Mamba for 3D Semantic Segmentation of Point Clouds

Kaushal Gumpula (kgumpula), Ricardo Buitrago Ruiz (ricardob),
Arya Shah (aryas), Hemit Shah (hemits)

https://github.com/r-buitrago/3DMamba

1 Motivation

Computational Costs of 3D Vision

3D computer vision tasks have historically been computationally expensive due to the additional spatial dimension of the data involved. Every stage from data collection, to preprocessing, training, and inference is impacted by the additional memory and compute required by 3D vision.

Structured State-Space Models for 3D Vision

Recently, structured state-space-model (SSM) architectures have garnered *attention* with the release of Mamba (Gu and Dao [2023]), a state-space model that incorporates selectivity while still being able to perform linear-time sequence modeling. Mamba has been shown to utilize nearly 90% less memory on GPUs and achieve 5× higher throughput than transformer-based models while also reaching the same level of performance with half the number of parameters.

Given that state-of-the-art 3D vision models for classification, object detection, and semantic segmentation also rely on transformer blocks and attention to achieve high levels of performance, Mamba could be extended to handle 3D vision tasks while being less computationally intensive.

More importantly, most work involving 3D point clouds focuses specifically on 3D objects, which have limited applications compared to 3D outdoor scenes. Taking advantage of the memory efficiency of Mamba, we aim to apply Mamba-based architectures to 3D semantic segmentation of outdoor scene datasets with more points in each point-cloud and greater sparsity within the 3D space occupied by the points.

Challenges with SSMs and Mamba for 3D Vision

One of the biggest challenges researchers currently face with applying Mamba to 3D vision is that 3D data is not compatible with Mamba's unidirectional causal modeling capabilities for 1D sequences. As a result, current research on performing 3D vision tasks with Mamba is heavily influenced by the chosen strategy for feeding 3D data as a 1D sequence to whichever Mamba-based architecture is implemented.

We believe that 3D to 1D serialization techniques are the key to unlocking SSM and Mamba's full capabilities on 3D data regardless of the final vision task.

2 Prior Work

Prior work in the space of 3D semantic segmentation (such as PointNet and its variants) has focused on applying models that consume point clouds directly and apply simple MLP layers (some with shared weights) along with learned affine transformations to produce the desired outputs (point cloud classifications or semantic segmentations). Some newer methods such as Point Transformer and its variants also incorporate transformers in their model architectures with great success.

Below we describe three very recent Mamba-based approaches to the 3D semantic segmentation task. The authors mainly apply their models to 3D object datasets such as ShapeNet, ModelNet40, and others.

Point Patch Mamba: Liang et al. [2024]

The authors use well-established methods for tokenizing point clouds using a combination of farthest point sampling and K-nearest neighbors before performing a re-ordering of the point patches to accommodate for Mamba's unidirectional modelling capabilities. The authors concatenate the tokens along each axis to create a sequence 3 times the original length. They then apply Mamba blocks to the entire sequence. Standard pretraining approaches are followed to train the encoder (point-masked auto-encoder approach). The original patches are embedded using PointNet before processing the token sequences through multiple Mamba blocks. Their decoder is a basic MLP.

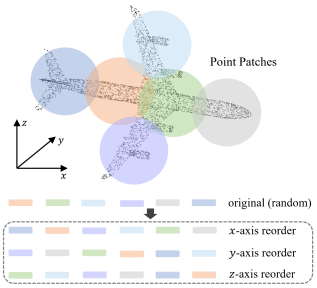


Figure 1: Patch serialization for PointMamba approach Liang et al. [2024]

Point Octree Mamba: Liu et al. [2024]

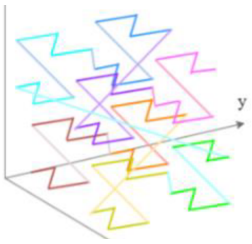


Figure 2: Octree ordering of points Liu et al. [2024]

Another work titled Point Mamba uses an octree based serialization method to convert the 3D point cloud data to a 1D sequence of points. This involves splitting the 3D space into 8 voxels at a time, and further splitting voxels until each voxel is only occupied by 1 point in the point-cloud. A z-order curve is then used to traverse the octree, with xyz-coordinates concatenated into a single sequence representing the entire point cloud. The authors claim this ordering strategy also preserves 3D spatial adjacency in the 1D sequence fed to Mamba. Point Octree Mamba's architecture also incorporates 4 stages of their "Point Mamba Block" followed by downsampling to produce encoded features. A classification or segmentation specific task head can then be applied on the encoded features, usually requiring upsampling for semantic segmentation.

