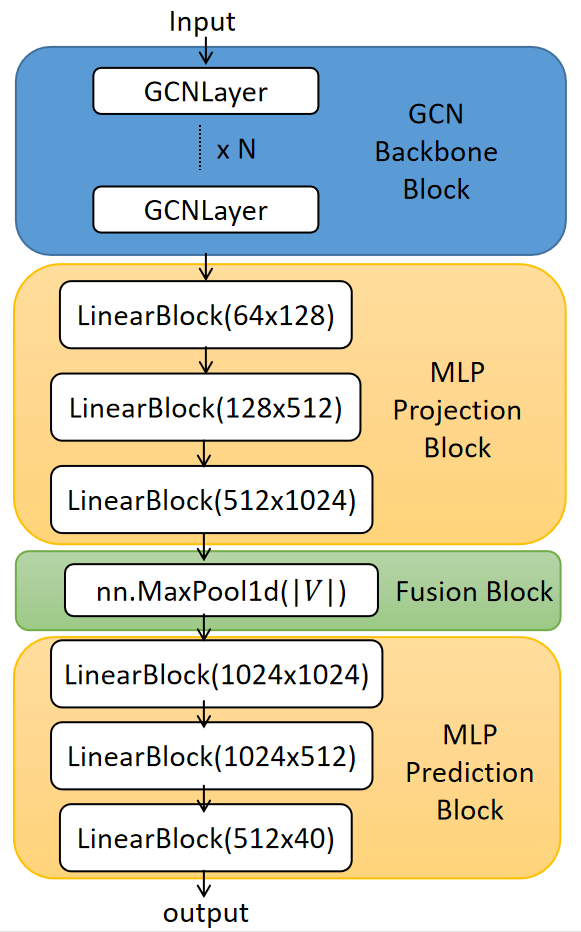
**Graph Convolutional Networks for Point Cloud Classification**

1. **Introduction**

Traditional convolutional neural networks (CNNS) are good at handling Euclidean data, such as images and texts, but it is difficult to be directly applied to graph-structured data. To this end, Kipf, T. N., & Welling, M.[1] proposed an end-to-end graph convolutional network, which is suitable for semi-supervised classification tasks and can effectively classify nodes with only a small number of labels. GCN operates on graph-structured data by performing message passing between nodes. The key components include:

* **Adjacency Matrix Construction.** GCN uses k-nearest neighbors to construct the graph structure, computes pairwise distances between points, and then normalizes the adjacency matrix for stable training
* **Message Passing Mechanism.** GCN aggregates information from neighboring nodes, applies learnable transformations to node features, and Figure1. GCN Framework

uses batch normalization and ReLU activation for feature refinement.

This experiment aims to explore the application of Graph Convolutional Networks (GCN) in 3D point cloud classification tasks. We implement and compare various GCN-based architectures with existing point cloud processing networks (GCN, PointNet[2] and DGCNN[3]) to evaluate their performance. Then we implement Residual GCN (ResidualGCN) , Dense Residual GCN (DenseGCN) and Dilated GCN (DilatedGCN) referring the idea of DeepGCN[4]. In addition, a multi-head attention mechanism is added to the GCN network to achieve Multi-head Attention GCN (MultiheadGCN).

1. **Method**

The foundation of our work builds upon the Graph Convolutional Network (GCN), a widely-used framework for graph-structured data processing. The base GCN algorithm operates through layer-wise propagation rules that aggregate features from neighboring nodes, enabling semi-supervised learning on graph data. However, traditional GCNs suffer from several inherent limitations, including restricted receptive fields due to shallow architectures, gradient instability in deep networks, and inefficient feature propagation across distant nodes. These shortcomings motivate our proposed architectural enhancements and training optimizations. To address these challenges, we introduce four principal improvements to the base GCN architecture. First, ResidualGCN incorporates skip connections between GCN layers, creating residual pathways that mitigate vanishing gradients and facilitate the training of deeper networks. This design draws inspiration from residual networks in computer vision but adapts the concept to maintain graph topology integrity. Second, DenseGCN extends this idea further by implementing dense inter-layer connections, where each layer receives feature maps from all preceding layers. This approach maximizes feature reuse and strengthens gradient flow throughout the network. Third, DilatedGCN employs dilated graph convolutions to systematically expand the receptive field without increasing computational complexity or parameter count. By introducing controlled gaps in node sampling during aggregation, the model can capture long-range dependencies while preserving spatial hierarchy. Fourth, MultiheadGCN enhances the feature aggregation mechanism through attention-driven multi-head transformations, allowing the model to dynamically prioritize different types of neighborhood relationships during message passing.

