university of groningen

faculty of science and engineering

Exploring the possibility, challenges and (dis-)advantages of deep ensemble methods in Point Cloud Completion

Jakob de Boer

university of groningen

faculty of science and engineering

University of Groningen

Exploring the Possibility, Challenges and (Dis-)Advantages of Ensemble Methods in Point Cloud Completion

MASTER'S THESIS

To fulfill the requirements for the degree of
Master of Science in Artificial Intelligence
at University of Groningen under the supervision of

Prof. Dr. Hamidreza Kasaei,
ir. Madhur Madhur,
ir. Frank Lammers

Jakob de Boer (s3803996)

March 20, 2026

Abstract

Reconstructing a complete 3D point cloud from an incomplete one is a critical challenge in numerous vision and robotics applications. Various point cloud completion methods have previously been introduced, each with its own strengths and weaknesses. In an attempt to make different methods complement one another, ensemble methods for Point Cloud Completion (PCC) tasks are proposed. Ensemble learning methods aim to enhance overall predictive performance beyond that achieved by any single constituent algorithm. In this thesis, six different deep-learning baseline PCC algorithms (PoinTr [1], PCN [2], GRNet [3], FoldingNet [4], SnowFlakeNet [5] and TopNet [6]) are combined using three novel ensembling criteria (Point Aggregation, Uniform Sampling and Performance-based Sampling) in both homo- and heterogeneous ensembles. Homogeneous ensemble base models are diversified with both random initialization and bagging. Parts of the large space of ensemble configurations are validated on the Completion3D dataset and a novel agricultural completion dataset. It is observed that ensembles using the point aggregation criterion improve chamfer distance beyond that of individual base learners up to 41.0%. Both random initialization and bagging successfully diversify base models, while randomly initialized base learners outperform the bagged alternative in both individual and ensembled contexts. Similar ensemble mechanisms are observed on the agricultural completion dataset.

Acknowledgments

This thesis would not have been possible without the support of several individuals. I want to express my gratitude to my supervisors, Hamidreza Kasaei, Madhur Madhur, and Frank Lammers, for their invaluable guidance and feedback, which greatly shaped this project.
I also appreciate everyone I met at Batenburg Beenen, especially in the R&D department and my fellow interns. Your support during presentations, insightful advice, and shared breaks made my experience very enjoyable.
Additionally, I want to thank the Center for Information Technology of the University of Groningen for their support and for providing access to the Hábrók high performance computing cluster. The computational resources provided were crucial for my experiments.
Lastly, I am grateful to everyone who offered support throughout this journey. Thank you for your encouraging words and for providing welcome distractions along the way.

Contents

Abstract I

Acknowledgements II

List of Figures V

List of Tables VII

1 Introduction 1
- 1.1 Research Questions . . . 1
- 1.2 Thesis Outline . . . 2

2 Background Literature 3
- 2.1 Ensemble learning . . . 3
  - 2.1.1 Traditional methods . . . 3
  - 2.1.2 Deep learning based methods . . . 4
- 2.2 Point cloud completion . . . 5
  - 2.2.1 Traditional methods . . . 5
  - 2.2.2 Deep learning based methods . . . 6
- 2.3 Ensemble learning in Point Cloud Completion . . . 8

3 Methods 9
- 3.1 Problem statement . . . 9
- 3.2 Learning algorithms . . . 9
  - 3.2.1 FoldingNet: Point cloud auto-encoder via deep grid deformation . . . 9
  - 3.2.2 PCN: Point Completion Network . . . 10
  - 3.2.3 TopNet: Structural point cloud decoder . . . 10
  - 3.2.4 GRNet: Gridding residual network for dense point cloud completion . . . 11
  - 3.2.5 SnowFlakeNet: Point cloud completion by snowflake point deconvolution with skip-transformer . . . 12
  - 3.2.6 PoinTr: Diverse Point Cloud Completion with Geometry-Aware Transformers 13
- 3.3 Learning strategies . . . 14
  - 3.3.1 Bagging . . . 14
  - 3.3.2 Random initialization . . . 14
- 3.4 Ensembling criteria . . . 15
  - 3.4.1 Point Aggregation . . . 15
  - 3.4.2 Uniform sampling . . . 15
  - 3.4.3 Performance-based sampling . . . 15
- 3.5 Performance Measure . . . 16
  - 3.5.1 Chamfer distance . . . 16

4 Experimental Setup 17
- 4.1 Datasets . . . 17
  - 4.1.1 Completion3D dataset . . . 17
  - 4.1.2 Agricultural dataset . . . 17
- 4.2 Base learner training . . . 18

4.3 Experiments . . . 19
4.3.1 Individual methods . . . 19
4.3.2 Homogeneous ensembles . . . 19
4.3.3 Heterogeneous ensembles . . . 19

5 Results **21**
5.1 Individual base learners . . . 21
5.2 Homogeneous ensembles . . . 23
5.2.1 Qualitative analysis . . . 23
5.2.2 Quantitative analysis . . . 24
5.3 Heterogeneous ensembles . . . 25
5.3.1 Qualitative analysis . . . 25
5.3.2 Quantitative analysis . . . 26
5.4 Agricultural dataset . . . 27

6 Conclusion & Discussion **30**
6.1 Research questions . . . 30
6.2 Discussion . . . 31
6.3 Future Work . . . 31

Bibliography **32**

A List of learning algorithm hyperparameters **38**

Appendices **38**

List of Figures

3.1 Ensemble methods are composed of a collection of base learners and a defined ensemble criterion. The base models are trained using a specific learning algorithm, which may also incorporate a learning strategy to differentiate homogeneous sets of base learners. . . . 9

3.2 The FoldingNet architecture. Both the 1st and 2nd folding are achieved by concatenating the codeword with the feature vectors, which are then processed by a 3-layer perceptron. Each perceptron operates independently on the feature vector of an individual point, effectively applying to the rows of the m-by-k matrix. Adapted from [4]. . . . 10

3.3 The PCN architecture. The encoder transforms the input point cloud X into a feature vector v. The decoder utilizes v to initially produce a coarse output Y_{coarse}, followed by a detailed output Y_{detail}. Each colored rectangle represents a matrix row, with the same color denoting identical content. Adapted from [2]. . . . 11

3.4 The TopNet architecture. The completion framework consists of a 2-stage point cloud encoder and a tree-structured decoder. The decoder's arrows are multilayer perceptron networks (MLPs), with MLPs of the same color sharing the same parameters. Adapted from [6]. . . . 12

3.5 The GRNet architecture. It includes (a) GRNet, (b) Gridding, (c) Gridding Reverse, (d) Cubic Feature Sampling, and (e) Gridding Loss. Adapted from [3]. . . . 12

3.6 (a) The overall architecture of SnowflakeNet includes three modules: feature extraction, seed generation, and point generation. (b) Details of the seed generation module. (c) Snowflake Point Deconvolution (SPD). Here, N, N_c, and N_i represent the number of points, while C and C' denote the number of point feature channels, which are 512 and 128, respectively. Adapted from [5]. . . . 13

3.7 The PoinTr architecture. The PoinTr pipeline begins by downsampling the input partial point cloud to derive the center points. Next, a lightweight DGCNN [7] extracts local features around these center points. After incorporating position embeddings into the local features, we employ a transformer encoder-decoder architecture to predict point proxies for the missing regions. Subsequently, a simple MLP and FoldingNet are utilized to complete the point cloud in a coarse-to-fine manner based on the predicted point proxies. Adapted from [1]. . . . 14

4.1 Samples from the Completion3D dataset. The partial point clouds are drawn in blue and the associated completed ground truth in green below. Samples from left to right are respectively from the car, chair, lamp, watercraft, couch, table, plane and cabinet category. . . . 17

4.2 Samples from the introduced synthetic agricultural dataset. The partial point clouds are drawn in blue and the associated completed ground truth in green below. . . . 18

4.3 Template mesh for the agricultural dataset. The template peduncle is simulated with soft-body physics and the fruits are randomly scaled to generate dataset samples. . . . 18

5.1 Boxplot displaying the distribution of chamfer distances across various learning algorithms combined with either learning strategy. Each algorithm's performance is represented by the mean chamfer distance, with error bars indicating variability. Results are collected on the Completion3D dataset. . . . 21

5.2 Qualitative results from different randomly initialized learning algorithms for eight validation samples from different categories from the Completion3D dataset. Different base learners produce different predictions. Qualitative results from bagged base learners look very similar. 23
5.3 Homogeneous ensemble outputs for each learning algorithm using the random initialization strategy and the aggregation criterion. While making the point cloud more dense, increasing the amount of base learners does not produce significant structural differences. Similar qualitative results are found for ensembles using the bagging learning strategy. 24
5.4 Comparison of different ensemble criteria over ensemble size for homogeneous ensembles. The aggregation criterion performs significantly better than both sampling based criteria. Results collected for various homogeneous ensembles on the Completion3D validation set. 25
5.5 Qualitative results for five heterogeneous ensemble configurations. Column by column, the ensemble is extended with the respectively labeled base learner. Increasing the amount of learning algorithms result in denser point clouds and combined structures seem to emerge. Created with randomly initialized base learners using the aggregation ensemble criterion on the Completion3D validation set. 26
5.6 Qualitative results from different randomly initialized learning algorithms for a sample from the agricultural dataset. Different base learners produce different predictions. 27
5.7 Boxplot displaying the distribution of chamfer distances across various learning algorithms trained with either Bagging or Random Initialization strategy. Results collected on the agricultural dataset. As opposed to the performance on the Completion3D dataset, GRNet performs better than PCN and TopNet. 28
5.8 Qualitative results for five heterogeneous ensemble configurations. Column by column, the ensemble is extended with the respectively labeled base learner. While the fifth (top) fruit is not visible in the partial point cloud, The largest ensemble seem to predict the correct position. Created with randomly initialized base learners using the aggregation ensemble criterion on the agricultural validation set. 29
6.1 To evaluate ensemble methods for point cloud completion in the agricultural domain, real-life experiments should be conducted such as depicted. Illustration of possible real-life experimental setting for evaluating PCC ensembles. Image adapted from https://www.ai.rug.nl/irl-lab/. 31

List of Tables

5.1 Performance comparison of different learning algorithms under two learning strategies: Random Initialization and Bagging, evaluated on the Completion3D dataset. The values represent average chamfer distance over 10 separately trained base learners. SnowFlakeNet outperforms all other learning algorithms under both learning strategies. 22
A.1 Hyperparameters configuration for the FoldingNet model for training. 38
A.2 Hyperparameters configuration for the GRNet model for training. 38
A.3 Hyperparameters configuration for the PCN model for training. 39
A.4 Hyperparameters configuration for the PoinTr model for training. 40
A.5 Hyperparameters configuration for the TopNet model for training. 41
A.6 Hyperparameters configuration for the SnowFlakeNet model for training. 41

1 Introduction

Traditionally, robots have been utilized in rigid, predictable contexts such as car manufacturing robots that sequentially execute hard-coded routines. These robots are specifically designed for the task at hand and there is little flexibility in the tasks that these robots can perform individually. The behavior of these robots is very limited in terms of converting their perception to a proper action. Nevertheless, these robots have significantly reduced the need for dangerous, monotonous or physically demanding human jobs.

Yet, a lot of these jobs can still not be performed by robots as they require extreme dexterity, or a perfect understanding of the complex environment in which is operated. A real-life example is the task of picking fruits. This task requires a range of different skills that are non-trivial for robots. First the robot needs to perceive and understand the environment to map and detect pick-worthy fruits. Then, the robot needs to carefully pick fruits between the fragile stems, branches and leaves without damaging anything else. For each of these steps, no simple if-this-then-that routine suffices to succeed at this task.

Instead of concentrating on a complete pipeline from perception to manipulation, this thesis is limited to perception. Challenges in perception arise from finding an accurate and complete representation of the environment that a robot operates in. Environments are usually captured using 3D point cloud representations. But due to occlusions, reflections or blind spots, the captured point-clouds are often incomplete [8].

The challenge of reconstructing incomplete three-dimensional shapes has previously led to the development of a diverse range of Point Cloud Completion (PCC) methods. More traditional methods complete shapes using primitive geometry- [9] or symmetry- [10] heuristics. Alternatively deep learning methods based on encoder-decoder- [11], transformer- [12], diffusion- [13] or combinations of these architectures were proposed. Regardless of the nature of each method, they all suffer and benefit from their individual characteristics [8].

This thesis investigates the potential of ensemble methods that are derived from multiple deep learning Point Cloud Completion methods to enhance the accuracy of the predicted complete point clouds. Ensemble methods, which combine the outputs of diverse learning algorithms, may mitigate the shortcomings of individual methods by integrating their strengths. Multiple PCC ensemble configurations are proposed and investigated on a household objects dataset and a novel agricultural dataset.

This idea is inspired on ensemble methods like random forests [14] or boosting [15] methods, which gain from a larger pool of models. Current point cloud combination methods have mostly focused on progressively mapping large (outdoor) environments [16] or low-level combinations of different sensors [17]. Yet, little research has been performed in the direction of combining several different methods for complementary point cloud completion. If and when certain PCC methods are composed, it is expected that the whole is greater than the parts.

1.1 Research Questions

The possibility, challenges and the potential (dis-)advantages of combining multiple PCC methods on household objects and a novel agricultural dataset is studied. To limit the scope of this thesis, only data-driven PCC methods are considered. The thesis focuses on the following sub-questions:

Q1. What are the (dis-)advantages of combining multiple Point Cloud Completion methods?

Q2. How can the output of different PCC algorithms be combined to form a single more complete point-cloud?

Q3. What is the relation between ensemble size and ensemble performance for homogeneous ensembles?

Q4. How does the base learner selection influence heterogeneous ensemble dynamics?

Q5. How do different learning strategies impact the results for homogeneous ensembles?

1.2 Thesis Outline

This thesis explores the intersection of ensemble learning and point cloud completion. Following the introduction, the background literature reviews existing methods in ensemble learning and point cloud completion, highlighting key traditional and deep learning techniques. In the methods section, the formal problem statement is defined alongside a detailed description of various learning algorithms, strategies, and performance measures. The experimental setup outlines the training of base learners and the datasets utilized for testing. Lastly, results are presented for both datasets, leading to a conclusion that summarizes contributions and suggests areas for future research.

2 Background Literature

2.1 Ensemble learning

Ensemble learning methods leverage multiple machine learning algorithms to produce predictions based on features derived from the diverse set of data projections. By integrating these predictions through various aggregation techniques, such as voting or averaging, ensemble approaches aim to enhance overall predictive performance beyond that achieved by any single constituent algorithm. Ensemble methods can be divided in both traditional and deep learning-based methods [18].

2.1.1 Traditional methods

The initial ideas for ensemble learning were proposed in works from Sheela [19], Kearns [20] and Schapire [21]. Sheela et al. introduced the idea of using multiple classifiers trained on different categories. These classifiers were combined to enhance the performance of the complete system [19]. Kearns et al. studied a similar problem related to the relationship between weak learners and strong learning algorithms observed in PCA models [20]. After which Schapire et al. investigated the possibility of employing multiple weak learning models to produce a high-precision model [21].

The design of ensemble learning methods consists of generating multiple unique weak learners and determining how to aggregate these model outputs to a single prediction. First, a homo- or heterogeneous set of learning algorithms are selected.

