Hochschule
Bonn-Rhein-Sieg
University of Applied Sciences

b-it Bonn-Aachen International Center for Information Technology

Master's Thesis

Multi-View Temporal Fusion in Semantic Segmentation

Manoj Kolpe Lingappa

Submitted to Hochschule Bonn-Rhein-Sieg,
Department of Computer Science
in partial fullfilment of the requirements for the degree
of Master of Science in Autonomous Systems

Supervised by

Prof. Dr. Nico Hochgeschwender
Prof. Dr. Sebastian Houben
M.Sc. Deebul Sivarajan Nair

Month 20XX

I, the undersigned below, declare that this work has not previously been submitted to this or any other university and that it is, unless otherwise stated, entirely my own work.

Date	Manoj Kolpe Lingappa

Abstract

Your abstract

Acknowledgements

Thanks to ....

Contents

List of Figures

List of Tables

1

Introduction

1.1 Motivation

Any task to make a prediction by combining data from different sources uses data fusion. Data fusion combines the information from multiple sources to achieve improved performance and inferences. According to Hall and Llinas [1] data fusion can be defined as "data fusion techniques combine data from multiple sensors and related information from associated databases to achieve improved accuracy and more specific inferences than could be achieved by the use of a single sensor alone." The living organisms fuse information from various sources and past data to make an informed decision [10]. Data fusion aims to reduce the prediction error probability and improve the model's reliability. Data from multiple sources can be fused at different levels, such as raw data, features, or decision levels. The data sources can be from various fields or different data types. Data fusion is described in different contexts and in other application areas. The most common areas include decision fusion and multisensor data fusion. [11]

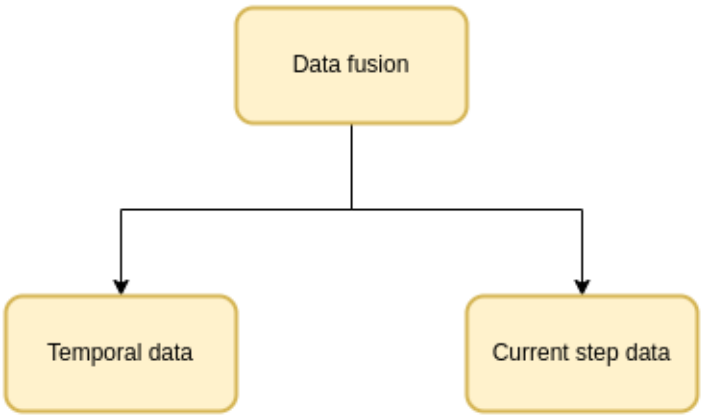


Figure 1.1: Data fusion categories based on timestamp

Data fusion is divided into temporal and current data based on the timestamp factor. Temporal data are the data collected from the former steps. In the current data fusion approach, the data are extracted

from the current step and are fused for improved prediction.Information fusion is applied in different fields such as time series prediction [12], video-based depth estimation [13], and segmentation [14].

Understanding surrounding regions and decision making of a human is based on the signals obtained from different sensors. However, with the knowledge of the past helps to better recognize the nearby activity or to make a educated choice. Thereby fusing the information from different sources and the former data adds to achieve a improved outcome. Temporal fusion is a process of fusing the information to the current step to make the prediction better at each timestamp. Common temporal data types include weather data, frames in a video sequence, and different sensor data. Semantic segmentation take advantage of the temporal fusion to make a better decision.

1.1.1 Temporal fusion

In a general setting the previous data is not utilized to make a current prediction, resulting in information loss. The rich features from the past can be utilized in the current step, thereby making a robust and efficient model prediction. Temporal fusion is one of the dimensions of data fusion, and different data sources collected over a period of time are fused for improvement in the prediction [15]. Classical multi-data fusion finds application in automatic target tracking, autonomous vehicle detection, surveillance systems, robotics, wearable devices, and manufacturing monitoring. In all of the mentioned areas, the data is collected over a period of time and contains important features. These temporal features are combined together to make an efficient prediction.

The temporal fusion can model the behavioral aspect of the collected data rather than just the current timestep data. The temporal fusion extracts the relationship between contextual and temporal proximity [16]. The temporal arrangements of events are captured, thereby incorporating the cause and effect phenomenon [16]. Temporal fusion can be commonly observed in human activity detection [17], context-aware mobile phones [19], and online batch process monitoring [18].

A 3D object detection approach on two popular datasets, KITTI[4] and nuScenes[5] take single-time step LIDAR data for prediction, resulting in the loss of valuable forecast and features computed during the previous step [3]. Using the rich features present in the successive frames to accommodate the past data at a time is extensively studied in neural network-based action recognition [19] [20] [21] [22] and video object detection [23] [24] [25] methods. The fusion of features from the previous step to the current step to improve the 3D object detection is studied in the Temp-Frustum Net architecture [3]. A Temporal Fusion Module (TFM) is proposed to combine the object-specific features. The temporal fusion method improved the average result by 6% [26]. Depth map using dynamic MRFs fuse Time-of-Flight (TOF) and passive stereo to get an enhanced depth map. The depth map estimation is extended to the temporal domain resulting in accurate depth maps [27]. Multi-camera video surveillance fuses the spatiotemporal frames from different sources to reliably find the motion trajectories [28]. A moving object is detected and segmented with the unmanned aerial vehicle (UAV) data by stacking the consecutive frames containing objects of interest resulting in constant object position and moving background, thereby improving the segmentation efficiency [29].

1.1.2 Semantic segmentation

Segmentation of images is an essential task of the visual understanding systems. It involves dividing the image into multiple segments. Image segmentation can be framed as classifying the individual pixels into a particular class or semantic labels. Segmentation can be classified as semantic segmentation, instance segmentation, and panoptic segmentation. Segmentation of images finds a broad range of application areas [30], such as medical for boundary extraction and tissue volume estimation, autonomous systems for detecting a boundary for path planning, and surveillance to track objects. Semantic segmentation is not only about the data but the problem segmentation addresses. For example, in a pedestrian detection system, pixels belonging to a person are categorized into a single class; however, for action recognition, the different parts of the body are classified into other classes. Instance segmentation [31] solves the problem of counting unique objects present in an image and is a common task in image retrieval tasks. Many traditional techniques have been developed to solve the segmentation problem [32]. For specialized tasks, different algorithms are developed [33]. Work by Shervin surveyed the various state-of-the-art segmentation algorithms [34]. However, many algorithms are developed, but few works are proposed for multi-view semantic segmentation. From the previous work, it is evident that temporal fusion improves the model's performance. This work aims to study the impact of temporal fusion of information in the latent space and cross-transfer the technology to the semantic segmentation task.

1.2 Challenges and Difficulties

Semantic segmentation benefited from the advancement of deep learning methods. Building a temporal fusion for semantic segmentation model is a challenging task due to the presence of high number of variables involved and different choices of fusion architectures. Common challenges involved are the

- Datasets
- Fusion architecture
- Computation cost
- Application areas

1.2.1 Dataset

In many application areas the deep learning model can be trained from scratch given that we have large datasets. However, for new domain there are not enough datasets available to train the model, in such cases transfer learning can be applied. In the transfer learning approach a model is trained on some data and the part of trained model weights are used for building a new application areas architecture. Many deep learning based models are trained on the ImageNet datasets and take the pretrained encoder weights which capture the features needed to do the segmentation thereby reducing the dependency on the requirement of large datasets. Image augmentation is the another approach to increase the number of data points. Data augmentation helps to create more data by applying transformation to the existing

small datasets so that variety of the input data is generated from the existing small datasets. Some of the common transformation on the input images are translation, reflection, rotation, warping, scaling, color space shifting, projecting onto the principle component. It helps to faster convergence and reduce the over-fitting probability, and improving the generalization capability of the model. For some task data augmentation showed improvement in the performance of the model. For temporal fusion there is a need of 2D datasets along with the pose of the camera. Pose information cane be fused at the latent space to improve the prediction efficiency of the model. There is a need for different kinds of datasets such as the still images, navigation datasets, Unmanned aerial vehicle (UAV) datasets to validate the model in different environment, helps to evaluate the model performance.

1.2.2 Fusion architecture

With the advancement of the deep learning more and more segmentation models are developed with improved efficiency and with variety of fusion architecture. Fusing of features are commonly used in the segmentation task. Adaption of fusion features in the increased depth deep learning model showed significant improvement in the prediction. U-net [35] model effectively use the already learned features by fusing the information from the encoder to the decoder. To tackle the decrease of initial image resolution at the output a RefineNet [36] network was proposed. Deeper layers captures the high-level semantic features is refined by fusing the fine-grained features from the earlier convolutions. Dense connection is employed in the many of the recent neural network architecture [37], [38], [39]. Choosing the appropriate fusion architecture depends on the factor of problem at hand available resources to solve the problem.

1.2.3 Computation cost

Many state of the art segmentation network requires high computation cost during training as well during the inference time. So the recent research is focused on decreasing the computation cost and also keeping the accuracy of the model high. To deploy the model in the low computational mobile devices simpler models needs to be developed that fits in the computation cost of the device. This can be done by compressing the model or using the knowledge distillation techniques to build the low computational model [34].

1.2.4 Real time inference for various application areas

Most of the recent top performing semantic segmentation models are based on the fully convolutional network [40]. For real time application or at the frame rate of the camera needs to have reasonable accuracy and prediction speed. Real time prediction is extremely critical in the autonomous driving and medical fields. However, most the fully convolutional network is not upto the mark with respect to the maximum requirements defined by application areas. Models with the dilated convolution improved the performance of the model however the benchmark can be still improved. ICNet takes multiple input sizes to capture objects of varying sizes to tackle the real time deployment [41].

1.3 Use cases

Semantic segmentation finds application in many areas of the computer vision. Some of them are listed below,

1.3.1 Autonomous driving and Robotics

Important components of the autonomous driving systems are the object recognition, object localization and segmentation. Semantic segmentation classify each pixels of the image into a particular class thereby identifying different classes such as street, traffic sign, trees, cars, sky, pedestrians or sidewalks. It is critical to classify each pixel with high accuracy due to the safety concerns. The rich information captured in last step can be used in the current step calculation to make a better prediction at the current computational step. With the development of the robotics system to perform a complex task the interaction with the environment also increased. So, there is a need to develop a robust system to understand the knowledge about the workspace.

1.3.2 Weed mapping using Unmanned Aerial Vehicle (UAV)

Mapping of the fields are essential for weed control and spraying applications. The presence of the weed can be mapped by unmanned aerial vehicle remote sensing technology. The targeted spraying onto the weed area helps to curb the weed growth by inspecting the weed map obtained from the UAV. The entire process involves real-time image processing hardware that integrates the map visualization, flight control, image collection [42]. To build a weed map semantic segmentation can be employed with reasonable performance and real time capability.

1.3.3 Real-Time Hand Gesture Recognition

Hand Gesture Recognition (HGR) is an essential component in human-computer interactions. With the advancement of vision-based HGR systems, HGR is widely used in the automotive sector, consumer electronics, home automation, etc. Important feature of the HGR is the real time performance. HGR should perform without any lag to control the location of the cursor. HGR is based on the semantic segmentation method to locate the position of the hand, therefore an efficient real time segmentation network needs to be developed [43].

1.4 Problem Statement and Contribution

Research question answered and contribution in the thesis work is listed below

1.4.1 Research question

RQ1 What are the works on state-of-the-art temporal fusion?

RQ2 How are the results from RQ1 compared with each other to perform temporal fusion?

RQ2.1 What are the results in comparison with different error metrics?

RQ3 How to cross-transfer the depth estimation temporal fusion technique to semantic segmentation?

RQ3.1 How do different loss criteria impact semantic segmentation perfor- mance?

RQ3.2 What is the semantic segmentation performance for different Gaussian kernels?

1.4.2 Contribution

- Literature review on the temporal fusion in the context of depth estimation and semantic segmentation
- Analysis of the state-of-the-art temporal fusion architectures
- Create a baseline of temporal fusion with sequence images
- Compare performances of state-of-the-art temporal fusion techniques with different error metrics
- Cross transfer the temporal fusion architecture to the segmentation task
- Performance of multi-view temporal fusion with different Gaussian kernels

1.5 Report outline

Theoretical background of deep learning, semantic segmentation, temporal fusion and their limitations is discussed in the Chapter 2. Datasets, preprocessing steps, experimental designs, training procedures and hardware configuration used to training and inferences are listed down in the Chapter 3. Evaluation of the temporal fusion architecture with different experimental settings, metrics and research questions are discussed in the Chapter 4. Deployment of the model in the android is described in the android deployment Chapter 5. Finally contribution of the thesis work, lesson learned and future work is explained in the conclusion chapter 6.

2

State of the Art

Introduction to the modern deep learning and their impact onto the various vision tasks are described in the Deep Learning section. Information fusion in the temporal domain to fuse information is explained in Temporal Fusion. State of the art segmentation of the input images, in particular semantic segmentation task is illustrated in the Semantic Segmentation section. State of the art segmentation in the classical era and in modern deep learning play crucial role in the temporally fused semantic segmentation. However, there is very little work of fusing the camera pose onto the segmentation task in temporal fashion. More details are discussed in the Semantic Segmentation. Finally chapter 2 is ended with the discussion on the limitations of the previous work with respect to the temporal fusion.

2.1 Deep Learning

Deep learning is a sub field of machine learning that aims to learn the features present in the data by utilizing the hierarchical architectures. The area deep learning falls in the artificial intelligence is depicted in the picture 2.1

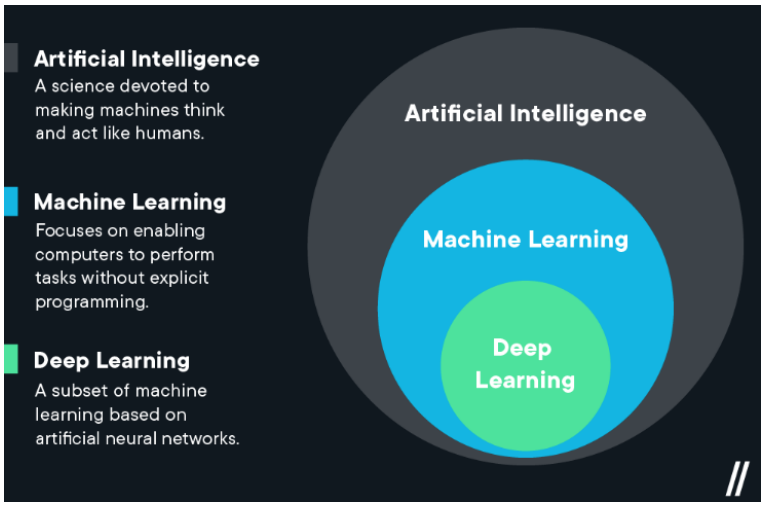


Figure 2.1: Deep learning in the artificial intelligence domain. Courtesy of [1]

Classical machine learning system uses the raw input and domain expert carefully represent the data as a feature vector from which the data is fed to the models to learn the patterns and classify into appropriate classes [44]. Deep learning is a representation learning that takes raw data and find the patterns in the data with different levels of representation in the multiple layers [44]. Deep learning can learn any complex representation of the data. For example, a image is represented as pixels and are fed to the neural network, at each layer of the network different feature are learned. In the first layer, higher level features such as edges at a specific orientation and location is determined. In the second layer motifs are learned and so on. The important aspect of the deep learning is that the features are not designed by the field expert rather than learned from the data with a specific set of learning procedures [44].

Many current state of the art learning models uses the deep learning approach to learn the complex function from data. Currently deep learning method can be found in image recognition [45], speech technologies [46], discovery of drug molecule [47] , understanding the particle accelerator data [48], DNA sequencing [49], ,and natural language processing [50].

Computer vision is the field of computer science that deals with replicating the functionalities of the human visual system. Traditionally computer vision solved the vision problem by finding the hand crafted features. However, the performance of the classical apporach is outperformed by the advancement of the deep learning based methods. Hand crafted feature descriptors such as Speeded Up Robust Features (SURF), Hough Transforms are used as feature vectors for the classical machine learning methods for learning [51]. Deep learning methods automatically learns the patterns from the data. Computer vision solves wide variety of problems in the perception domain. Latest approaches helps to solve the detection [52], [53], classification [54], image synthesis and segmentation tasks [55].

Temporal data are the time varying information and can be commonly observed in financial portfolio management, accounting, medical records, inventory management, data from airline, hotel, train industries contains time component with it [56]. Video data are constructed from combining time variant frames and is a common example of a temporal data. Temporal fusion deals with combining of the past information into the current step computation with a aim of improved performance.

In general approach segmentation is done frame by frame or by skipping in between frames and computing the segmentation on the nth frame. Temporal fusion can be applied in these settings to perform improved segmentation task by combining the past rich information in the current step.

2.2 Temporal Fusion

Temporal fusion can be defined as the process of fusing the temporal data onto the current step with a aim of improving the performance of the model. Temporal data can be observed in many fields such as social media, healthcare, accounting, agriculture, transportation, physics, crime data, traffic dynamics and climate science [57]. Temporal data can be encountered with different data types, Common data types are video, audio, tabular data, and sensors data. Forecasting is a common application of temporal fusion. Multi horizon forecasting is a important problem in the domain of time series. Multi horizon forecast allow the user to optimize the process across the entire path. A novel Temporal fusion Transformer (TFT) [58] is a attention based DNN architecture for forecasting by fusing the important past features into

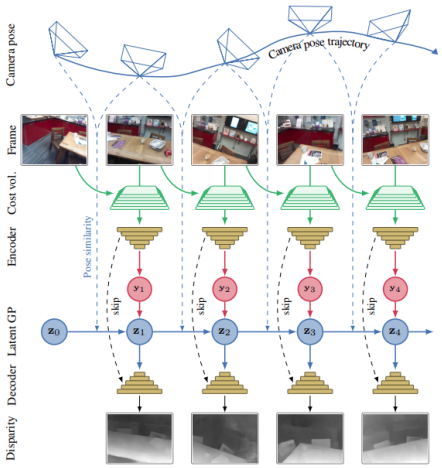


Figure 2.2: Mulit view stereo architecture for depth estimation. Courtesy of [2]

the current step. Temporal fusion play a major role in the improved video action recognition. Temporal fusion helps in two ways, firstly by understanding the temporal data the accuracy of the recognition for the dynamic action is improved, secondly removing the redundant temporal data saves the computation overhead. A temporal fusion network known as the AdaFuse, fuses the current and past features with a goal of improved accuracy and efficiency [59]. A temporal non parametric fusion aims to fuse the temporal pose data to the computation of the depth map thereby improving accuracy and efficiency [2]. The architecture of the multi view stereo can be depicted in the Fig 2.2. An online multi view depth prediction approach where the depth estimated in the previous step is fused onto the current step in a sensible manner. The network is named as DeepVideoMVS and it is based on the encoder decoder architecture. A ConvLSTM is placed at the latent space to fuse the information from the previous step. The proposed approach outperformed all the existing state of the art multi view stereo method evaluated on the standard metrics [60]. A Multiple Fusion Adaptation (MFA) method improves the segmentation accuracy on a unlabeled datasets. Three fusion approach was proposed under MFA, cross model fusion, temporal fusion and novel online-offline pseudo labels. The MFA produced improved semantic segmentation result of 58.2% and 62.5% on GTA5-to-Cityscapes and SYNTHIA-to-Cityscapes respectively [61].

2.3 Semantic Segmentation

Humans can perceive the surrounding environment and make sense of it with high accuracy. Due to the advancement of computer vision these capabilities are transferred to the machines, performing even better than humans. Today, we have computer vision models that can detect objects, find shapes, track the object movement and perform action based on the data. Computer vision is most commonly used in the autonomous driving cars, aerial mapping, surveillance applications, virtual reality and augmented reality and so on. One of the common problem in computer vision is labeling the each pixels of the image to a particular categories. Also known as the segmentation. Mathematically image segmentation can be defined as

If I is set of all image pixels of a image, then segmentation generate unique regions $S_1, S_2, S_3, S_4, ...S_n$ such that combining all these regions will return I.

Image segmentation can be classified into three categories Semantic segmentation, Instance segmentation and Panoptic segmentation. Semantic segmentation finds the shape, size and form of the objects in addition to their location. Instance segmentation finds one more parameter of number of unique object present in the image. Panoptic segmentation is the combination of the semantic and instance segmentation. The difference between all the types of semantic segmentation can be observed in the Fig 2.3.

