**Task 1: PointCloud**

**Introduction**

This report describes the implementation and evaluation of a PointNet2 model for classifying point clouds. The model incorporates Set Abstraction (SA) layers and a Global Set Abstraction (GSA) layer, which helps capture both local and global features within the point cloud. The primary objective of this experiment was to classify point clouds accurately, using a well-structured architecture tailored for point-based data processing.

**Model Architecture:**

The model architecture is based on the PointNet2 framework, featuring hierarchical feature learning through Set Abstraction (SA) modules and a Global Set Abstraction (GSA) module. The specific architecture includes:

1. Set Abstraction (SA) Modules:
   * The first SA module (sa1\_module) uses a PointNetConv layer with a local neural network (MLP) that has layers with dimensions [3, 64, 64, 128].
   * The second SA module (sa2\_module) has an MLP with layers [131, 128, 128, 256].
2. Global Set Abstraction (GSA) Module:
   * The GSA module aggregates global features using an MLP with layers [259, 256, 512, 1024].
3. Classification Head:
   * The final classification head consists of an MLP with layers [1024, 512, 256, 10], which outputs class scores for 10 categories.

Overall, this model contains 1,463,626 parameters, balancing complexity with computational efficiency.

**Training Procedure:**

The model was trained over 10 epochs. For each epoch, training and validation accuracy, along with training loss, were recorded. The training process involved optimizing the network to minimize the classification loss, improving both the model's convergence and accuracy.

* Training Accuracy improved consistently across epochs, starting at 68.8% in the first epoch and reaching 95.7% by the last.
* Validation Accuracy also showed gradual improvement, achieving a final accuracy of 90.6%.

**Loss and Accuracy Progression:**

* Epoch 1: Loss - 0.9912, Train Acc - 0.6875, Val Acc - 0.7814
* Epoch 5: Loss - 0.1894, Train Acc - 0.9267, Val Acc - 0.8794
* Epoch 10: Loss - 0.1094, Train Acc - 0.9570, Val Acc - 0.9057

**Results:**

The final model achieved a high classification accuracy, with a training accuracy of 95.7% and a validation accuracy of 90.6%. The learning curve indicates effective learning over the epochs, as seen in the **A graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of a graph of

Description automatically generated**

**Analysis of Classified and Misclassified Point Clouds**

**A diagram of a table and a desk

Description automatically generated with medium confidence**

In analyzing the misclassified examples, some patterns emerged. Misclassifications often occurred with point clouds that had similar shapes or patterns across classes. This suggests that while the model effectively distinguishes most classes, certain point clouds present ambiguity due to structural overlap or lack of distinctive features in the chosen dataset.

**Task 2: Graph Classification**

**Method:**

Our methodology was straightforward: we developed dataloaders for both training and validation purposes. Utilizing cross-entropy loss as our loss function, we observed a class imbalance in the data, with the number of "ones" being twice the number of "zeros." To address this issue, we introduced a weight parameter into the loss function. Subsequently, we initiated a cross-validation process to evaluate various architectures and their corresponding parameters.

**Architectures:**

In our project, we conducted experiments utilizing the following architectures:

* Graph Attention Network (GAT), as presented in the course material.
* Graph Sample and Aggregate (GraphSAGE).
* Gaussian-enhanced GAT.
* Gaussian-enhanced GraphSAGE.

Gaussian models introduce the concept of uncertainty by learning Gaussian embeddings for nodes. Rather than relying on fixed embeddings, each node is represented by a mean and variance. This approach empowers the model to capture inherent uncertainty in node representations. To facilitate sampling of embeddings, the reparameterization trick is employed. Additionally, KL divergence regularization is applied to discourage excessively large variances, thus promoting the model's ability to generate confident predictions.