Multiple different **learning algorithms** can be selected to form a heterogeneous ensemble. Alternatively, a homogeneous ensemble contains learners based on a single learning algorithm [22]. To differentiate base models in homogeneous ensembles, **learning strategies** have to be adopted. This ensures that base models are unique and that equal learning algorithms generate different predictions. Various learning strategies have been proposed.

Breiman et al. introduced the concept of bagging [23]. By resampling the dataset with replacements into multiple different datasets, different base models are generated. Tests conducted on both real and simulated datasets using classification and regression trees demonstrated that bagging can significantly enhance accuracy. A crucial factor in this process is the instability of the prediction method. When minor changes to the learning set lead to considerable variations in the constructed predictor, bagging can effectively improve accuracy. This success with bagging gave rise to various variations of bagging. An approach termed dagging was proposed by Ting et al. Instead of sampling datasets with replacements, disjoint training datasets are generated by sampling without replacements [24]. Another alternative to bagging is wagging which was proposed in by Bauer et al. To generate training datasets, training samples are randomly weighed for sampling. These random weights are initialized using Gaussian or Poisson distributions with a mean of zero [25]. Boosting was introduced by Freund et al., who also proposed manipulating training data to differentiate base models. By iteratively reweighing training samples according to their training accuracy and resampling training sets, subsequent base models iteratively focus on difficult samples [15]. A combination of Wagging and Boosting was proposed by Webb et al. named multiboosting [26]. That uses boosting for assigning weights to training samples and uses wagging for updating these weights.

A popular learning strategy for Neural Networks (NN) is to use random network initializations. These random initializations tend to deliver different NN models.

To generate a single unified output from the base models, an **ensemble criterion** is used. Ensemble criteria can roughly be divided in three categories: Weighting methods, meta-learning methods and ensemble selection methods [22].

Weighting methods: Weighting methods rely on the idea of assigning weights to the base model outputs to produce a single output. Classification models usually produce a one-hot-encoding output. For these classification problems, the majority voting criterion can be used to combine outputs. This criteria dictates that the output of the ensemble is the class with the most 'votes' from base models. This is an example of a weighting method were each method is weighted equal. For regressions, where the output is a rational number, this weighting concept is applied by averaging all base model outputs [27]. Conversely, base model outputs can be weighed according to their respective performance on a validation set [22].

Ensemble selection methods: Ensemble selection methods work by training several different base models and then selecting a subset of most effective base learners. This idea was introduced by Zhou et al., as they found that a subset of base learners may perform better than the complete set of base learners [28]. An idea proposed by Martinez et al. was to rank base learners by the associated accuracy. Then the set of base models is pruned based on its respective rank [29]. Selecting a subset of learners may not always perform optimally, but ensemble selection methods do have the additional benefit of lower space- and time-complexity [22].

meta-learning methods: The core idea of meta-learning methods is to reduce generalization errors by applying machine learning to the meta-knowledge captured by each individual base learner and the dataset. An example of these meta-learning methods is the Mixture of Experts method (MoE) proposed by Masoudnia et al. [30]. By dividing the problem to sub-problems and training respective base learners, a set of 'experts' is created. Another model is then trained on these experts, samples and labels to produce an output.

2.1.2 Deep learning based methods

The increased popularity of deep learning and deep neural networks also sparked the creation of ensembles consisting of multiple deep learning models. Deep ensembles were initially proposed by Lakshminarayanan et al. [31]. Neural networks can be used as a learning algorithm to function in the traditional framework, but depending on the context, deep learning methods may also be used to replace the ensemble criterion [22].

In the context of wind power forecasting, Wang et al. proposed to implement multiple deep CNN's as as a group of base models. These CNN's were trained on features from raw wind power data which were converted into 2D images. The CNN's varied in the number of hidden layers, the number of feature maps per layer and the input size. Additionally, each learner was trained on different wind power features. The ensembling criterion was based on wavelet reconstruction. This ensemble method was evaluated using actual wind farm data from China. Their results indicated that uncertainties in wind power data were more effectively captured with this approach, leading to competitive performance [32].

To forecast electricity load demands, Qiu et al. proposed an ensemble using empirical mode decomposition and deep belief networks. Using empirical mode decomposition the time series of load demands were decomposed in several intrinsic mode functions. For each of the intrinsic mode functions a deep belief network was trained to construct the set of base learners. The outputs of the base learners were combined by averaged summation or linear combinations to form the ensemble criterion. The electricity load demand datasets from the Australian Energy Market Operator (AEMO) were utilized to evaluate the effectiveness of the proposed approach. The simulation results highlighted the advantages of this method when compared to nine other forecasting techniques. [33].

2.2 Point cloud completion

Point cloud completion is part of the field of Object completion, which is part of the greater field of Shape completion. Shape completion is concerned with predicting the complete geometric from a partial shape of any 3D data format. This data type may be voxels, point clouds and meshes [34].

Aside from Object completion, Shape completion additionally includes semantic scene completion and semantic instance completion. The task of semantic scene completion is about predicting complete shapes of all objects in a scene and labeling these objects. Semantic instance completion aims to detect the instances in a scene/object and predict their complete geometry. These tasks are concerned with multiple objects in a scene and the semantics of the objects. Object completion is different as it focuses on predictions on single objects and is not concerned with semantically labeling this object.

Object point clouds can be captured by stereoscopic depth cameras, LiDAR or 3D laser scanners. These sensors generate a 3D representation consisting of multiple points in Cartesian space (x-, y- and z-coordinate) from the surface of the perceived object. Depending on the used sensor, these points may also carry additional information such as color.

Point cloud scans frequently suffer from incompleteness due to various factors: Complex morphological textures can result in sensor blind spots. Surfaces that exhibit mirror-like properties may prevent accurate scanning in those areas. Additionally, occlusions caused by the object itself or by external objects can further restrict the completeness of the point cloud data [34].

Point cloud completion methods are primarily divided in three different categories [34] [35]: Traditional, learning based- and alternative methods. The latter will not be discussed in this thesis.

2.2.1 Traditional methods

Traditional methods optimize parameters of shape models to fit the partial point clouds and generate a completed output. These traditional methods can be subdivided in alignment-based, geometry-based, interpolation-based and matching-patches-based methods [35].

Matching-patches-based: Matching-patches-based methods function on the idea that missing regions of the input point clouds are likely similar to the given partial region. Therefore, similar points or patches from the input point cloud are matched to the missing parts. When a match is found, the patch from the input is used to fill the holes. Various examples of these methods have been proposed ([36], [37], [38] [39]). These methods work well when similar patches can be found in the partial point cloud. Yet, these methods are sensitive to noise and finding a proper match might hinder predictions if no match is found.

Geometry-based: Geometry-based methods complete shapes based on geometric properties. Most objects exhibit certain structural properties. These properties can be exploited to complete the partial point cloud. Various geometric cues have previously been proposed among symmetry [40] and regularity [41]. Several alternative geometry-based methods with different geometric cues have been proposed ([42], [43], [44], [45]). Geometry-based methods work well on objects that display strong global structures. They exhibit limited performance on irregularly shaped objects.

Alignment-based: Alignment-based methods rely on matching the partial input shape to an object from a dataset. This dataset contains a large amount of completed objects and the objects from this dataset are compared to the partial point cloud to see if the models align. If these objects align, the shape is completed using the template shape [46]. To increase the flexibility of this approach, methods where multiple template shapes are combined have also been proposed by Schnabel et al. [9]. Alternatively, instead of using complex models as a template dataset, (multiple) primitive shapes have also been used to complete partial point clouds. Other similar alignment-based methods have also been proposed [47] [48]. Alignment-based methods function well when the input objects are part

of the shape database. When the input object is not part of the shape database, the predictions may be misleading.

Interpolation-based: Interpolation-based methods are more suitable for surface reconstruction, but are still applicable to point cloud completion. These methods fill missing point cloud regions based on smooth interpolation over the missing areas [49]. Various methods have been proposed that fill so-called gaps by using curve-completion algorithms [50] or Poisson surface reconstruction [51]. Interpolation-based methods work very well for small holes in the partial point cloud. Nevertheless, the method often fails for partial point clouds with larger missing portions. Additionally, the optimization has a large computational cost at inference.

2.2.2 Deep learning based methods

Deep-learning-based completion algorithms can be roughly divided in six different architectures: *graph-based*, *generative-model-based*, *point-based*, *convolution-based*, *transformer-based* and alternative methods [35]. These architectures may be applied in supervised, weakly-supervised [52] and unsupervised [53] manners. Only supervised techniques will be discussed in this thesis. In this section, we first briefly discuss the graph-based and generative models that are outside the scope of this thesis, then the point-based, convolution-based and transformer-based methods are discussed, which form the basis of our ensemble methods.

Graph-based: Graph-based methods consider individual points in point clouds as vertices in a graph. Edges are generated between neighboring points, after which these graphs are handled by graph convolutions. These graph convolutions gather the information from the spatial neighborhoods of each point and generate a new graph. Graph-based PCC methods will not be discussed in this thesis. For a more comprehensive overview of graph-based point cloud completion methods, the reader may refer to [54].

Generative-model-based: Three types of generative-model based architectures can be distinguished: Generative adversarial networks (GANs) Variational-autoencoders (VAEs) and diffusion models. These methods often generate outputs inspired on the training data. Generative-model-based PCC methods will not be discussed in this thesis. For a more comprehensive overview of GAN- and VAE-based point cloud completion methods, the reader may refer to [8].

Point-based: Point-based methods rely heavily on the efforts of PointNet and PointNet++. PointNet [55] is a method that learns features directly from point clouds. The point cloud is used as an input, which is up-sampled through a few multilayer perceptron (MLP) layers. Max pooling is performed over all points in every dimension which results in a global feature of 1×1024. Due to this architecture, it only obtains global features from the point cloud. Local features are not integrated. PointNet++ from [56] improved on this. By introducing a multi-scale structure inspired by PointNet, the network is able to output higher-level features over larger scales.

Yuan et al. [2] introduced the PCN algorithm for achieving dense point cloud completion. PCN operates on an encoder–decoder architecture and employs a coarse-to-fine strategy. The encoder extracts the global feature vector of the point cloud using an extended network based on PointNet. This global feature vector enables shape completion of the dense point cloud through a two-stage decoder. In the first stage, fully connected (FC) layers and reshape operations generate rough point clouds. The second stage utilizes the folding network [4] to integrate global feature vectors to create dense point clouds.

Based on the encoder from PCN, Tchapmi et al. proposed TopNet [6]. This approach incorporated a hierarchical tree structured decoder, which takes the geometric information into account. This generates a structured point cloud without relying on any specific structure or topology. The root

node of the decoder holds the global feature vector derived from the encoder, which is then processed by a multilayer perceptron to regress the coordinates of the point cloud. While both PCN and TopNet achieve a certain degree of point cloud completion, they struggle with accurately reconstructing object details.

In general, Point-based methods suffer from noisy inputs during training which limit the performance. Additionally, their inference speed is relatively slow and most methods lack fine-grained outputs. While it maintains the permutation invariance inherited from PointNet, this also limits the understanding of interdependence between the points. That leads to loss of local features in the output.

Convolution-based: Convolutional Neural Networks (CNNs) achieved great performance on 2D images. Based on these successes, 3D convolution-based methods were proposed. As convolutions are not directly applicable to point clouds, most methods depend on voxelization. This process transforms point clouds into binary voxels. Due to this process, geometry of details and fine-grained information may be lost. These voxelized inputs are often fed to a convolutional encoder that transforms the input into latent space, from which an output is generated by a deconvolutional decoder.

Xie et al. proposed a three-dimensional grid structure as an intermediate representation aimed at organizing unordered point clouds [3]. They developed the Grid Residual Network (GRNet), which features both gridding and de-gridding layers. These layers enable GRNet to efficiently convert point clouds into 3D grids while maintaining the intrinsic structural information. Additionally, a differentiable trilinear feature sampling layer designed to extract information from neighboring points was introduced. This aims to preserve contextual knowledge effectively.

In general, convolution-based methods face limitations due to the backpropagation algorithm's inefficiency in handling the large amounts of training data required. This issue is compounded by the substantial memory demands associated with high-resolution voxelizations. These methods also often suffer from translation invariance which may reduce performance when objects are translated to different positions. Lastly, pooling layers can cause a significant loss of valuable information by overlooking the relationships between the whole and its individual parts.

Transformer-based: Transformers initially gained interest in the field of natural language processing, after which it was successfully introduced in 2D computer vision. Later it was utilized in various point cloud processes [8]. Transformer-based networks usually perform well on irregular data and they incorporate representation learning and an attention mechanism. A downside of this architecture is the required computational resources as a result from the large amount of model parameters.

By regarding point cloud completion as a set-to-set translation, Yu et al. [1] introduced PoinTr based on a transformer encoder-decoder structure. The input point cloud is treated as a set of disordered points with position embeddings by converting the point cloud to a series of point proxies. Then, the transformer generates the output pointcloud. A geometry-aware block that simulates local geometric relations was added. This block improves the usage of the inductive bias of 3D geometric structures of the pointcloud.

Alternatively, Xiang et al. [5] applied the transformer-based structure to the decoder in SnowFlakeNet. The proposed Snowflake Point Deconvolution (SPD) layers generate complete point clouds by snowflake-like growth of points. Each SPD layer, child-points are generated by dividing parent-points. By introducing a skip-transformer in the SPD, point division modes are learned. The attention mechanism from the skip-transformer summarizes the splitting patterns from the previous SPD layer, to apply to the next SPD layer. This allows the SnowFlakeNet to produce highly detailed geometries.

2.3 Ensemble learning in Point Cloud Completion

Ensemble methods have previously been proposed in various point cloud processing methods. Classification of point clouds with ensemble methods was explored by Levi et al. [57]. They found that the proposed ensemble method significantly improved the robustness of top classification networks. In similar contexts, Zhang et al. proposed PointHop [58]. They concluded that their method offers state-of-the-art classification performance whilst requiring lower training complexity.

Semantic segmentation with ensemble methods was investigated by Atik et al. [59]. The proposed SegUNet3D is an ensemble approach inspired on both U-Net and SegNet methods. It was found that the SegUNet3D performed better than alternative methods. Similar results were observed by Chen et al. for 3D photogrammetry point clouds with a comparable method [60].

The relevance of ensemble methods for point cloud registration was successfully demonstrated by Yuan et al. [61] [62]. For point cloud completion, there is little to no literature present that display the potential of ensemble methods.

Tesema et al. observes that different PCC methods have their own pros and cons. They suggest that two methods may be used simultaneously to complement one another [35]. Adapting benefits from both methods and tuning them for specific circumstances is suggested to be important. Similarly, Zhang et al. [63] suggest a combination of different network types by proposing the use of composite neural networks for PCC. Composite Neural Networks are rooted directed acyclic graphs that combine multiple pre-trained neural network models. By leveraging the benefits of a variety of network types, PCC performance may be improved.

3 Methods

A variety of learning algorithms, learning strategies and ensembling criteria are described in this section. Any combination of the learning algorithms, learning strategies and ensemble criteria generates a valid ensemble. This allows a large number of ensembling possibilities. The ensemble configurations selected for experiments are described in the experimental setup section. An overview of the architecture of ensemble methods can be observed in Figure 3.1.