Figure 2.3: Semantic and Instance segmentation example. Courtesy of [3]

2.3.1 Classical Semantic Segmentation

Most commonly used traditional segmentation techniques are threshold based technique [62], histogram-based bundling, region-growing [63], k-means clustering, watersheds, active contours, graph cuts, conditional and Markov random fields [64], sparsity based methods [65]. However, in the recent years deep learning (DL) yielded a new generation of image segmentation models with state of the art performance.

2.3.2 Deep Learning based Semantic Segmentation

Deep learning based segmentation network can be classified into following categories [4]

- Fully convolutional networks
- Convolutional models with graphical models

- Encoder-decoder based models
- Multi-scale and pyramid network based models
- R-CNN based models (for instance segmentation)
- Dilated convolutional models and DeepLab family
- Recurrent neural network based models
- Attention-based models
- Generative models and adversarial training
- Convolutional models with active contour models

Deep learning based computer vision model most commonly use the convolutional neural network [66], recurrent neural network (RNNs), and Long short term memory (LSTM), encoder-decoder [6] and generative adversarial networks (GANs) based networks [4]. The master thesis work is concentrated on the encoder-decoder based deep learning models. Encoder-Decoder based network are a two stage network that learns to map from input point to the output point. In the encoder stage the input data is compressed into a latent space representation $z = f(x)$ and decoder decompress the latent space representation to the output $a = g(z)$ [67]. Latent representation of the input data in compressed form. It can be commonly observed in image-to-image translation problem as well as in sequence-to-sequence models in NLP. A reconstruction loss $L(y, \hat{y})$ is defined at the output that measure the differences between the ground truth output y and corresponding reconstruction $\hat{y}$. Autoencoders are the special version of the encoder-decoder models that have similar input and output.

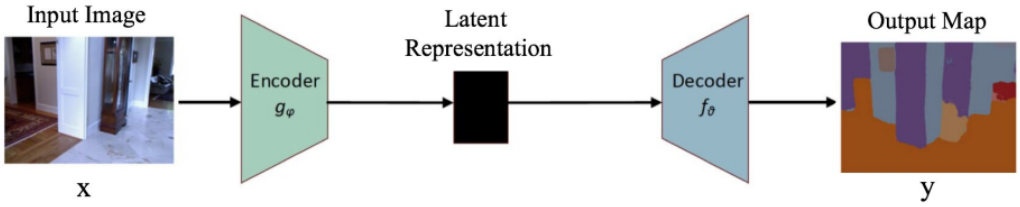


Figure 2.4: Simple encoder-decoder architecture. Courtesy of [4]

Most of the segmentation network are encoder-decoder based architecture. A novel semantic segmentation network was proposed by Noh et al [5]. The network is based on the deconvolution. The encoder network is based on the VGG 16-layer network and the decoder network takes the latent space encoding and outputs the pixel wise class probabilities. The segmentation mask and pixel-wise class labels are predicted by the deconvolutional and unpooling layers. The network generated a accuracy of 72.5 % on the PASCAL VOC 2012 dataset.

Badrinarayanan et al proposed a convolutional encoder-decoder architecture for image segmentation called as SegNet [6]. The architecture of the SegNet described in the Figure 2.6

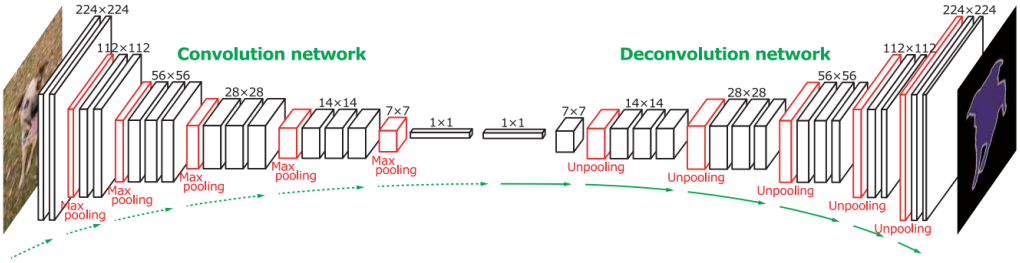


Figure 2.5: Simple encoder-decoder architecture. Courtesy of [5]

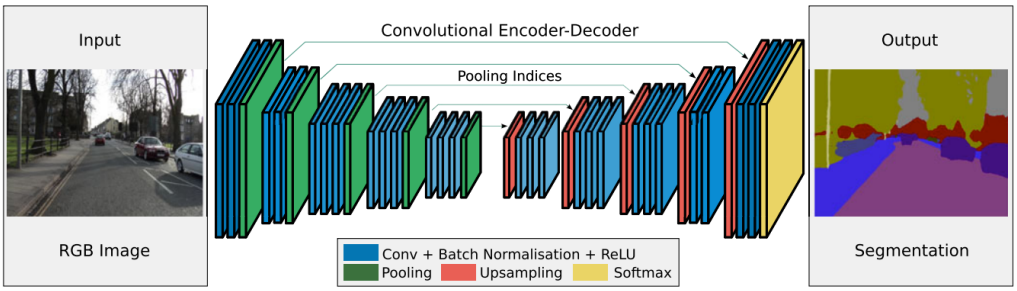


Figure 2.6: SegNet architecture. Courtesy of [6]

The encoder part of the SegNet consist of 13 convolutional layers in the VGG16 network, and followed by the pixel wise classification layer. Decoder upsample the low resolution feature maps in a unique fashion. Non-linear upsampling is performed by using the pooling indices computed in the max-pooling step of the encoder. This process of reusing the encoder output helps to eliminate the need for learning to up-sample. Dense feature maps are generated by convolving with the trainable filters. To account for the uncertainty involved with the encoder-decoder network, scene segmentation is proposed [68]. HRNet [69] is the recently developed high resolution network by connecting the high to low resolution convolutions streams in parallel and exchanging information between different resolutions. HRNet maintain high resolution representation through the encoding process. Many recent architecture use HRNet as the backbone. Other encoder decoder segmentation models are Stacked Deconvolutional Network [70], Linknet [71], W-net [72].

Many segmentation models are developed for medical application and among those U-Net [7] and V-Net [73] are the famous architecture. These architecture are now used outside of the medical domain.

Ronneberger et al [7] proposed a segmentation model to perform semantic segmentation on medical microscopy images. The architecture of the U-Net is described in Fig 2.7. The context is captured by the contracting part and localization of the target area is identified by the expanding decoder path. The network heavily dependent on the annotated images efficiently. The encoder part has a 3x3 convolutions features extractor, similar to the FCN-like architecture. The decoder part increases the dimensions and reducing the number of feature maps. The feature map from the encoder is mapped to the upscaled decoder to retain the pattern information. A 1x1 convolution at the output process the feature maps to

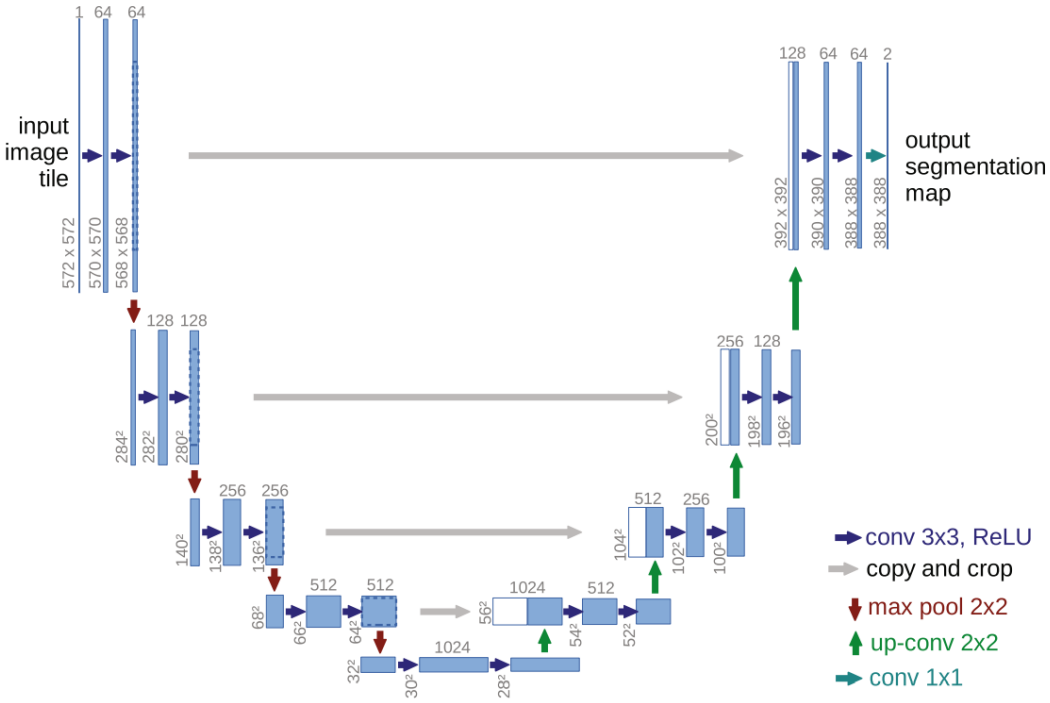


Figure 2.7: Unet architecture. Courtesy of [7]

generate segmentation output by categorizing each pixels of the input image to a particular class. Original U-Net was trained on the electronic microscopic images and outperformed by a large margin on the ISBI challenge. The network is fast and produce result on 512x512 image in less than a second on the modern GPU [7], [4]. In a sequence data the information from the previous frames can be utilized to segment the current frame with a aim of improved performance in comparison to the segmentation without the temporal fusion.

2.4 Temporal Fusion in Semantic Segmentation

Semantic segmentation of sequence data aims to assign pixel-wise semantic labels to the video frames. It is a important task in the visual understanding [74]. Strong representation of the feature map are important for the segmentation task. One of the common approach in the video segmentation is to perform the image segmentation to each frame independently. However the temporal information of the dynamic scenes are not captured by this approach. A common solution to the problem is to apply semantic segmentation to the each and every frame and add additional layer on top to capture the temporal data to extract the better features [75], [76], [77]. However, such approach doesn't help to improve the performance as feature needs to be computed at each and every frame. So a good approach is to apply the segmentation at key frames and reuse the already computed features for the other frames [78], [79]. A new highly efficient and low accuracy neural network based model is developed for semantic video

segmentation called as Temporally Distributed Network (TDNet) [8]. In TDNet feature extraction is distributed evenly across the sequential frames to eliminate the re-computation and then these features are combined together using the Attention Propagation Module (APM) to get the strong features for accurate segmentation [8]. The pictorial representation of the same is described in the Fig 2.8.

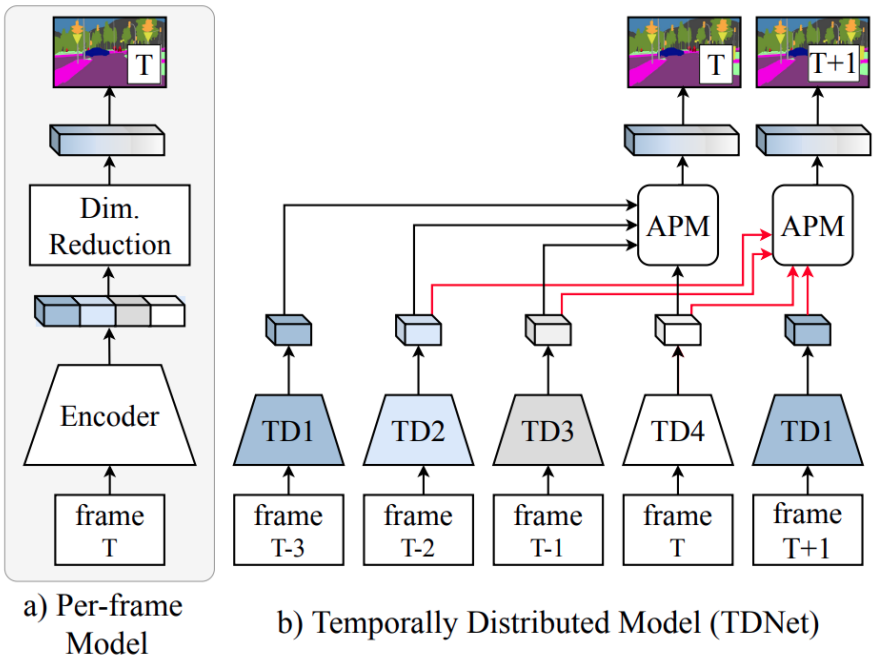


Figure 2.8: TDNet. Courtesy of [8]

2.5 Limitations of Previous Work

The perception system of the modern ADAS uses segmentation to understand the surrounding environment by capturing the surrounding environment with the help of modern cameras. The high FPS data collected by the camera are in continuous sequence where the every frame is related to its previous frames. In general setting segmentation is performed on these frames to understand the object and their boundaries, number of objects, types of objects present in the frame. A work by Hou et al [2] integrate the camera pose data onto the computation of the depth maps, however similar strategy is not studied for a segmentation task. Also study of temporal data fusion in the latent space using LSTM is not studied in any of the previous work. This thesis aims to study the impact of temporal pose data onto the computation of the semantic segmentation and taking the previous frame data onto the current frame semantic segmentation task using LSTM network is studied.

3

Methodology

Methodology section contains the details about the dataset used for performing the experiments, preprocessing of the dataset, and experimental design.

3.1 Dataset

To conduct the experiments Scannet and Virtual KITTI 2 are used. Scannet is a indoor scenes. These dataset contains the continuous video sequence data and explained in details in the below section.

3.1.1 ScanNet ??

ScanNet is a video sequence dataset with 1513 scanes captured in 707 distinct spaces. The dataset contains totally 2.5M RGB-D images. The dataset originally designed for the task of indoor scenes 3D reconstruction including 3D objec classification, semantic voxel labeling and CAD model retrieval **??**, **??**. The dataset contains the details of the 3D camera poses, surface reconstruction, and instance-level semantic segmentation. One sample of the data is shown in the figure

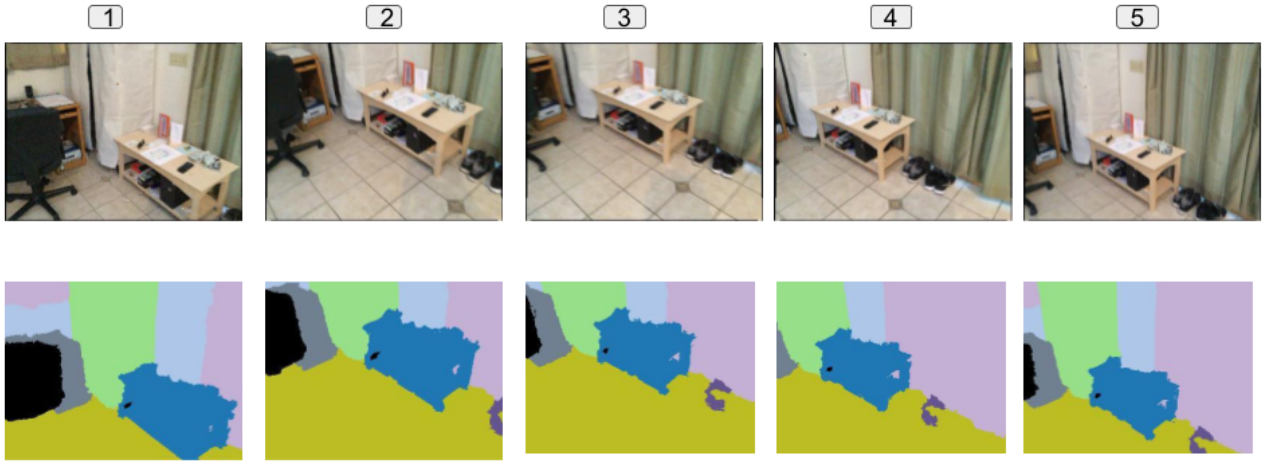


Figure 3.1: Sample of Scannet dataset rgb and semantic label

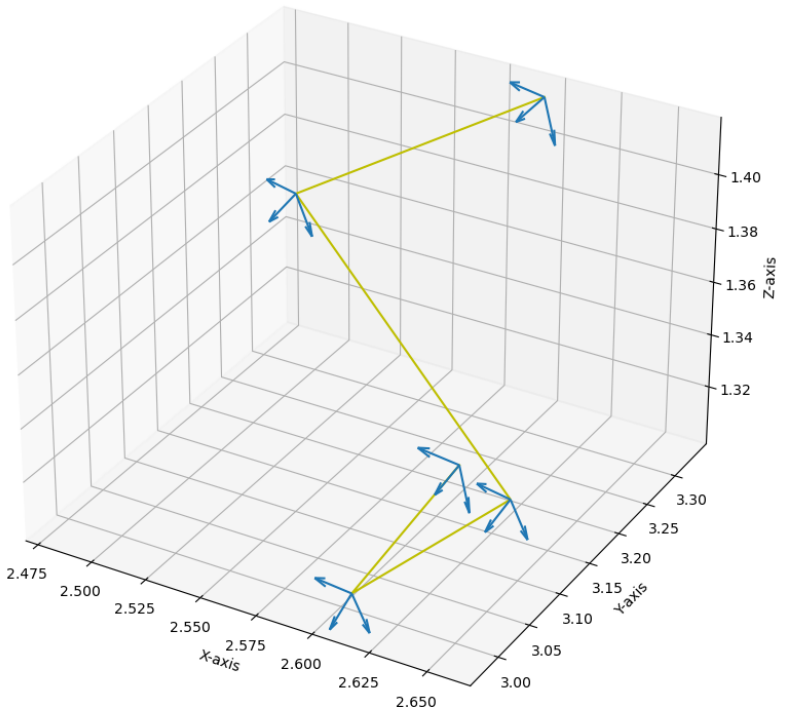


Figure 3.2: Sample of Scannet dataset pose

The 3D representation of the sample data is presented in the figure3.2. There are totally 40 classes in the dataset with different varieties under each categories. The complete list of the classes is presented in the Table B.1, B.2, and B.3. Dataset is collected by 20 users at different countries locations. Since the dataset is huge, a subset of the dataset is taken for performing the experiments. Totally 186 video sequence data is taken for conducting the experiments. These video sequence data are further split into 149 sequence for training the model and 36 sequences for testing. The distribution of the scannet dataset labels is shown in the figure 3.3. There is high number of pixels in the dataset belonging to the Wall, Other, Floor and Chair categories. The dataset represent the indoor scenes with different varieties. Hence experiments are conducted by combining the low pixel distribution class into a single categories to balance the pixel distribution.

3.1.2 Virtual KITTI 2 ??

Virtual KITTI dataset is first of its kind released to the public domain with driving applications. It is a synthetic dataset representing the simulation of the vehicle in a different environmental conditions. And it is a cost effective alternative to the real-world data. Virtual KITTI 2 is the next generation dataset with improved photo-realistic and featured version of the original virtual KITTI dataset. The virtual

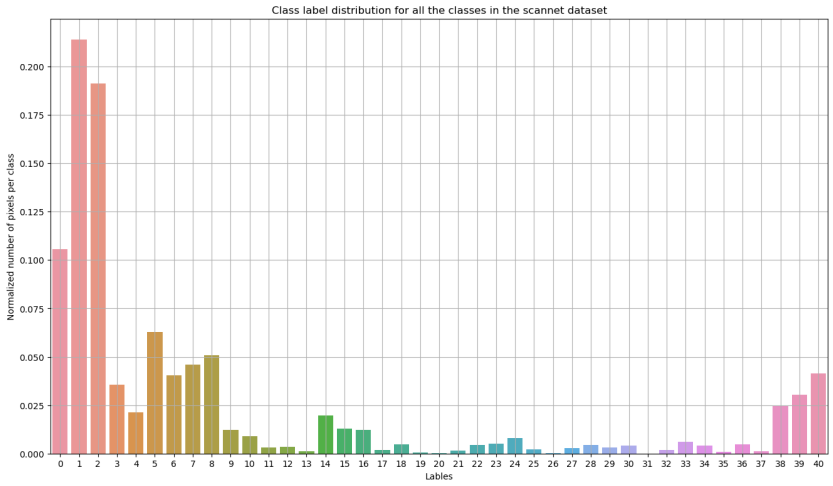


Figure 3.3: Scannet dataset class distribution

KITTI 2 contains the stereo camera views to expand the application areas. The dataset contains same 5 sequences clones with camera 0 representing the same dataset as virtual KITTI and camera 1 is 0.5327m to its right. Each camera contains RGB, class segmentation, instance segmentation, depth, forward and backward optical flow and forward and backward scene flow images. Each sequence contains camera parameters, vehicle color, pose and bounding boxes. One sample from the sequence and corresponding pose representation in the 3D plane is shown in Figure 3.4 and 3.5 respectively.