**Experiments:**

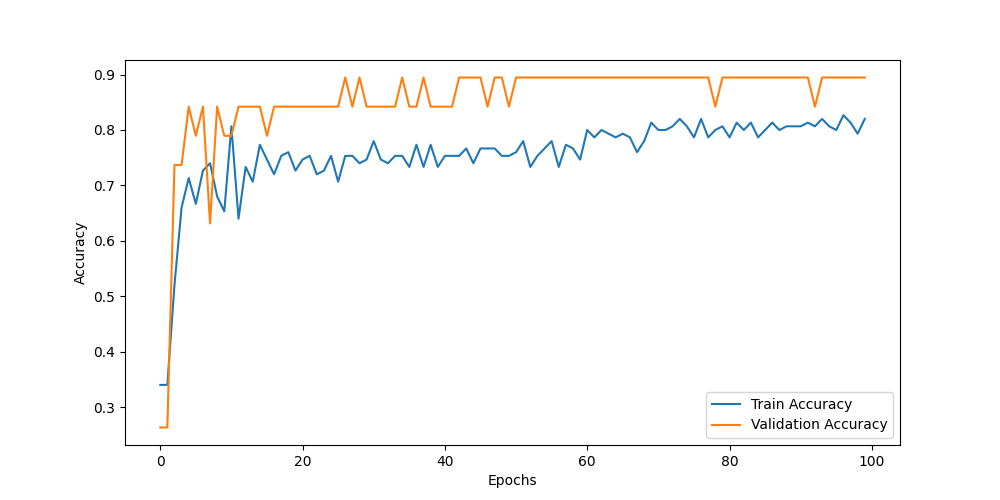
In our experimentation, we applied cross-validation across a comprehensive set of hyperparameters to identify the optimal configuration for model performance. We systematically varied the learning rate (lr), weight decay, batch size, number of layers (n\_layer), hidden units in the aggregation layers (agg\_hidden), and fully connected layers (fc\_hidden). Additionally, we experimented with three options for the global pooling method (agg\_method), assessing sum, average and max pooling.

This cross-validation approach allowed us to evaluate the impact of each parameter setting on model stability and performance, enabling a fine-tuned balance between accuracy and generalization.

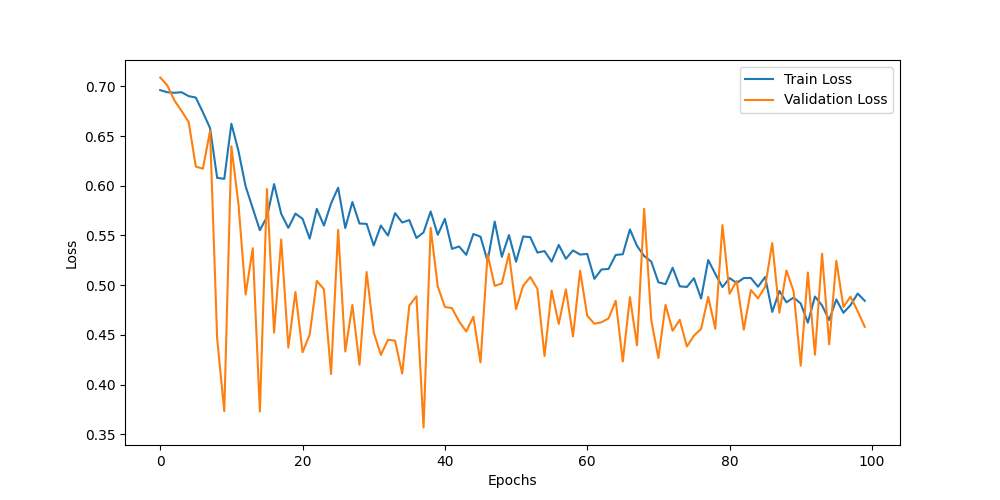
**Results:**

Through our experiments, the Gaussian models provided decent results by reaching accuracies of 0.95-1; however, they were unstable, and thus we decided to refrain from using them. Ultimately, we selected GraphSAGE with dropout, which demonstrated stable and high performance. The optimal parameters for this model were an aggregation hidden size of 128, a max pooling aggregation method, 100 training epochs, a fully connected hidden layer size of 128, a learning rate of 0.0007, seven layers, and a batch size of 15. This configuration proved effective, providing a consistent balance between accuracy and generalizability across our validation dataset.

And this model was used to predict on the test dataset.



***Accuracy on Training and Validation***



***Loss on Training and Validation***