is robust to noise and outliers.

Corollary 3 -

Assume ψ is countable. Then there exists a function $f : \psi \to R^{\infty}$ so that for $h(X) = max_{x \epsilon X} f(x)$, $h(X_1) = h(X_2)$ if and only if X_1 *and* X_2 have the same underlying set.

Datasets

- We considered Bioinformatic datasets like:

1. MUTAG :-

- It has 188(No.of graphs) mutagenic aromatic and heteroaromatic nitro compounds with 7 discrete labels and 2 different classes.

2. PROTEINS :-

- It is a dataset where nodes are secondary structure elements (SSEs) and there is an edge between two nodes if they are neighbors in the amino-acid sequence or in 3D space. It has 3 discrete labels, representing helix, sheet or and 2 different classes.

3. NCI1 :-

- It is a dataset made publicly available by the National Cancer Institute (NCI) and is a subset of balanced datasets of chemical compounds screened for ability to suppress or inhibit the growth of a panel of human tumor cell lines, having 37 discrete labels and 2 different classes.

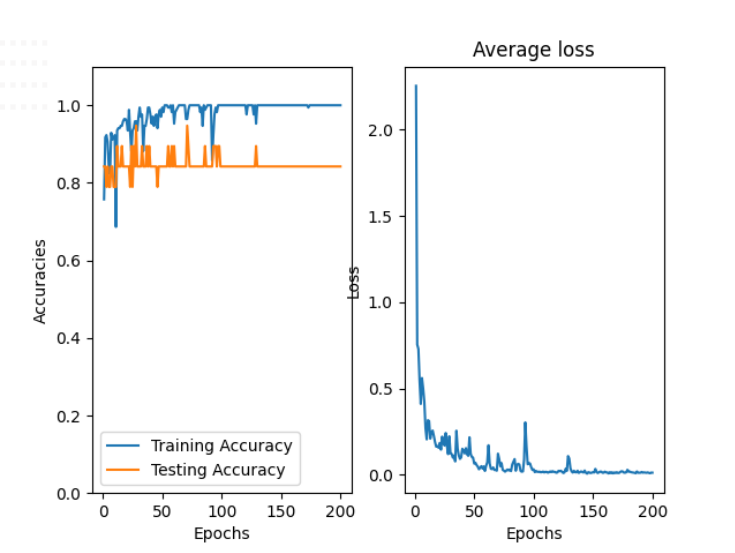
4. PTC :-

- It is a dataset of 344 chemical compounds that reports the carcinogenicity for male and female rats, and it has 19 discrete labels and 2 different classes.

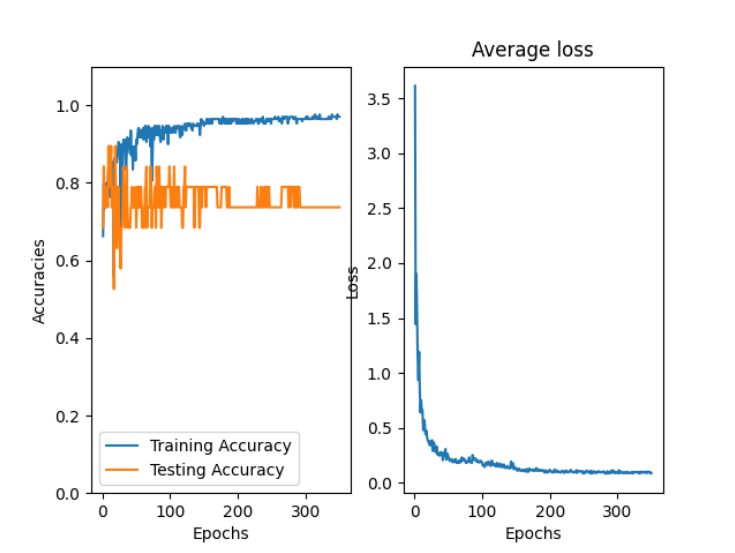
Results

MUTAG:

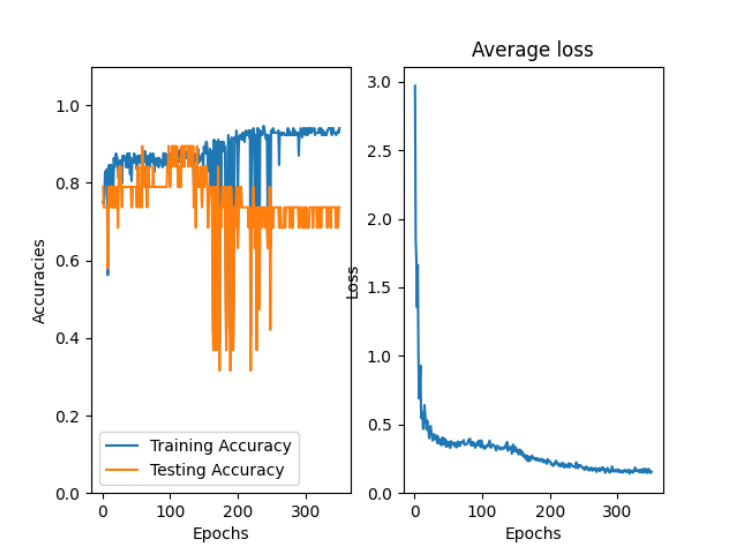
Sum-Aggregation



Max-Pooling



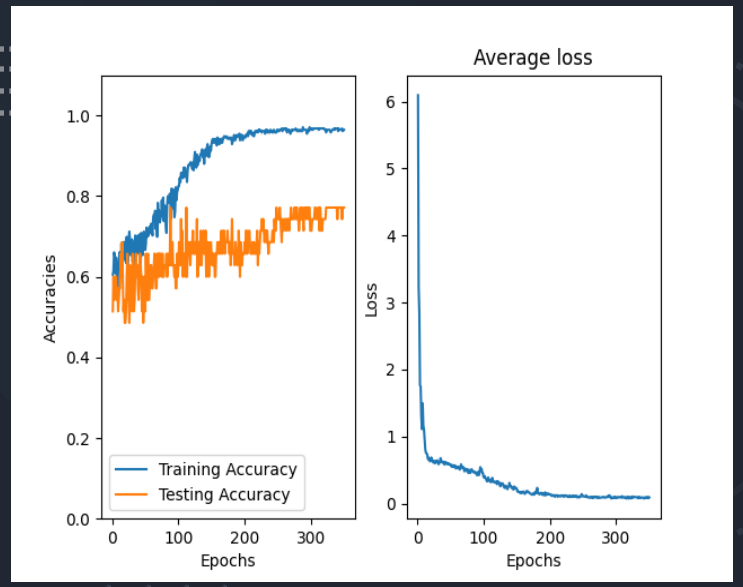
Mean-Aggregation



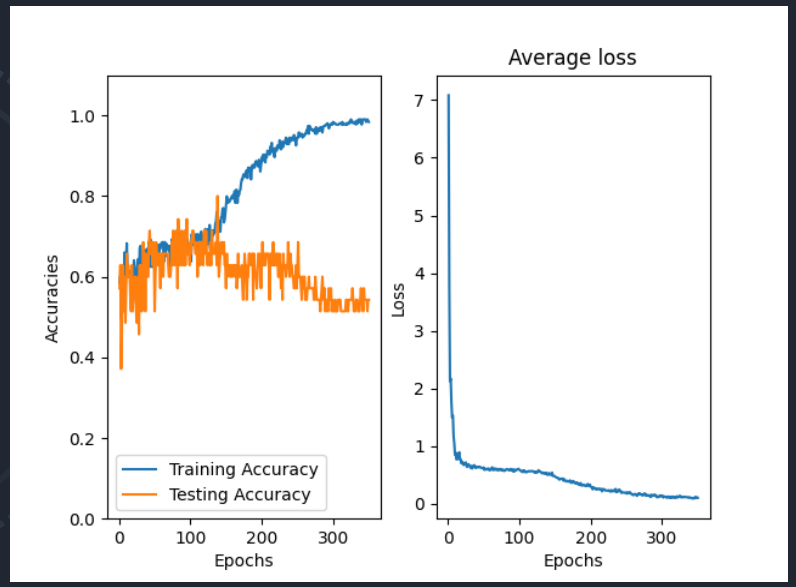
Results

PTC:

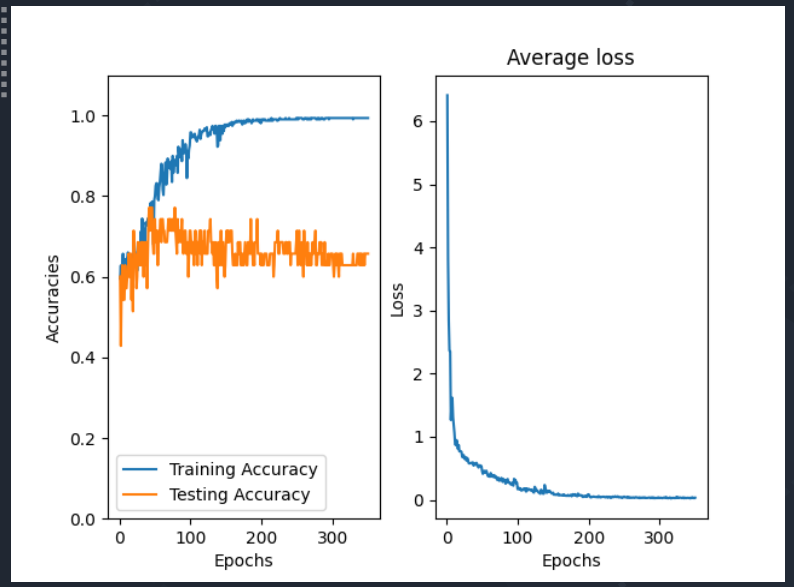
Max-pooling



Mean-Aggregation



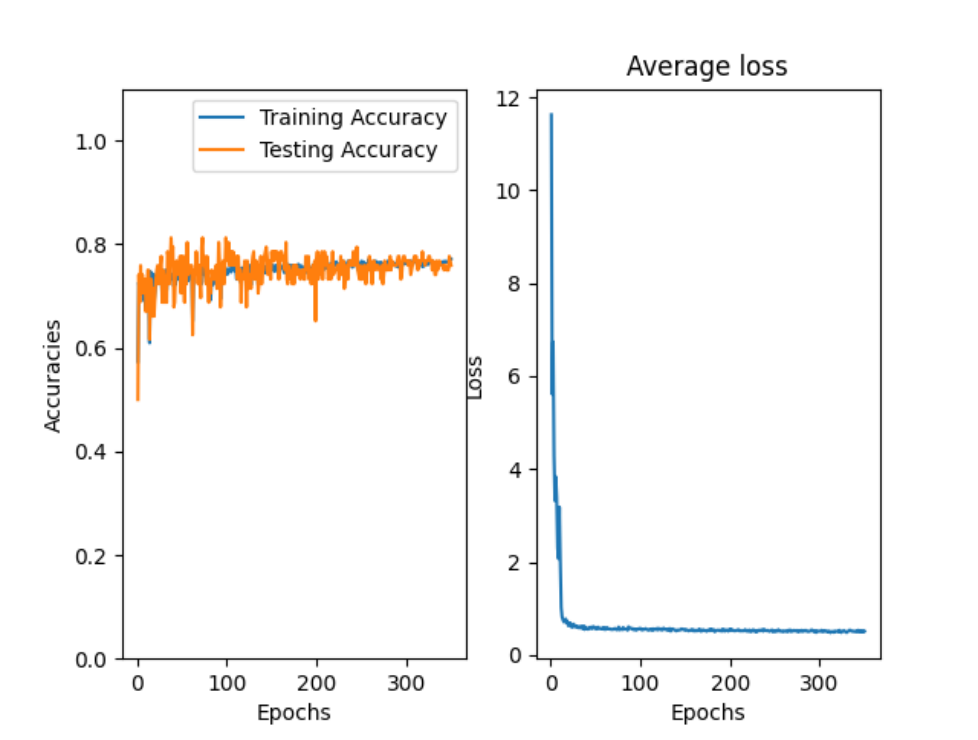