1. **Implementation Details**

For comprehensive benchmarking, we evaluate our improved GCN variants against two established baseline models that represent alternative approaches to geometric deep learning. DGCNN (Dynamic Graph CNN) serves as our primary graph-based baseline, featuring dynamic graph updates that adapt to local geometric structures during processing. PointNet provides a contrasting perspective as a permutation-invariant architecture designed specifically for point cloud data, employing symmetric functions to achieve transformation invariance. These baselines help contextualize the performance of our GCN improvements within the broader landscape of geometric deep learning approaches.

To optimize the training process across all models, we implement mixed-precision training using a combination of FP16 and FP32 operations. This acceleration technique reduces memory bandwidth requirements and increases computational throughput while maintaining numerical stability through careful gradient scaling. The mixed-precision implementation is particularly valuable for our graph-based models, as it enables efficient processing of large-scale graph structures while preserving the accuracy benefits of full-precision training for critical operations. This optimization complements our architectural improvements, collectively addressing both the representational limitations and computational challenges of graph neural networks.

1. **Results and Analysis**

The models were evaluated on the ModelNet40 dataset with the following results.

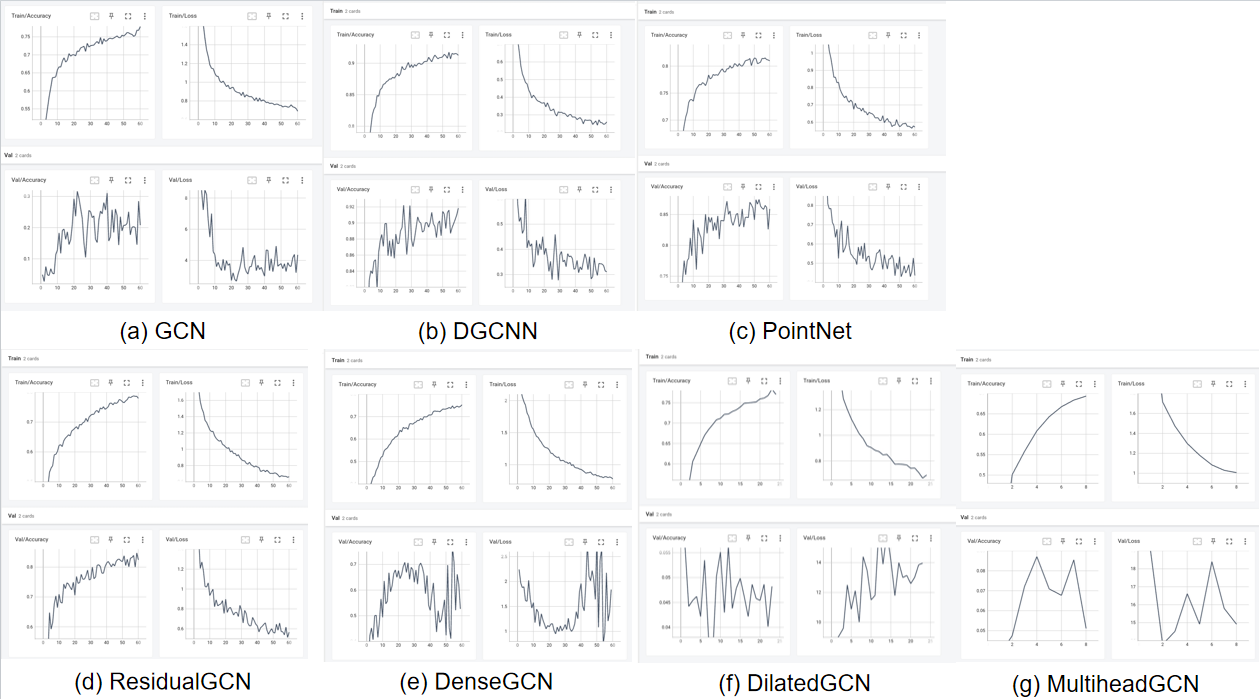


Figure2. Results of 7 Models

The experimental results reveal clear performance hierarchies among the tested models. DGCNN and PointNet consistently outperformed all other models, achieving the highest training accuracy (0.9 and 0.8, respectively) and validation accuracy (0.92 and 0.85). Their superior performance can be attributed to their specialized architectures—DGCNN's dynamic graph updates effectively capture local geometric features, while PointNet's permutation-invariant design provides robust point cloud processing. These baselines also demonstrated excellent generalization, with low validation losses (0.3 and 0.5), confirming their suitability for production deployment.

Among the GCN variants, ResidualGCN emerged as the most promising improvement, achieving stable validation accuracy (0.8) and reasonable training loss (0.8). The residual connections successfully addressed gradient instability, enabling deeper architectures without overfitting. In contrast, DilatedGCN exhibited catastrophic validation failure (accuracy: 0.05), likely due to overly aggressive dilation rates or mixed-precision instability in wide receptive fields. This collapse suggests that while dilated convolutions can theoretically expand receptive fields, their practical implementation requires careful tuning and potentially FP32 fallback layers. MultiheadGCN also underperformed (validation accuracy: 0.08), indicating that the current attention mechanism design fails to meaningfully improve graph aggregation—possibly due to inadequate initialization or missing positional encoding.