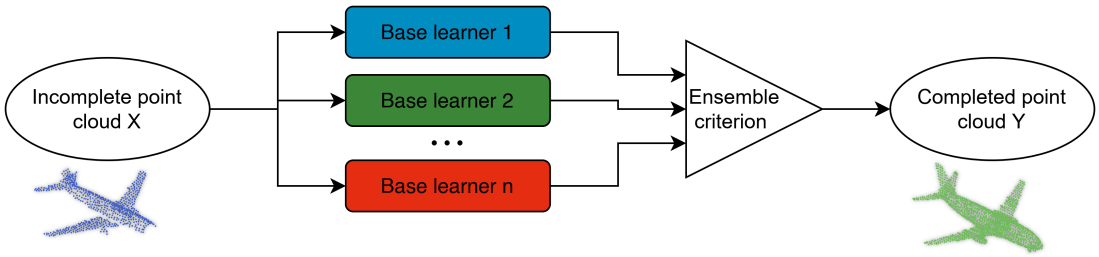

Figure 3.1: Ensemble methods are composed of a collection of base learners and a defined ensemble criterion. The base models are trained using a specific learning algorithm, which may also incorporate a learning strategy to differentiate homogeneous sets of base learners.

3.1 Problem statement

To formalize the Point Cloud Completion task, the following definition proposed by Yuan et al. is used [2]: Let X be a set of 3D points (x, y, z) which are found on the observed surface of an object. Let Y be a set of 3D points uniformly sampled from both the observed and unobserved surfaces from the object. Then, the point cloud completion task can be described as predicting Y given X. As pointed out by Yuan et al., this definition does not imply that X is necessarily a subset of Y. This loosens the constraint of points from set X and Y having to correspond. Note that in this thesis, points are only defined by their x, y and z coordinates and do therefore not contain any associated features such as color.

3.2 Learning algorithms

To train base learners, various deep PCC methods have been selected and implemented as operable learning algorithms (FoldingNet [4], PCN [2], TopNet [6], GRNet [3], SnowFlakeNet [5] and PoinTr [1]). To limit the scope of this thesis, no traditional methods are selected. The methods are presented in chronological order of publication.

3.2.1 FoldingNet: Point cloud auto-encoder via deep grid deformation

FoldingNet was proposed by Yang et al. [4]. It follows an auto-encoder structure consisting of a graph-based encoder and a folding-based decoder. The complete architecture can be observed in Figure 3.2.

The graph-based encoder is inspired on the work from Shen et al., which is a concatenation of MLPs and 16-nearest-neigbor graph-based max-pooling layers [64]. For every point in the input point cloud, the local covariance is computed based on one-hop neighbors of each point. Together with the point positions, this is fed to a 3-layer MLP as a per-point function. The output of the MLP

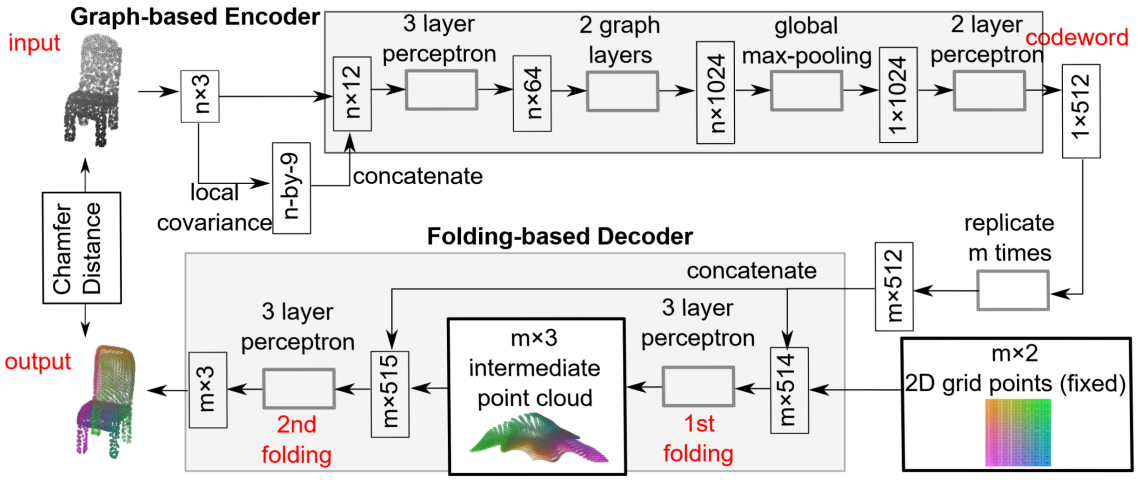

Figure 3.2: The FoldingNet architecture. Both the 1st and 2nd folding are achieved by concatenating the codeword with the feature vectors, which are then processed by a 3-layer perceptron. Each perceptron operates independently on the feature vector of an individual point, effectively applying to the rows of the m-by-k matrix. Adapted from [4].

is fed to two max-pooling graph layers which compute the local signature from the graph structure. This output is fed to the folding-based decoder. Two 3-layer MLPs are used to warp a fixed 2D grid towards the shape of the input. The output of the encoder is replicated 2025 times and concatenated with an 2025×2 matrix containing 2025 grid points on a square centered at the origin. Using a row-wise processing of a 3-layer MLP, concatenating that output with the encoders' output and feeding it into another 3-layered MLP, the reconstructed point cloud is obtained.

3.2.2 PCN: Point Completion Network

The Point Completion Network (PCN) was introduced by Yuan et al. [2]. The PCN is an encoder-decoder network. The input point cloud is fed to the encoder, which produces an 1024-dimensional feature vector. This is fed to the decoder, which transforms this feature vector to both a coarse and detailed completed point cloud. The complete architecture can be observed in Figure 3.3.

The used encoder is an expanded version of PointNet [55]. This ensures a permutation invariance and allows it to deal with variable number of input points. After a PointNet layer, the input point cloud is fed to a shared MLP with two linear layers and ReLU activation. This produces a set of point feature vectors. Over this set, point-wise maxpooling is performed to obtain a 1024-dimensional global feature. This global feature and the set of point feature vectors is fed to another PointNet layer producing an augmented point feature matrix. That is passed through a shared MLP and point-wise max pooling. This makes the complete encoder produce a feature vector of 1024 dimensions. The decoder processing this feature vector is a combination of a fully-connected decoder and a folding-based decoder [4]. The fully-connected decoder performs well at predicting a sparse point cloud, while the folding-based decoder performs well at predicting a representative surface for the local geometry.

3.2.3 TopNet: Structural point cloud decoder

TopNet was introduced by Tchapmi et al. [6]. It features an encoder-decoder architecture where the encoder is identical to the one proposed in the Point Completion Network [2]. Instead of using

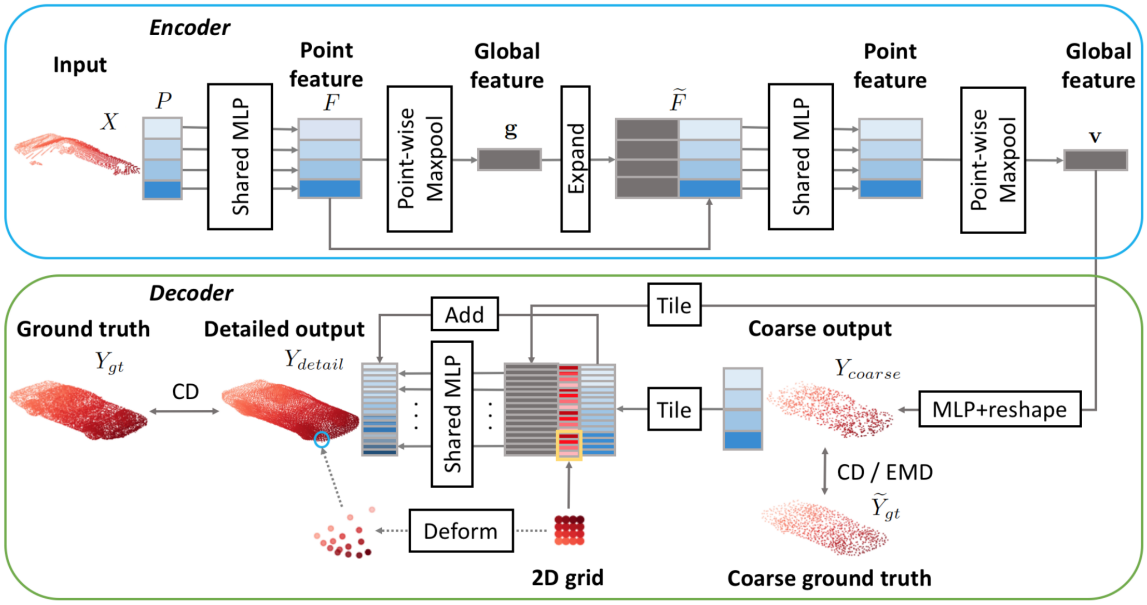

Figure 3.3: The PCN architecture. The encoder transforms the input point cloud X into a feature vector v. The decoder utilizes v to initially produce a coarse output Y_{coarse}, followed by a detailed output Y_{detail}. Each colored rectangle represents a matrix row, with the same color denoting identical content. Adapted from [2].

a combination of a fully-connected decoder and a folding-based decoder, TopNet utilizes a decoder from a collection of MLPs that are arranged in a tree structure. It learns the embedding of child nodes from the embedding of the parent node and global embedding from the encoder. The MLP at the root of the tree retrieves the global embedding from the encoder and generates a feature vector, which is passed to the child MLPs together with the global embedding. The final level of MLPs in the decoder generate feature vectors that represent the points for the output point cloud. MLPs that are on the same level in the tree share parameters. An overview of the model architecture can be observed in Figure 3.4.

3.2.4 GRNet: Gridding residual network for dense point cloud completion

The Gridding Residual Network (GRNet) was introduced by Xie et al. [3]. The network architecture includes a gridding, 3D convolutional, gridding reverse, cubic feature sampling and MLP layers. The complete architecture can be observed in Figure 3.5.

The gridding layer transforms the unordered and irregular input point cloud to a regular 3D grid. In standard voxelization, the value of a grid cell is set to one if there exists a point from the input point cloud within the bounds of that cell. A value of zero is set if no point exists in the bounds of a cell. This may overlook small details of the object. Instead, the gridding layer determines a non-binary cell value based on the neighborhood of the grid cell. The resulting 3D grid is fed to a 3D Convolutional Neural Network (3D CNN) with skip connections inspired by a 3D encoder-decoder with U-net connections. The idea is that this 3D CNN performs the completion of the object. This output is transformed by the gridding reverse layer to form a coarse point cloud. It does so by generating a single point coordinate for each grid cell by a weighted combination of the eight vertices coordinates and values from the cell. Using cubic feature sampling, point features are extracted from the feature maps from the first three transposed convolutional layers of the 3D CNN. 2048 points

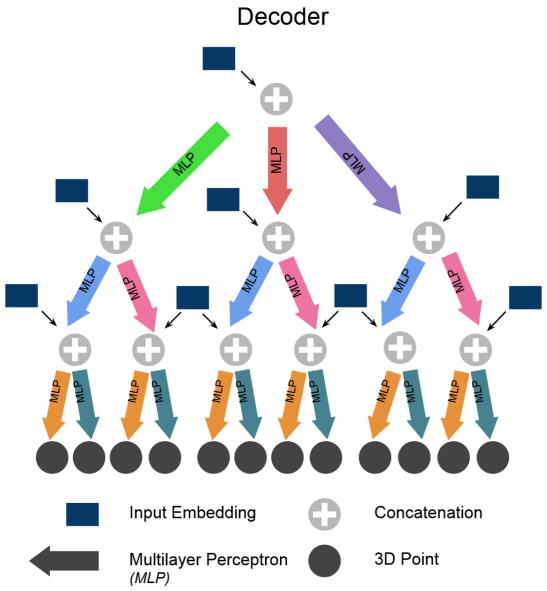

Figure 3.4: The TopNet architecture. The completion framework consists of a 2-stage point cloud encoder and a tree-structured decoder. The decoder's arrows are multilayer perceptron networks (MLPs), with MLPs of the same color sharing the same parameters. Adapted from [6].

are randomly sampled from the coarse point cloud, producing a feature map of size 2048×1792. Lastly, an MLP is utilized to retrieve details from the coarse point cloud by learning the residual offsets between the coordinates of points in the coarse and the final completed point cloud. After four fully connected layers, the output is reshaped to 16384×3 corresponding to a point cloud with 16384 points.

3.2.5 SnowFlakeNet: Point cloud completion by snowflake point deconvolution with skip-transformer

SnowFlakeNet was introduced by Xiang et al. [5]. It is composed of three different modules: Feature extraction, seed generation and point generation. The complete architecture can be observed in Figure 3.6.

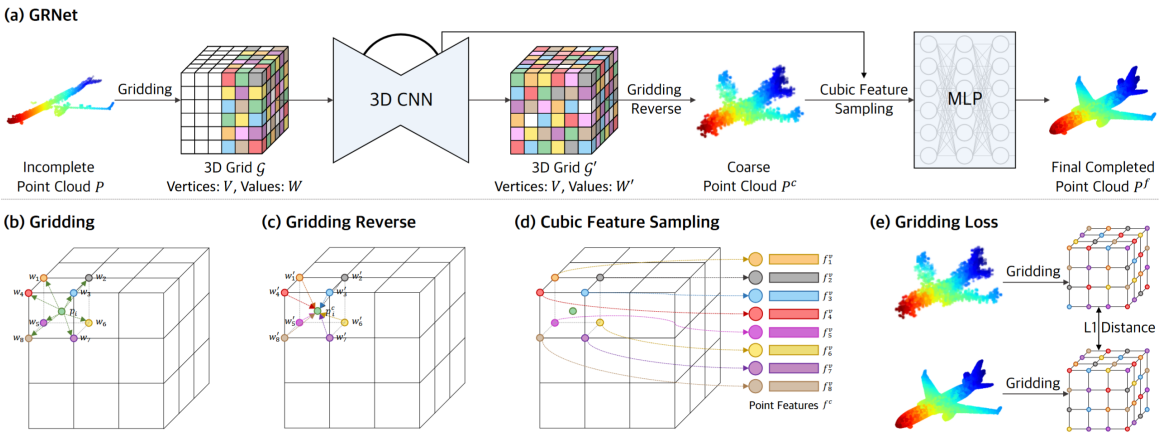

Figure 3.5: The GRNet architecture. It includes (a) GRNet, (b) Gridding, (c) Gridding Reverse, (d) Cubic Feature Sampling, and (e) Gridding Loss. Adapted from [3].

