CS 282 Project report - *Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image*

Teng Xu
2021233339
ShanghaiTech University
xuteng@shanghaitech.edu.cn

Kaixin Yao
2019533072
ShanghaiTech University
yaokx@shanghaitech.edu.cn

Peiheng Cai
2019533182
ShanghaiTech University
caiph1@shanghaitech.edu.cn

Heyang Li
2019533157
ShanghaiTech University
lihy3@shanghaitech.edu.cn

Jie Fu
2019533142
ShanghaiTech University
fujie@shanghaitech.edu.cn

Abstract

In this project, we have successfully re-implemented the NeurIPS 2021 Paper *Voxel-based 3D Detection and Reconstruction of Multiple Objects from a Single Image*. The main idea, approach and the academic contribution are introduced in this report. Besides, our insights and efforts on this paper are discussed. The major goal of this paper is to recover 3D shape and transformation of multiple objects given a single image, which is a popular and long-existing task in the field of Computer Vision and Machine Learning. Many existing works leverage learning methods to regress 3D shape of objects, but with different 3D representations. In this work, the authors have developed a novel hybrid voxel-SDF-based 3D representation. This hybrid representation allows 3D reconstruction in finer details and faster inference. The authors have also designed a pipeline, to extract 3D features from 2D images, and then predict 3D bounding box, heatmap, and course-to-fine 3D shape from the 3D features.

1 Introduction

Inferring 3D information from 2D image is a long-existing and still challenging task in Computer Vision. Related tasks including 3D human or object detection and reconstruction from monocular images, has become increasingly popular within the community. Common approaches to tackle this problem include optimization-based methods and learning-based methods. With the advances in machine learning, especially deep learning, many learning-based methods showed significant success in the recent 5 to 10 years. Among those, various methods are proposed to predict either 3D shape(reconstruction) or 3D bounding boxes(detection) of objects. However, when trying to reconstruct the complicated real-world scenes, one always has to combine two methods together. In other words, both semantic information and geometric information must be considered and extracted jointly from 2D images, in order to fulfill the needs of recovering the scene.

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

The main goal of this work is to propose a comprehensive pipeline, to recover 3D information including 3D bounding boxes and 3D geometry from a single monocular image of a scene containing multiple objects. To achieve this, the authors combined detection tasks and reconstruction tasks together, and designed a novel voxel-SDF-based representation of objects.

The main contribution of this paper can be concludes as:

- The authors has proposed a hybrid 3D object detection and reconstruction pipeline with a single 2D image as input.
- For detection, the authors designed a novel detector called CenterNet-3D, which predicts 3D locations of object center, and produces 3D heatmap and bounding-box of objects presented in the image. This method takes in 3D features to avoid 2D depth ambiguity and showed increased performance.
- For reconstruction, the authors proposed a coarse-to-fine method based on a novel local PCA-SDF geometry representation. Such representation brings 10x speed-up during inference and provides greater details of local 3D shape.

As the final project of CS 282, we summarize our contribution as:

- We successfully re-implement the pipeline proposed by the authors, and successfully reproduce the results shown in the paper. Note that the authors open-sourced only a portion of their codes and data, which is not sufficient to run the pipeline.
- We introduce, analyze, and discuss the methods proposed in this paper in detail.
- We discuss our ideas and insights on this paper, and its relationship with the topics introduced in Machine Learning lectures.

2 Methods

As shown in Fig. 1, the overall pipeline of this work can be separated into 3 parts, each will be introduced in Sec. 2.1, Sec. 2.2, Sec. 2.3.

To begin with, the 2D image is sent to the feature extraction module, where 3D voxel features are computed. For the complete volume, each voxel will be assigned 3D features with the corresponding 2D image pixels by re-projecting the volume back to the 2D image. As in this task we only have a single image as input, voxels along the camera ray will be projected into the same position and therefore get the same features for multiple different voxels at different depths. The authors leveraged the positional encoding strategy to reduce the influence caused by this. As positional encoding will let different voxels have their positional information embedded into their feature and therefore will not be completely the same even if they are projected to the same 2D pixel.