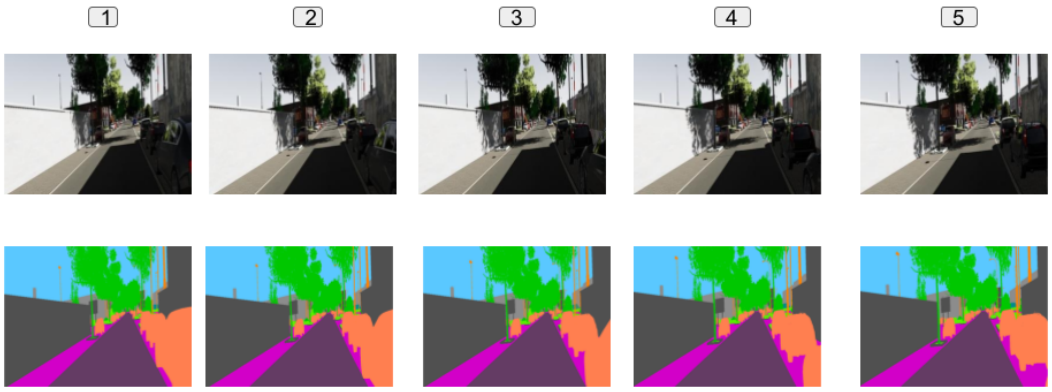


Figure 3.4: Sample of Virtual Kitti 2 dataset

3.2 Data Collection and Preprocessing

The RGB-D scannet dataset can be downloaded from the python script sent to the user from the dataset development group, after submitting the agreement terms. The description and format of the dataset can be found at the github **??**. Totally there are 1500 scans. Any specific scans, file types, can be

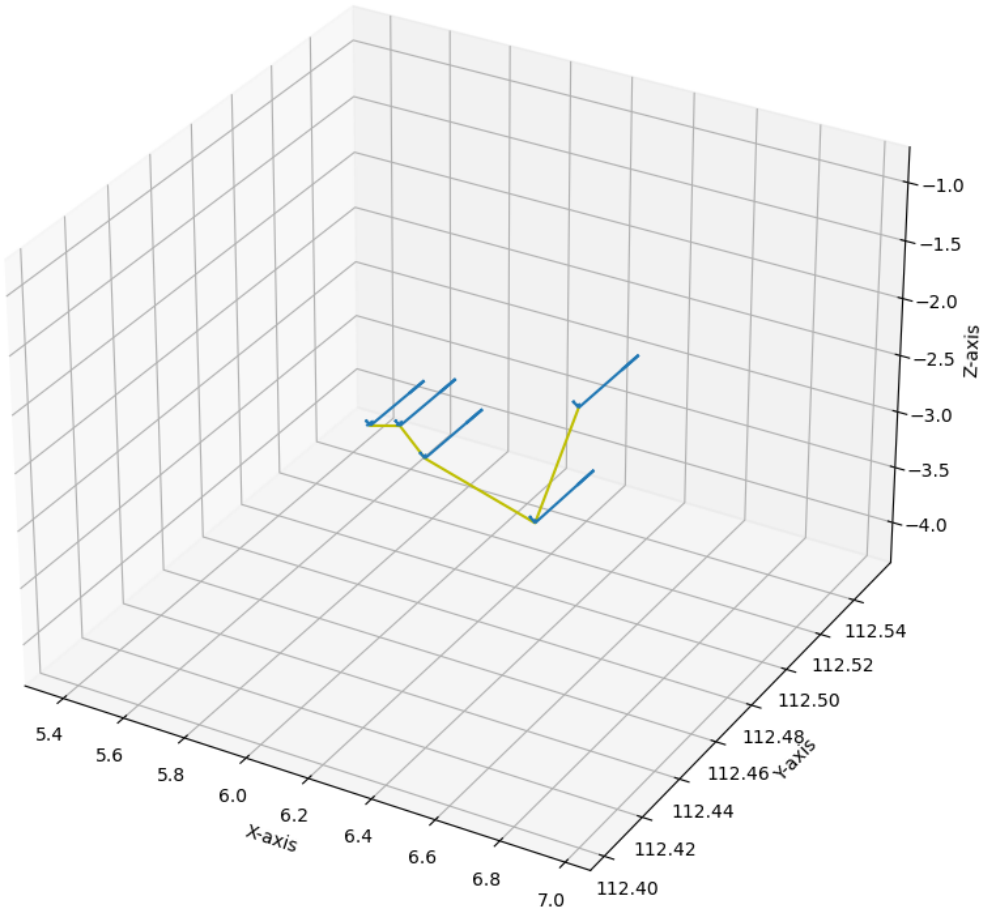


Figure 3.5: Sample of Virtual Kitti 2 dataset

downloaded. There are multiple information regarding a scans in a file. These files are further processed to color, label, pose and depth data for each scans. More details is documented in the github **??** repository. The raw rgb image is normalized before passing it into the model for quick convergence of learning. The input image is a 3 channel rgb data in .jpg format, the label is a single channel image in .png format with 40 classes in total, and pose data in text format. The pose data contains the rotation matrix and transnational vector in homogeneous matrix format. The vkitti dataset is also in the similar format. However, there are total of 15 classes in vkitti dataset. The raw rgb image, label and pose data is shown in the figure 3.6.

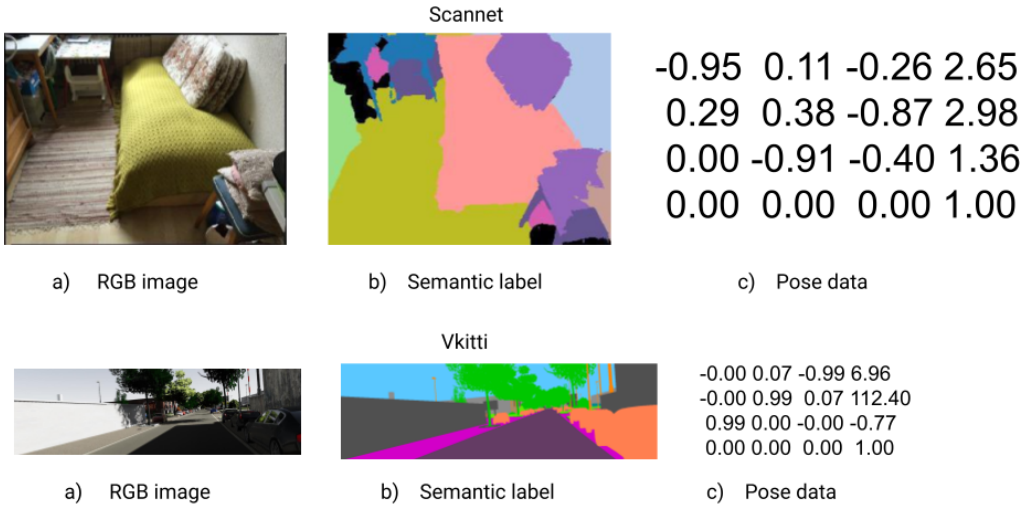


Figure 3.6: RGB, Label and Pose dataset sample of scannet and vkitti data

3.3 Experimental Design

The aim of the experiment is to incorporate temporal fusion in the latent space to understand the impact of past frames information fusion in the future frame segmentation prediction. The Unet model **??** perfectly suites to this experiment. Unet model consist of a encoder and a decoder network with latent space encoding in between. Three types of experiment were conducted with the unet model. In the first type plain vanilla unet model is taken in the second type of experiment unet model with gaussian process, third type is the unet model with Long Short Term Memory (LSTM). In the last two experiment the latent space encoding is subjected to temporal fusion by the gaussian process, and LSTM.

3.3.1 U-Net Vanilla model

U-Net vanilla model is a simple semantic segmentation model, with encoder-decoder architecture. Encoder is a contracting path and decoder is a expanding path. The input image is fed into the encoder, made up of convolutional neural net. The architecture consist of two 3x3 2d convolutional layer followed by a rectified linear unit (ReLU) and batchnorm2d with 2d downsampling Maxpool layer. At every step of the convolutional block the number of out channels/ feature channel is doubled. Starting with 64 features in the first convolutional block output to 1024 feature at the downsampled fifth layer or the latent space encoding. The upsampling layer consist of a 2d convolution with halfed feature map at every step followed by concatenation of the corresponding feature map from the encoder path. The convolution2d consist of two 3x3 convolutions each followed by a ReLU function. At the final layer of the decoder, a 2d convolutional layer was introduced to map the previous convolutional block output to the predicted class labels. Totally there are 23 convolutional block in the entire network.

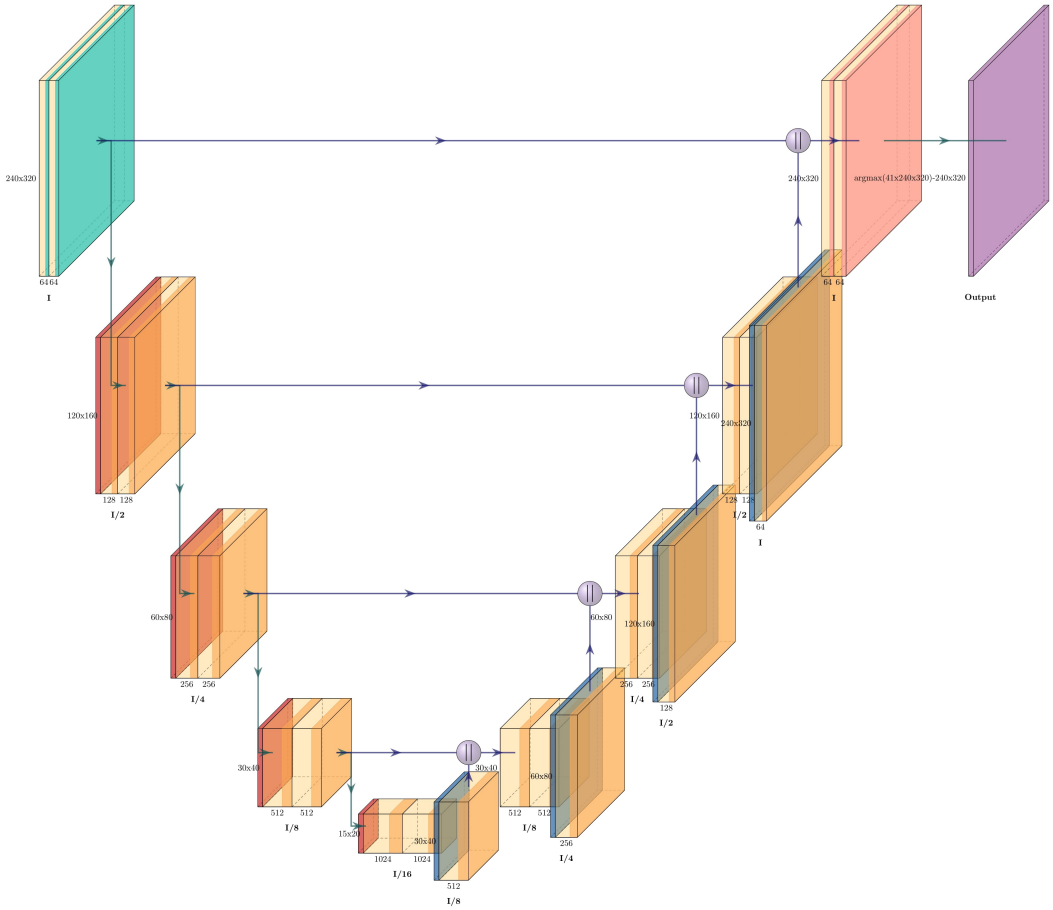


Figure 3.7: RGB, Label and Pose dataset sample of scannet and vkitti data

3.3.2 U-Net with Gaussian process

In the second type of experiment the latent space encoding of the U-Net model is taken and subjected to the gaussian process. The gaussian process make sure that the frames that are close to each other have similar latent space encodings. This is done by building the kernel of the gaussian process with distance matrix. A probabilistic prior on the latent space is defined that accounts for the prior knowledge that poses that are very close have similar latent space encoding than the poses that are very far from each other. This information is encoded in the covariance matrix. The distance measure between the camera poses is calculated with a metric to define the closeness. The distance between the camera poses is calculate with the help of work by Yuxin Hou et al **??** [02] and Mazzotti et al [97] **??**. This measure the distance between the rigid body poses. The pose-distance measure between the two camera poses P_i and P_j is defined in equation 3.1. Where t is the translation vector, R is the rotational vector, I is the identity matrix and $tr()$ is the trace of the matrix.

$$D[P_i, P_j] = \sqrt{||t_i - t_j||^2 + \frac{2}{3}tr(I - R_i^T R_j)} \tag{3.1}$$

The covariance function is defined from this distance matrix $D[P_i, P_j]$. The covariance function is chosen from the Matern class **??**, **??** [97] [98] in equation 3.2.

$$k(P, P') = \gamma^2(1 + \frac{\sqrt{3}D[P, P']}{l}exp(-\frac{\sqrt{3}D[P, P']}{l})) \tag{3.2}$$

The hyperparameter γ^2 and l define the magnitude and length-scale of the processes. Independent GP priors to all values in Z_i, and the output of the encoder is assumed to be the noise corrupted version of the expected latent space encodings. The GP regression model is defined from this initialization settings and defined in the equation 3.4 3.3 [02]. The architecture of the unet model with gaussian process is depicted in the figure 3.8.

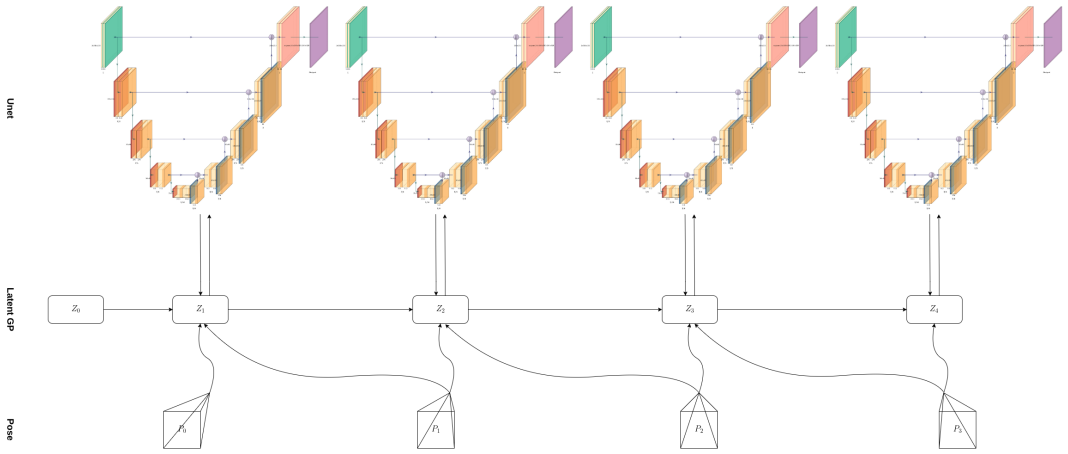


Figure 3.8: RGB, Label and Pose dataset sample of scannet and vkitti data

$$z_j(t) \sim GP(0, k(P[t], P[t'])) \tag{3.3}$$

$$EncoderOutput = z_j(t_i) + \epsilon_{j,i}, \epsilon_{j,i} \sim N(0, \sigma^2) \tag{3.4}$$

3.3.3 U-Net with Long Short Term Memory (LSTM)

In the third type of experiment, the U-Net model is subjected to the temporal fusion with the help of Long Short Term Memory (LSTM) by passing the latent space encoding onto the LSTM convolutional cell. The overlapping information in the consecutive frame data is passed onto the future frame prediction by incorporating the convolutional LSTM cell with the aim of improved performance and model the spatio temporal relations in comparison to the vanilla model. The partial scene geometry information from the previous frame is used in current step frame prediction. A hidden state propagation approach from LSTM

method is used to pass the latent space information. The adaption of temporal fusion with LSTM is inspired from the work of Arda Düzçeker et al **??**. The author introduced a convolutionalLSTM cell in the latent space to learn the shared information between the frames to predict the depth of objects present in a video sequence data. The convolutional LSTM cell is inspired from **??**. The original LSTM version is explained in the article **??**. The hidden state H and cell stae C is initialized to a value. Let X denotes the output of the bottleneck encoder network, then the logic to compute the hidden state and current state can be calculated as below (Courtesy of **??**)

$$i_t = \sigma(w_{xi} * X_t + w_{hi} * H_{t-1}) \tag{3.5}$$

$$f_t = \sigma(w_{xf} * X_t + w_{hf} * H_{t-1}) \tag{3.6}$$

$$o_t = \sigma(w_{xo} * X_t + w_{ho} * H_{t-1}) \tag{3.7}$$

$$g_t = ELU(layernorm(w_{xg} * X_t + w_{hg} * H_{t-1})) \tag{3.8}$$

$$C_t = layernorm(f_t \bigodot C_{t-1} + i_t \bigodot g_t) \tag{3.9}$$

$$H_t = o_t \bigodot ELU(C_t) \tag{3.10}$$

Where $*$ denotes the convolution, $\bigodot$ is the Hadamard product, σ is the sigmoid activation function, w are convolution filter weights. The pictorial representation of the same is presented in the Figure 3.9. Each frame of the sequence video is passed onto the bottleneck U-net model in order and prediction is made for individual frames. The output from the encoder network is passed through LSTM cell. The output from first LSTM cell act as the input to the next LSTM cell computation, thereby it carries the learned information forward from one frame computation to the next frame computation.

3.4 Training and Evaluation Pipeline

Datasets are splitted into training and evaluation set. During the training phase the image, label and pose paths are added into a list in order and then passed onto the dataloader. The image data is read and normalized. Experiment is conducted with different number of classes by merging the classes. This is done in labels processing stage. A model is initialized with different parameter settings. The processed image, label and pose data is passed onto the initialized model. A loss criteria is defined to compute the losses. Loss is computed from the ground truth value and the model prediction. Back propagation from the loss is calculated followed by the optimization step. In the final stage the loss and model is saved. The dataloader loads data in batches. The process is repeated until all the data in the dataloader iterate through entire data once. The entire process is repeated for different number of epochs. Model performance is monitored during each run of the epoch and stopped once the loss reaches a small values and model fits to the training data.