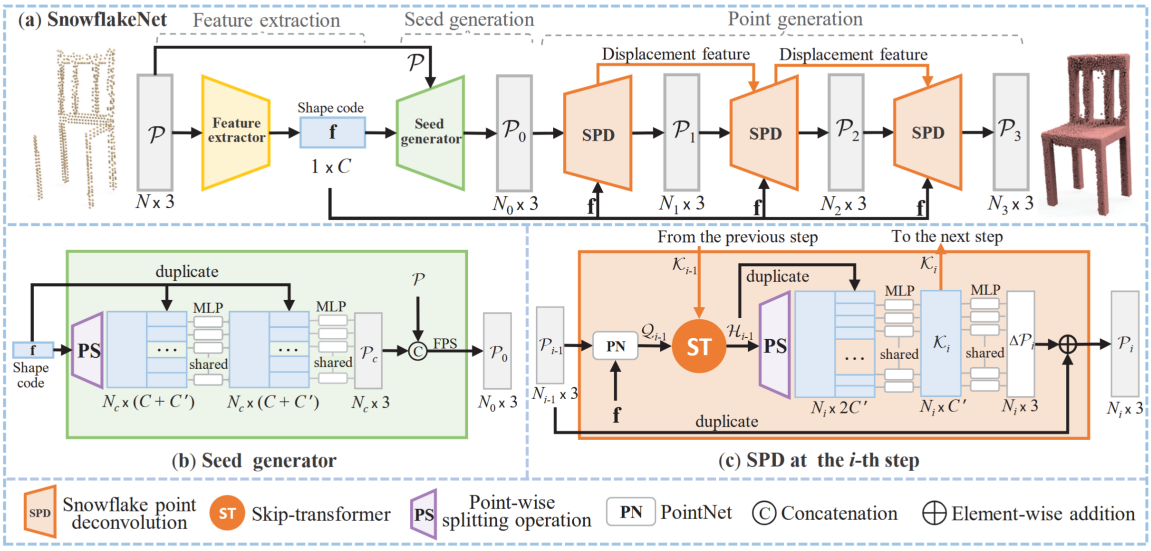

Figure 3.6: (a) The overall architecture of SnowflakeNet includes three modules: feature extraction, seed generation, and point generation. (b) Details of the seed generation module. (c) Snowflake Point Deconvolution (SPD). Here, N, N_c, and N_i represent the number of points, while C and C' denote the number of point feature channels, which are 512 and 128, respectively. Adapted from [5].

The feature extractor is designed to derive a shape code that obtains the global structure and detailed local pattern of the target object. It is based on three layers of set abstraction for aggregating point features from local to global. A point transformer is used for retrieving local shape context. Then, the seed generation module is used to produce a coarse complete point cloud that include the geometry and structure of the completed point cloud. The seed generation module starts with point-wise splitting to produce point features. These point features are integrated with the shape code through an MLP to form the coarse point cloud. Subsequently, the point generation module performs three iterations of so-called Snowflake Point Deconvolution (SPD). Each iteration, the point cloud from the former iteration is split by up-sampling factors. The SPDs work together to create a rooted tree structure that adheres to local patterns for each seed point. To enable consecutive SPDs to split points coherently, a novel skip transformer is used that captures the shape context and the spatial relationships between parent points and their split points.

3.2.6 PoinTr: Diverse Point Cloud Completion with Geometry-Aware Transformers

PoinTr was proposed by Yu et al. [1]. They reformulated PCC as a set-to-set translation problem and proposed to use a transformer encoder-decoder architecture. The input point cloud is converted to a set of feature vectors referred to as point proxies. These point proxies represent the local regions from the point clouds. The point proxies are fed to a transformer-based encoder-decoder architecture, which produces the point proxies for the missing regions. Figure 3.7 displays an architecture overview.

The encoder-decoder architecture contains multi-head self-attention layers in both the en- and decoder. The encoder's self-attention layer first updates the input point proxies with long- and short-range information. Then, the decoder uses both self-attention and cross-attention mechanisms for learning structural knowledge. The self-attention layer improves the local features with global features. The cross-attention layers discover the relationship between queries and the encoder's output. Dynamic query embeddings are used to produce the point proxies for the missing areas. This aims to establish a more flexible decoder for different types of objects.

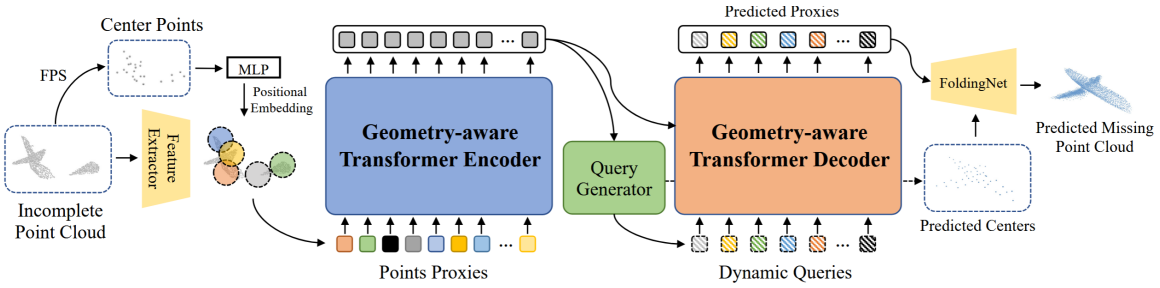

Figure 3.7: The PoinTr architecture. The PoinTr pipeline begins by downsampling the input partial point cloud to derive the center points. Next, a lightweight DGCNN [7] extracts local features around these center points. After incorporating position embeddings into the local features, we employ a transformer encoder-decoder architecture to predict point proxies for the missing regions. Subsequently, a simple MLP and FoldingNet are utilized to complete the point cloud in a coarse-to-fine manner based on the predicted point proxies. Adapted from [1].

3.3 Learning strategies

To differentiate base learners that employ the same learning algorithm, learning strategies are used. Two common learning strategies for (deep) ensembles are described and explored.

3.3.1 Bagging

Bagging, or bootstrap aggregating, is a learning strategy that differentiates learning algorithms by training each model on a random subset of the original training data.

For this thesis a similar approach to Breiman et al. [23] is utilized. Given a training dataset $\mathcal{L}$ containing data in the format $\{(y_n, x_n), n = 1, ...N\}$ where x and y respectively correspond to a partial input and a completed point cloud sample. By taking repeated bootstrap samples $\{\mathcal{L}^{(B)}\}$ from $\mathcal{L}$, replicate datasets are formed. Each dataset contains N samples that are randomly drawn with replacement from $\mathcal{L}$ that share the same underlying distribution as $\mathcal{L}$.

Given a PCC learning algorithm $\varphi(x, \mathcal{L})$ that predicts a reconstructed point cloud y for sample x, a sequence of unique PCC predictors $\{\varphi(x, \mathcal{L}^{(B)})\}$ can be constructed from the replicate datasets.

3.3.2 Random initialization

Random network initialization in ensemble learning involves initializing each model with different random weights or configurations, which enhances diversity among models.

Training a Neural Network (NN) entails optimizing a non-convex cost function that is defined over the network's parameters. During this process, network parameters are adjusted to minimize the cost function using backpropagation [65]. During backpropagation a training sample is fed in the NN, after which the loss between the output and the expected output is computed. Then, the network weights are updated using gradient descent over that loss. For efficient backpropagation, the initialization weights are important [66]. Initializing all weights to 0 would result in a weight symmetry that breaks the learning process. Setting the weights too large may result in the exploding gradients problem that also breaks learning. Therefore, initial network weights should be set to small random values.

When different random initialization weights are chosen, different base models are produced. Random initialization in the context of deep ensemble learning have previously been explored successfully by Lakshminarayanan et al. [31] and Lee et al. [67].

3.4 Ensembling criteria

To combine the output from various base learners, three ensembling criteria are defined. These are designed to summarize the collective knowledge of the different outputs into one ensemble output. An example of ensembling criteria for a regression problems is averaging. For classification problems this may be majority voting. In the context of point cloud completion, an ensembling criteria C is defined as a function $X_{combined} = C(X_1, X_2, ..., X_n)$, where $X_i = \{(x_j, y_j, z_j), ...\}$ for $j = 1, ..., m_i$ where m_i is the number of points in X_i.

3.4.1 Point Aggregation

The point aggregation ensembling criteria can be defined as follows:

$$C_{pagg}(X_1, X_2, ..., X_n) = \cup_{i=1}^{n} X_i$$

This ensembling criteria aggregates all of the points from the constituent point clouds to form the ensemble output. Due to the nature of this criteria, the amount of points in the output is a sum of the total amount of points in the input point clouds. Depending on the amount of base learners this amount may grow very large. If this is not desirable, there are alternative criteria to reduce the amount of points in the ensembled point cloud.

3.4.2 Uniform sampling

The uniform sampling ensembling criteria can be defined as follows:

$$C_{unis}(X_1, X_2, ..., X_n) = \mathcal{U}(\cup_{i=1}^{n} X_i, k)$$

Where $\mathcal{U}$ is the uniform sampling operator and k is the amount of points to sample. Sampling occurs without replacement. This ensembling criteria aggregates all of the points from the constituent point clouds and uniformly samples k amount of points. Therefore, this criterion does not take base learner performance into account. This may limit criterion effectiveness when a poor base learner is integrated.

3.4.3 Performance-based sampling

This ensembling criteria first aggregates all of the points from the constituent point clouds after which k amount of points are sampled. Instead of uniformly sampling, points from the constituent point clouds are weighted according to the base learners performance. Based on this weighting, k amount of points are sampled.

Given a set of n base models that produce a set of reconstructed point clouds X_1, X_2, ..., X_n. Each of these base models is associated with a mean chamfer distance over the training set: X_1, X_2, ..., X_n.

These mean distances are transformed into a selection probability p_i for each of the base model outputs using:

$$p_i = \sum_{j=1}^{n} \frac{X_j}{X_i}$$

This inverts the chamfer distance to increase selection probability for better performing base models. Then the weight is normalized to produce probabilities. The performance-based sampling criterion can then be defined as:

$$C_{pebas}(X_1, X_2, ..., X_n, p_1, p_2, ..., p_n, k) = \mathcal{W}(\cup_{i=1}^{n} X_i, k)$$

Where $\mathcal{W}$ is a weighted sampling operator that samples k points. The selection probability is used k times to select a constituent point cloud, from which a random point is sampled without replacement.

3.5 Performance Measure

3.5.1 Chamfer distance

To evaluate the performance of the ensembles, the Chamfer Distance (CD) [68] is used, as it is the most popular metric for completion tasks [35]. It features a relatively simple calculation complexity and a short computation time. Additionally, the number of points in both point clouds does not need to match. The CD represents the mean distance of nearest points between two point clouds. The chamfer distance is defined between two point clouds X_1 and X_2 as in Equation 1:

$$CD(X_1, X_2) = \frac{1}{|X_1|} \sum_{x \in X_1} \min_{y \in X_2} \|x - y\|_2^2 + \frac{1}{|X_2|} \sum_{y \in X_2} \min_{x \in X_1} \|y - x\|_2^2 \tag{1}$$

Where $\|...\|_2^2$ refers to the Euclidean distance and x and y respectively represents a point in pointcloud X_1 and X_2. For each point in X_1, the distance to the closest point in X_2 is computed, aggregated and averaged. The same procedure is performed from X_2 to X_1 and the result of both average closest distances is summed to form the chamfer distance.

4 Experimental Setup

In this section the base learner training, experimental configurations and used datasets are discussed. To investigate the dynamics of ensemble methods, various ensemble configurations are described.

4.1 Datasets

All of the experimental configurations from Section 4.3 are validated on two different datasets. The Completion3D dataset highlights the performance of ensemble methods on human-made objects. The agricultural dataset focuses on PCC performance on a single-category agricultural organic shapes.

4.1.1 Completion3D dataset

The Completion3D dataset contains 29774 partial and complete point clouds, from which 800 samples are reserved for validation. It contains samples from eight different human-made object-categories (Plane, Cabinet, Car, Chair, Lamp, Couch, Table, and Watercraft). An overview of the objects in the Completion3D dataset can be observed in Figure 4.1. Both the partial and complete point clouds contain 2048 points, which are sampled from CAD models from the ShapeNet dataset [69]. The partial 3D point clouds are constructed by back-projecting a random partial view of the object to 2048 points in the x-, y- and z-space. The ground truth is constructed by uniformly sampling 2048 points over the complete objects surface. The Completion3D dataset is selected to evaluate the performance of PCC ensemble methods on everyday objects. As opposed to other popular datasets (PCN dataset [2], ShapeNet-55 [69], ShapeNetSem [70]), the training samples from Completion3D contain both partial and complete shapes. This removes the need to partialize complete shapes.

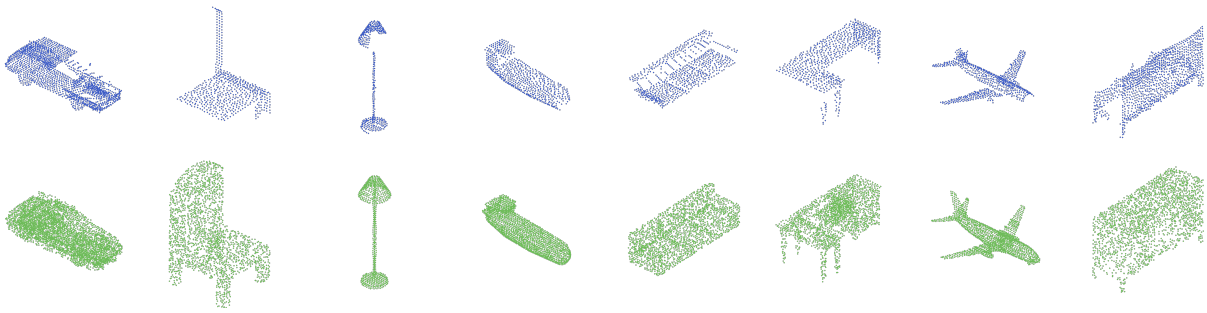

Figure 4.1: Samples from the Completion3D dataset. The partial point clouds are drawn in blue and the associated completed ground truth in green below. Samples from left to right are respectively from the car, chair, lamp, watercraft, couch, table, plane and cabinet category.

4.1.2 Agricultural dataset

To evaluate the performance of the ensemble methods on organic, non-human-made objects, an agricultural dataset is introduced. It contains 3500 partial and complete point clouds, from which 500 samples are reserved for validation. Both the partial and complete point clouds contain 2048 points. A visualization of the dataset samples can be observed in Figure 4.2.

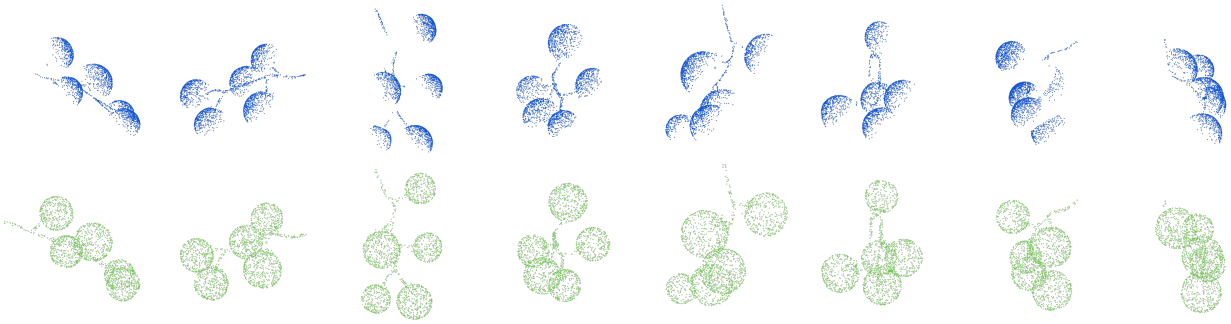

Figure 4.2: Samples from the introduced synthetic agricultural dataset. The partial point clouds are drawn in blue and the associated completed ground truth in green below.