Point Cloud Mamba: Zhang et al. [2024]

This paper proposes the Consistent Traverse Serialization strategy to ensure that adjacent points in the 1D sequence are also adjacent in spatial position in 3D. The specific encoding function they use is not very intuitive, but it is able to achieve their desired properties when mapping the 3D point cloud to a 1D sequence. Unlike Liang et al. [2024], they do not use point patches as tokens but rather the points themselves, taking inspiration from PointNet's approach. Their architecture relies on an affine transformation module, followed by several Mamba blocks, before performing down-sampling of points. This process is repeated in 4 stages. Due to the down-sampling of points between layers, their decoder architecture must up-sample through interpolation followed by MLP layers in 4 stages as well. Point Cloud Mamba also tests on indoor scene data available in S3DIS and achieves SOTA overall accuracy and mIoU for the semantic segmentation task.

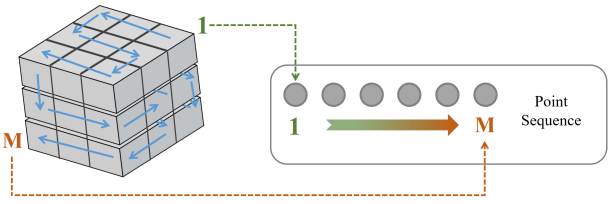


Figure 3: Consistent Traversal Serializations strategy utilized by Zhang et al. [2024].

Bidirectional Mamba

Note that in all of the above Mamba based approaches, the authors extend the original Mamba block to perform bidirectional sequence modeling. They pass in both the forward and reverse orders of the sequences generated using their serialization methods. In Point Cloud Mamba, they also generate the 1D sequence of the points in all 6 possible orderings of the xyz axes to provide the Mamba blocks an opportunity to learn casual relationships in all directions in both the forward and reverse order for each spatial dimension.

3 Idea

3.1 What is the idea?

In this project, our aim is to create a Mamba-based 3D vision architecture that can be applied to semantic segmentation of sparse 3D point cloud data. For this task, we incorporate approaches used for tokenization of 3D input data in SOTA architectures for 3D semantic segmentation. To achieve this, we also needed to carefully reconsider previous approaches used to move 3D data into a 1D sequence that can be causally modeled by Mamba. As will be discussed in detail later in the report, many 3D Point Cloud-based Mamba architectures use unique serialization strategies to transform the 3D coordinate space into a 1D sequence. We discuss the benefits and drawbacks of these approaches. Additionally, we provide a performant architecture, Parallel Point Cloud Mamba, that improves upon the way these architectures transform and pass sequences of 3D point cloud data into Mamba blocks.

3.2 Why does your idea make sense intuitively?

The main motivation for using Mamba for 3D tasks is computational efficiency. State-of-the-art architectures such as attention have quadratic complexity in the sequence length, which becomes a bottleneck, especially as 3D point clouds grow in size. In contrast, Mamba offers a linear-time approach that can solve this problem.

However, Mamba requires 3D data to be serialized into a 1D sequence. Again, we believe that this challenge can be overcome by a good serialization strategy (Zhang et al. [2024], Liu et al. [2024], Liang et al. [2024]).

Intuitively, humans process visual information extremely efficiently. Rather than processing each pair of visual elements in our field of view (which is analogous to what attention does in quadratic time), we are able to detect "causal" relationships between the objects in front of us by analyzing each object at a time (in a "linear" sequential manner), by keeping track of the other objects we have seen in memory. As 3D datasets become denser, and larger, we believe that the fundamental differences in "correlative" (attention) vs "causal" (Mamba) modeling will favour the choice of Mamba over transformers.

Transformers also cannot operate directly over point cloud information as most datasets often have $O(10^5)$ points with at least xyz-coordinates and additional features. Patching of point clouds is required in transformer architectures for computational feasibility. Mamba based approaches can operate over entire point clouds directly as it is well-suited to handle sequences of up to $O(10^6)$ tokens (Gu and Dao [2023]).

3.3 How does it relate to prior work in the area?

As mentioned previously, several works have been published earlier in 2024 applying Mamba with great success to the task of point-cloud analysis, but specifically on object datasets or indoor scene datasets. Both of these datasets generally contain much denser point-clouds than outdoor scene datasets collected via LIDAR sensors. We aim to generalize Mamba architectures for outdoor 3D scene understanding.