The extracted 3D voxel features are then sent to the detection module and reconstruction module, separately. For the detection module, the features are used to detect 3D keypoints and a 3D heatmap based on classification results showing the probabilities of the location of the 3D object center. For the reconstruction module, a hybrid voxel-SDF based representation is used. The coarse-level reconstruction is a voxel-based representation, whereas the fine-level reconstruction is a novel local PCA-SDF representation.

2.1 3D Voxel Feature Learning

The module 3D voxel feature learning, which is the basis of the next two modules, is designed to extract 3D voxel features of the images including both context and positional information, making the features more discriminative. Authors use $\mathbf{V}_i \in \mathbb{R}^{X \times Y \times Z}$ with size r to separate the scene space and get the 3D grid with size $Xr \times Yr \times Zr$.

First, read RGB information of Image and get $\mathbf{I} \in \mathbb{R}^{W_I \times H_I \times 3}$, apply ResNet-34 to extract D-channels multi-scale 2D feature maps $\mathbf{F} \in \mathbb{R}^{W_F \times H_F \times D}$.

Then, transform 3D voxel center into 2D image center with camera calibration according to perspective projection:

$$[u \cdot d, v \cdot d, d]^T = \mathbf{P}[x, y, z, 1]^T = \mathcal{K}(\mathcal{R}, t)[x, y, z, 1]^T$$

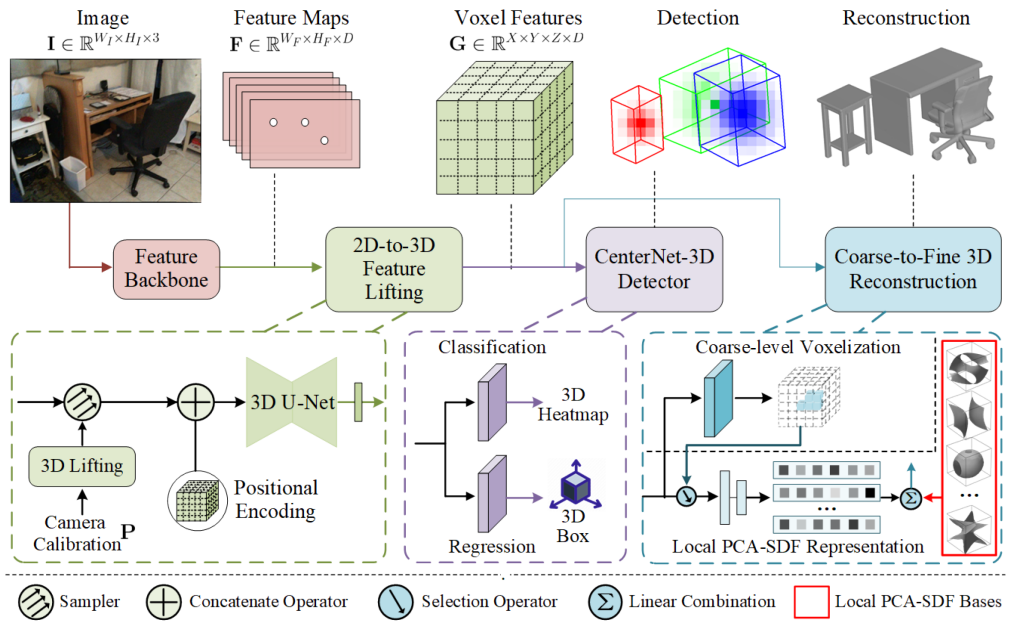

Figure 1: **Pipeline of the paper.** The authors has proposed a detection-reconstruction framework which can be divided into 3 parts: 3D Feature Learning, CenterNet-3D Detector, and Coarse-to-Fine 3D Reconstruction. Given a single 2D image, the corrsponding 2D Feature Maps are generated through the feature backbone module. The 2D features are projected into 3D voxels using the 2D-to-3D feature lifting module. The 3D voxel features are then used for the detection module and reconstruction module to produce 3D Heatmap, 3D Bounding Box and 3D shapes. During the 3D reconstruction process, the hybrid Voxel-SDF methods are adopted, and a novel local PCA-SDF representation are used to enhance local details of generated coarse-level 3D voxels.