The dataset is generated from a mesh template bunch of five fruits that is constructed with Computer-Aided Design (CAD) (see Figure 4.3). This virtual bunch is subjected to a physics simulation where the vine is modeled as a soft-body. The simulation exploits the soft-body flexibility of the peduncle, allowing the generation of a large range of bunch configurations. To further increase the diversity of the samples in the dataset, the fruits of the bunch are individually scaled with a random percentage of $\pm 5\%$. The completed point clouds are generated by uniformly sampling 2048 points over the surface of the sample mesh. The partial point cloud is generated by capturing a virtual depth-image from which a point cloud consisting of 2048 points is constructed. It has been decided to limit the dataset to 5-fruit trusses as this is most common for real crops.

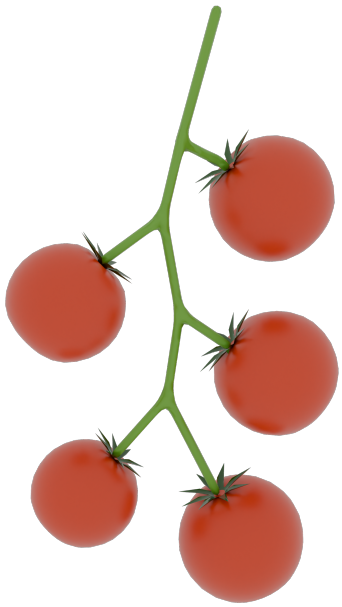

Figure 4.3: Template mesh for the agricultural dataset. The template peduncle is simulated with soft-body physics and the fruits are randomly scaled to generate dataset samples.

4.2 Base learner training

For each of the individual learning algorithm, a hyperparameter optimization has been performed using Optuna [71]. During this parameter optimization, various learning rates and weight decays were validated on the Completion3D dataset. A complete list of final hyperparameters per learning algorithm can be found in Appendix A.

All base models are trained on the GPU partition of the Habrok computer cluster from the University of Groningen. These nodes are equipped with Intel Xeon Platinum 8358 CPUs and Nvidia A100 GPU accelerator cards with 40 GB RAM. In total, 240 base models were trained in ± 1000 hours. The code associated with training and inferring the learning algorithms can be found on GitHub[1], which is heavily inspired on the work from Yu et al. [1]. Base model training was verified by checking the convergence of training loss for each model.

[1] https://github.com/Jakob4613/BB_MT_JAKOB

4.3 Experiments

The experiments aim to uncover the contrasts or find the similarities between various ensemble configurations and the individual point cloud completion methods. The ensemble configurations vary in the selection from six unique learning algorithms, two learning strategies and three ensembling criteria.

4.3.1 Individual methods

All six individual learning algorithms are validated to serve as a baseline with respect to the ensemble methods. They do not require a learning strategy or ensemble criteria, as the performance is measured directly over the model output with respect to the completed ground truth. This baseline assessment enables a comparison of how the ensemble methods improve upon the individual learner's performance by integrating their outputs. Furthermore, by establishing a solid baseline, we may identify the strengths and weaknesses of each algorithm, allowing us to fine-tune them for specific applications and better understand how different characteristics contribute to the overall ensemble effectiveness.

4.3.2 Homogeneous ensembles

Homogeneous ensembles are constructed by training different base models using a single learning algorithm [22]. To diversify the base learners, any of the introduced learning strategies are used. For each of the six learning algorithms, 10 base learners are trained using both the random initialization and the bagging learning strategy. This results in 120 base models that can be used to form homogeneous ensembles when they are combined with any of the three proposed ensembling criteria. This research is limited to 10 base learners per learning algorithm as limited time resources are available.

For each of the six learning algorithms, 3-, 6- and 10-base learner homogeneous ensembles are formed with every proposed learning strategy and ensembling criteria. Given the three sizes of ensembles and the six combinations of learning strategies and ensemble criteria, 18 ensembles are tested for each learning algorithm.

By comparing the result of varying ensemble sizes with the individual baselines, the effect of ensemble size on ensemble performance can be determined. It is expected that when ensembling has a positive effect on the performance, this effect further grows as the ensemble size increases past 10 base learners. Additionally, these experiments serve to determine differences between the bagging, and random initialization learning strategy.

4.3.3 Heterogeneous ensembles

Heterogeneous ensembles are based on different learning algorithms to generate distinct base learners [22]. Given the six learning algorithms that are implemented in this research, 57 unordered unique heterogeneous combinations of learning algorithms exist. Given the three ensembling criteria, this would result in 171 possible ensembles. Instead of validating all 171 ensembles, it is proposed to limit the study to 3- and 6-base learner heterogeneous ensembles. This reduces the amount of heterogeneous combinations to 21. For each of the proposed heterogeneous combination, every ensemble criterion is tested. This results in 63 heterogeneous ensemble experiments.

By evaluating the different heterogeneous combinations of base learners with respect to the individual base learner's performance, the effect of individual methods within an ensemble can be studied. This may also yield insight in potential complementary behavior of two or more base learners.

By combining the results from both homo- and heterogeneous results, the performance of the different ensemble criteria can be compared across ensemble configurations. Additionally, it can be observed if heterogeneous or homogeneous ensembles perform significantly different.

5 Results

This section presents the results produced by the proposed experiments on the Completion3D dataset. Results are presented both qualitatively and quantitatively and significant relations are demonstrated. The results for the individual base learners are summarized first, as these serve as a baseline to compare with different ensemble configurations. Subsequently, results for homogeneous and heterogeneous ensemble mechanisms are presented. The section concludes with results for the agricultural dataset. Figures presented in this section displaying point-clouds can also be observed at https://sites.google.com/student.rug.nl/s3803996-master-thesis, where a rotational motion is added to the pointclouds to improve structural understanding.

5.1 Individual base learners

Six learning algorithms have been trained 10-fold using both the bagging and random initialization learning strategy to form 120 base models. The resulting performance on the Completion3D validation set can be observed in Figure 5.1. SnowFlakeNet performs best and PCN performs the worst. It can also be observed that both learning algorithms result in chamfer distance variance between models sharing the learning algorithm. This indicates that both learning strategies successfully generate different base models. As a result of the bagging strategy, model performances seem more varied with respect to the random initialization strategy. The TopNet learning algorithm produces the most performance-varied set of base learners for both learning strategies. Models trained with the random initialization strategy outperform bagged models for every learning algorithm. The associated mean chamfer distance over 10 separately trained base learners can be found in Table 5.1.

Qualitative results for different randomly initialized base learners on eight samples from distinctive

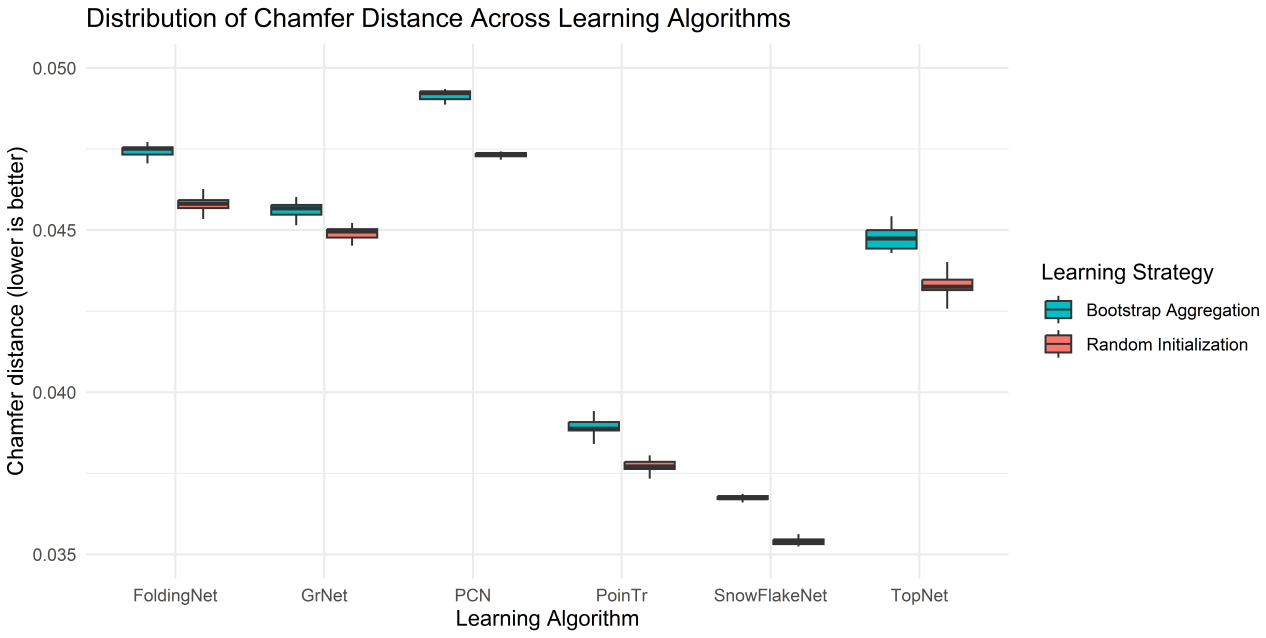

Figure 5.1: Boxplot displaying the distribution of chamfer distances across various learning algorithms combined with either learning strategy. Each algorithm's performance is represented by the mean chamfer distance, with error bars indicating variability. Results are collected on the Completion3D dataset.

	FoldingNet	GRNet	PCN	PoinTr	SnowFlakeNet	TopNet
Random Initialization	0.0458	0.0449	0.0473	0.0377	**0.0354**	0.0433
Bagging	0.0474	0.0456	0.0492	0.0389	**0.0367**	0.0447

Table 5.1: Performance comparison of different learning algorithms under two learning strategies: Random Initialization and Bagging, evaluated on the Completion3D dataset. The values represent average chamfer distance over 10 separately trained base learners. SnowFlakeNet outperforms all other learning algorithms under both learning strategies.

categories from the Completion3D dataset are displayed in Figure 5.2.

FoldingNet produces a completed point cloud by folding a 2D grid. This grid-like folding can be recognized by the consistent grid-like patterns of points that form the completed shape. This strategy limits FoldingNet capability to produce gaps between the watercraft's masts and rigging. A similar shortcoming can be observed for the lamp, where the highly detailed arm structures are too complex with respect to the folding flexibility. This results in completed point clouds that roughly envelope object shapes.

As opposed to FoldingNet, GRNet is able to produce gaps between the watercrafts mast and rigging. Furthermore, It can be noted that none of the predictions include points that are aligned in a single line. This limits the GRNet to produce correct predictions for the watercrafts rigging or the cable suspending the lamp. This may be an artefact generated by the de-gridding layer, where intermediate voxelized outputs are de-gridded into neighboring points.

PCN features the same encoder architecture as TopNet. This may explain why both methods do not produce the required gap between the mast and the rigging of the watercraft. This may indicate that the required gap is not encoded in the latent space produced by the encoder. While both methods feature the same encoder, they incorporate a different decoder. As the pointclouds produced by PCN and TopNet appear differently, the difference in decoder shows to have an effect on the base learners output. PCN's predictions are characterized by small clusters of points that are aligned in small planes. The collection of these small pointcloud planes then form the completed object. Alternatively, Points in the prediction from TopNet seem more evenly spaced out.

Both PoinTr and SnowFlakeNet perform rather similar. Both methods produce similar shapes for each of the objects. This is most obvious for the car object, where both methods mispredict the back end of the pickup truck with a similar diagonal shape. Both methods also produce part of the detailed structure of the lamp arm structures as opposed to the other methods.

The partial point cloud for both the chair and the table contain a large missing area. While the ground truth for the chair shows that the chair stands on two interconnected posts; most of the base learners produce an emphasized four-legged structure, while also producing some noise in the surrounding area. The tables produced by the different base learners look relatively similar, although none of the predictions match the ground truth shape exactly. Both of these observations may partly be a result of data bias. For example from the fact that only four-legged chairs are present in the dataset.

Based on the qualitative results of base model predictions, it can be observed that the set of base models exhibit prediction diversity. This base learner diversity is an important condition for heterogeneous ensemble methods to function. For homogeneous ensembles, this diversity is a result of the different learning strategies. This base model diversity is barely observable in qualitative results, but was observed quantitatively in Figure 5.1.

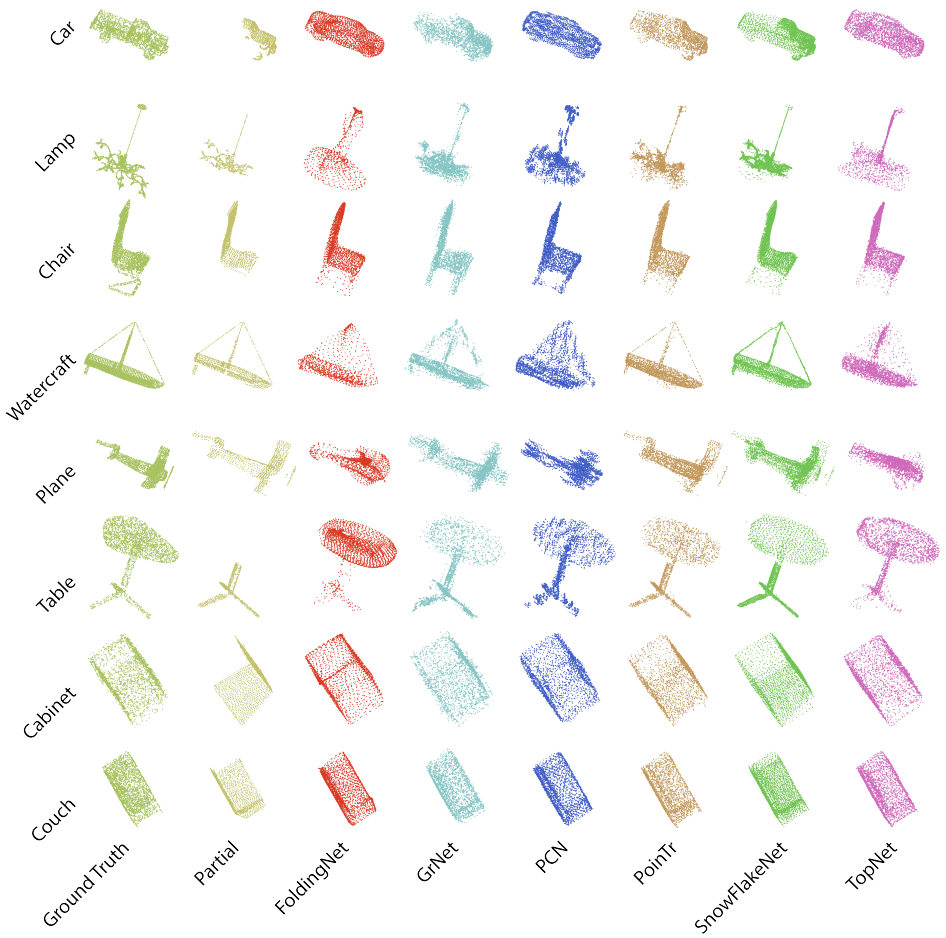

Figure 5.2: Qualitative results from different randomly initialized learning algorithms for eight validation samples from different categories from the Completion3D dataset. Different base learners produce different predictions. Qualitative results from bagged base learners look very similar.