Furthermore, as Mamba is seen as a less compute-intensive and memory-intensive replacement for the transformer block, it should be possible to take existing expensive transformer-based architectures for 3D semantic segmentation, scene completion, and reconstruction, and replace transformer blocks with Mamba. We hope this reduces the cost of training and inferencing these models. Note that transformer-based point cloud architectures (namely Point Transformer models), are the current SOTA when it comes to 3D semantic segmentation of outdoor scene point clouds such as those in the nuScenes dataset Caesar et al. [2020].

4 Evaluating Existing Serialization Strategies

Mamba is currently limited to unidirectional modeling of 1D to 1D sequences Gu and Dao [2023]. To leverage Mamba for 3D semantic segmentation, effective serialization methods are essential as we need to go from 3D space to 1D space. We provide a brief overview of each model's serialization strategies and architecture performance with results on the mini split of the nuScenes dataset.

NuScenes is an outdoor scene dataset comprised of 1000 driving scenes from Boston and Singapore. It contains 32 classes and 1.4 billion annotated LiDAR points across 40,000 pointclouds, where each point consists of the spatial coordinates x, y, z plus two features from the LiDAR sensors. Each point is annotated with a class, and the task is to perform semantic segmentation of these point clouds Caesar et al. [2020].

4.1 Point Octree Mamba

After we evaluated Point Octree Mamba on the nuScenes mini dataset, it proved to be a very poor approach for serializing sparse 3D point clouds of outdoor scenes into a 1D sequence. It did not seem to be well suited to the task of 3D semantic segmentation of outdoor scenes as it was unable to break past an accuracy of 22% on the training points. We believe the limitations of Point Octree Mamba for semantic segmentations of sparse 3D point clouds of outdoor scenes are a result of their octree-based ordering strategy. Their traversal strategy over the 3D space when considering a sparser point cloud generally fails to preserve the necessary locality-based causal information.

4.2 Point Patch Mamba

Point Patch Mamba's serialization strategy was to traverse the point cloud patches in x, y, z-order directions and concatenate the sequences of each traversal forwards and backwards. However, in doing so, Point Patch Mamba's training suffered from heavy memory constraints; for scale, loading in 32k points (from pointclouds) at once in a batch would take around 12GB of memory on the GPU.

Even after getting around the memory issues, Point Patch Mamba's performance on training converged fairly quickly. In the end, Point Patch Mamba's performance on its native part segmentation tasks did not translate as well to semantic segmentation on nuScenes point clouds.

4.3 Point Cloud Mamba

We also implemented the Point Cloud Mamba (PCM) model architecture from Zhang et al. [2024]. This model is an encoder of the point cloud, so we also implemented a segmentation head which concatenates the global features of the cloud with point-wise features to perform segmentation, as in Qi et al. [2016]. We provide a diagram of it in Figure 4.

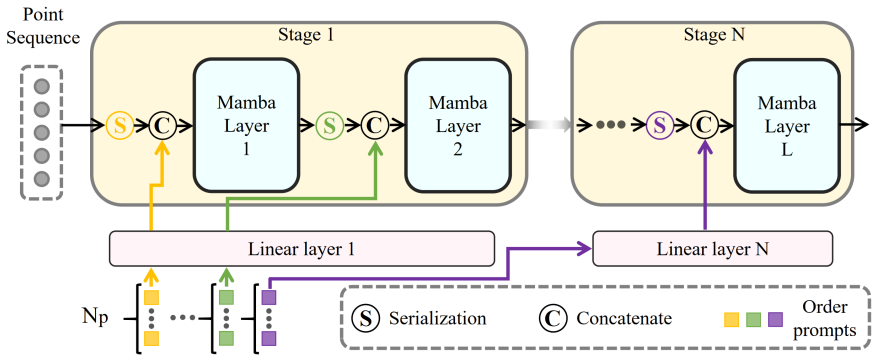


Figure 4: Diagram of Point Cloud Mamba's architecture taken from Zhang et al. [2024].

PCM takes a slightly different approach to previous models, as it iteratively introduces spatial reorderings rather than concatenating them all at once at the beginning. Referencing Figure 4, we can see that a spatial ordering is created, passed in through a Mamba block to create features, and then reordered again through a different serialization method. As the network gets deeper, PCM iteratively continues this process to finally create global features that are pooled and then passed to the segmentation head decoder.