After this, 2D feature maps are transformed into 3D voxel features $\mathbf{G} \in \mathbb{R}^{X \times Y \times Z \times D}$.

Finally, add voxel center position to the voxel features and utilize 3D U-Net to blend positional information into the extracted 3D voxel features. The features are $\mathbf{G} \in \mathbb{R}^{X \times Y \times Z \times D}$to $\mathbf{G} \in \mathbb{R}^{X \times Y \times Z \times (D+3)}$.

Thus, this module transform images with RGB information $\mathbf{I}$ into 3D voxel features $\mathbf{G}$ that are relatively effective.

2.2 CenterNet-3D Detector

In this paper, a novel CenterNet-3D head is introduced to make use of 3D voxel features for regression. Compared to conventional CenterNet-based methods that used 2D features, it detects more accurately.

This part takes the voxel features $\mathbf{G}$ as input and predicts a 3D heatmap and 3D bounding boxes.

For the heatmap part, the output $\mathcal{Y} \in \mathbb{R}^{X \times Y \times Z \times C}$, where C is number of object categories. The value of each voxel in $\mathcal{Y}$ is the probability that the 3D centroid of a certain object category exists at the voxel center. The final positions are obtained by computing local maxima and filtering via a threshold, denoted as $\tilde{c}_{3d} = [x_c, y_c, z_c]^T$.

For the bounding box part, the output is a combination of multiple sets of information. It is represented as $\tau = [\delta_{x_c}, \delta_{y_c}, \delta_{z_c}, \delta_h, \delta_w, \delta_l, sin\theta, cos\theta)]^T$. The first three $\Delta c_{3d} = [\delta_{x_c}, \delta_{y_c}, \delta_{z_c}]^T$ defines a compensation offset of the center of object because of disretization error(the position may locate between two adjacent voxels). The next three is $[\delta_h, \delta_w, \delta_l]^T$ represents corresponding transformations; with $[\overline{h}, \overline{w}, \overline{l}]^T]$, a pre-calculated category-wise average box, the size of bounding box is $[l, h, w]^T = [\overline{l} \cdot e^{\delta_l}, \overline{h} \cdot e^{\delta_h}, \overline{w} \cdot e^{\delta_w}]^T$. The last two gives the rotation angle of bounding box around y-axis(perpendicular to the floor).

With the outputs, the bounding box $\mathcal{B} \in \mathbb{R}^{3 \times 8}$ can be defined with 8 corners by

$\mathcal{B} = R_\theta[\pm l/2, \pm h/2, \pm w/2]^T + c_{3d}, \quad c_{3d} = \tilde{c}_{3d} + \Delta c_{3d}$ where $R_\theta \in \mathbb{R}^{3\times3}$ is the rotation matrix.

2.3 Coarse-to-Fine 3D Reconstruction using local PCA-SDF

The authors proposed a coarse-to-fine pipeline, takes in the learned 3D voxel features, and predicts the coarse-level voxel-based volume, and then a fine-level representation using a novel local implicit function local PCA-SDF.

Coarse-Level Voxelization. Given the learned 3D voxel features, the coarse-level reconstruction is a voxelization process. The voxels are represented with binary values, either occupied or non-occupied. The voxels are a coarse-level representation of the 3D geometry of a given object. The coarse voxels are then sent to the local PCA-SDF module to reconstruct at a finer-level.

Local PCA-SDF Shape Representation. The implicit functions such as Neural Radiance Fields or Signed Distance Function(SDF) are risingly popular methods to represent 3D geometry. However, directly applying these methods to the voxel representation will not lead to instant increase of performance, because it's still a challenging problem to learn a global representation directly.

In order to solve this issue, the authors made an argument that, local features, instead of global ones, must be fully exploited. Another key observation that the authors has made is that the local features, on a voxel level, can be classified into several major patterns, no matter what kind of object is. Therefore, the authors designed a novel implicit function PCA-SDF to represent local geometry features.