Sum-Aggregation



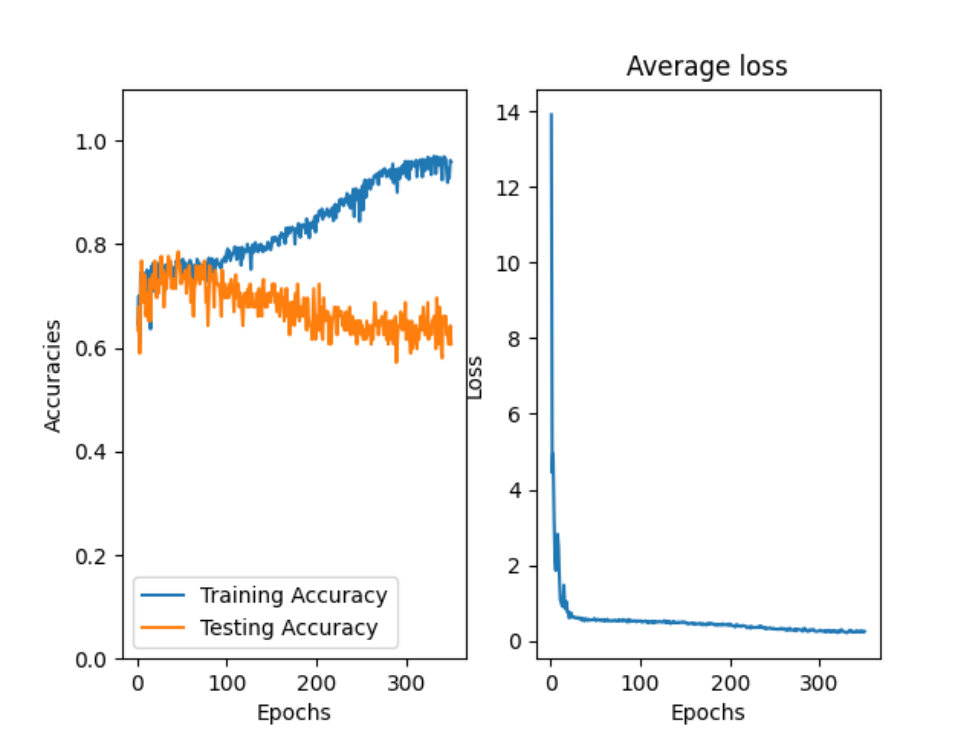
Results

PROTEINS:

Max-pooling



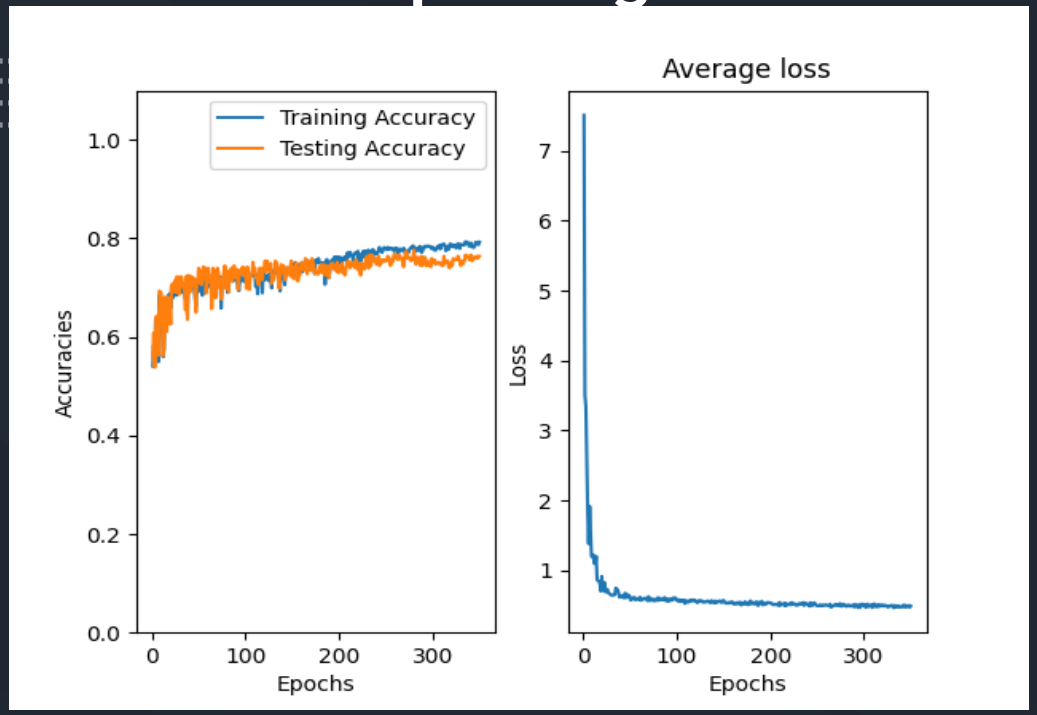
Sum-Aggregation



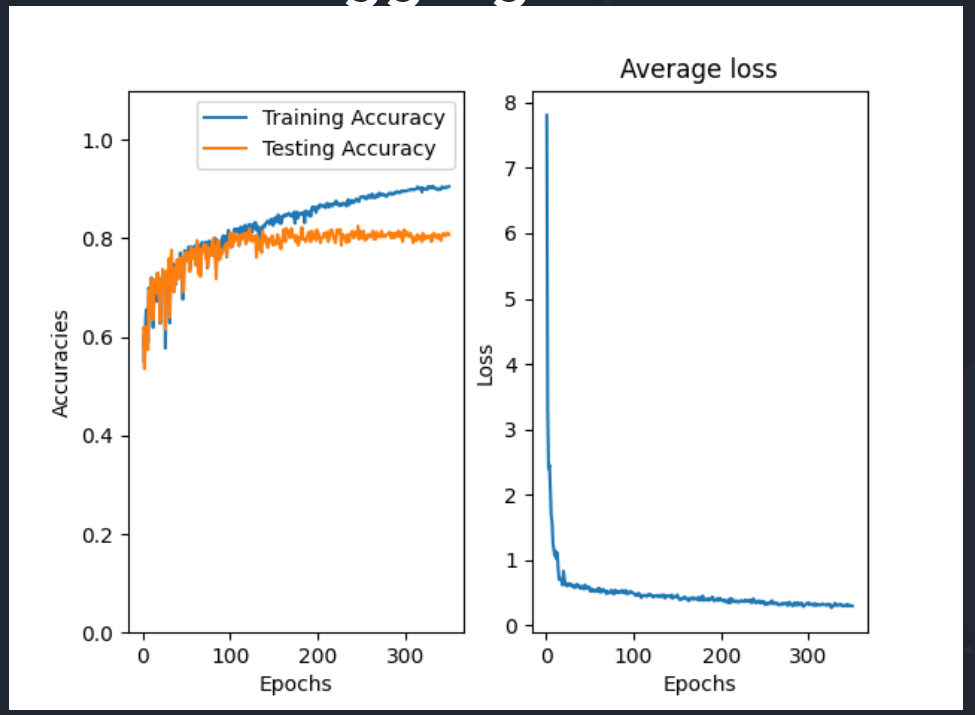
Results

NCI1:

Max-pooling



Sum-Aggregation



Thank you