5.2 Homogeneous ensembles

5.2.1 Qualitative analysis

Qualitative results for 3- 6- and 10-base learner homogeneous ensembles with the aggregation ensemble criterion can be found in Figure 5.3, where results for a chair sample are presented. The partial point cloud shows a very limited part of the chair, where two legs, an armrest and the backrest are completely omitted.

Each individual base learner (top row) seems to sufficiently produce the structure of a chair comparable to the ground truth. Only GRNet does not predict a second armrest and PCN generates a slightly warped chair leg. As the aggregation criterion is used, increasing the amount of base learners multiplies the amount of points in the point cloud. This makes the pointclouds increasingly more dense.

For most ensembles the increase in base learners does not result in significant structural changes. However, for the GRNet ensemble the increase of base learners enables the ensemble to produce the missing second armrest. Additionally, the warped chair leg predicted by the single PCN base learner straightens out when more base learners are added.

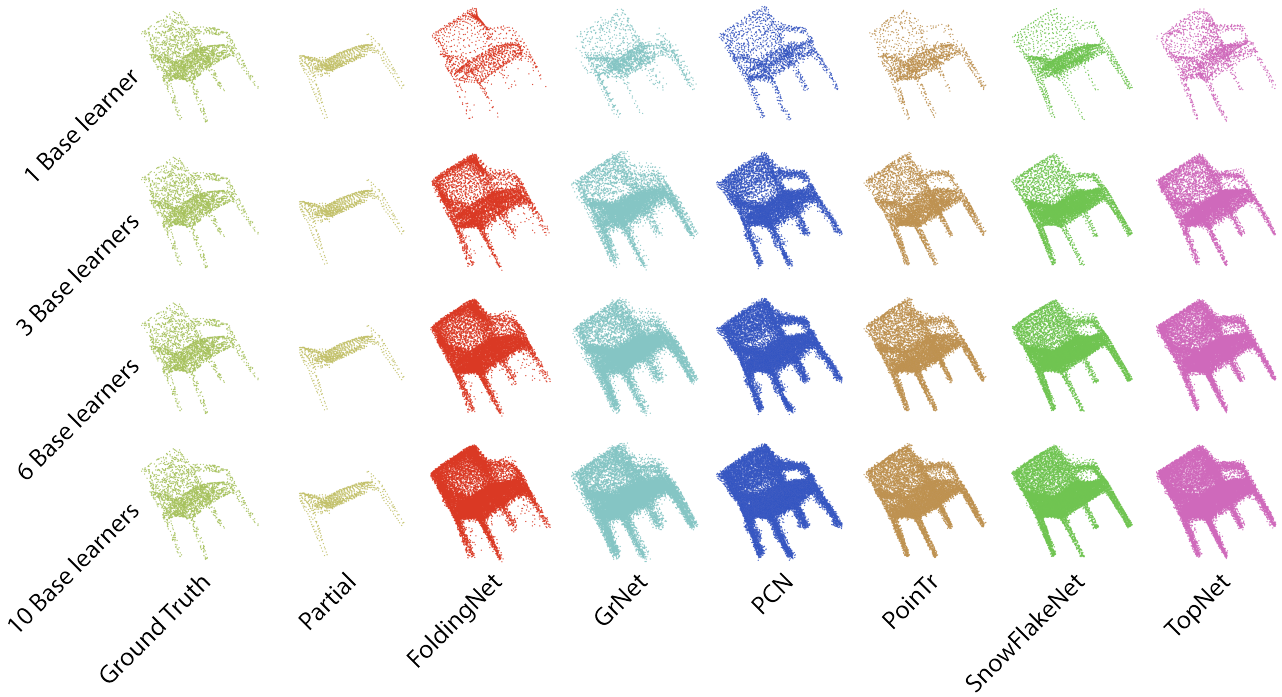

Figure 5.3: Homogeneous ensemble outputs for each learning algorithm using the random initialization strategy and the aggregation criterion. While making the point cloud more dense, increasing the amount of base learners does not produce significant structural differences. Similar qualitative results are found for ensembles using the bagging learning strategy.

5.2.2 Quantitative analysis

The effect of ensemble size for different homogeneous ensemble configurations can be observed in Figure 5.4. It is found that using the aggregation ensemble criterion, increasing the amount of base learners reduces the chamfer distance. This increase seems to decrease as more base learners are added to the ensemble. This trend of improvement with respect to the individual base learner performance seem comparable for each combination of learning algorithms and learning strategies. The biggest improvement found is a chamfer distance reduction of 41.0% for GRNet with the random initialization learning strategy.

For both the sampling-based sampling criteria, the relative improvement for randomly initialized base learners with respect to bagged base learners is also found. However, no significant improvements in performance are found when more base learners are added to any homogeneous ensemble using any of the sampling based criteria.

Additionally, no difference is found between both sampling criteria for homogeneous ensembles. This behavior is expected, as training losses of the individual base learners are not very diverse. For the performance-based sampling criterion, this means that the training losses are very similar. Which makes the sampling probabilities of the individual base learner predictions also very similar. This makes the performance-based sampling criterion behave as a near-uniform sampling criterion. This theory is reinforced by a one-way ANOVA.

To compare different ensemble criteria, a one-way ANOVA is performed. This reveals a significant difference in chamfer distances across ensemble criteria ($F(2,663) = 407.1, p < 2e-16$). Post-hoc comparisons indicate that both uniform sampling and performance-based sampling yield significantly worse performance than aggregation ($p < 0.001$), while no significant difference was found between

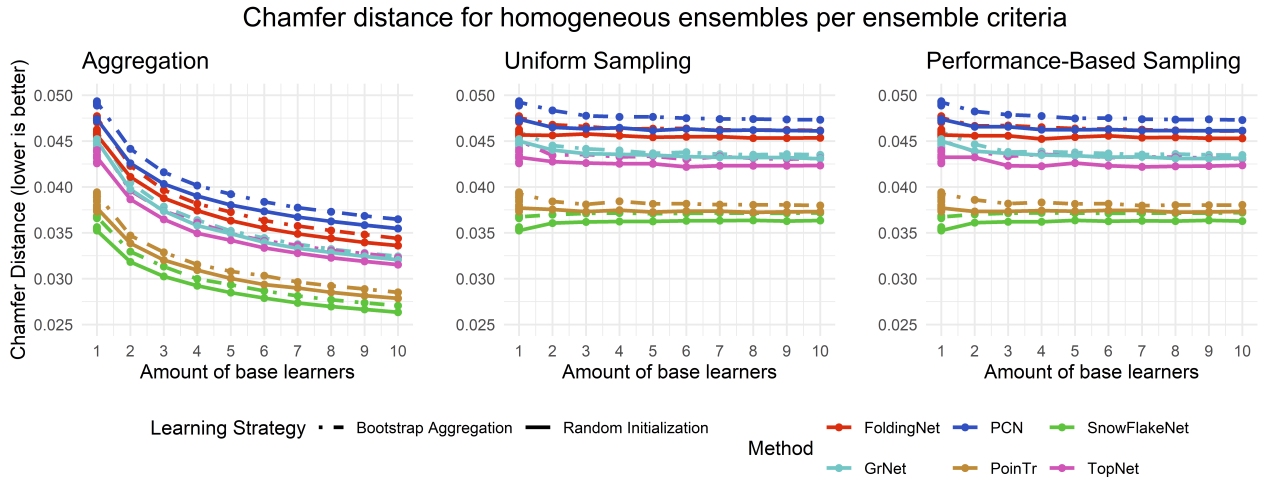

Figure 5.4: Comparison of different ensemble criteria over ensemble size for homogeneous ensembles. The aggregation criterion performs significantly better than both sampling based criteria. Results collected for various homogeneous ensembles on the Completion3D validation set.

performance-based sampling and uniform sampling ($p = 0.423$). This indicates that both sampling methods differ statistically significant from the aggregation ensemble criteria, while no difference is found between both sampling-based criteria.

Lastly, the random initialization learning strategy seems to outperform bagging-based ensemble configurations for each of the ensemble criteria and learning algorithms. A paired T-test revealed a significant difference in chamfer distances between random initialization and bagging strategies over all homogeneous ensembles, with a t-value of -37.561, degrees of freedom df$= 161$, and a p-value of $< 2.2e-16$, indicating that the random initialization strategy performs significantly better than bagging.

5.3 Heterogeneous ensembles

5.3.1 Qualitative analysis

Pointclouds predicted by various heterogeneous ensemble configurations are displayed in Figure 5.5. As the range of heterogeneous ensemble configurations and dataset samples is too large to effectively visualize, a sample from each dataset category is displayed for five heterogeneous ensemble configurations. These five configurations involve increasingly more methods to display the effect of ensemble size for heterogeneous methods. The order of including base learners was not specifically selected and can be considered random.

It can be observed that the combination of different learning algorithms may emphasize certain structures from objects, while other structures start to become more fuzzy. The back rest and seat of the chair become more accentuated, while the chair legs structure become softer.

For the plane category sample, an interesting combination of structures is observed. There, it seems that the GRNet individually introduces points that represent the propellor that other methods fail to produce.

For both the cabinet and couch, each base learner contributes a very similar shape. For these cases, ensembling does not necessarily deliver structural improvements, but it does make the point cloud more dense.

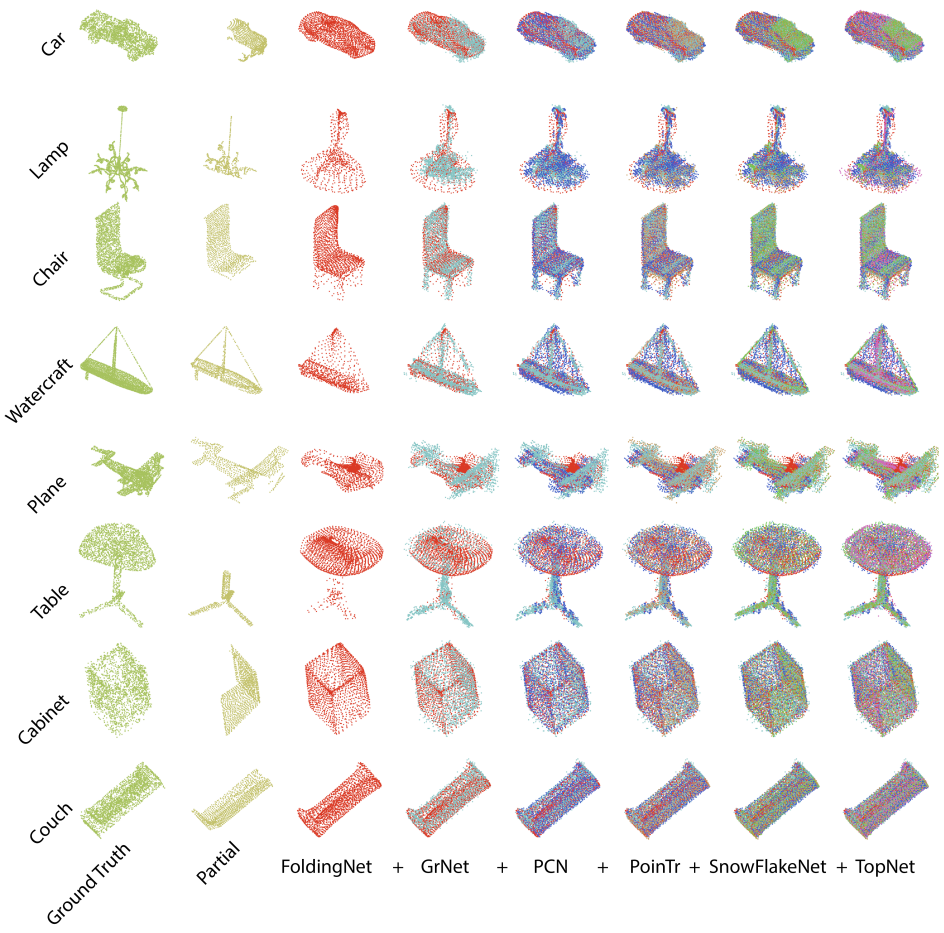

Figure 5.5: Qualitative results for five heterogeneous ensemble configurations. Column by column, the ensemble is extended with the respectively labeled base learner. Increasing the amount of learning algorithms result in denser point clouds and combined structures seem to emerge. Created with randomly initialized base learners using the aggregation ensemble criterion on the Completion3D validation set.

5.3.2 Quantitative analysis

A linear regression analysis was performed to investigate the impact of base learner inclusion on ensemble performance in comparison with the mean heterogeneous ensemble performance. The model incorporates a main effect for the inclusion of each learning algorithm. Additionally, all pairwise interaction terms are integrated to expose complementing base learners. Results reveal that all main effects improve performance, with PoinTr (Estimate$= -0.0115, p < 2e-16$) and SnowFlakeNet (Estimate$= -0.0136, p < 2e-16$) demonstrating the most substantial improvements. This is in line with the results from individual base learners. Interaction terms also show significant effects. The interaction between PoinTr and SnowFlakeNet (Estimate$= 0.0055, p < 2e-16$) reduces performance, suggesting that these two learning algorithms may impede each other . Other significant impeding interactions include GRNet with PoinTr (Estimate$= 0.0024, p = 4.18e-08$) and GRNet with SnowFlakeNet (Estimate$= 0.0031, p = 8.23e-12$). These are only significant effects with respect to the mean heterogeneous ensemble performance instead of individual baselines. Therefore, it is important to note that for the aggregation employed heterogeneous ensembles, 80.7% of the en-

sembles outperform the best individual base learner (SnowFlakeNet). None of the sampling-based criteria outperform this baseline.

Opposing the observations on ensemble criteria for homogeneous ensembles, it is expected that the sampling-based criteria will perform significantly different, as a result of different base model training performances. An ANOVA revealed this significant effect of ensemble criterion ($F(1.07, 60.07) = 1027.38, p < 0.001$). Post-hoc pairwise t-tests were subsequently conducted with Bonferroni adjustment, showing that aggregation outperformed both uniform sampling and performance-based sampling ($p < 0.001$ for both comparisons), while uniform sampling also significantly outperformed performance-based sampling ($p < 0.001$).

5.4 Agricultural dataset

The agricultural dataset differs by design in various aspects in comparison with the Completion3D dataset. Unlike eight categories of human-made objects, the agricultural dataset only contains objects from a single category. Additionally, the partializing process of point clouds is performed more consistently for the agricultural samples as a side-view perspective is assumed for every sample. These dataset differences deliver different, but comparable results.

Qualitative results for each learning algorithm are found in Figure 5.6. It can be observed that learning algorithms show very similar prediction effects in comparison to results on Completion3D. The FoldingNet shows a folded grid that seems too rigid to follow the complex stem structures. The GRNet shows a similar fuzzyness around the fruit surfaces. PCN produces comparable planes of points that substitute the completed prediction. PoinTr and SnowFlakeNet reproduce the ground truth relatively well. This is also reflected in the quantitative results.

Quantitative results for the individual base models can be observed in figure 5.7. It can be found that all individual base models perform worse with respect to the exact chamfer distances as found in figure 5.1 from the Completion3D dataset. A small fraction of this difference may be a result from different point cloud normalization techniques. Agricultural samples are normalized in a unit-sphere, while Completion3D samples are seemingly normalized to a smaller scaled cube. This could systematically result in larger chamfer distances for the agricultural dataset.