During the evaluation process the evaluation dataset path is stored in the list. The individual list contains Image, Label and Pose path information. Since the dataset is a continuous sequence data, the path of the data stored in the list are in order. In most of the cases there is a overlapping information

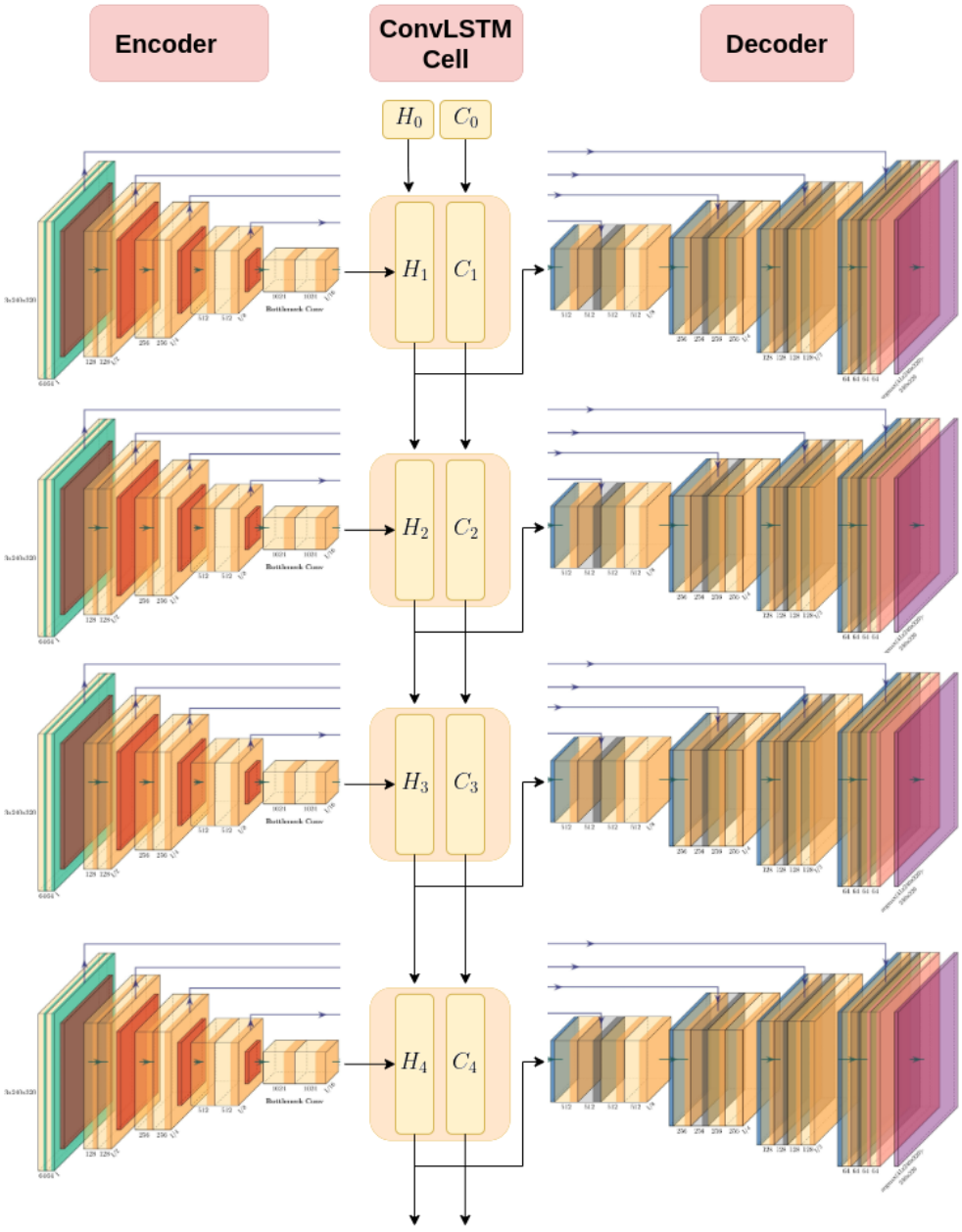


Figure 3.9: Unet with ConvLSTM cell

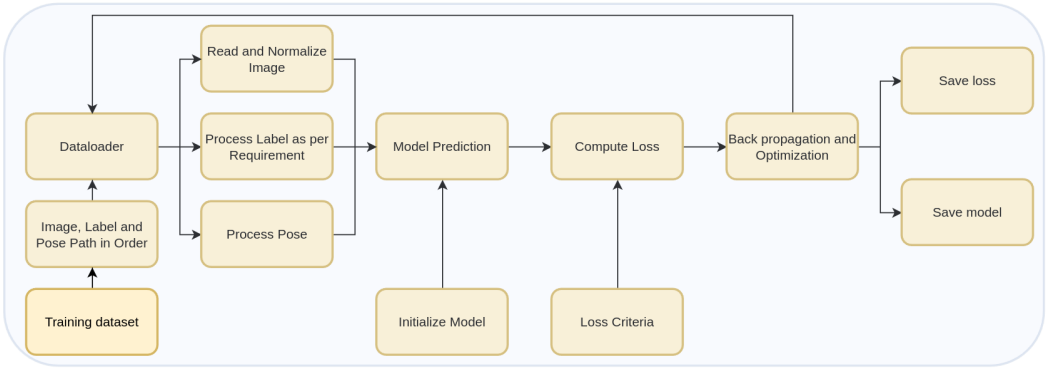


Figure 3.10: Training pipeline

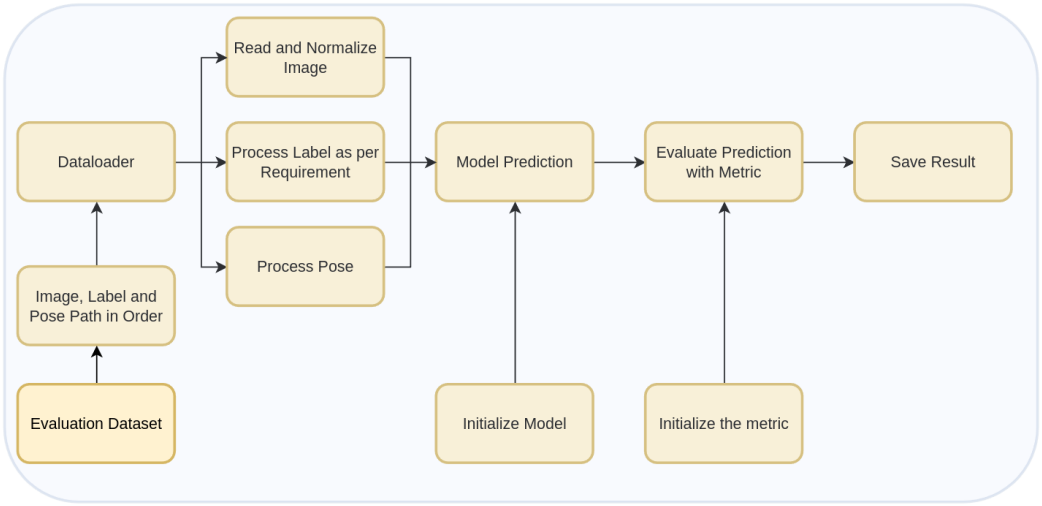


Figure 3.11: Evaluation pipeline

between the consecutive frames. In the evaluation dataset are passed onto the dataloader. Images are normalized from this dataloader and labels are processed as per the requirement set by the experiments. Pose information is in text format. The text data is read and passed to a list and returned by the dataloader. The trained model is loaded and the processed data from the dataloader is passed onto the model. The predicted output from the model is evaluated against the evaluation metric and the results are saved.

3.5 Hardware Configuration

NVIDIA GeForce RTX 3070 Laptop GPU, 8192 MB Tesla T4, 12680 MB, Google Colab Nvidia Tesla V100 PCIe GPU with 5120 Cuda cores and 640 Tensor cores, 16 GB HBM2 memory University cluster

4

Evaluation and Experimental Result

Evaluation and experimental result chapter contains the metrics used to evaluate the conducted experiment and the discussion on the research questions. Results of research question is answered in the subsequent sections with listing of the result in a table. Three research questions answered in the section are the results of state of the art temporal fusion techniques, How are the results in comparison to each other and finally cross transfer the temporal fusion techniques from depth estimation to the semantic segmentation. The model is trained and evaluated on three datasets Scannet [80], Virtual kitti [81] and VIODE [82] datasets.

4.1 Evaluation Metric

The proposed model needs to be validated to understand the impact of the newly trained model. To validate the model different evaluation metrics are proposed. The description of the proposed model is listed below.

4.1.1 Pixel Accuracy

Pixel accuracy is commonly defined as percent of pixels in a image correctly classified into a particular class. Accuracy is defined as below

$$\textbf{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

where

TP = True Positive

TN = True Negative

FP = False Positive

FN = False Negative

Per class mean pixel accuracy (mPA)

Per class mean pixel accuracy is the average of pixel accuracies of all the classes.

Pixel accuracy (PA) and Mean pixel accuracy (mPA) can also be defined as below [83]

$$\textbf{PA} = \frac{\sum_{j=1}^{k} n_{jj}}{\sum_{j=1}^{k} t_j}$$

$$\textbf{mPA} = \frac{1}{k} \frac{\sum_{j=1}^{k} n_{jj}}{\sum_{j=1}^{k} t_j}$$

where n_{jj} is the total number of pixels both classified and labeled as class j.

4.1.2 IoU

Intersection over Union (IoU) also known as the Jaccard index is a method to quantify the overlapping between the target mask and the predicted output. In other words, it is number of pixels common between the target and prediction masks by the total number of pixels exist between both the masks [84]. pictorial representation of the same is presented in Fig 4.1

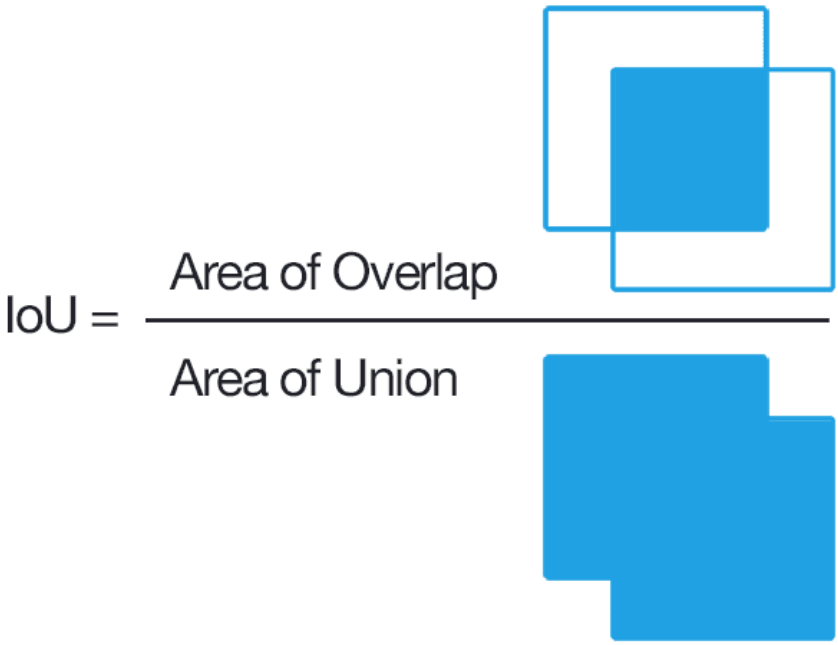


Figure 4.1: IoU. Courtesy of [9]

IoU is calculated for each class separately and then averaged over all the classes to provide mean IoU.

$$\textbf{IoU} = \frac{\sum_{j=1}^{k} n_{jj}}{\sum_{j=1}^{k}(n_{ij} + n_{ji} + n_{jj})}, i \neq j$$

where n_{ij} is the pixels which are labeled as class i but classified as class j and n_{ji} is the total number of pixels labeled as class j, but classified as class i. [83]

$$\textbf{mIoU} = \frac{1}{k} \sum_{j=1}^{k} \frac{n_{ij}}{n_{ij} + n_{ji} + n_{jj}}, i \neq j$$

Frequency weighted IoU (FwIoU). It is a metric derived from the mIoU which weighs each class importance depending on appearance frequency using t_j [83]

$$\textbf{FwIoU} = \frac{1}{\sum_{j=1}^{k} t_j} \sum_{j=1}^{k} t_j \frac{n_{jj}}{n_{ij} + n_{ji} + n_{jj}}, i \neq j$$

4.2 Hypothesis

Semantic segmentation is a vital task in the computer vision domain that assigns labels to every pixel in an image. Semantic segmentation is applied in 2D images, Video data, and 3D or Volumetric data. The video sequence data contains multiple image frames stacked together. Given the high frame rate of the video, the consecutive frames are related to each other due to the overlapping regions in the successive frames. In most semantic segmentation tasks, the segmentation is performed on each frame or the keyframes. Past information can be fused into current step computation by taking the learned features from previous frames [8].

The work hypothesizes that the performance improves by fusing the past information onto the current frame segmentation computation by the following approaches.

(i) Subjecting the latent space encoding of the Unet model to the Gaussian process by taking the pose of the current frame and the previous frame, the performance improves in comparison to segmentation without temporal fusion since the overlapping information present in the past is used for computation of the current frame segmentation.

(ii) Placing the ConvLSTM cell at the latent space encoding of the Unet model to transfer the geometry information from the past to the current step helps improve semantic segmentation performance.

The approach is cross-transferred from the depth estimation task [2] [13].

4.3 RQ1: What are the works on state-of-the-art temporal fusion in semantic segmentation?

Segmentation is the process of assigning pixel labels for a given image. Segmentation can be observed in different context such as Video instance segmentation (VIS), Semantic segmentation of the key frames

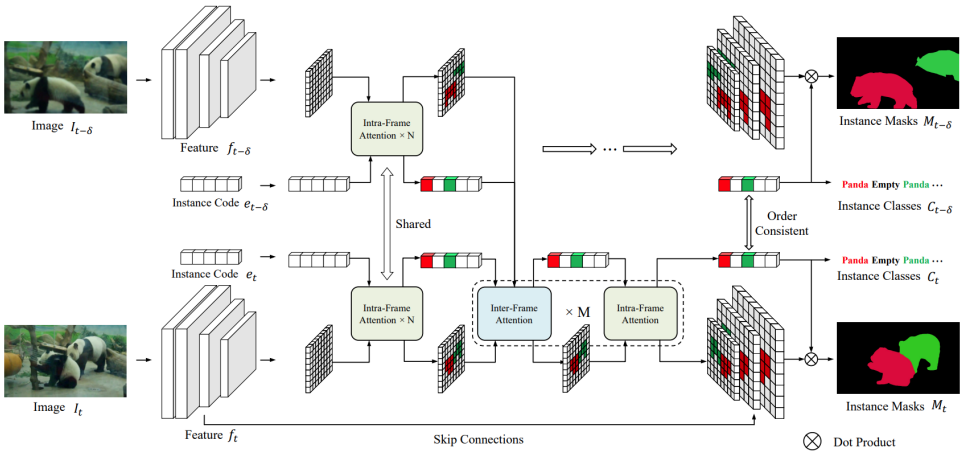


Figure 4.2: VIS proposed framework. Courtesy of [87]

of a video sequence, Video panoptic segmentation, MRI image segmentation, autonomous driving scene understanding. Youtube - Video Instance Segmentation (VOS), Scannet are the common datasets in the semantic segmentation domain.

Youtube VIS data Youtube - Video Instance Segmentation (VIS) is a dataset that extends the image instance segmentation from the image to the video domain. The dataset aims to solve the problem of simultaneous detection, segmentation and tracking problem. Scannet is an RGB-D video dataset containing 2.5 millions views with more than 1500 scans. A paper by Xiang Li et.al [87] explains the temporal fusion for online video instance segmentation. The author introduced the concept of an online video framework with novel aware temporal fusion method. A cropping free temporal fusion approach to model the temporal consistency between video frames. A bottom-up online transformer based network is used to solve the VIS problem. A transformer layer is introduced in the convolutional neural network (CNN) to include the instance information. A attention layer between the frames used to extract the instance code for the current frame. A skip connection is used to use low level contextual information and use dynamic convolution to generate the segmentation map. CNN feature map and the latent code combined together helps to jointly represent instance aware features. A instance code is a LxD vector to VIS task, where L is the maximum detected instances number in a frame D is the feature instance for each instance. Inter and intra frame attention module for fusing the temporal information. The network architecture is presented in Fig 4.2. Three types of attention code-to-code(c2c), code-to-pixel(c2p), and pixel-to-code(p2c) used to construct the relation between the instance code and feature map. Inter frame used to build the temporal correlation and combine contextual features across frames. Inter frame c2c, c2p and p2c are used to construct the temporal information.

Miran Heo et al [88] worked on the video instance segmentation via Object Token Association (VITA). The work effectively understand the video by the object centric tokens. The VITA model requires a frame-level detector. The frame level detector localizes instances using masks without the need of bounding boxes. Two features is generated for the frame-level predictions specifically: a) dynamic 1x1 convolution

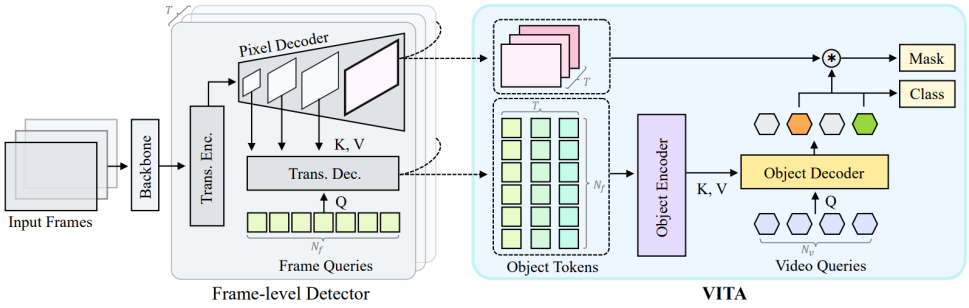


Figure 4.3: The frame level detector take frame queries and mask features and generate the embeddings and pass onto the VITA model for mask prediction. The object-aware knowledge in the spatial scenes is captured by construction of the temporal interactions between the frame queries. Finally mask trajectories are obtained form the VITA model. Courtesy of [88]

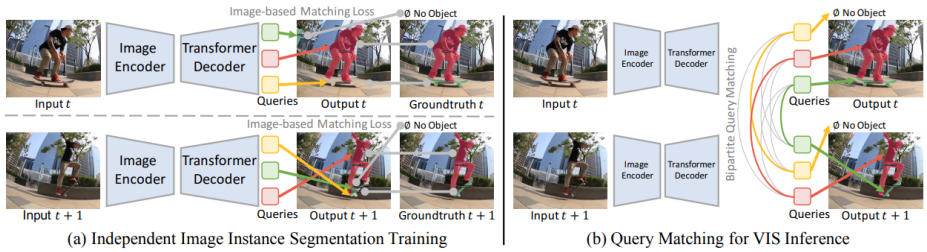


Figure 4.4: (a) MinVIS trained on query based segmentation individually for every frame. (b) Inference of the video instance segmentation from the segmented image using bipartite matching of th query embeddings. Courtesy of [89]

weight from the frame queries f b) per-pixel embeddings from the pixel decoder. Dot product between the two embeddings are taken in the frame-level predictions. The end-to-end video instance segmentation method VITA is divided into three stages. Firstly, VITA works with the frame-level detector in a manner that is frame independent. No computation between the frames are involved. The frame queries that holds the object centric information are collected throughout the video sequence and object encoder is used to build the video level information between the frames. And thirdly the decoder combines the information from the frame and video queries which is finally used to find the object mask in the video frames.

A minimal video instance segmentation (MinVIS) by De-An Huang et al [89] does not require video based training and can be applied directly to image containing sparse image instance segmentations annotations. MinVIS is a two stage approach: 1) Independent image instance segmentation on each frame 2) Instance association between the frames. Image Instance Segmentation has three unique main components a)Encoder to extract features from the image b) Transformer decoder, used to process the output of image encoder to update the query embeddings c) Prediction head takes the query embeddings to predict the output. Association between the instance segmented frames is calculated by bipartite matching of the respective query embeddings.

Mask classification in image segmentation problem is solved by the state of the art segmentation

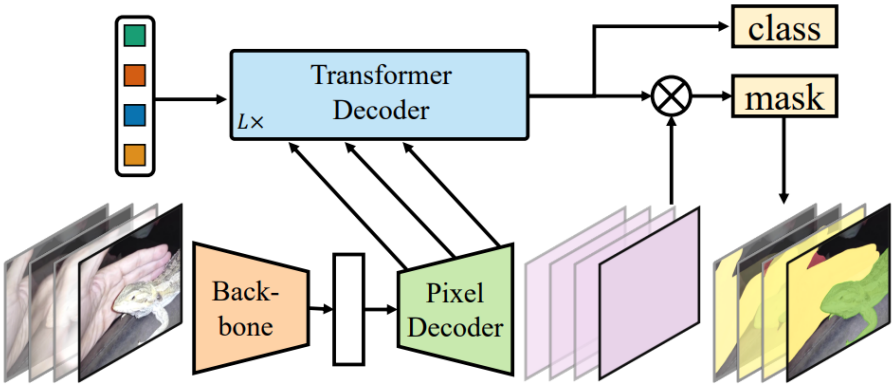


Figure 4.5: A mask2former with video instance segmentation. Courtesy of [90]

models. B.Cheng et al [90] proposed a method to process directly the video rather than the image. This is achieved by feeding the Mask2Former with 3D spatio-temporal features and predict the 3D volume to track each object instances across time. For data with single frame the Mask2Former works as a general image segmentation model. For data with more than 1 frame the model segments and tracks instances across frames.

4.4 RQ2: How are the results from RQ1 compared with each other to perform temporal fusion?

All of the above mentioned approaches are trained and tested on the Youtube VIS datasets. The performance of the models are tabulated in table 4.1.