The PCA-SDF representation is a Principal Component Analysis(PCA) on the voxel-level SDF. The local SDF is decomposed into linear combinations of the shape bases. This local shape prior model enables fine details reconstruction and also efficient training procedure. Formally, for each voxel $\mathbf{V_i}$, a finer grid $\mathbf{q_i} \in \mathbb{R}^{k\times k\times k\times 3}$ is defined, where the grid is $k \times k \times k$ length-width-height and 3 for $x - y - z$. And the corresponding SDF $\mathbf{s} \in \mathbb{R}^{k\times k\times k\times 1}$ is calculated. The PCA is applied to the set of local SDFs $\{\mathbf{s}\}$. Then we can get local shape bases, the PCA bases, $\mathcal{S}_B \in \mathbb{R}^{k\times k\times k\times l_b}$, where l_b is the number of bases. The latent code$\mathbf{z}_i$, are normalized PCA weights $\hat{\mathbf{z}}_i$ to be regressed by a MLP, and the final fine-level representation, SDF $\mathcal{S}_i$, can be calculated by $\mathcal{S}_i = \mathcal{S}_B\hat{\mathbf{z}}_i$.

3 Experiments and Results

3.1 Datasets

Since the author of the article does not directly open source the dataset used and only provides some vague descriptions of the data pre-processing, we use a dataset that we have re-collected based on our own understanding of the article input. We use the well-known ShapeNetcorev2_h5_2048 dataset as our largest dataset, which contains real images, 3D CAD reconstructed point cloud models and pre-calibrated parameters, labels, etc. To better test our work, we first simply take the chair as our main training set, and collect a total of xxx sets of data about the chair As our input, we used our own way of calibration, and the data set is displayed as shown in Fig. 2

3.2 Implementation Details

In this section, we introduce our re-implementation approaches in detail.

3.2.1 3D Voxel Feature Learning

The first module is 3D voxel feature learning which consists of 3 parts: feature extraction, 2D-to-3D Lifting, and 3D voxel feature aggregation as the authors mentioned in the paper.

Feature Extraction. In the supplementary material, the authors mention that they use a ResNet-34 without bottleneck layers as the backbone to extract features. ResNet is an effective neural network adopting skip connection in order to make the network reach a very deep level, thus elevating performance.

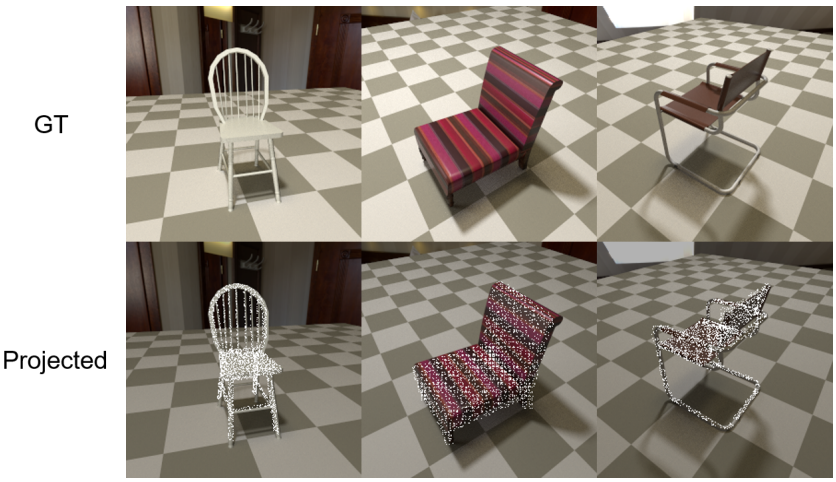

Figure 2: **From top to bottom are: the captured image, the projection of the model point cloud.**

The input data is $256 * 256 * 3$ given by reading images with `imread` while the output data are $feat8$, $feat16$, and $feat32$, respectively corresponding to the reshaped result of Block #2, Block #3, and Block #4.