We found this model to be the best performing out of the baseline serialization approaches we tried. After training a model with 22M parameters for 10 epochs in a NVIDIA RTX 6000 Ada Generation GPU, we obtained a test top-1 accuracy of **73.82**. Both the training and test accuracy stagnated after 10 epochs.

As it is common in the task of semantic segmentation, the nuScenes dataset is highly imbalanced, which is why we show accuracy numbers per class in Figure 10. There are some classes which are not present in our fraction of the full dataset, and some others which constitute very small fractions of our training data. Consequently, the accuracy for these classes is quite low.

4.4 Improving Serialization of 3D Point Clouds

After adapting three Mamba-based 3D semantic segmentation models on nuScenes, there were several common architecture design choices that we noticed.

Models like Point Patch Mamba and Point Octree Mamba directly concatenate unique traversals of the set of 3D points. However, Mamba is a sequence-to-sequence model that expects a 1D sequence that is causally related. When concatenating sequences together in different orderings, there is a lack of consistency in the input sequences of these architectures. By naively concatenating sequences together, we hurt Mamba's ability to learn the data as neighboring tokens from different serializations have no causal relationship.

While PCM does not concatenate all the orderings at the beginning and keeps representations consistent, continually serializing the same embedding vector through different Mamba blocks makes it difficult for the model to learn as the orderings are predetermined and fixed. After enough iterations, we expect that the information gained from this process actually washes out important spatial relationships between the points. This is also reflected in ablation studies done on serialization methods from the original PCM paper.

Strategy	OA (%)	mAcc (%)
{"z"}× 9	86.78	84.67
{"hilbert"} × 9	86.78	84.68
{"xyz"} × 9	86.71	85.00
{"xyz", "yzx", "zxy"} × 3	86.88	85.11
{"xyz", "xzy", "yxz", "yzx", "zxy", "zyx", "xyz", "yzx", "zxy"}	87.10	85.51
{"xyz", "xzy", "yxz", "yzx", "zxy", "zyx", "hilbert", "z", "z-trans"}	87.20	85.54

Figure 5: Ablation study on varying serialization strategies on S3DIS from Zhang et al. [2024].

As we can see from Figure 5, regardless of the type of serialization strategy used, the performance of the PCM model is virtually the same. Considering the first three strategies are naively running the same serialization strategy 9 times consecutively, it is clear that the PCM model is deriving its performance from the feature space and not the spatial relationships between the point embeddings themselves.

From our exploration of these models, we came to the conclusion that making point cloud models deeper by continually chaining Mamba blocks led to a degradation in performance. Additionally, the concatenation of input sequences was also not a viable solution as it destroys the consistent ordering of the input points that Mamba implicitly expects. To solve these issues, we took inspiration from multi-headed attention. Rather than using the concatenated sequences from different serialization strategies, we used the Mamba blocks to act independently on unique reorderings. By doing this, we allow each Mamba block to learn different feature representations unique to each serialization strategy.

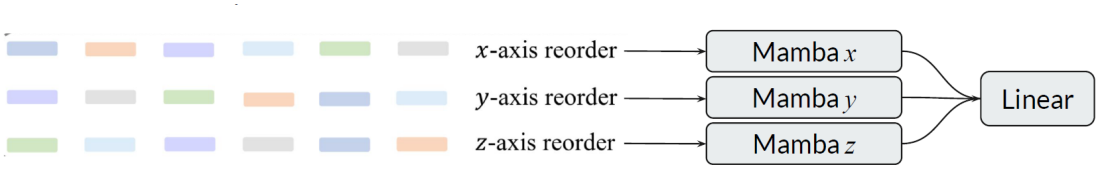


Figure 6: Instead of concatenating the reordered sequences together, we opt to pass the sequences through Mamba blocks in parallel. This preserves the consistency of the input sequence passed into the Mamba blocks. This approach also allows for better pooling of information across different serialization strategies.

5 Parallelizing PCM

To implement these changes, we build upon the architecture of Point Cloud Mamba. We will refer to our novel architecture as Parallel PCM which is visualized in Figure 7. As explained earlier, the crucial architecture change we make is to parallelize the Mamba blocks within each stage of Sequential PCM. Instead of passing two orderings in sequence through two Mamba blocks, we pass in two serialization strategies in parallel and then concatenate and project down their features (as done in multi-headed attention Vaswani et al. [2023]).