Method	AP	AP50	AP75	AR1	AR10
HIATF for VIS 2019 [87]	41.3	61.5	43.5	42.7	47.8
VITA ResNet-50 2019 [88]	49.8	72.6	54.5	49.4	61.0
VITA ResNet-101 2019 [88]	51.9	75.4	57.0	49.6	59.1
VITA Swin-L 2019 [88]	63.0	86.9	67.9	56.3	68.1
VITA 2021 [88]	45.7	67.4	49.5	40.9	53.6
Min VIS 2019 [89]	61.6	83.3	68.6	54.8	66.6
Min VIS 2021 [89]	55.3	76.6	62.0	45.9	60.8
Mask2former 2019 [90]	60.4	84.4	67.0	-	-
Mask2former 2021 [90]	52.6	76.4	57.2	-	-

Table 4.1: A normal caption

4.5 RQ3: How to cross-transfer the temporal fusion technique to semantic segmentation?

U-net model is developed to categorize the biomedical images. Currently the model is used in the other application areas as well. The original network is mainly based on the data augmentation strategy. The network consist of a encoder and decoder along with the interconnection between different layers of encoder and decoder. The expanding path of the U-net model helps to capture the location of the objects in the image. The vanilla U-net model code is referenced from the Kaggle competition [85]. The result presented below is for vanilla U-net model trained on the scannet dataset [80]. The model is trained for 300 epochs. The plain vanilla model is trained with a neural network based unet model. Experiment is conducted in three ways

- Considering all the classes
- Containing only two classes by combining the low pixel distribution classes together (Wall and Other)
- Containing three classes by combining classes with low pixel distribution (Wall, Furniture, and Other)

The parameter used to train a plain unet model is tabulated in **??**.

4.5.1 Combining scannet dataset classes for experiment

There are 41 classes [B.1 , B.2 , B.3] present in the entire Scannet dataset.

To conduct experiment 185 indoor video sequence data is considered. The distribution of the entire scannet data classes is presented in the Fig 4.6. Out of 185 sequences 149 sequences are taken for training and the remaining 36 sequences for testing. The distribution of pixels in the training data and testing data represented in Fig 4.7.

The pixel distribution of scannet dataset classes are not uniform. Experiments were conducted in three ways, in first approach all the classes were considered for conducting the experiment and no combining of the classes. In the second approach all the classes other than wall class is combined together to assigned a class label of ”zero” and the ”Wall” class is chosen as the second class. The distribution of the pixels is depicted in the Fig 4.8. The Wall, Furniture and Other classes combined together have high number of pixel distribution, hence all the furniture class are combined together as class 2nd, Wall as the 1st classes and remaining classes are combined together to 0th class as Other classes, resulting in three unique classes. The distribution of the three classes is presented in the Fig 4.9.

4.5.2 Distance and Kernel matrix

A distance matrix is a square matrix that represent the distance between the pair of objects. Distance matrix can be constructed from the pose information of the captured video sequence. Closer the two

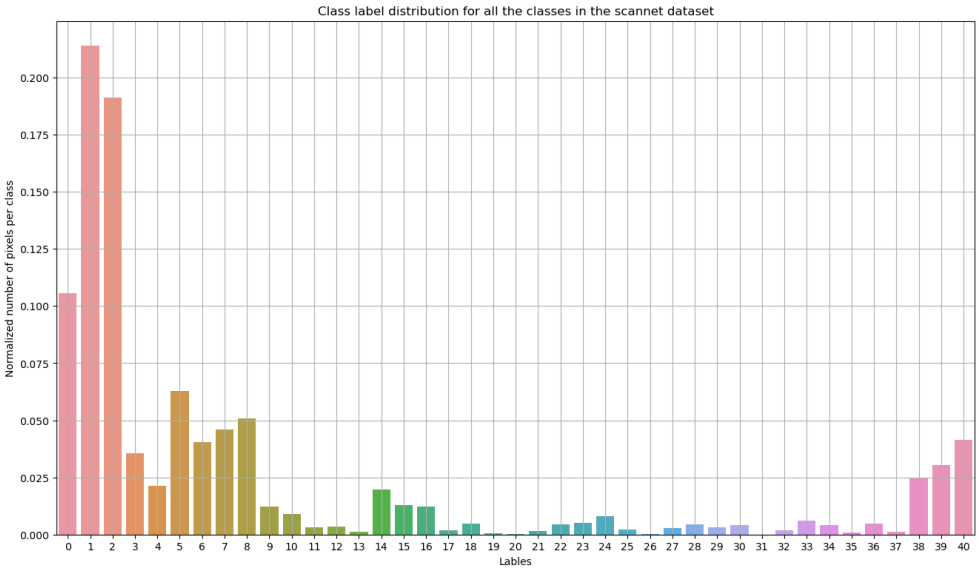
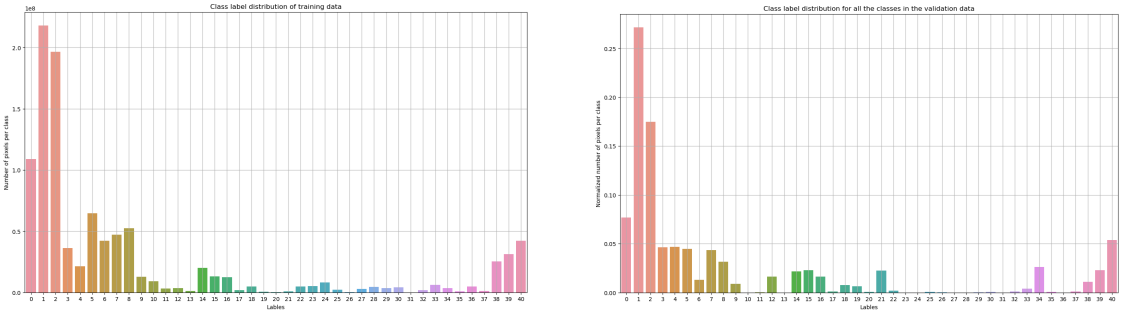


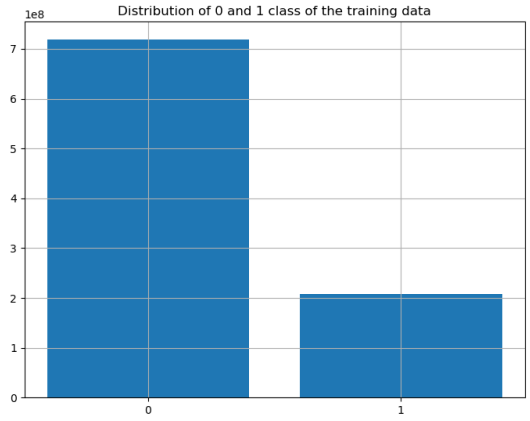
Figure 4.6: Per class pixel distribution of the entire scannet dataset



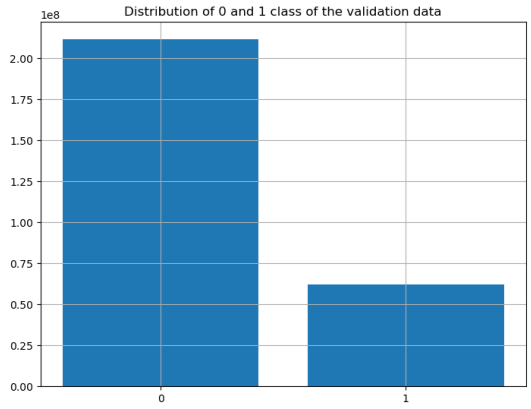
(a) Per class pixel distribution of the training dataset pixel class label

(b) Per class pixel distribution of the testing dataset pixel class label

Figure 4.7: Pixel distribution for the scannet data containing all the classes

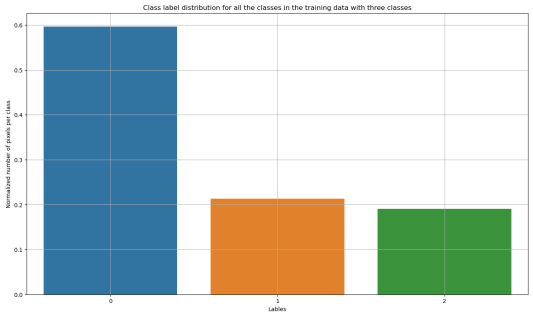


(a) Per class pixel distribution of the training dataset pixel class label

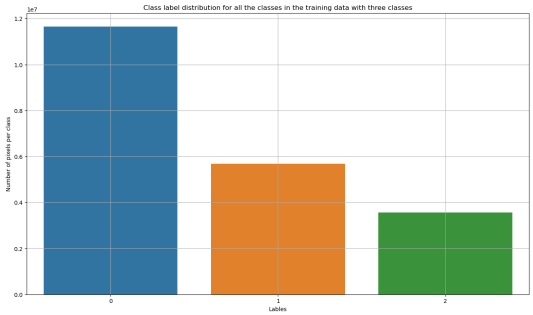


(b) Per class pixel distribution of the testing dataset pixel class label

Figure 4.8: Pixel distribution for the scannet data for two classes



(a) Per class pixel distribution of the training dataset pixel class label



(b) Per class pixel distribution of the testing dataset pixel class label

Figure 4.9: Pixel distribution for the scannet data for three classes

frames are in a sequence the distance between them is smaller, same can be represented with the matrix plot. Two types of distance matrix is plotted in the Fig 4.11 and 4.12. In the first type of plot, a ordered frames pose is represented in the x and y axis. The distance of pose with respect to itself is zero and as the frame number is increased the distance keeps increasing. In the second type of unordered pose plot the frames pose are shuffled and presented as the matrix plot. These pictures represent the distance of one frame to the other.

(a) Ordered set of images

(b) Unordered set of images label

Figure 4.10: Ordered and Unordered set of images

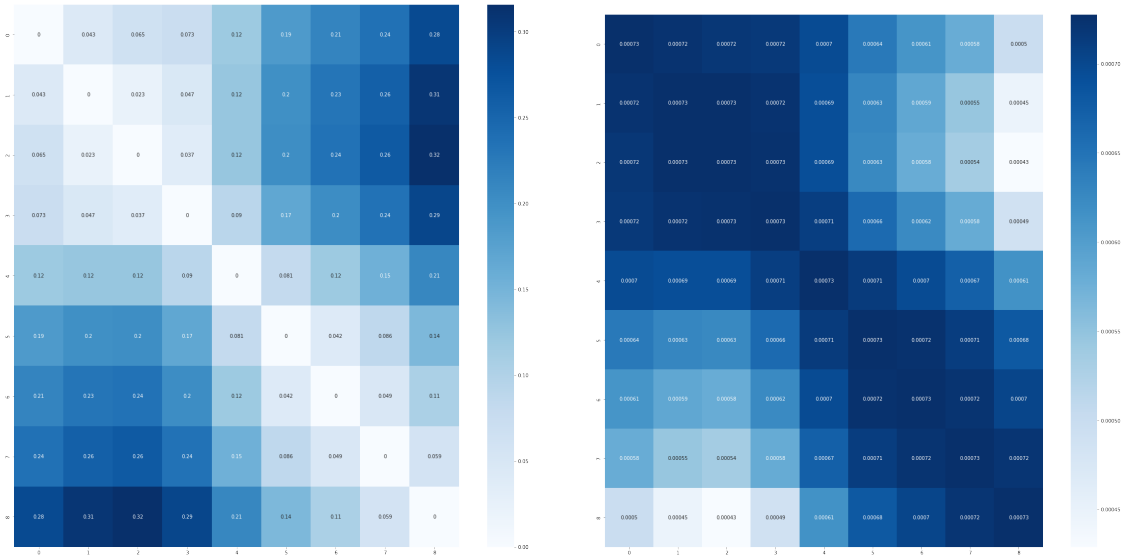
The distance matrix is used to construct the kernel of the Gaussian process. Kernel represent the similarity or correlation between the two data points. Closer data points in the distance space have high correlation and further away data points have low correlation, and same can be observed in the plotting of the kernal matrix as heatmap in Fig 4.11 and 4.12. The corresponding ordered and unordered frames images are presented in 4.10.

4.5.3 Experiment1.1: U-Net Vanilla, GP and LSTM model considering all the classes

Below sections explains the results of the conducted experiments. The model is trained and tested with the scannet dataset without merging the classes. The models performance is evaluated with Pixel accuracy, Mean pixel accuracy, meanIoU, IoU and FwIoU metrics.

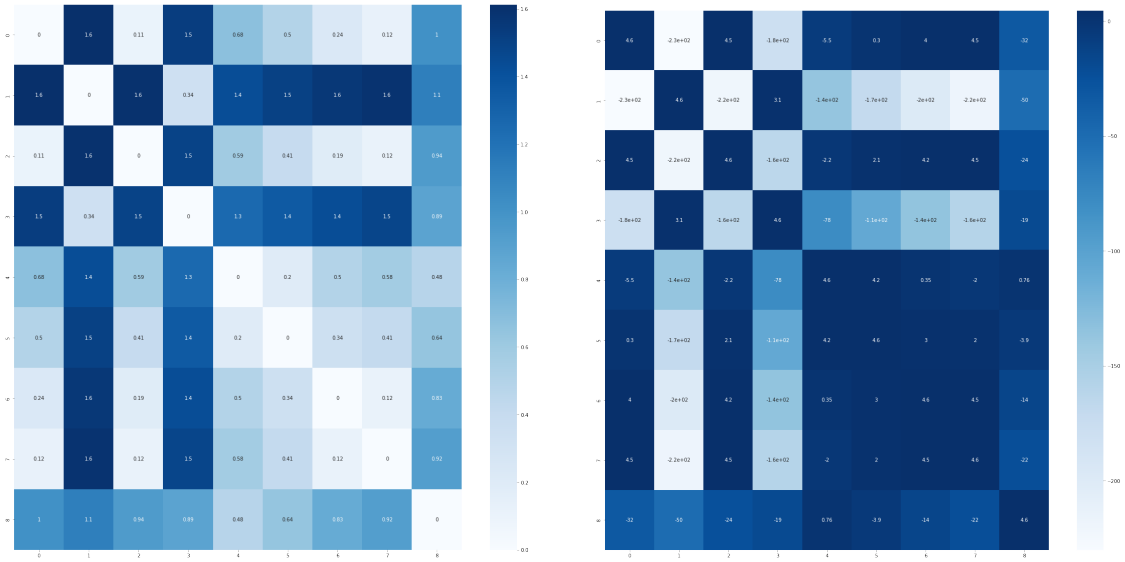
Experiment1.1.2: U-Net vanilla model

The pixel accuracy is 50.96% for the validation dataset. The is due to the presence of large number of classes with unequal pixel distribution. However, the IoU for the Wall, Floor and Mattress is high at 0.56, 0.608, 0.382 due to the high pixel distribution for these classes. Frequency weighted IoU take into account the unequal pixel distribution and the values stand at 0.3531. The predicted and the ground truth pixel distribution is presented in the Fig 4.13. From the bar plot it is evident that predicted "Wall" class distribution is high compared to the ground truth. The Wall and Floor class distribution is similar as the baseline and predicted class labels.



(a) Distance matrix depicted as heatmap (b) Kernel matrix depicted as heatmap label

Figure 4.11: Distance matrix and Kernel matrix for ordered set of images

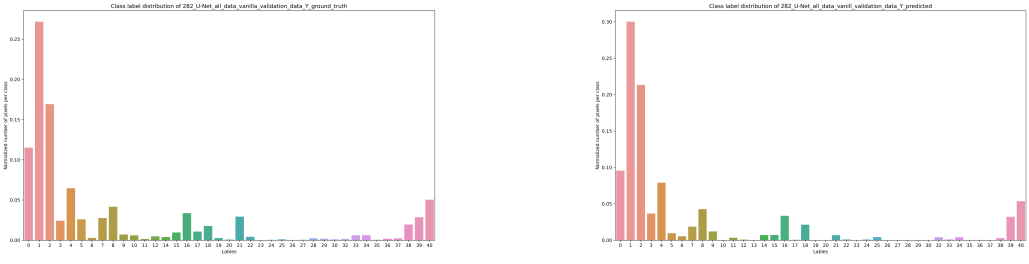


(a) Distance matrix depicted as heatmap (b) Kernel matrix depicted as heatmap label

Figure 4.12: Distance matrix and Kernel matrix for unordered set of images

Metric	Value
Pixel Accuracy	0.5096
Pixel Mean accuracy	0.1907
meanIOU	0.1102
IoU	[1.8345e-01, 5.6047e-01, 6.0881e-01, 1.6476e-01, 3.8241e-01, 1.1697e-01, 2.3122e-02, 8.9410e-02, 2.5848e-01, 1.6661e-01, 2.0180e-03, 2.0504e-01, 4.5985e-02, nan, 4.2167e-02, 1.1143e-01, 2.3969e-01, 1.2545e-02, 1.1324e-01, 2.7658e-03, 0.0000e+00, 7.0415e-02, 7.0606e-02, 0.0000e+00, 0.0000e+00, 1.2208e-01, 1.4680e-02, 1.1862e-04, 2.2112e-04, 6.5610e-03, 4.3742e-03, nan, 7.6680e-02, 3.5784e-02, 1.1516e-01, 5.6912e-02, 2.9310e-04, 3.7764e-02, 2.2634e-02, 1.1269e-01, 2.2094e-01]
FwIoU	0.3531

Table 4.2: A normal caption



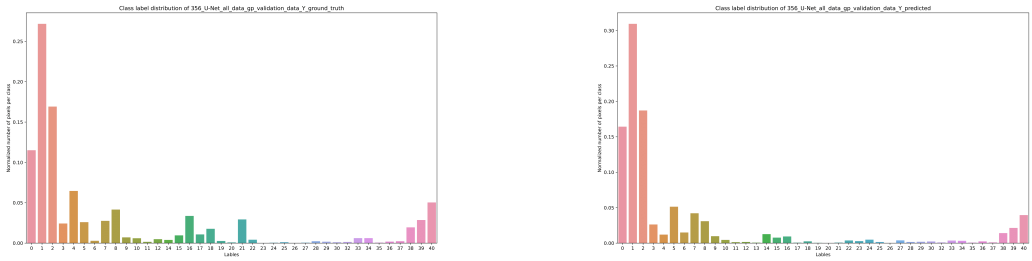
(a) Per class pixel distribution of the ground truth pixel class label

(b) Per class pixel distribution of the predicted pixel class label

Figure 4.13: Pixel distribution for the ground truth and predicted scannet data for vanilla unet model

Metric	Value
Pixel Accuracy	0.5184
Pixel Mean accuracy	0.1679
meanIOU	0.1161
IoU	[1.7087e-01, 5.1271e-01, 5.9998e-01, 2.1256e-01, 4.1160e-01, 1.5834e-01, 3.8634e-02, 2.3669e-01, 1.6056e-01, 1.1568e-01, 7.9677e-02, 1.0454e-02, 2.4003e-02, 0.0000e+00, 1.2199e-01, 4.3193e-02, 3.3956e-01, 6.6473e-02, 1.4712e-01, 2.8003e-03, 6.0475e-05, 2.6127e-01, 5.7962e-02, 0.0000e+00, 3.4611e-04, 1.6519e-02, 0.0000e+00, 4.3417e-04, 4.4221e-02, 6.6478e-03, 1.2108e-02, nan, 5.3272e-02, 5.8480e-02, 2.2352e-01, 4.2175e-02, 8.4644e-02, 1.1630e-04, 4.9106e-02, 1.0338e-01, 1.7687e-01]
FwIoU	0.3497

Table 4.3: A normal caption



(a) Per class pixel distribution of the ground truth pixel class label for gp model

(b) Kernel matrix depicted as heatmap label

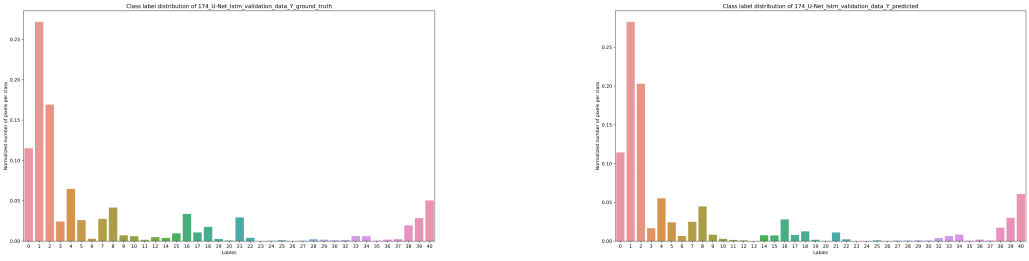
Figure 4.14: Per class pixel distribution of the predicted pixel class label for gp model

Experiment1.1.2: U-Net gp model

In the gp model experiment the information from the previous frame is passed onto the current frame segmentation computation. The accuracy is slightly better than the vanilla model with a increase of 1% with the temporal fusion GP prior integration. The latent space encoding is subjected to the Gaussian process and then passed onto the decoder for segmentation computation. The mean pixel accuracy is similar to the vanilla model with not much improvements. The presence of large number of classes is causing the drop in the performance. As expected ”Wall”, ”Floor”, and ”Furniture” class have high IoU values due to the presence of high pixel distribution.