Lifting 2D to 3D. Since the process of photography is mapping the 3D world into a 2D photo, then the same process can also be used to map the 3D voxel center into 2D image center. we use the camera calibration including a perspective matrix $\mathbf{P}$ where intrinsic calibration matrix K and extrinsic parameters are given data to achieve this objective.

After computing the corresponding relationship between the voxel center and the image center, 2D features can be mapped to 3D by bi-linear interpolation. Inputting $W_F * H_F$ features achieved from ResNet and the image center projected by camera calibration, using `grid_sample` to complete bi-linear interpolation, finally, we get the output voxel features.

3D Voxel Features Aggregation. Positional encoding can fix the problem that voxels along a camera ray will receive the same 2D feature, which may add difficulties to 3D detection and reconstruction in that different information is smeared. Thus, we follow the authors by concatenating 3D features and voxel center position and utilize 3D U-Net to integrate the position information, so that the features can be more discriminative.

3.2.2 CenterNet-3D Detector

The paper said that a novel CenterNet-3D detector for generating heatmap and bounding boxes, but no implementation details were mentioned in the paper. The only hint for the part is the size of the convolution kernel mentioned in the supplemental file, but the job simply cannot be done with two convolution layers without more specific processing. Considering the whole structure of the net in the paper, feature extraction and processing are finished before the data comes to the detector, so this part should correspond to the 'head' part of deep neural networks. As a deviant of CenterNet, we referred to other CenterNet implementations for help.

In the traditional CenterNet, the head part for each output consists of two 2D convolution layers. In this paper, the outputs are 3D bounding boxes, so 3D convolution layers should be used instead. The number of channels for heatmap output should be the number of categories of objects, which is decided by the dataset. The output for regression should have 8 channels: 3 for center offset to remedy the discretization error of voxels, 3 relative error of size on $x - y - z$ corresponding to a template, and 2 for rotation angle of the bounding box. Different activation functions are used to bound results into corresponding ranges.

Normalization, activation function, and data filling are used according to implementations of another similar work *SMOKE: Single-Stage Monocular 3D Object Detection via Keypoint Estimation*. Ordinary normalization is replaced with group normalization to be less sensitive to batch size and more robust.

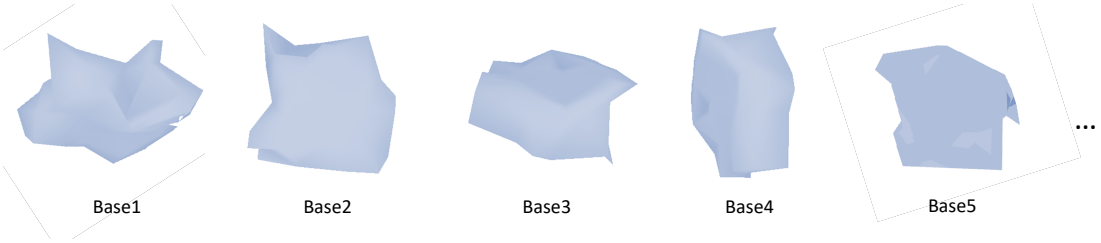

Figure 3: **Visualized PCA bases examples.**These bases are generated by a learnt local PCA-SDF model (in our implementation, we restrain the number of bases into $\mathbf{l_B}$=25, here 5 out of 25 are shown), each can represent PCA-SDF basis within a voxel, which means using the linear combinations of these bases, we are able to recover the shape with fewer parameters in a more efficient way.

3.2.3 Coarse-to-Fine 3D Reconstruction using local PCA-SDF

As mentioned in the paper, we have re-implement the two reconstruction stages, namely, coarse-level and fine-level reconstruction.

Coarse-level. Following the pipeline proposed by the authors, the 3D voxel feature $\mathbf{G}$ can be used to obtain a coarse-level voxelization $\tilde{\mathbf{V}}$, which represents the space with binary occupancy values. So we implemented this part with a network with upsampling to get an estimate of voxelization with more local information, which is good for generating voxels. With the voxelization we have, now we can further reconstruct the object with more details, which is fine-level reconstruction.