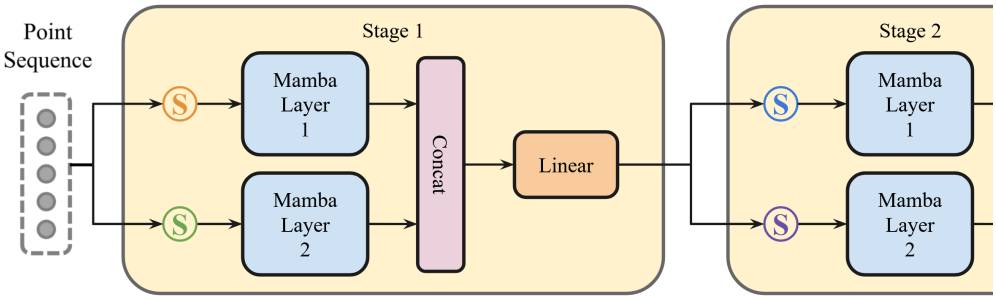


Figure 7: Our implementation of Parallel PCM using base diagram taken from Zhang et al. [2024].

Afterwards, we can extend the model to be deeper by taking the same approach as Sequential PCM to serialize the output features from each stage, but instead using our strategy to pass in spatial orderings in parallel to the Mamba blocks in the next stage.

6 Results

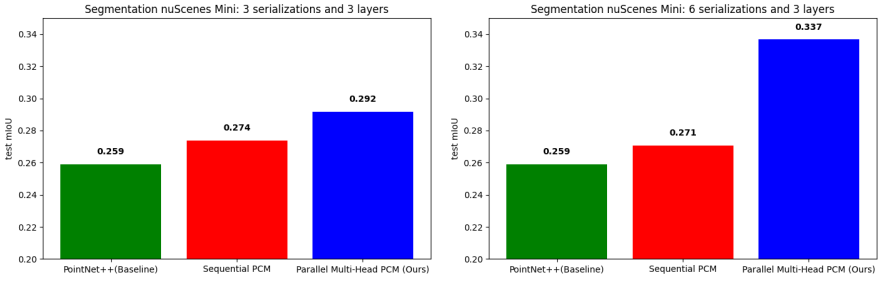
6.1 nuScenes Mini

Given our computational resources, it is infeasible to train or even store the full volume of nuScenes data. For that reason, we first trained and evaluated our models on nuScenes Mini, which is a subset that comprises 404 points clouds, each with around 35,000 points, on which we performed a 90%-10% train-test split. We trained and evaluated the following three models:

- **PointNet++ (baseline)** (Qi et al. [2017]): This model is an extension on PointNet Qi et al. [2016], which was a pioneering deep learning architecture capable of achieving strong performance on indoor point clouds. PointNet++ uses a hierarchical structure to gradually refine features at different scales, which allows it to capture more detailed geometric information, making it more effective for tasks like semantic segmentation of point clouds. We use this as our baseline because it is a well-known and popular model, and because it can run efficiently on outdoor point clouds. In contrast, we could not use PointTransformer (Zhao et al. [2021]) on outdoor point clouds because given our GPU power and the quadratic memory limitations of transformers we could not run it on sequences of length 35,000.
- **Sequential PCM** (Zhang et al. [2024]): This is the model we introduced in section 4.3, which had the strongest performance in our midway report.
- **Parallel PCM (our contribution):** A refinement over Sequential PCM, which uses parallel Mamba blocks instead of sequential ones, as explained in section 5.

The results for nuScenes-Mini are shown in figure 8. In the left figure, the results of PCM with 3 serializations are shown. We can see that both Sequential and Parallel PCM outperform PointNet++, yet the performance of Parallel PCM is similar to the performance of Sequential PCM. However, when we increase the number of serializations, we observe that the Parallel PCM increases its performance by a lot (around 15%), whereas Sequential PCM's performance doesn't increase. This is one of the most important findings of the report, which validates our hypothesis that sequentially passing multiple serializations to Mamba is not optimal. Given that the serializations represent the same points in different orders, the gradients must propagate

through a sequence of layers with very similar inputs, which make it more difficult to learn. In contrast, when the serializations are passed in parallel to Mamba, the gradient has a direct path to each block's weights, which helps learning.



(a) 3 stages (layers), each with 3 serializations.