Metric	Value
Pixel Accuracy	0.5426
Pixel Mean accuracy	0.3509
meanIOU	0.1411
IoU	[0.1690, 0.5573, 0.5949, 0.1428, 0.1170, 0.1984, 0.1930, 0.1278, 0.2010, 0.0778, 0.0071, 0.0514, 0.0216, 0.0070, 0.0426, 0.0610, 0.1063, 0.0020, 0.0907, 0.0017, 0.0000, 0.0114, 0.0422, 0.0015, 0.0380, 0.0209, 0.0000, 0.0362, 0.0710, 0.0010, 0.0182, 0.0000, 0.0174, 0.1382, 0.0534, 0.0077, 0.0558, 0.0016, 0.0646, 0.1776, 0.1697]
FwIoU	0.7643

Table 4.4: A normal caption



(a) Per class pixel distribution of the ground truth pixel class label for lstm model

(b) Kernel matrix depicted as heatmap label

Figure 4.15: Per class pixel distribution of the predicted pixel class label for lstm model

Experiment1.1.2: U-Net lstm model

The scene geometry from the past frame is passed onto a current frame in a efficient manner with the lstm model. The ConvLSTM cell at the latent space compress and store the past information in its states. The change in the viewpoint is captured by the hidden state and passed onto the next hidden state for the computation of the segmentation map. From the result table **??** it is evident that the accuracy has improved by 6% and 5% in comparison with the vanilla and gp models respectively. Mean accuracy has increased by a quite a big margin. The IoU values are also improved by a large margin. IoU for the "Wall" class is highest with 0.61 in comparison with vanilla and gp approach. However, for the cabinet class gp performed well. Table, bookshelf, Counter, desk class gp outperformed the vanilla and lstm model.

Accuracy, meanaccuracy, meanIoU, FWIoU comparison is plotted as the bar graph in 4.16. The accuracy is highest for the lstm model in comparison with the vanilla and gp. However, the mean accuracy is lowest for the gp model with respect to other models. It is evident from the experiment that by incorporating the temporal fusion in the semantic segmentation for the continuous sequence data the performance improves. The overlapping and the past information are learned, stored and passed onto to the computation of the future frames semantic segmentation. The temporal dependency between the frames are exploited to efficiently perform the segmentation by taking pose into account in Gaussian

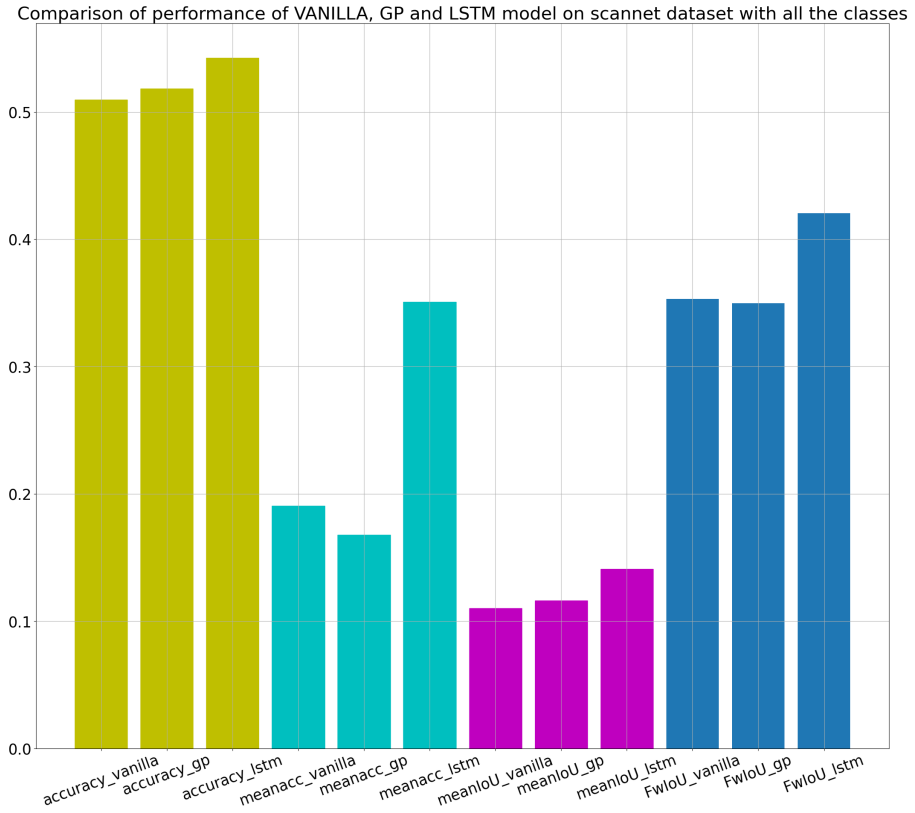


Figure 4.16: Comparison of VANILLA, GP and LSTM model performance based on metric

process and compressed information in the hidden state of the ConVlstm cell. The Gaussian process works in the fact that if the two frames are very close to each other, represented by the pose information, there is a high probability of overlapping regions in the consecutive frames. Output of the decoder is based on the input to the decoder. The output of the encoder is passed onto the gaussian process, it takes into account the current and previous pose information and outputs a latent encoding based on the distance between the current and the previous frame. Hence, if two frames are close to each other then the latent output of the Gaussian process are similar resulting in information transfer from the past learnt latent representation to the current latent output resulting in improved performance. The comparison of the raw RGB, ground truth and predicted segmentation map is shown in Fig 4.17. It can be observed from the figure that Vanilla and LSTM model performed reasonably well in comparison with the GP model. However, overall metric wise LSTM out performed vanilla and gp.

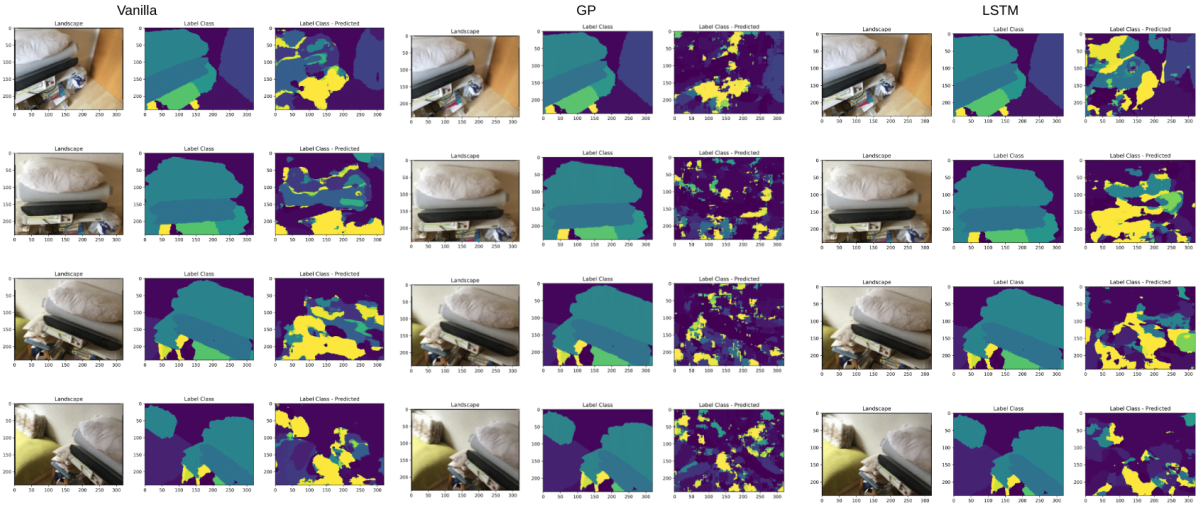


Figure 4.17: Comparison of VANILLA, GP and LSTM model performance

4.5.4 Experiment1.2: U-Net Vanilla, Temporally fused gp model and LSTM model considering two classes

In the second experiment on the scannet dataset all the classes are combined together except the wall class resulting in wall and other class. Table 4.5 compares the results of vanilla, GP and LSTM model performance. In this case the pixel accuracy is similar in case of vanilla and the lstm model, however the accuracy dropped for the gp model. Similar observation can be made with the mean accuracy. However the meanIoU value is high for the lstm model at 0.28 and vanilla and gp produced a result of 0.14 and 0.23. The frequency weighted intersection over union is high for the lstm model followed by vanilla and gp model. Comparison of model output is presented in the Fig 4.18. In this case the wall and other categories in the test image are segmented reasonably well by the LSTM model. The metric bar comparison is shown in the Fig 4.19. It is evident from the plot that the mean accuracy is good for the vanilla and gp and lstm performance is below the other models. However, the meanIoU and FWIoU is high for the lstm model.

Metric	Vanilla	GP	LSTM
Pixel Accuracy	0.4342	0.3764	0.8549
Pixel Mean accuracy	0.6232	0.5625	0.7837
meanIOU	0.145	0.2318	0.6593
IoU	[0.2737, 0.2809]	[0.2377, 0.2259]	[0.8280, 0.4903]
FwIoU	0.2793	0.2284	0.7643

Table 4.5: Performance of Vanilla model with respect to different metric and two classes

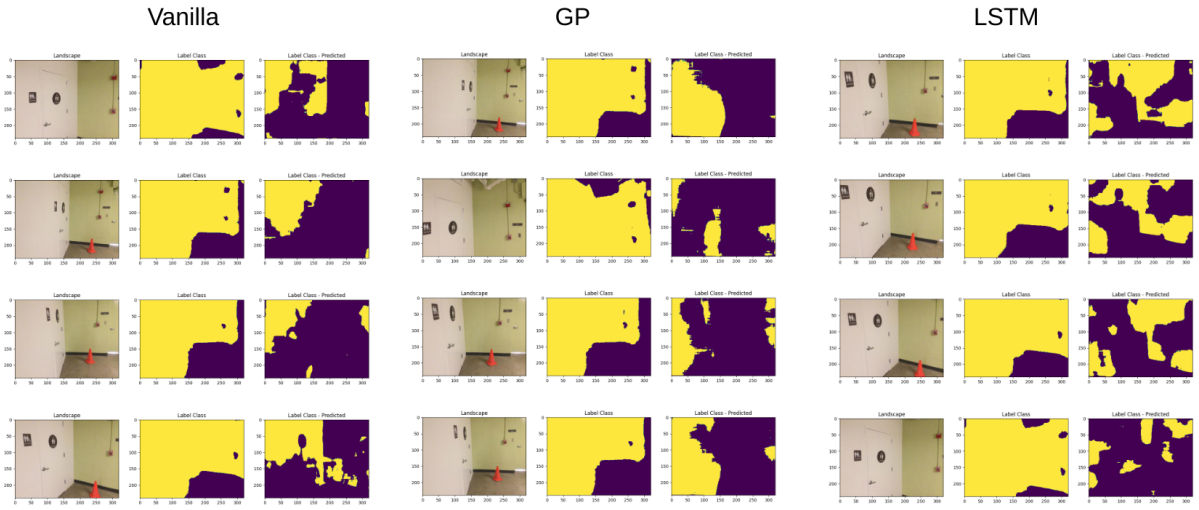


Figure 4.18: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm model

4.5.5 Experiment1.3: U-Net Vanilla, temporally fused gp and lstm model considering three classes

In this experiment wall, furniture and other+combined classes are considered due to high pixel distribution, resulting in three unique classes. There is a huge jump in the pixel accuracy for the lstm and gp model in comparison to vanilla model. The mean pixel accuracy is high at 0.55 in comparison to vanilla model performance at 0.31. A increase of 24%. The meanIoU value also stand high at 0.40 for lstm model in comparison to 0.18 for vanilla model, a difference of 22%. The FwIoU value increased by 25% for the lstm model. The effect of temporal fusion can be clearly seen with these metrics. Temporal fusion improved the performance by a huge margin. The gp model also performed well in comparison with vanilla and lstm. In this experiment the pixel classes are well balanced in comparison with the two class approach, where the difference is high. The comparison of results are presented in the table Table 4.6. Temporal fusion in action can be seen in Fig 4.20. The comparison of model performance with a bar plot is depicted in Fig 4.21

Metric	Vanilla	GP	LSTM
Pixel Accuracy	0.3289	0.5731	0.6266
Pixel Mean accuracy	0.3135	0.4779	0.6920
meanIOU	0.1820	0.3356	0.4870
IoU	[0.2732, 0.1686, 0.1042]	[0.5178, 0.3066, 0.1823]	[0.6810, 0.4661, 0.2998]
FwIoU	0.2160	0.4033	0.6245

Table 4.6: Performance of Vanilla model with respect to different metric and two classes

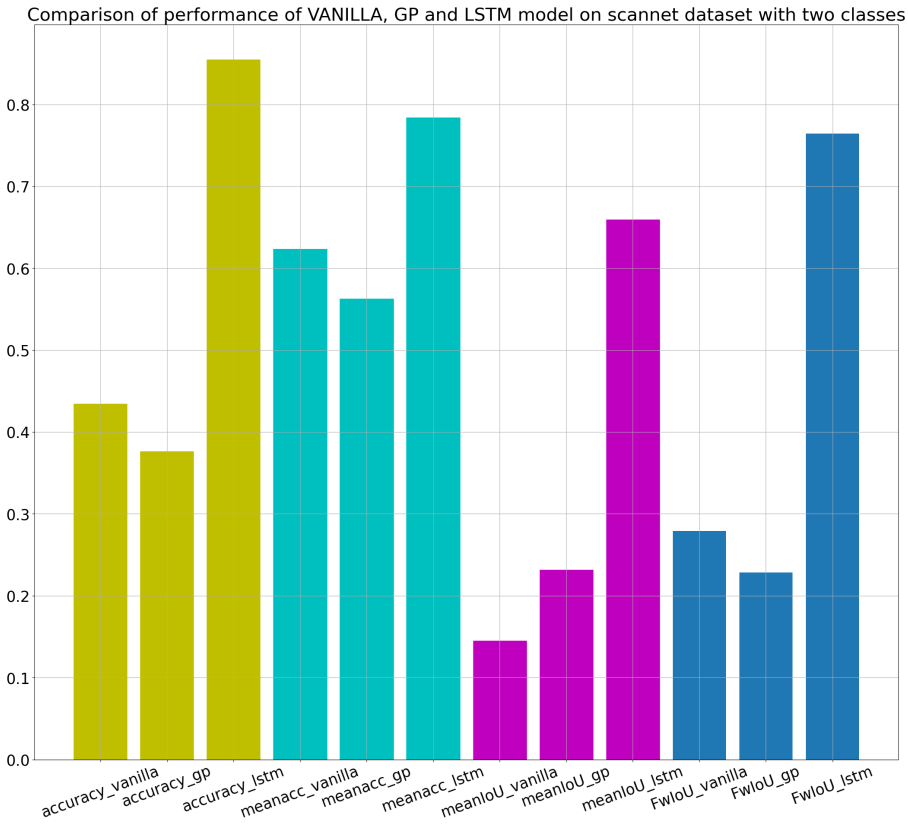


Figure 4.19: Comparison of VANILLA, GP and LSTM model performance based on metric for scannet two classes

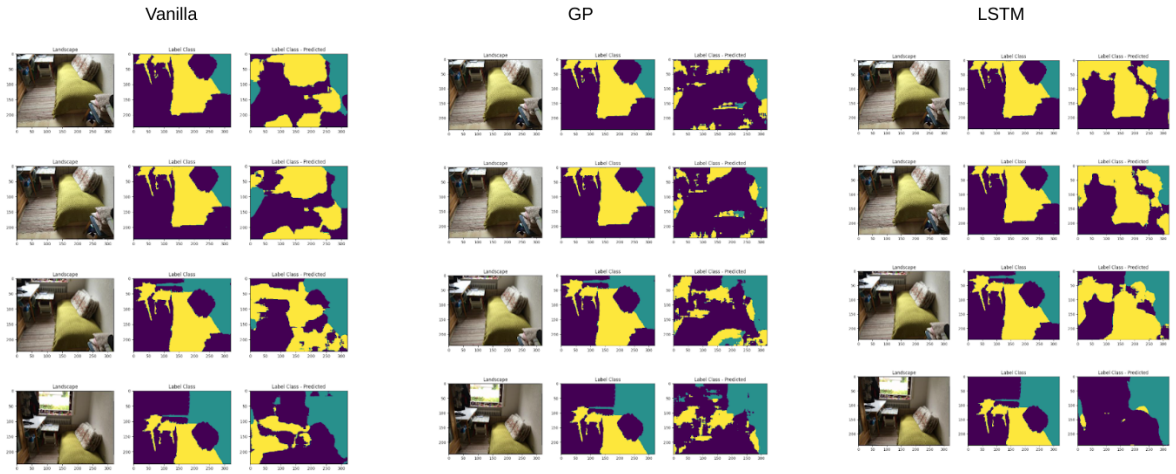


Figure 4.20: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm model

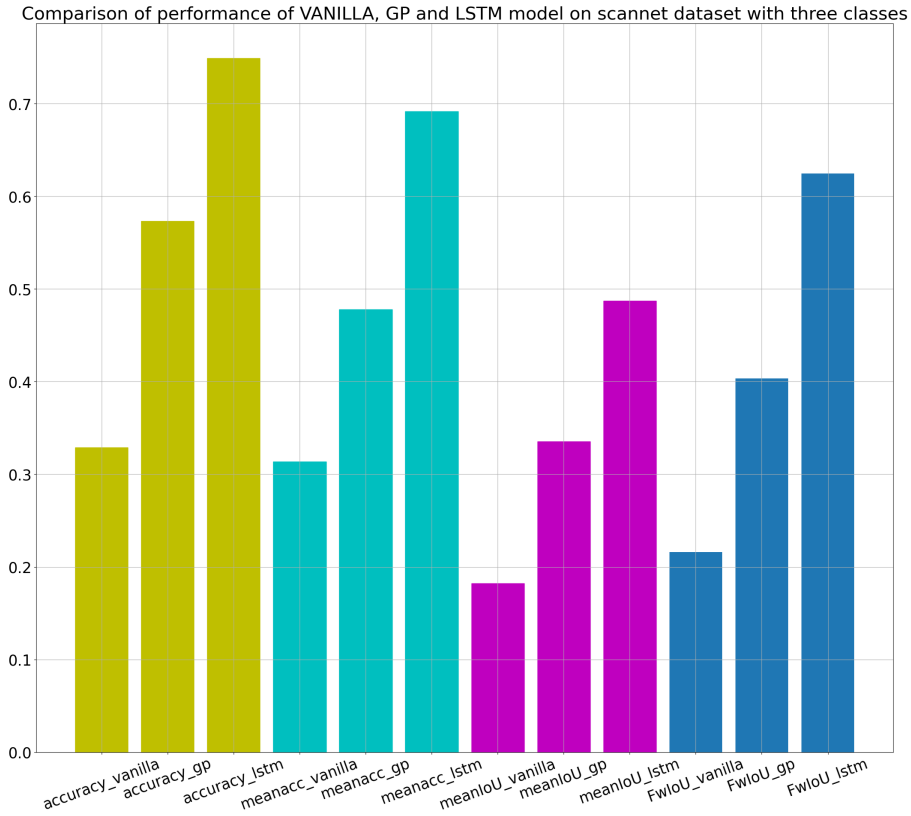


Figure 4.21: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm model

4.6 Experiment 2.1: Experiment with vkitti dataset with clone and sunset as the testing data

Similar to the experiments conducted on the scannet dataset, experiments were conducted on the vkitti dataset. Vkitti is a dataset containing a virtual car moving in a city scape and recorded with different configuration. There are totally 15 classes in the entire dataset. The results were tabulated in the Table 4.7. There are totally 10 categories in the entire dataset. In this experiment clone and sunset are taken as the testing data and trained on the remaining 8 classes. The result of the experiments were tabulated in the Table 4.7. The Vanilla, GP and LSTM model performance are compared in the table. A small improvement in the performance can be observed due to the similarity between the all the different categories. All the categories are similar, only the environment changes. In this experiment GP performed well in comparison to the vanilla and LSTM model. The accuracy of the model is high 0.98. Temporal fusion can be observed with mean pixel accuracy, FwIoU and meanIoU metric with improvement in result. The model is evaluated with the testing data and the results are calculated for every batch and plotted as the box plot in the Fig 4.23. For all of the metric the GP produced better result in comparison other models. IoU for the traffic light, pole was low with respect to the other classes. The performance of vanilla and the LSTM model is similar. Four test images were taken and run through vanilla, GP and LSTM model and the ground truth and the predictions are compared side by side. Vanilla and GP model perfectly segment the truck, however the LSTM model not properly segment the truck. However, car, pole, road, sky categories are perfectly categorized by these models. The side by side comparison is presented in the Figure 4.22.