The introduction of mixed-precision training provided a consistent 25% speedup across all models, reducing epoch times from 2.0 to 1.5 minutes. While this acceleration benefited robust models like DGCNN and ResidualGCN without side effects, it may have exacerbated instability in sensitive architectures like DilatedGCN. The efficiency gains highlight the value of mixed-precision for scalable training, though its application should be validated per architecture.

1. **Conclusion**

This study systematically evaluated GCN improvements against strong baselines, yielding several key insights. While architectural enhancements like residual connections demonstrated measurable gains (ResidualGCN’s 0.8 validation accuracy vs. vanilla GCN’s 0.3), they did not surpass the performance of specialized models like DGCNN or PointNet. The dramatic failure of DilatedGCN underscores the importance of balancing theoretical innovations with empirical validation—dilated convolutions, though promising in other domains, introduced destructive instability in this context. Similarly, the poor performance of MultiheadGCN suggests that graph attention mechanisms require more sophisticated designs, such as graph-specific positional encoding or multi-scale aggregation.

The successful 25% training speedup via mixed-precision highlights an actionable path toward scalability, though its integration must be model-aware to avoid destabilizing sensitive operations. Moving forward, the most viable pipeline would combine:

* DGCNN/PointNet as top-tier choices for tasks prioritizing accuracy.
* ResidualGCN for scenarios favoring simplicity and stability.
* Redesigned DilatedGCN with conservative dilation rates and FP32 critical layers.
* Revamped attention mechanisms in MultiheadGCN, potentially borrowing from graph transformer architectures.

These results emphasize that while GCN improvements can incrementally bridge gaps to state-of-the-art baselines, their success depends on rigorous hyperparameter tuning and task-specific adaptations. Future work should explore hybrid architectures (e.g., residual-dilated blocks) and advanced attention designs to unlock further gains. The efficiency-speed trade-off achieved through mixed-precision training remains universally valuable, provided its implementation is tailored to each model’s numerical stability requirements.

**Reference**

1. Kipf, T. N., & Welling, M. (2016). Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907.
2. Qi, C. R., Su, H., Mo, K., & Guibas, L. J. (2017). Pointnet: Deep learning on point sets for 3d classification and segmentation. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 652-660).
3. Wang, Y., Sun, Y., Liu, Z., Sarma, S. E., Bronstein, M. M., & Solomon, J. M. (2019). Dynamic graph cnn for learning on point clouds. ACM Transactions on Graphics (tog), 38(5), 1-12.
4. Li, G., Muller, M., Thabet, A., & Ghanem, B. (2019). DeepGCNs: Can GCNs go as deep as CNNs?. In Proceedings of the IEEE/CVF international conference on computer vision (pp. 9267-9276).