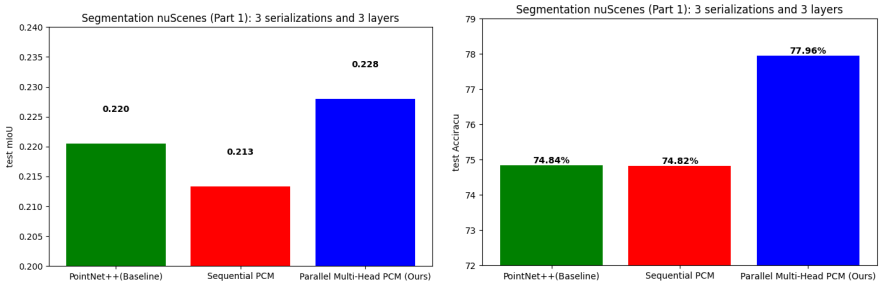
(b) 3 stages (layers), each with 6 serializations.

Figure 8: Test mIoU for NuScenes Mini test split. Ablations on number of serializations per stage/layer.

6.2 nuScenes (Blob 1)

Now, instead of using the nuScenes Mini subset, we downloaded one of the blobs of the full nuScenes dataset. Originally, we tried training for this full blob, but the training time was extremely slow (over half an hour per epoch), so we took a subset of this part, comprised of 1000 point clouds. We used a 90%-10% train-test split and trained the 3 aforementioned models. The results of this experiment are shown in Figure 9.

In the Blob 1 dataset, we observe that the test mIOU is lower across all models compared to the Mini dataset. The happens because the dataset consists of a more varied collection of scenes, as opposed to nuScenes Mini, which only consists of a limited number of similar point clouds. Furthermore, the extreme class imbalance makes it hard for models to learn, which explains why all models perform similarly. For that reason, we decided to further explore the problem of class imbalance and attempt different solutions.



(a) Test mIOU

(b) Test Accuracy

Figure 9: Performance Metrics for nuScenes Part 1. 3 stages (layers) with 3 serializations.

6.3 Class Imbalance

nuScenes is a dataset which is known for its extreme high imbalance (Caesar et al. [2020], Lee et al. [2022], Nagesh et al. [2022]). The class imbalance ratio for some instances is 1:10k, which makes this dataset notoriously hard for segmentation. When training our models in nuScenes Part 1, we found that the models diverged easily because they just focus on the majority classes.

To deal with this problem, we implemented three methods to deal with class imbalance: Frequency Weighted Softmax, Focal loss, and Gradient Clipping. We explain them in the appendix C.1. In this section, we show the effect of these strategies, compared to "Nothing" (training without any strategy). Figure 11 shows the

effect the strategies have on the test curve for Parallel PCM and 11. As we can see, if we don't apply any class imbalance strategy, the loss diverges. This happens because the dataset is so imbalanced that the model falls into a local minima where it just outputs one of the majority classes. As seen in figure 11, the model without a class imbalance strategy simply outputs "Flat Driveable Surface" all the time, which is one of the most frequent classes.

This divergence is solved with any of our training strategies. As we can see in 11, the test mIOU curve does not diverge with these strategies. The best test mIOU is obtained with Gradient Clipping, yet interestingly, the strategies have different effects on the model performance, as seen in 11. The Frequency Weighted Softmax is the one that produces decent accuracy for the most number of classes. This makes sense, given that even if such classes appear very frequently, the model will be heavily penalized if it missclassifies them, which is why it tries to have decent accuracy in most classes (at the cost of the performance in more frequent classes). Focal loss and Gradient Clipping, in contrast, don't have this kind of penalization, so they keep focusing on the most frequent classes.

Unfortunately, the problem with these strategies is that they reduce the gradient signal that gets back-propagated to the weights. For example, focal loss reduces the value of the cross entropy loss for the most frequent classes. But these predictions turn out to be the great majority of the points, which is why the focal loss is very small, thus introducing the problem of vanishing gradients. Therefore, we conclude that simply modifying the loss or training procedure is not enough to deal with class imbalance, a minority-class augmentation method must be used to achieve state of the art performance.