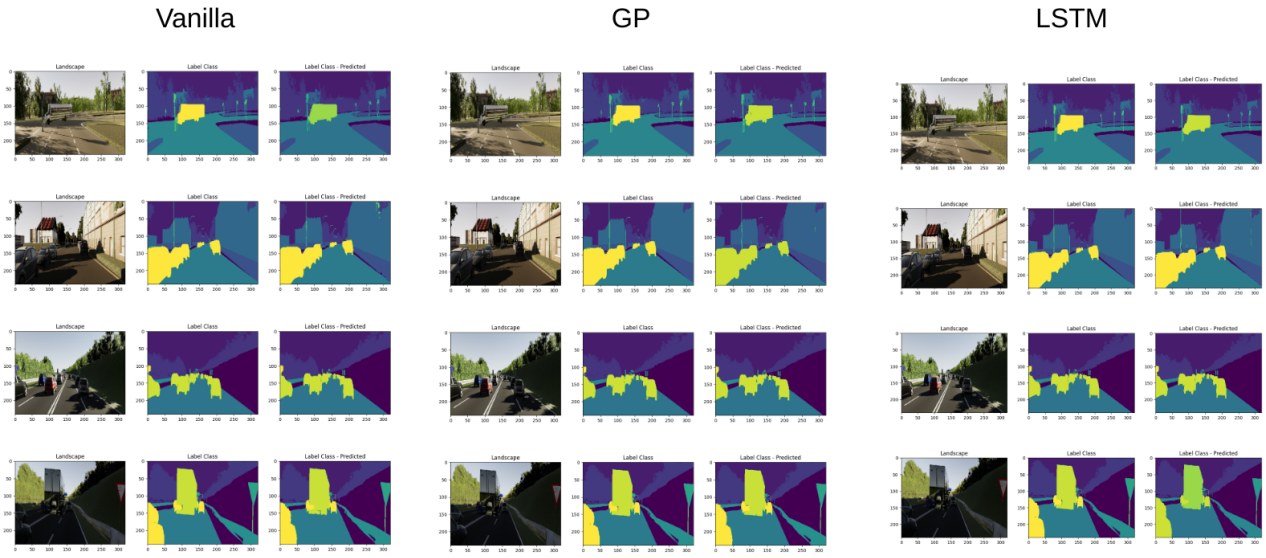


Figure 4.22: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm

Metric	Vanilla	GP	LSTM
Pixel Accuracy	0.9835	0.9857	0.9843
Pixel Mean accuracy	0.9572	0.9637	0.9596
meanIOU	0.9241	0.9364	0.9279
IoU	[0.9714, 0.9731, 0.966, 0.9377, 0.9296, 0.9894, 0.9663, 0.9265, 0.8165, 0.7835, 0.8359, 0.9474, 0.9633, 0.9299]	[0.9752, 0.9741, 0.9685, 0.955, 0.9393, 0.9918, 0.9714, 0.9413, 0.822, 0.8195, 0.8686, 0.9618, 0.9705, 0.9497]	[0.9732, 0.9736, 0.9664, 0.9482, 0.9289, 0.9906, 0.9693, 0.9313, 0.8135, 0.7837, 0.8492, 0.9595, 0.9653, 0.9373]
FwIoU	0.9679	0.9721	0.9694

Table 4.7: Performance of Vanilla model with respect to different metric and two classes

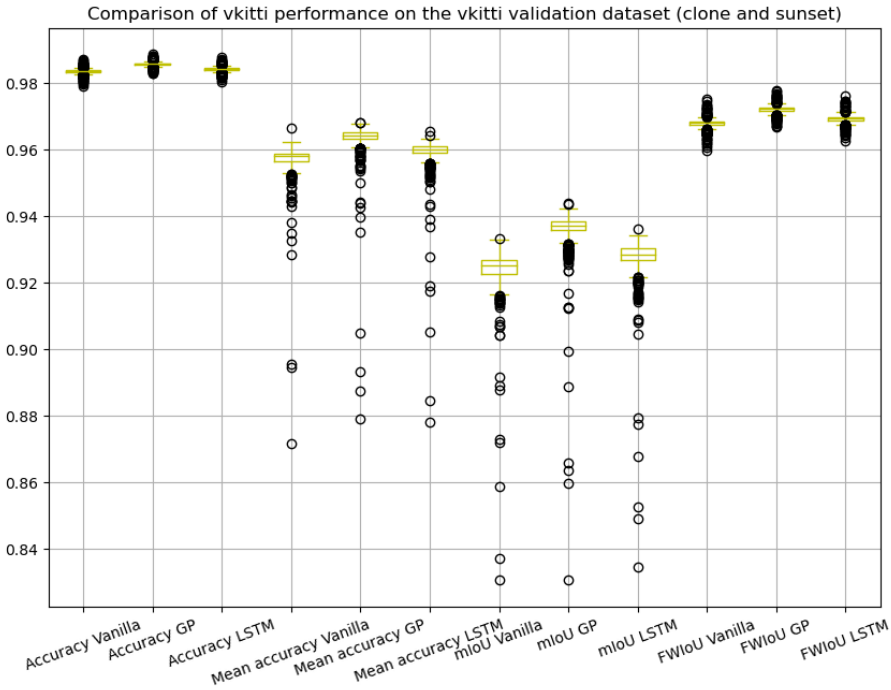


Figure 4.23: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm

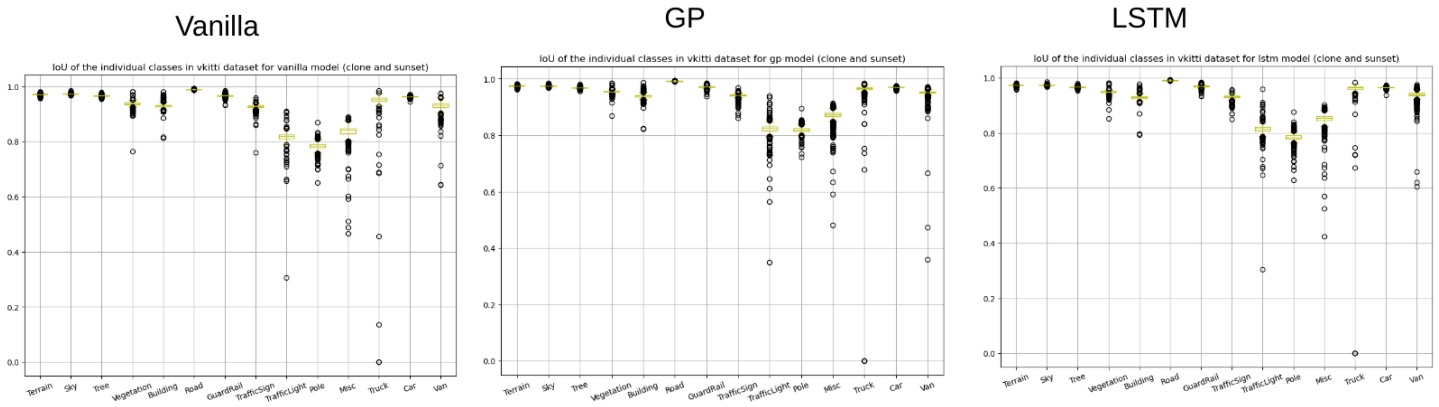


Figure 4.24: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm

Metric	Vanilla	GP	LSTM
Pixel Accuracy	0.9419	0.9329	0.9491
Pixel Mean accuracy	0.7797	0.7878	0.8083
meanIOU	0.6884	0.6977	0.7206
IoU	[0.9128, 0.9417, 0.92, 0.6876, 0.7247, 0.958, 0.8576, 0.6196, 0.4536, 0.4217, 0.3476, 0.5452, 0.8674, 0.3803]	[0.8961, 0.8985, 0.8949, 0.7113, 0.7408, 0.9505, 0.8602, 0.6862, 0.5612, 0.4683, 0.3427, 0.4788, 0.8506, 0.4285]	[0.9078, 0.9565, 0.9296, 0.7427, 0.7816, 0.96, 0.8669, 0.6828, 0.4898, 0.4464, 0.4182, 0.6187, 0.8843, 0.4025]
FwIoU	0.8959	0.8788	0.9074

Table 4.8: Performance of Vanilla model with respect to different metric and two classes

4.7 Experiment 2.2: Experiment with vkitti dataset with clone, sunset, rain, 15-deg-right, and 30-deg-left as the testing data

In the second type of experiment on vkitti data, five('clone/', 'sunset/', 'rain/', '15-deg-right/', '30-deg-left/') categories out of 10 categories are taken for testing. The model is trained on five (15-deg-left/','fog/', 'overcast/', 'morning/', '30-deg-right/') categories. The result of the experiments are tabulated in the Table 4.8. It is a side by side comparison of the model performance based on different metric. The performance improvement can be observed with the menIoU metric. The meanIoU improved by 1% and 3% for gp and lstm with respect to the vanilla model. Similar characteristics can be observed with the mean pixel accuracy. However, the pixel accuracy remained constant for the all the three model. The side by comparison of the raw rgb image, ground truth and predicted segmentation map is shown in the Fig 4.25.

The box plot of the vanilla, gp and lstm model performance is shown in the Fig 4.26. The pixel accuracy, and FwIoU is high for the lstm model and for the gp and vanilla model the performance is comparable. Mean pixel accuracy and meanIoU are high for the lstm model. Traffic light, Pole and Misc

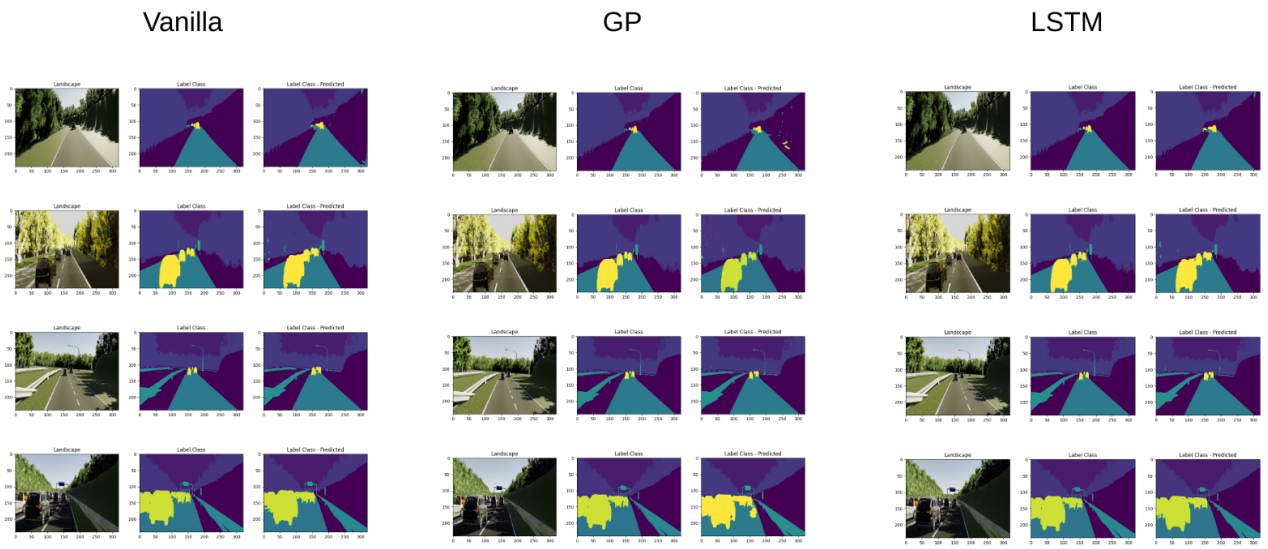


Figure 4.25: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm

class IoU are low in vanilla, gp and lstm models. The same is shown in the Fig 4.27. Hence from the result table we can prove that by incorporating the temporal fusion data for the semantic segmentation improves the performance.

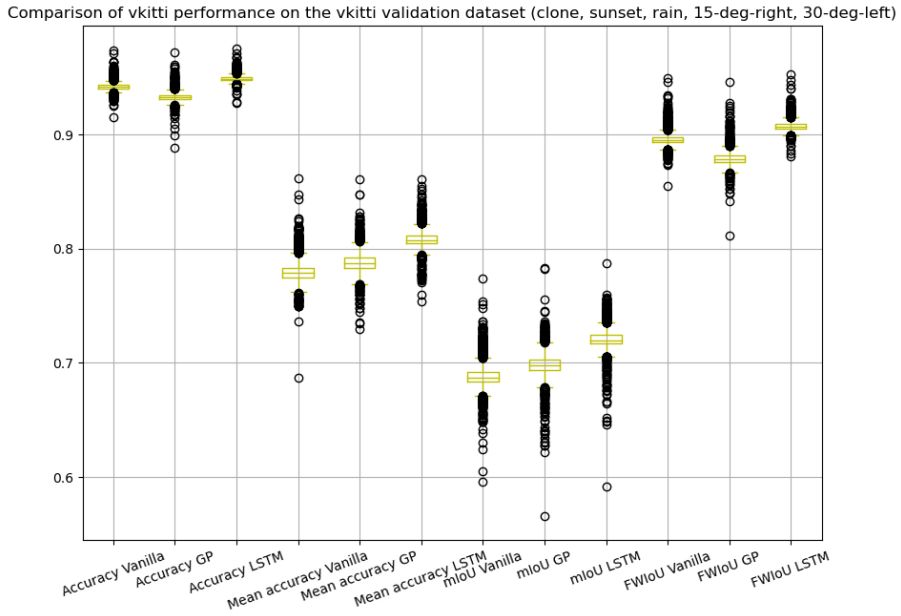


Figure 4.26: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm

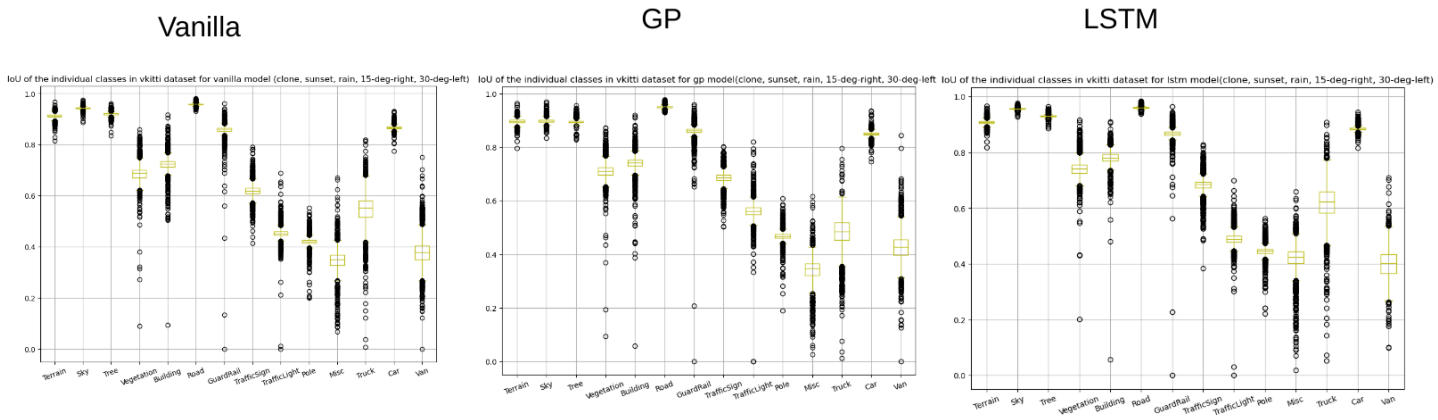


Figure 4.27: Plotting of raw input image, ground truth and predicted output for vanilla, gp and lstm

5

Android Deployment

5.1 Framework

Describe results and analyse them

5.2 Pipeline

5.3 Deployment and Results

6

Conclusions

6.1 Contributions

6.2 Lessons learned

6.3 Future work

A

Design Details

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 240, 320]	1,792
ReLU-2	[-1, 64, 240, 320]	0
BatchNorm2d-3	[-1, 64, 240, 320]	128
Conv2d-4	[-1, 64, 240, 320]	36,928
ReLU-5	[-1, 64, 240, 320]	0
BatchNorm2d-6	[-1, 64, 240, 320]	128
MaxPool2d-7	[-1, 64, 120, 160]	0
Conv2d-8	[-1, 128, 120, 160]	73,856
ReLU-9	[-1, 128, 120, 160]	0
BatchNorm2d-10	[-1, 128, 120, 160]	256
Conv2d-11	[-1, 128, 120, 160]	147,584
ReLU-12	[-1, 128, 120, 160]	0
BatchNorm2d-13	[-1, 128, 120, 160]	256
MaxPool2d-14	[-1, 128, 60, 80]	0
Conv2d-15	[-1, 256, 60, 80]	295,168
ReLU-16	[-1, 256, 60, 80]	0
BatchNorm2d-17	[-1, 256, 60, 80]	512
Conv2d-18	[-1, 256, 60, 80]	590,080
ReLU-19	[-1, 256, 60, 80]	0
BatchNorm2d-20	[-1, 256, 60, 80]	512
MaxPool2d-21	[-1, 256, 30, 40]	0
Conv2d-22	[-1, 512, 30, 40]	1,180,160
ReLU-23	[-1, 512, 30, 40]	0
BatchNorm2d-24	[-1, 512, 30, 40]	1,024
Conv2d-25	[-1, 512, 30, 40]	2,359,808
ReLU-26	[-1, 512, 30, 40]	0
BatchNorm2d-27	[-1, 512, 30, 40]	1,024
MaxPool2d-28	[-1, 512, 15, 20]	0
Conv2d-29	[-1, 1024, 15, 20]	4,719,616
ReLU-30	[-1, 1024, 15, 20]	0
BatchNorm2d-31	[-1, 1024, 15, 20]	2,048
Conv2d-32	[-1, 1024, 15, 20]	9,438,208
ReLU-33	[-1, 1024, 15, 20]	0
BatchNorm2d-34	[-1, 1024, 15, 20]	2,048
ConvTranspose2d-35	[-1, 512, 30, 40]	4,719,104
Conv2d-36	[-1, 512, 30, 40]	4,719,104
ReLU-37	[-1, 512, 30, 40]	0
BatchNorm2d-38	[-1, 512, 30, 40]	1,024
Conv2d-39	[-1, 512, 30, 40]	2,359,808
ReLU-40	[-1, 512, 30, 40]	0
BatchNorm2d-41	[-1, 512, 30, 40]	1,024
ConvTranspose2d-42	[-1, 256, 60, 80]	1,179,904
Conv2d-43	[-1, 256, 60, 80]	1,179,904
ReLU-44	[-1, 256, 60, 80]	0
BatchNorm2d-45	[-1, 256, 60, 80]	512
Conv2d-46	[-1, 256, 60, 80]	590,080
ReLU-47	[-1, 256, 60, 80]	0
BatchNorm2d-48	[-1, 256, 60, 80]	512
ConvTranspose2d-49	[-1, 128, 120, 160]	295,040
Conv2d-50	[-1, 128, 120, 160]	295,040

Table A.1: Unet vanilla model architecture

ReLU-51	[-1, 128, 120, 160]	0
BatchNorm2d-52	[-1, 128, 120, 160]	256
Conv2d-53	[-1, 128, 120, 160]	147,584
ReLU-54	[-1, 128, 120, 160]	0
BatchNorm2d-55	[-1, 128, 120, 160]	256
ConvTranspose2d-56	[-1, 64, 240, 320]	73,792
Conv2d-57	[-1, 64, 240, 320]	73,792
ReLU-58	[-1, 64, 240, 320]	0
BatchNorm2d-59	[-1, 64, 240, 320]	128
Conv2d-60	[-1, 64, 240, 320]	36,928
ReLU-61	[-1, 64, 240, 320]	0
BatchNorm2d-62	[-1, 64, 240, 320]	128
Conv2d-63	[-1, 41, 240, 320]	23,657