The rank of learning algorithm performances is also different between the two datasets. For the Completion3D dataset the TopNet outperforms the GRNet. For the agricultural dataset instead, the GRNet outperforms the TopNet. The remaining methods rank identical with respect to the Completion3D dataset.

Homogeneous ensembles show the same performance mechanisms with respect to learning strategies and ensembling criteria. Random initialization ensembles perform better than the bagged al-

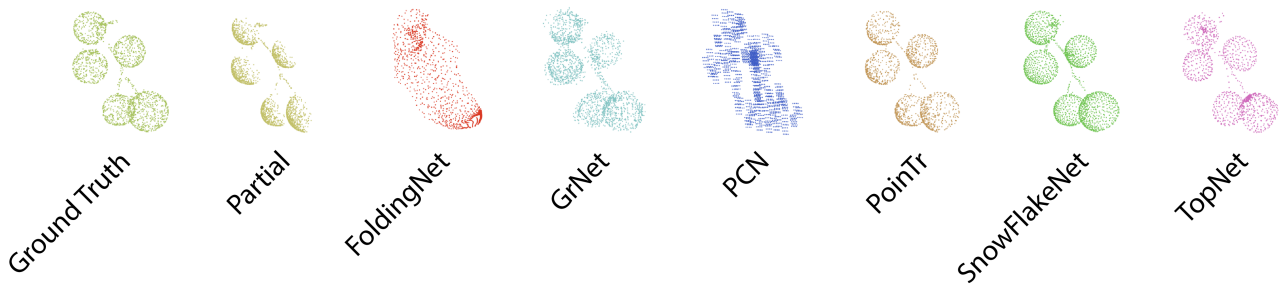

Figure 5.6: Qualitative results from different randomly initialized learning algorithms for a sample from the agricultural dataset. Different base learners produce different predictions.

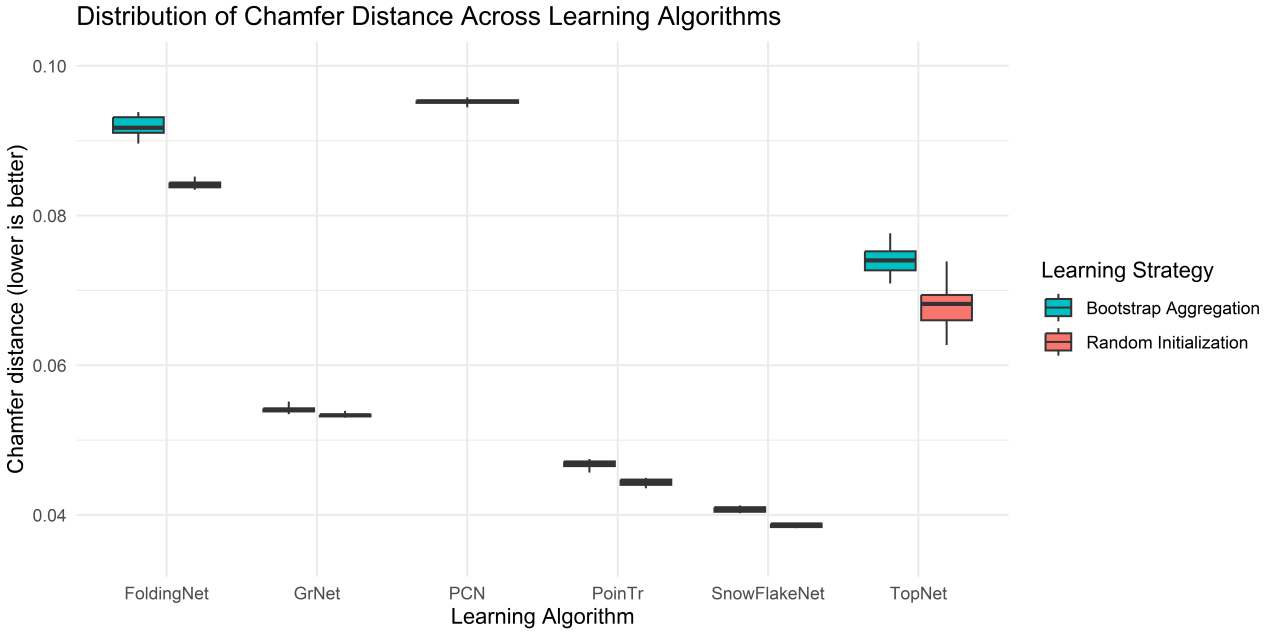

Figure 5.7: Boxplot displaying the distribution of chamfer distances across various learning algorithms trained with either Bagging or Random Initialization strategy. Results collected on the agricultural dataset. As opposed to the performance on the Completion3D dataset, GRNet performs better than PCN and TopNet.

ternative and only the aggregation criterion improves ensemble performance beyond individual base learners. Ensembles do not benefit from the sampling-based ensembling criteria for the agricultural dataset.

When comparing the relative improvements for homogeneous ensembles between the two datasets, it can be observed that the mean total decrease of chamfer distance over all learning algorithms is slightly higher for the agricultural dataset (42.4%) with respect to the Completion3D dataset (37.3%). Indicating that homogeneous ensembles yield better improvements on the agricultural dataset.

Qualitative results are very comparable to results found on the Completion3D dataset as observed in Figure 5.3. Ensembles produce denser clouds, but no structural improvements are found for homogeneous ensembles.

Qualitative results of heterogeneous ensembles on the agricultural dataset are displayed in Figure 5.8. As the range of heterogeneous ensemble configurations is too large to effectively visualize, a single validation sample is displayed for five heterogeneous ensemble configurations. These five configurations involve increasingly more methods to display the effect of ensemble size for heterogeneous methods.

In this figure, a complex sample is visualized, as the partial point cloud has only captured four of the five present fruits. As the ensemble increases in size, the structure of the invisible fruit slowly starts to emerge. Additionally, the point clouds become more dense, as the aggregation criterion is used.

As opposed to the performance increase on the Completion3D dataset of 80.7%, only 7.0% of heterogeneous ensembles improves on the best baseline (SnowFlakeNet). This may be an effect of the smaller variance between dataset samples as there is only a single category of objects. This may render the structural diversity between member method predictions redundant. Similarly to the Completion3D dataset, none of the sampling-based criteria ensembles outperformed this baseline.

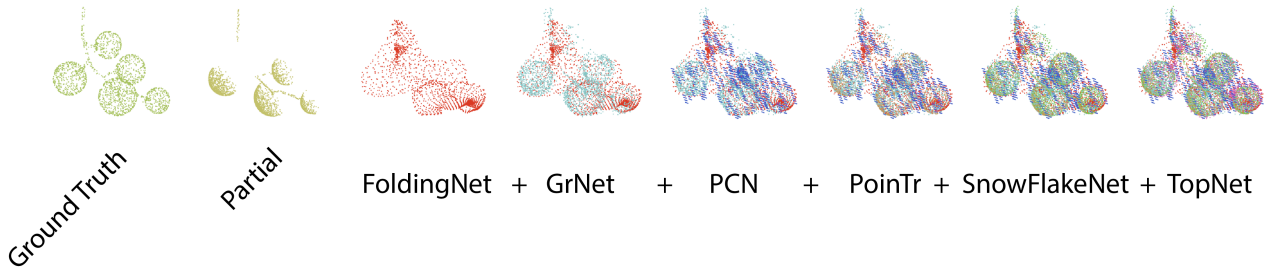

Figure 5.8: Qualitative results for five heterogeneous ensemble configurations. Column by column, the ensemble is extended with the respectively labeled base learner. While the fifth (top) fruit is not visible in the partial point cloud, The largest ensemble seem to predict the correct position. Created with randomly initialized base learners using the aggregation ensemble criterion on the agricultural validation set.

6 Conclusion & Discussion

This thesis aimed to explore the possibility, challenges and (dis-)advantages of deep ensemble methods in Point Cloud Completion. In order to define a conclusion on this topic, the five previously posed research questions are discussed.

In short, this thesis makes three primary contributions. First, deep-learning–based ensemble methods are introduced for point cloud completion, employing three novel ensemble criteria. Second, it is shown that the best-performing homogeneous ensemble surpasses the best-performing heterogeneous model, which in turn outperforms the top individual baseline. Third, a novel agricultural PCC dataset is generated and used to validate both individual and ensemble PCC methods. Together, these contributions advance deep ensemble methods for point cloud completion in the agricultural domain.

6.1 Research questions

How can the output of different PCC algorithms be combined to form a single more complete point-cloud?
Three different ensemble criteria have been presented and explored. The aggregation criterion merges all candidate point clouds and successfully reduces the chamfer distance with respect to individual base learners for some ensemble configurations. Both the uniform sampling and performance-based sampling fail to reduce the chamfer distance for most ensemble configurations.
What are the (dis-)advantages of combining multiple Point Cloud Completion methods?
The best performing homogeneous ensemble outperformed the best performing heterogeneous model, which in turn outperformed the best individual baseline. Thus, combining different point cloud completion methods in an ensemble proved to improve performance. Nevertheless, training a broad range of different base learners takes significantly more time- and computational resources. For homogeneous methods this requires implementing only a single learning algorithm and training that algorithm multiple times. However, for heterogeneous methods, multiple learning algorithms should be implemented and trained. In practice this may be time-consuming. Homogeneous methods may therefore be more suited in contexts where computational resources are limited. But depending on the context the required additional resources for ensembles may not justify the increase in performance with respect to individual base learners.
What is the relation between ensemble size and PCC performance for homogeneous ensembles?
Homogeneous ensembles incorporating the aggregation criterion show a significant improvement when more base learners are added. This improvement decreases as more base learners are added. No significant improvement is found for the alternative ensemble criteria.
How does the base learner selection influence heterogeneous ensemble dynamics?
It was observed qualitatively that different learning algorithms display individual characteristics. It was hoped that experimenting with different combinations of base learners would expose complementary behavior. However, this was not specifically found. Heterogeneous ensembles improved on the best individual baseline for the Completion3D dataset, although this effect diminished for the agricultural dataset. This indicates that heterogeneous ensemble methods with a large variety of base learners may generalize better on diverse datasets.
How do different learning strategies impact the results for homogeneous ensembles?
It was found that the random initialization learning strategy outperformed bagging for every homogeneous ensemble configurations. These results correspond with earlier results from both Lakshminarayanan et al. [31] and Lee et al. [67] where it was also concluded that training individual networks with random initialization is sufficient and that bootstrapping data can hurt ensemble performance.

Figure 6.1: To evaluate ensemble methods for point cloud completion in the agricultural domain, real-life experiments should be conducted such as depicted. Illustration of possible real-life experimental setting for evaluating PCC ensembles. Image adapted from https://www.ai.rug.nl/irl-lab/.

6.2 Discussion

Various limitations should be noted in relation to these conclusions. Although the chamfer distance is a universally acknowledged performance metric in PCC, it features various shortcomings identified by numerous authors. Tatarchenko et al. showed that the chamfer distance is very sensitive to outliers [72] and Wu et al. states that the Chamfer Distance may not effectively account for variations in local density distribution [73]. Fei et al. [8] observe that some paired point clouds encounter the issue of differing scales. For this reason, the comparison between Completion3D and the agricultural dataset is limited. Secondly, the selection of learning algorithms is relatively narrow. Although six different deep PCC methods are selected, there may be various different learning algorithms that would significantly alter ensemble behavior. Especially when they are combined with more tailor-made ensemble criteria that take the individual learning algorithm's biases into account. Lastly, the amount, diversity and complexity of the ensemble criteria is limited. The implemented ensemble criteria are relatively naive and only serve as a starting point for more elaborate criteria.

6.3 Future Work

It can be concluded that the ensemble criterion is a key influence in determining the success of ensemble configurations. Therefore, it is suggested to further develop novel ensemble criteria. Directions may include region-bound majority voting, where individual base learner prediction points may locally be discarded if no other ensemble members predict points in a similar structure.

One could also investigate ensemble criteria that fuse point clouds such that no output point is part of the ensemble member predicted points. Points from multiple point clouds may be averaged together such that differently sized structures are combined.

Lastly, depending on the size of the in- and output point clouds of base learners, one may introduce multiple stages of completion. This may be done by iteratively performing different base learners on the initial partial point cloud. When the sequence of architectures is trained end-to-end, this may resemble a composite neural network as suggested by Zhang et al. [63].

Aside from the possible theoretical improvements, it should be noted that the results presented in this thesis are based on synthetic data. In order to fully evaluate PCC ensembles, experiments should also be conducted in real life. Figure 6.1 illustrates such an experiment, where an RGB-D camera captures real fruits in an agricultural setting. Training real-life ensembles may require more accurate synthetic datasets that may better reflect the context in which PCC is applied. Alternatively this may reduce performance as observed by Li et al. [74].

Bibliography

[1] X. Yu, Y. Rao, Z. Wang, Z. Liu, J. Lu, and J. Zhou, "Pointr: Diverse point cloud completion with geometry-aware transformers," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 12498–12507, 2021.

[2] W. Yuan, T. Khot, D. Held, C. Mertz, and M. Hebert, "Pcn: Point completion network," in *2018 international conference on 3D vision (3DV)*, pp. 728–737, IEEE, 2018.

[3] H. Xie, H. Yao, S. Zhou, J. Mao, S. Zhang, and W. Sun, "Grnet: Gridding residual network for dense point cloud completion," in *European conference on computer vision*, pp. 365–381, Springer, 2020.

[4] Y. Yang, C. Feng, Y. Shen, and D. Tian, "Foldingnet: Point cloud auto-encoder via deep grid deformation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 206–215, 2018.

[5] P. Xiang, X. Wen, Y.-S. Liu, Y.-P. Cao, P. Wan, W. Zheng, and Z. Han, "Snowflakenet: Point cloud completion by snowflake point deconvolution with skip-transformer," in *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 5499–5509, 2021.

[6] L. P. Tchapmi, V. Kosaraju, H. Rezatofighi, I. Reid, and S. Savarese, "Topnet: Structural point cloud decoder," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 383–392, 2019.

[7] Y. Wang, Y. Sun, Z. Liu, S. E. Sarma, M. M. Bronstein, and J. M. Solomon, "Dynamic graph cnn for learning on point clouds," *ACM Transactions on Graphics (tog)*, vol. 38, no. 5, pp. 1–12, 2019.

[8] B. Fei, W. Yang, W.-M. Chen, Z. Li, Y. Li, T. Ma, X. Hu, and L. Ma, "Comprehensive review of deep learning-based 3d point cloud completion processing and analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 12, pp. 22862–22883, 2022.

[9] R. Schnabel, P. Degener, and R. Klein, "Completion and reconstruction with primitive shapes," in *Computer graphics forum*, vol. 28, pp. 503–512, Wiley Online Library, 2009.

[10] H. Yan, Z. Li, K. Luo, L. Lu, and P. Tan, "Symmcompletion: High-fidelity and high-consistency point cloud completion with symmetry guidance," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 39, pp. 9094–9102, 2025.

[11] X. Wu, X. Wu, T. Luan, Y. Bai, Z. Lai, and J. Yuan, "Fsc: Few-point shape completion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26077–26087, 2024.

[12] Y. Chang and K. Wang, "Multi-stage refinement network for point cloud completion based on geodesic attention," *Scientific Reports*, vol. 15, no. 1, p. 3570, 2025.