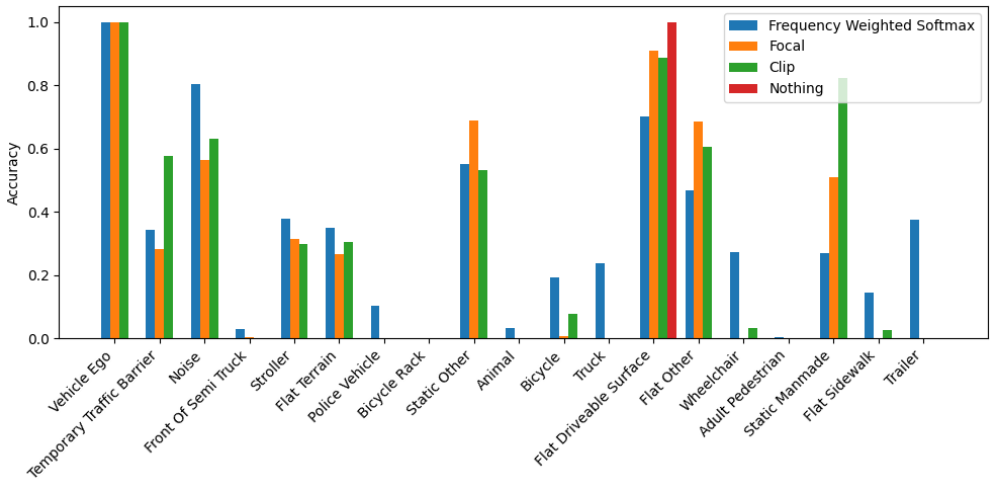


Figure 10: Accuracy by class for different class imbalance strategies. (Model: PCM Parallel).

A Summary

We adapted three Mamba-based 3D semantic segmentation models to train on the nuScenes dataset and evaluated each model's 3D to 1D serialization method. We also evaluated the performance of each model on nuScene's sparse 3D point cloud representations of outdoor scenes via accuracy and mIoU metrics.

After evaluating multiple serialization strategies, we conclude that providing Mamba multiple serializations results in the best performance on 3D semantic segmentation of point clouds. Building on Sequential PCM Zhang et al. [2024], we develop a novel Parallel PCM architecture. Unlike Sequential PCM, our method scales with the number of serializations provided to each stage comprised of multiple Mamba blocks which act in parallel on unique 3D to 1D serializations of our input point cloud data.

B Future Directions

One future direction is to go for a mixture of experts approach with Mamba Blocks. By assigning a Mamba expert to each spatial ordering, we can consider a weighted prediction of experts that can decide what spatial orderings are most useful to consider for a given sequence of 3D points.

Additionally, with our promising results from utilizing Mamba for semantic segmentation, we also want to explore using Mamba for the semantic scene completion task in a multi-stage architecture similar to the models mentioned previously that are the current SOTA. Considering that the VoxFormer architecture itself requires a depth map as a part of its two-stage architecture, we could even consider applying Mamba to the 3D reconstruction task using stereo views, as this task is currently dominated by a transformer-based model known as MVSFormer++(Cao et al. [2024]).

Drawing inspiration from Flash Attention, we can also develop a very efficient Parallel Point Cloud Mamba hardware-aware implementation since the Mamba blocks are being run in parallel on different serializations of the same input sequence. Our novel contribution of Multi-headed Mamba blocks can also be applied to any sequence-to-sequence tasks, including NLP and 2D vision tasks.

Bibliography

Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuscenes: A multimodal dataset for autonomous driving. In *CVPR*, 2020.

Chenjie Cao, Xinlin Ren, and Yanwei Fu. Mvsformer++: Revealing the devil in transformer's details for multi-view stereo, 2024.

K. Ruwani M. Fernando and Chris P. Tsokos. Dynamically weighted balanced loss: Class imbalanced learning and confidence calibration of deep neural networks. *IEEE Transactions on Neural Networks and Learning Systems*, 33(7):2940–2951, 2022. doi: 10.1109/TNNLS.2020.3047335.

Albert Gu and Tri Dao. Mamba: Linear-time sequence modeling with selective state spaces, 2023.

Daeun Lee, Jongwon Park, and Jinkyu Kim. Resolving class imbalance for lidar-based object detector by dynamic weight average and contextual ground truth sampling, 2022.

Dingkang Liang, Xin Zhou, Xinyu Wang, Xingkui Zhu, Wei Xu, Zhikang Zou, Xiaoqing Ye, and Xiang Bai. Pointmamba: A simple state space model for point cloud analysis, 2024.

Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection, 2018.

Jiuming Liu, Ruiji Yu, Yian Wang, Yu Zheng, Tianchen Deng, Weicai Ye, and Hesheng Wang. Point mamba: A novel point cloud backbone based on state space model with octree-based ordering strategy, 2024.