Fine-level. The main idea of this stage is to use the SDFs of a given regular lattice(a split of a voxel as defined in Sec. 2.3) to recover the shape of each voxel, therefore reconstructing the whole object. According to the paper, applying the PCA(Principal Component Analysis) to such large-scale features can speed up the progress of training and avoid learning too much parameters, so we trained a PCA model with all the lattices with SDFs to get the bases and mean. However, this is not enough. Computing the SDF is way too slow and costly, so based on the work of DeepSDF, the author uses latent code to solve this problem. And with this part, we implemented a simple network to get the latent code(SDF features) from voxel feature $\mathbf{G}$ to recover the shape of each single pixel. Finally, after two stages of reconstruction, we are able to get a very fine model with details and meet the requirement of real-time applications.

3.3 Training

For each training session, our training input is the images in the dataset and the corresponding camera calibration internal and external parameters, and our training output is the sdf surface generated by the object point cloud information from the ShapeNet dataset through the PCA model and the latent constraint, in our experiments, we used the graphics card Nvidia RTX 2080 Ti to train 200 epochs were used to train to get the model results we wanted.

3.4 Result

To make the results more intuitive, we have listed the original image of the network as input, the ground truth model of the object and our results to check how good or bad our results are. As we can see from the Fig. 4, the classification of the objects in the general space is generally correct, but we can also see some small flaws, such as the overall contour of the object is not smooth, and there are cases of broken object classification, which can be considered in our subsequent work to further improve the results.

4 Discussion and Conclusion

In this section, we gave a discussion of the relationship between this project and the CS 282 course. Also, We conclude our contribution in this project and the academic contribution made by the authors in Sec. 4.2.

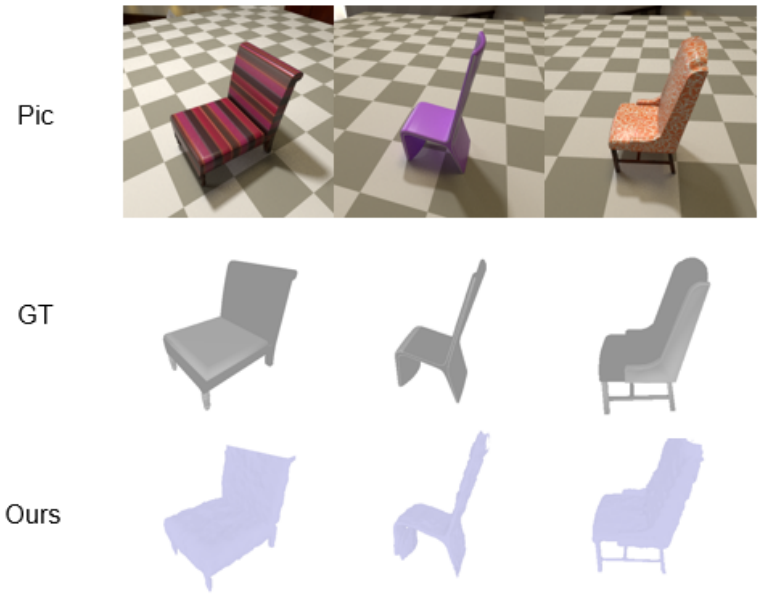

Figure 4: **From top to bottom are: the captured image, the model GT, our results.**

4.1 Discussion

This work, just like many other works in the field of Computer Vision and Deep Learning, all have Machine Learning theories embedded in their most fundamental part. We can discuss them topics by topics.

Types of Machine Learning. In this project, we used supervised learning and unsupervised learning. The training process of our major neural network is supervised learning, and the training of the local shape prior using principal component analysis(PCA) is unsupervised learning.

Components of learning. In this work, the fundamental learning process is the same as the learning diagram. The 3D detection and reconstruction task is basically try to fit the Unknown target function $f : \mathcal{X} \to \mathcal{Y}$, i.e. the implicit shape function, SDF. The training examples are used. The hypothesis set is defined by the architecture of the deep neural networks. With the Learning Algorithm, we finally get the final hypothesis $g \approx f$

Expected Risk Minimization. The training process of the deep neural network is the process of the Expected Risk Minimization. As it minimizes the predefined loss on training samples.