[13] Y. Kasten, O. Rahamim, and G. Chechik, "Point cloud completion with pretrained text-to-image diffusion models," *Advances in Neural Information Processing Systems*, vol. 36, pp. 12171–12191, 2023.

[14] L. Breiman, "Random forests," *Machine learning*, vol. 45, pp. 5–32, 2001.

[15] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of computer and system sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[16] M. Abdelazeem, A. Elamin, A. Afifi, and A. El-Rabbany, "Multi-sensor point cloud data fusion for precise 3d mapping," *The Egyptian Journal of Remote Sensing and Space Science*, vol. 24, no. 3, pp. 835–844, 2021.

[17] Y. Cui, R. Chen, W. Chu, L. Chen, D. Tian, Y. Li, and D. Cao, "Deep learning for image and point cloud fusion in autonomous driving: A review," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 722–739, 2021.

[18] X. Dong, Z. Yu, W. Cao, Y. Shi, and Q. Ma, "A survey on ensemble learning," *Frontiers of Computer Science*, vol. 14, no. 2, pp. 241–258, 2020.

[19] B. Dasarathy and B. Sheela, "A composite classifier system design: Concepts and methodology," *Proceedings of the IEEE*, vol. 67, no. 5, pp. 708–713, 1979.

[20] M. Kearns, "Learning boolean formulae or finite automata is as hard as factoring," Tech. Rep. TR-14-88, Harvard University, Aiken Computation Laboratory, 1988.

[21] R. E. Schapire, "The strength of weak learnability," *Machine learning*, vol. 5, no. 2, pp. 197–227, 1990.

[22] Y. Yang, H. Lv, and N. Chen, "A survey on ensemble learning under the era of deep learning," *Artificial Intelligence Review*, vol. 56, no. 6, pp. 5545–5589, 2023.

[23] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[24] T. W. K., "Stacking bagged and dagged models," *Proceedings of ICML'97*, pp. 367–375, 1997.

[25] E. Bauer and R. Kohavi, "An empirical comparison of voting classification algorithms: Bagging, boosting, and variants," *Machine learning*, vol. 36, no. 1, pp. 105–139, 1999.

[26] G. I. Webb, "Multiboosting: A technique for combining boosting and wagging," *Machine learning*, vol. 40, no. 2, pp. 159–196, 2000.

[27] J. Mendes-Moreira, C. Soares, A. M. Jorge, and J. F. D. Sousa, "Ensemble approaches for regression: A survey," *Acm computing surveys (csur)*, vol. 45, no. 1, pp. 1–40, 2012.

[28] Z.-H. Zhou, J. Wu, and W. Tang, "Ensembling neural networks: many could be better than all," *Artificial intelligence*, vol. 137, no. 1-2, pp. 239–263, 2002.

[29] G. Martinez-Munoz, D. Hernández-Lobato, and A. Suárez, "An analysis of ensemble pruning techniques based on ordered aggregation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, pp. 245–259, 2008.

[30] S. Masoudnia and R. Ebrahimpour, "Mixture of experts: a literature survey," *Artificial Intelligence Review*, vol. 42, no. 2, pp. 275–293, 2014.

[31] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," *Advances in neural information processing systems*, vol. 30, 2017.

[32] H.-z. Wang, G.-q. Li, G.-b. Wang, J.-c. Peng, H. Jiang, and Y.-t. Liu, "Deep learning based ensemble approach for probabilistic wind power forecasting," *Applied energy*, vol. 188, pp. 56–70, 2017.

[33] X. Qiu, Y. Ren, P. N. Suganthan, and G. A. Amaratunga, "Empirical mode decomposition based ensemble deep learning for load demand time series forecasting," *Applied soft computing*, vol. 54, pp. 246–255, 2017.

[34] Z. Zhuang, Z. Zhi, T. Han, Y. Chen, J. Chen, C. Wang, M. Cheng, X. Zhang, N. Qin, and L. Ma, "A survey of point cloud completion," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 17, pp. 5691–5711, 2024.

[35] K. W. Tesema, L. Hill, M. W. Jones, M. I. Ahmad, and G. K. Tam, "Point cloud completion: A survey," *IEEE Transactions on Visualization and Computer Graphics*, vol. 30, no. 10, pp. 6880–6899, 2023.

[36] A. Sharf, M. Alexa, and D. Cohen-Or, "Context-based surface completion," *ACM Trans. Graph.*, vol. 23, p. 878–887, Aug. 2004.

[37] Z. Cai, C. Wang, C. Wen, and J. Li, "3d-patchmatch: An optimization algorithm for point cloud completion," in *2015 2nd IEEE International Conference on Spatial Data Mining and Geographical Knowledge Services (ICSDM)*, pp. 157–161, IEEE, 2015.

[38] D. Doria and R. J. Radke, "Filling large holes in lidar data by inpainting depth gradients," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 65–72, IEEE, 2012.

[39] K. Sarkar, K. Varanasi, and D. Stricker, "Learning quadrangulated patches for 3d shape parameterization and completion," in *2017 International Conference on 3D Vision (3DV)*, pp. 383–392, IEEE, 2017.

[40] M. Kazhdan, T. Funkhouser, and S. Rusinkiewicz, "Symmetry descriptors and 3d shape matching," in *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 115–123, 2004.

[41] O. Kroemer, H. B. Amor, M. Ewerton, and J. Peters, "Point cloud completion using extrusions," in *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*, pp. 680–685, IEEE, 2012.

[42] T. Rumezhak, O. Dobosevych, R. Hryniv, V. Selotkin, V. Karpiv, and M. Maksymenko, "Towards realistic symmetry-based completion of previously unseen point clouds," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 2542–2550, 2021.

[43] S. Friedman and I. Stamos, "Online facade reconstruction from dominant frequencies in structured point clouds," in *2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–8, IEEE, 2012.

[44] A.-L. Chauve, P. Labatut, and J.-P. Pons, "Robust piecewise-planar 3d reconstruction and completion from large-scale unstructured point data," in *2010 IEEE computer society conference on computer vision and pattern recognition*, pp. 1261–1268, IEEE, 2010.

[45] M. Sung, V. G. Kim, R. Angst, and L. Guibas, "Data-driven structural priors for shape completion," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 6, pp. 1–11, 2015.

[46] M. Pauly, N. J. Mitra, J. Giesen, M. H. Gross, and L. J. Guibas, "Example-based 3d scan completion.," in *Symposium on geometry processing*, vol. 23, p. 32, Vienna, 2005.

[47] C.-H. Shen, H. Fu, K. Chen, and S.-M. Hu, "Structure recovery by part assembly," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 6, pp. 1–11, 2012.

[48] Y. Li, A. Dai, L. Guibas, and M. Nießner, "Database-assisted object retrieval for real-time 3d reconstruction," in *Computer graphics forum*, vol. 34, pp. 435–446, Wiley Online Library, 2015.

[49] X. Zhang, Y. Feng, S. Li, C. Zou, H. Wan, X. Zhao, Y. Guo, and Y. Gao, "View-guided point cloud completion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 15890–15899, 2021.

[50] B. B. Kimia, I. Frankel, and A.-M. Popescu, "Euler spiral for shape completion," *International journal of computer vision*, vol. 54, no. 1, pp. 159–182, 2003.

[51] M. Kazhdan, M. Bolitho, and H. Hoppe, "Poisson surface reconstruction," in *Proceedings of the fourth Eurographics symposium on Geometry processing*, vol. 7, 2006.

[52] D. Stutz and A. Geiger, "Learning 3d shape completion from laser scan data with weak supervision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1955–1964, 2018.

[53] Y. Cai, K.-Y. Lin, C. Zhang, Q. Wang, X. Wang, and H. Li, "Learning a structured latent space for unsupervised point cloud completion," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 5543–5553, 2022.

[54] D. Li, C. Lu, Z. Chen, J. Guan, J. Zhao, and J. Du, "Graph neural networks in point clouds: A survey," *Remote Sensing*, vol. 16, no. 14, 2024.

[55] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, "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, 2017.

[56] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "Pointnet++: Deep hierarchical feature learning on point sets in a metric space," *Advances in neural information processing systems*, vol. 30, 2017.

[57] M. Y. Levi and G. Gilboa, "Epic: Ensemble of partial point clouds for robust classification," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 14475–14484, 2023.

[58] M. Zhang, H. You, P. Kadam, S. Liu, and C.-C. J. Kuo, "Pointhop: An explainable machine learning method for point cloud classification," *IEEE Transactions on Multimedia*, vol. 22, no. 7, pp. 1744–1755, 2020.

[59] M. E. Atik and Z. Duran, "An efficient ensemble deep learning approach for semantic point cloud segmentation based on 3d geometric features and range images," *Sensors*, vol. 22, no. 16, p. 6210, 2022.

[60] M. Chen, A. Feng, K. McCullough, P. B. Prasad, R. McAlinden, and L. Soibelman, "3d photogrammetry point cloud segmentation using a model ensembling framework," *Journal of Computing in Civil Engineering*, vol. 34, no. 6, p. 04020048, 2020.

[61] M. Yuan, K. Fu, Z. Li, Y. Meng, A. Shen, and M. Wang, "Robust point cloud registration via random network co-ensemble," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 34, no. 7, pp. 5742–5752, 2024.

[62] M. Yuan, A. Shen, Y. Ma, J. Du, Q. Huang, and M. Wang, "Boosting 3d point cloud registration by orthogonal self-ensemble learning," *IEEE Transactions on Artificial Intelligence*, 2025.

[63] K. Zhang, A. Zhang, X. Wang, and W. Li, "Deep-learning-based point cloud completion methods: A review," *Graphical Models*, vol. 136, p. 101233, 2024.

[64] Y. Shen, C. Feng, Y. Yang, and D. Tian, "Mining point cloud local structures by kernel correlation and graph pooling," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4548–4557, 2018.

[65] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[66] M. V. Narkhede, P. P. Bartakke, and M. S. Sutaone, "A review on weight initialization strategies for neural networks," *Artificial intelligence review*, vol. 55, no. 1, pp. 291–322, 2022.

[67] S. Lee, S. Purushwalkam, M. Cogswell, D. Crandall, and D. Batra, "Why m heads are better than one: Training a diverse ensemble of deep networks," *arXiv preprint arXiv:1511.06314*, 2015.

[68] H. Fan, H. Su, and L. J. Guibas, "A point set generation network for 3d object reconstruction from a single image," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 605–613, 2017.

[69] A. X. Chang, T. Funkhouser, L. Guibas, P. Hanrahan, Q. Huang, Z. Li, S. Savarese, M. Savva, S. Song, H. Su, J. Xiao, L. Yi, and F. Yu, "Shapenet: An information-rich 3d model repository," 2015.

[70] M. Savva, A. X. Chang, and P. Hanrahan, "Semantically-enriched 3d models for common-sense knowledge," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pp. 24–31, 2015.

[71] T. Akiba, S. Sano, T. Yanase, T. Ohta, and M. Koyama, "Optuna: A next-generation hyperparameter optimization framework," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2019.

[72] M. Tatarchenko, S. R. Richter, R. Ranftl, Z. Li, V. Koltun, and T. Brox, "What do single-view 3d reconstruction networks learn?," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 3405–3414, 2019.

[73] T. Wu, L. Pan, J. Zhang, T. Wang, Z. Liu, and D. Lin, "Density-aware chamfer distance as a comprehensive metric for point cloud completion," *arXiv preprint arXiv:2111.12702*, 2021.

[74] L. Li and H. Kasaei, "Enhanced view planning for robotic harvesting: Tackling occlusions with imitation learning," in *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 13146–13152, IEEE, 2025.

A List of learning algorithm hyperparameters

	Parameter	Value
Optimizer	Type	Adam
	Learning Rate (lr)	0.0001
	Weight Decay	0
Scheduler	Type	StepLR
	Step Size	100
	Gamma	0.5
Model	Number of Predictions (num_pred)	2048
	Encoder Channels	1024
Miscellaneous	Total Batch Size (total_bs)	32
	Steps per Update	1
	Max Epoch	300
	Considered Metric	L1 Chamfer Distance

Table A.1: Hyperparameters configuration for the FoldingNet model for training.

	Parameter	Value
Optimizer	Type	Adam
	Learning Rate (lr)	0.0001
	Weight Decay	0
Scheduler	Type	StepLR
	Step Size	100
	Gamma	0.5
Model	Number of Predictions (num_pred)	2048
	Gridding Loss Scales	128
	Gridding Loss Alphas	0.1
Miscellaneous	Total Batch Size (total_bs)	8
	Steps per Update	1
	Max Epoch	300
	Considered Metric	L1 Chamfer Distance

Table A.2: Hyperparameters configuration for the GRNet model for training.

	Parameter	Value
Optimizer	Type	Adam
	Learning Rate (lr)	0.0001
	Weight Decay	0
Scheduler	Type	StepLR
	Step Size	100
	Gamma	0.5
Model	Number of Predictions (num_pred)	2048
	Encoder Channels	1024
Miscellaneous	Total Batch Size (total_bs)	32
	Steps per Update	1
	Max Epoch	300
	Considered Metric	L1 Chamfer Distance

Table A.3: Hyperparameters configuration for the PCN model for training.

	Parameter	Value
Optimizer	Type	AdamW
	Learning Rate (lr)	0.0005
	Weight Decay	0.0005
Scheduler	Type	LambdaLR
	Decay Step	42
	Learning Rate Decay	0.9
	Minimum Decay	0.02
BNM Scheduler	Type	Lambda
	Decay Step	42
	Batch Normalization Decay	0.5
	Batch Normalization Momentum	0.9
	Lowest Decay	0.01
Model	Number of Predictions (num_pred)	2048
	Number of Queries (num_query)	224
	KNN Layer	1
	Transformation Dimension (trans_dim)	384
Miscellaneous	Total Batch Size (total_bs)	48
	Steps per Update	1
	Max Epoch	600
	Considered Metric	L1 Chamfer Distance

Table A.4: Hyperparameters configuration for the PoinTr model for training.

	Parameter	Value
Optimizer	Type	Adam
	Learning Rate (lr)	0.001
	Weight Decay	0
Scheduler	Type	StepLR
	Step Size	100
	Gamma	0.5
Model	Node Features	8
	Encoder Features	1024
	Number of Predictions (num_pred)	2048
	Encoder Channels	1024
	Number of Levels (nlevels)	8
Miscellaneous	Total Batch Size (total_bs)	32
	Steps per Update	1
	Max Epoch	300
	Considered Metric	L1 Chamfer Distance

Table A.5: Hyperparameters configuration for the TopNet model for training.

	Parameter	Value
Optimizer	Type	Adam
	Learning Rate (lr)	0.001
	Weight Decay	0
Scheduler	Type	GradualWarmup
	Step Size	100
	Gamma	0.5
	Multiplier	1
	Total epoch	400
Model	Feature Dimension (dim_feat)	512
	Num_pc	256
	Num_p0	512
	Radius	1
	Up Factors	2, 2
Miscellaneous	Total Batch Size (total_bs)	64
	Steps per Update	1
	Max Epoch	600
	Considered Metric	L1 Chamfer Distance

Table A.6: Hyperparameters configuration for the SnowFlakeNet model for training.