Sushruth Nagesh, Asfiya Baig, Savitha Srinivasan, Akshay Rangesh, and Mohan Trivedi. Structure aware and class balanced 3d object detection on nuscenes dataset, 2022.

Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation, December 2016. URL `http://arxiv.org/abs/1612.00593`.

Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space, 2017.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

Yan Yan, Yuxing Mao, and Bo Li. Second: Sparsely embedded convolutional detection. *Sensors*, 18(10), 2018. ISSN 1424-8220. doi: 10.3390/s18103337. URL `https://www.mdpi.com/1424-8220/18/10/3337`.

Tao Zhang, Xiangtai Li, Haobo Yuan, Shunping Ji, and Shuicheng Yan. Point could mamba: Point cloud learning via state space model, 2024.

Hengshuang Zhao, Li Jiang, Jiaya Jia, Philip Torr, and Vladlen Koltun. Point transformer, 2021.

C Appendix:

C.1 Class Imbalance Strategies

Data augmentation techniques are commonly applied to nuScenes training (Yan et al. [2018], Lee et al. [2022]). Nevertheless, these methods require a large amount of training instances of minority classes, because otherwise the same instances will be repeated several times, which leads to neural network memorization. In turn, this requires large amounts of storage (over a hundred GBs) which we didn't have.

Therefore, we instead tried the following strategies that can be easily implemented:

- **Gradient Clipping:** Although not designed specifically to tackle class imbalance, we observed that loss landscape was very abrupt due to the presence of big majority classes, so we tried gradient clipping to avoid falling into undesired local minima.
- **Focal Loss** (Lin et al. [2018]): This loss gives less importance to the classes for which the predicted probability matches well with the ground truth (in our case, the majority classes).
- **Weighted Softmax** (Fernando and Tsokos [2022]: A softmax where the values for each class are weighted based on the frequency of the class in the datasets. The classes that are more frequent are weighted less. We implemented this dynamically (computed the weights on each batch) and used the log of the inverse of the frequencies in the batch as weights.

The effect of these strategies is shown in Figures C.1, 11, and 10.

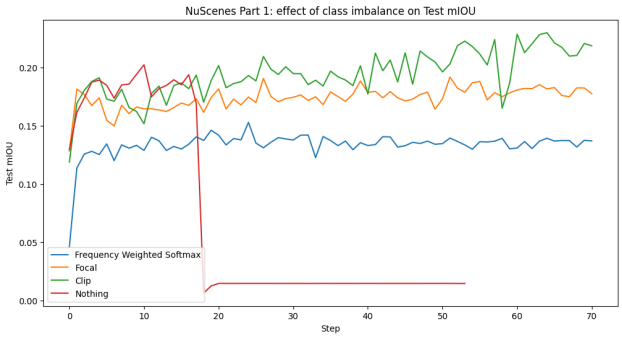


Figure 11: Test mIOU for different class strategies to tackle the class imbalance issue. (Model: PCM Parallel).

Class Imbalance Histogram

A histogram of the class imbalance present in our problem is shown in 12. Note the logarithmic scale.

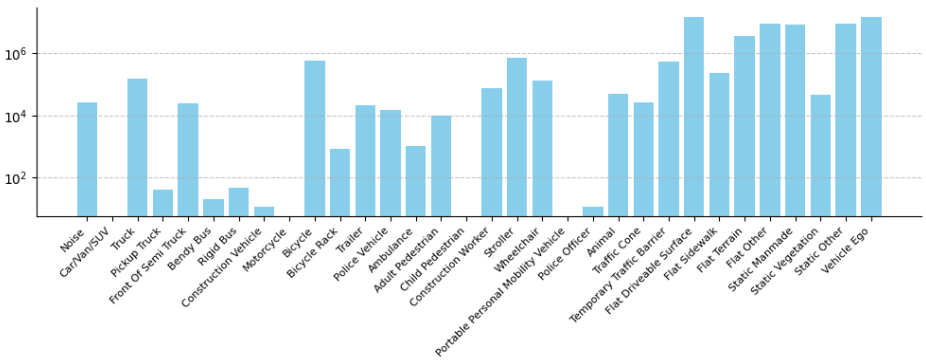


Figure 12: Number of Appearances of Each Class in subset of nuScenes. Note the logarithmic scale.