Noise and Error. The design of the loss of the neural network reflects the measurement of the noise and error. In this case, it has no preference on false positive or false negative.

MSE vs. MAE. The L1 loss and L2 loss used in this work by deep neural networks are MAE and MSE.

Stochastic Gradient Descent. In this work, the variant of the Stochastic Gradient Descent method, Adam, is used as the optimizer of the deep neural network. In fact, nearly all of the recent solutions to these tasks, either optimization-based or learning-based, has optimization methods such as SGD, Adam or LBFGS embedded.

Learning Rate Selection. Although backtracking line search is not used in this work, learning rate selection policy, rather than fixed LR, is still effective. In this work, `MultiStepLR` is used, decreasing learning rate during the training process. This is a common and widely-used and effective

techniques in deep learning communities, as decreasing step size will better approach the desired local minima.

Perceptron. In this work, one of the most common neural network, Multi-Layer Perceptron(MLP), is used. Different from a single Perceptron and the Perceptron Learning Algorithms(PLA) taught in the Machine Learning Lectures, the Multi-Layer Perceptron is composed of multiple layers of Perceptron, and each Perceptron is the basic unit of the neural network. Actually, as the most fundamental neural network, MLP became popular again in the recent three years. This is because of the significant success of implicit models since 2020, such as Neural Radiance Fields(NeRF), and Neural Signed Distance Functions(Neural SDFs).

Theory of Generalization Most of the deep learning tasks claim the abilities of generalization as one of their major goals, including this one. The VC inequality, as shown in Eqn. 1,

$$\mathbb{P}[\sup_{h\in\mathcal{H}} |E_{in}(h) - E_{out}(h)| > \epsilon] \leq 4m_{\mathcal{H}}(2N)e^{-\frac{1}{8}\epsilon^2 N} \tag{1}$$

gives us the fundamental explanation of how the generalizability of deep neural network is achieved. As the size of the training dataset grows, with the fixed tolerance and hypothesis class, there are less probability that the out-sample error E_{out} deviates from the in-sample error E_{in}. So in deep learning tasks, we try to increase the data size to increase the generalizability. Also, we carefully design the neural network architecture, which changes the hypothesis class$\mathcal{H}$, to increase the generalizability.

Overfitting and Regularization In this work, together with most of the deep learning tasks, we added regularization terms to our loss functions in order to decrease the probability of overfitting. This is also the concept of **Structural Risk Minimization**. We also divide our dataset into three parts, training set, validation set, and test set. The validation process are applied in order to select a better model, and to decrease the effects of overfitting. During training, various methods are also adopted to inspect whether the overfitting is occurred, which is also an extension of our course concepts.

4.2 Conclusion

In conclusion, we have successfully re-implemented the pipeline proposed by the authors, and successfully reproduce the results shown in the paper. Note that the authors open-sourced only a portion of their codes and data, which is not sufficient to run the pipeline. We have introduced, analyzed, and discussed the methods proposed in this paper in detail. Also, we have discussed our ideas and insights on this paper, and its relationship with the topics introduced in Machine Learning lectures.

In this paper, the authors have proposed a hybrid voxel-SDF-based 3D object detection and reconstruction pipeline with a single 2D image as input. For detection, the authors designed a novel detector called CenterNet-3D, which predicts the 3D locations of the object center, and produces a 3D heatmap and bound-box of objects presented in the image. This method takes in 3D features to avoid 2D depth ambiguity and showed increased performance. For reconstruction, the authors proposed a coarse-to-fine method based on a novel local PCA-SDF geometry representation. Such representation brings 10x speed-up during inference and provides greater details of the local 3D shape. However, the limits of this paper are that a camera calibration matrix is necessary for feature extraction, which is not ideal in real-world scenarios. Also, the object category that this pipeline is still limited (less than 20 categories in this work), as obtaining such a large-scale dataset with rich object categories is still a challenging problem.