Table A.2: Unet vanilla model architecture

B

Parameters

Id	Classes
1	Wall, Shower Walls, Closet Wall, Shower Wall, Pantry Wall, Closet Walls, Bath Walls, Pantry Walls, Door Wall,
2	Floor, Shower Floor, Closet Floor,
3	Cabinet, Kitchen Cabinet, Kitchen Cabinets, File Cabinet, Bathroom Vanity, Cabinets, Bathroom Cabinet, Cabinet Doors, Open Kitchen Cabinet, File Cabinets, Trash Cabinet, Media Center,
4	Bed, Mattress, Loft Bed, Sofa Bed, Air Mattress,
5	Chair, Office Chair, Armchair, Sofa Chair, Stack Of Chairs, Folded Chair, Folded Chairs, Massage Chair, Recliner Chair, Rocking Chair, Stack Of Folded Chairs,
6	Couch, Sofa,
7	Table, Coffee Table, End Table, Dining Table, Folded Table, Round Table, Side Table, Air Hockey Table,
8	Door, Doorframe, Doors, Bathroom Stall Door, Closet Doors, Closet Door, Shower Door, Mirror Doors, Cabinet Door, Glass Doors, Sliding Door, Closet Doorframe,
9	Window,
10	Bookshelf, Bookshelves,
11	Picture, Poster, Painting, Pictures,
12	Kitchen Counter, Counter, Bathroom Counter,
13	Blinds,
14	Desk,
15	Shelf, Organizer Shelf, Pantry Shelf, Closet Shelf,
16	Curtain, Curtains,
17	Dresser,
18	Pillow, Pillows, Couch Cushions, Cushion,
19	Mirror,
20	Mat, Yoga Mat,
21	Clothes, Clothing, Cloth, Sock, Kitchen Apron, Costume, Socks,
22	Ceiling, Closet Ceiling,
23	Books, Book, Music Book,
24	Refrigerator, Mini Fridge, Cooler,
25	Tv,
26	Paper, Papers,
27	Towel, Towels, Hand Towel,
28	Shower Curtain,
29	Box, Boxes, Mailboxes, Mailbox, Storage Box, Pizza Box, Boxes Of Paper, Jewelry Box, Cat Litter Box, Covered Box, Pizza Boxes,
30	Whiteboard,
31	Person, Legs,
32	Nightstand,
33	Toilet, Urinal,
34	Sink,
35	Lamp, Lamp Base, Desk Lamp, Wall Lamp, Table Lamp, Ceiling Lamp, Night Lamp,
36	Bathtub,
37	Bag, Paper Bag, Messenger Bag, Ikea Bag, Duffel Bag, Bag Of Coffee Beans, Grocery Bag, Golf Bag, Garbage Bag, Coffee Bean Bag, Trash Bag, Cosmetic Bag, Shopping Bag, Food Bag,

Table B.1: Classes and ids of the Scannet dataset

Id	Classes
38	Board, Stove, Light, Bathroom Stall, Bar, Light Switch, Ceiling Light, Range Hood, Blackboard, Rail, Bulletin Board, Ledge, Shower, Windowsill, Dishwasher, Stair Rail, Stairs, Handicap Bar, Column, Oven, Pillar, Structure, Shower Head, Projector Screen, Staircase, Fireplace, Breakfast Bar, Hand Rail, Water Fountain, Kitchen Island, Pipes, Shower Control Valve, Handrail, Step, Dart Board, Grab Bar, Railing, Stair, Soap Bar, Studio Light, Shower Doors, Boards, Frame, Garage Door, Platform, Elevator, Wood Beam, Banister, Curtain Rod, Chandelier, Stovetop, Glass,
39	Trash Can, Radiator, Recycling Bin, Ottoman, Bench, Tv Stand, Wardrobe Closet, Trash Bin, Seat, Closet, Ladder, Piano, Water Cooler, Stand, Washing Machine, Rack, Washing Machines, Wardrobe Cabinet, Clothes Dryer, Ironing Board, Keyboard Piano, Music Stand, Furniture, Crate, Clothes Dryers, Drawer, Footrest, Piano Bench, Foosball Table, Footstool, Compost Bin, Tripod, Treadmill, Chest, Folded Ladder, Drying Rack, Pool Table, Heater, Toolbox, Beanbag Chair, Dollhouse, Ping Pong Table, Clothing Rack, Podium, Luggage Stand, Rack Stand, Futon, Book Rack, Seating, Workbench, Easel, Luggage Rack, Headboard, Display Rack, Crib, Bedframe, Closet Wardrobe, Wardrobe, Bunk Bed, Magazine Rack, Furnace, Stepladder, Baby Changing Station, Flower Stand, Display,

Table B.2: Classes and ids of the Scannet dataset

Id	Classes
40	Object, Monitor, Backpack, Plant, Toilet Paper, Shoes, Keyboard, Bottle, Stool, Computer Tower, Telephone, Cup, Jacket, Microwave, Paper Towel Dispenser, Suitcase, Laptop, Printer, Soap Dispenser, Fan, Tissue Box, Blanket, Copier, Soap Dish, Laundry Hamper, Storage Bin, Coffee Maker, Decoration, Clock, Mouse, Basket, Dumbbell, Bucket, Sign, Speaker, Container, Shower Curtain Rod, Tube, Storage Container, Paper Towel Roll, Ball, Laundry Basket, Cart, Dish Rack, Purse, Bicycle, Tray, Plunger, Paper Cutter, Toilet Paper Dispenser, Bin, Toilet Seat Cover Dispenser, Guitar, Fire Extinguisher, Pipe, Vacuum Cleaner, Plate, Cd Case, Bowl, Closet Rod, Scale, Broom, Hat, Guitar Case, Water Pitcher, Laundry Detergent, Hair Dryer, Divider, Power Outlet, Coffee Kettle, Toaster, Shoe, Alarm Clock, Water Bottle, Case Of Water Bottles, Toaster Oven, Coat Rack, Storage Organizer, Machine, Fire Alarm, Vent, Power Strip, Calendar, Toilet Paper Holder, Potted Plant, Stuffed Animal, Luggage, Headphones, Candle, Projector, Dustpan, Rod, Globe, Step Stool, Vending Machine, Ceiling Fan, Swiffer, Jar, Hamper, Poster Tube, Case, Carpet, Thermostat, Coat, Smoke Detector, Flip Flops, Banner, Clothes Hanger, Whiteboard Eraser, Iron, Instrument Case, Toilet Paper Rolls, Soap, Block, Wall Hanging, Toothbrush, Shirt, Cutting Board, Vase, Exercise Machine, Shorts, Tire, Teddy Bear, Bathrobe, Faucet, Thermos, Rug, Tupperware, Shoe Rack, Beer Bottles, Salt, Dispenser, Remote, Carton, Slippers, Soda Stream, Toilet Brush, Cooking Pot, Stapler, Scanner, Elliptical Machine, Kettle, Metronome, Dumbell, Rice Cooker, Sewing Machine, Flowerpot, Nerf Gun, Binders, Quadcopter, Pitcher, Hanging, Mail, Hoverboard, Water Heater, Spray Bottle, Rope, Plastic Container, Soap Bottle, Sleeping Bag, Frying Pan, Oven Mitt, Pot, Hand Dryer, Shampoo Bottle, Hair Brush, Tennis Racket, Display Case, Boiler, Bananas, Carseat, Helmet, Umbrella, Coffee Box, Envelope, Wet Floor Sign, Controller, Dolly, Shampoo, Paper Tray, Changing Station, Poster Printer, Screen, Crutches, Stack Of Cups, Toilet Flush Button, Trunk, Plastic Bin, Car, Shaving Cream, Shredder, Statue, Hose, Bike Pump, Coatrack, Bear, Humidifier, Toothpaste, Mouthwash Bottle, Poster Cutter, Food Container, Camera, Card, Mug, Cardboard, Flag, Magazine, Exit Sign, Rolled Poster, Wheel, Blackboard Eraser, Organizer, Doll, Laundry Bag, Sponge, Lotion Bottle, Can, Lunch Box, Food Display, Storage Shelf, Sliding Wood Door, Pants, Wood, Bottles, Washcloth, Cups, Exercise Ball, Roomba, Bike Lock, Briefcase, Bath Products, Star, Map, Ipad, Traffic Cone, Toiletry, Canopy, Paper Organizer, Barricade, Cap, Dumbbell Plates, Cooking Pan, Santa, Boat, Kinect, Plastic Storage Bin, Dishwashing Soap Bottle, Xbox Controller, Banana Holder, Ping Pong Paddle, Airplane, Conditioner Bottle, Tea Kettle, Toilet Paper Package, Wall Mounted Coat Rack, Film Light, Chain, Sweater, Kitchen Mixer, Water Softener, Trolley, Loofa, Shower Faucet Handle, Toy Piano, Fish, Electric Panel, Suitcases, Tape, Plates, Alarm, Fire Hose, Toy Dinosaur, Cone, Hatrack, Subwoofer, Fire Sprinkler, Photo, Barrier, Stacks Of Cups, Beachball, Folded Boxes, Contact Lens Solution Bottle, Folder, Mail Trays, Slipper, Sticker, Lotion, Buddha, File Organizer, Paper Towel Rolls, Fuse Box, Knife Block, Cd Cases, Stools, Hand Sanitzer Dispenser, Teapot, Pen Holder, Tray Rack, Wig, Switch, Plastic Containers, Night Light, Notepad, Mail Bin, Elevator Button, Gaming Wheel, Drum Set, Coffee Mug, Baby Mobile, Diaper Bin, Stepstool, Paper Shredder, Dress Rack, Cover, Exercise Bike, Kitchenaid Mixer, Soda Can, Tap, Cable, Binder, Towel Rack, Medal, Telescope, Baseball Cap, Battery Disposal Jar, Mop, Tank, Mail Tray, Centerpiece, Stick, Dryer Sheets, Bycicle, Clip, Postcard, Display Sign, Paper Towel, Boots, Tennis Racket Bag, Clothes Hangers, Starbucks Cup,
0	Other

Table B.3: Classes and ids of the Scannet dataset

References

[1] Michael Middleton. Deep learning vs. machine learning — what's the difference?, 2021.

[2] Yuxin Hou, Juho Kannala, and Arno Solin. Multi-view stereo by temporal nonparametric fusion. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 2651–2660, 2019.

[3] Your complete guide to image segmentation. `https://www.telusinternational.com/articles/guide-to-image-segmentation#:~:text=Different%20types%20of%20image%20segmentation%20tasks,types%20of%20image%20segmentation%20tasks.` Accessed: 2022-08-22.

[4] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[5] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. Learning deconvolution network for semantic segmentation. In *Proceedings of the IEEE international conference on computer vision*, pages 1520–1528, 2015.

[6] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE transactions on pattern analysis and machine intelligence*, 39(12):2481–2495, 2017.

[7] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[8] Ping Hu, Fabian Caba, Oliver Wang, Zhe Lin, Stan Sclaroff, and Federico Perazzi. Temporally distributed networks for fast video semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8818–8827, 2020.

[9] Adrian Rosebrock. Intersection over union (iou) for object detection, 2016.

[10] Danilo P Mandic, Dragan Obradovic, Anthony Kuh, Tülay Adali, Udo Trutschell, Martin Golz, Philippe De Wilde, Javier Barria, Anthony Constantinides, and Jonathon Chambers. Data fusion for modern engineering applications: An overview. In *International Conference on Artificial Neural Networks*, pages 715–721. Springer, 2005.

[11] Federico Castanedo. A review of data fusion techniques. *The scientific world journal*, 2013, 2013.

[12] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[13] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 15324–15333, 2021.

[14] Minghan Li, Shuai Li, Lida Li, and Lei Zhang. Spatial feature calibration and temporal fusion for effective one-stage video instance segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11215–11224, 2021.

[15] Ko Ming Hsiao, Geoff West, Svetha Venkatesh, and Mohan Kumar. Temporal data fusion in multisensor systems using dynamic time warping. In *SENSORFUSION 2005: Workshop on Information Fusion and Dissemination in Wireless Sensor Networks*, pages 1–9. IEEE, 2005.

[16] Ko Ming Hsiao, Geoff West, Svetha Venkatesh, and Mohan Kumar. Temporal data fusion in multisensor systems using dynamic time warping. In *SENSORFUSION 2005: Workshop on Information Fusion and Dissemination in Wireless Sensor Networks*, pages 1–9. IEEE, 2005.

[17] Andreas Krause, Daniel P Siewiorek, Asim Smailagic, and Jonny Farringdon. Unsupervised, dynamic identification of physiological and activity context in wearable computing. In *ISWC*, volume 3, page 88, 2003.

[18] Jong-Min Lee, ChangKyoo Yoo, and In-Beum Lee. On-line batch process monitoring using a consecutively updated multiway principal component analysis model. *Computers & Chemical Engineering*, 27(12):1903–1912, 2003.

[19] Wei Han, Pooya Khorrami, Tom Le Paine, Prajit Ramachandran, Mohammad Babaeizadeh, Honghui Shi, Jianan Li, Shuicheng Yan, and Thomas S Huang. Seq-nms for video object detection. *arXiv preprint arXiv:1602.08465*, 2016.

[20] Kai Kang, Hongsheng Li, Junjie Yan, Xingyu Zeng, Bin Yang, Tong Xiao, Cong Zhang, Zhe Wang, Ruohui Wang, Xiaogang Wang, et al. T-cnn: Tubelets with convolutional neural networks for object detection from videos. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(10):2896–2907, 2017.

[21] Guanghan Ning, Zhi Zhang, Chen Huang, Xiaobo Ren, Haohong Wang, Canhui Cai, and Zhihai He. Spatially supervised recurrent convolutional neural networks for visual object tracking. In *2017 IEEE international symposium on circuits and systems (ISCAS)*, pages 1–4. IEEE, 2017.

[22] Zhichao Lu, Vivek Rathod, Ronny Votel, and Jonathan Huang. Retinatrack: Online single stage joint detection and tracking. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14668–14678, 2020.

[23] Okan Köpüklü, Xiangyu Wei, and Gerhard Rigoll. You only watch once: A unified cnn architecture for real-time spatiotemporal action localization. *arXiv preprint arXiv:1911.06644*, 2019.

[24] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016.

[25] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016.

[26] Emeç Erçelik, Ekim Yurtsever, and Alois Knoll. Temp-frustum net: 3d object detection with temporal fusion. In *2021 IEEE Intelligent Vehicles Symposium (IV)*, pages 1095–1101. IEEE, 2021.

[27] Jiejie Zhu, Liang Wang, Jizhou Gao, and Ruigang Yang. Spatial-temporal fusion for high accuracy depth maps using dynamic mrfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(5):899–909, 2009.

[28] Gang Wu, Yi Wu, Long Jiao, Yuan-Fang Wang, and Edward Y Chang. Multi-camera spatio-temporal fusion and biased sequence-data learning for security surveillance. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 528–538, 2003.

[29] Michael Teutsch and Wolfgang Krüger. Spatio-temporal fusion of object segmentation approaches for moving distant targets. In *2012 15th International Conference on Information Fusion*, pages 1988–1995. IEEE, 2012.

[30] David Forsyth and Jean Ponce. *Computer vision: A modern approach.* Prentice hall, 2011.

[31] Jifeng Dai, Kaiming He, and Jian Sun. Instance-aware semantic segmentation via multi-task network cascades. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3150–3158, 2016.

[32] King-Sun Fu and JK Mui. A survey on image segmentation. *Pattern recognition*, 13(1):3–16, 1981.

[33] W Ladys law Skarbek and Andreas Koschan. Colour image segmentation a survey. *IEEE Transactions on circuits and systems for Video Technol-ogy*, 14(7):1–80, 1994.

[34] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[35] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[36] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1925–1934, 2017.

[37] Simon Jégou, Michal Drozdzal, David Vazquez, Adriana Romero, and Yoshua Bengio. The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pages 11–19, 2017.

[38] Forrest Iandola, Matt Moskewicz, Sergey Karayev, Ross Girshick, Trevor Darrell, and Kurt Keutzer. Densenet: Implementing efficient convnet descriptor pyramids. *arXiv preprint arXiv:1404.1869*, 2014.

[39] Maoke Yang, Kun Yu, Chi Zhang, Zhiwei Li, and Kuiyuan Yang. Denseaspp for semantic segmentation in street scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3684–3692, 2018.

[40] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. *arXiv preprint arXiv:1412.7062*, 2014.

[41] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European conference on computer vision (ECCV)*, pages 405–420, 2018.

[42] Jizhong Deng, Zhaoji Zhong, Huasheng Huang, Yubin Lan, Yuxing Han, and Yali Zhang. Lightweight semantic segmentation network for real-time weed mapping using unmanned aerial vehicles. *Applied Sciences*, 10(20):7132, 2020.

[43] Chen-Chiung Hsieh, Dung-Hua Liou, and David Lee. A real time hand gesture recognition system using motion history image. In *2010 2nd international conference on signal processing systems*, volume 2, pages V2–394. IEEE, 2010.

[44] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

[45] Clement Farabet, Camille Couprie, Laurent Najman, and Yann LeCun. Learning hierarchical features for scene labeling. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1915–1929, 2012.

[46] Geoffrey Hinton, Li Deng, Dong Yu, George E Dahl, Abdel-rahman Mohamed, Navdeep Jaitly, Andrew Senior, Vincent Vanhoucke, Patrick Nguyen, Tara N Sainath, et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups. *IEEE Signal processing magazine*, 29(6):82–97, 2012.

[47] Lauv Patel, Tripti Shukla, Xiuzhen Huang, David W Ussery, and Shanzhi Wang. Machine learning methods in drug discovery. *Molecules*, 25(22):5277, 2020.

[48] T Ciodaro, D Deva, JM De Seixas, and D Damazio. Online particle detection with neural networks based on topological calorimetry information. In *Journal of physics: conference series*, volume 368, page 012030. IOP Publishing, 2012.

[49] Jinny X Zhang, Boyan Yordanov, Alexander Gaunt, Michael X Wang, Peng Dai, Yuan-Jyue Chen, Kerou Zhang, John Z Fang, Neil Dalchau, Jiaming Li, et al. A deep learning model for predicting next-generation sequencing depth from dna sequence. *Nature communications*, 12(1):1–10, 2021.

[50] Julia Hirschberg and Christopher D Manning. Advances in natural language processing. *Science*, 349(6245):261–266, 2015.

[51] Niall O'Mahony, Sean Campbell, Anderson Carvalho, Suman Harapanahalli, Gustavo Velasco Hernandez, Lenka Krpalkova, Daniel Riordan, and Joseph Walsh. Deep learning vs. traditional computer vision. In *Science and information conference*, pages 128–144. Springer, 2019.

[52] Sharada P Mohanty, David P Hughes, and Marcel Salathé. Using deep learning for image-based plant disease detection. *Frontiers in plant science*, 7:1419, 2016.

[53] Zhongchun Han and Anfeng Xu. Ecological evolution path of smart education platform based on deep learning and image detection. *Microprocessors and Microsystems*, 80:103343, 2021.

[54] Shrey Srivastava, Amit Vishvas Divekar, Chandu Anilkumar, Ishika Naik, Ved Kulkarni, and V Pattabiraman. Comparative analysis of deep learning image detection algorithms. *Journal of Big Data*, 8(1):1–27, 2021.

[55] Shervin Minaee, Yuri Y Boykov, Fatih Porikli, Antonio J Plaza, Nasser Kehtarnavaz, and Demetri Terzopoulos. Image segmentation using deep learning: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 2021.

[56] Christian S Jensen and Richard T Snodgrass. Temporal data management. *IEEE Transactions on knowledge and data engineering*, 11(1):36–44, 1999.

[57] Gowtham Atluri, Anuj Karpatne, and Vipin Kumar. Spatio-temporal data mining: A survey of problems and methods. *ACM Computing Surveys (CSUR)*, 51(4):1–41, 2018.

[58] Bryan Lim, Sercan Ö Arık, Nicolas Loeff, and Tomas Pfister. Temporal fusion transformers for interpretable multi-horizon time series forecasting. *International Journal of Forecasting*, 37(4):1748–1764, 2021.

[59] Yue Meng, Rameswar Panda, Chung-Ching Lin, Prasanna Sattigeri, Leonid Karlinsky, Kate Saenko, Aude Oliva, and Rogerio Feris. Adafuse: Adaptive temporal fusion network for efficient action recognition. *arXiv preprint arXiv:2102.05775*, 2021.

[60] Arda Duzceker, Silvano Galliani, Christoph Vogel, Pablo Speciale, Mihai Dusmanu, and Marc Pollefeys. Deepvideomvs: Multi-view stereo on video with recurrent spatio-temporal fusion. In

Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 15324–15333, 2021.

[61] Kai Zhang, Yifan Sun, Rui Wang, Haichang Li, and Xiaohui Hu. Multiple fusion adaptation: A strong framework for unsupervised semantic segmentation adaptation. *arXiv preprint arXiv:2112.00295*, 2021.

[62] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[63] Nobuyuki Otsu. A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, 9(1):62–66, 1979.

[64] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11):1222–1239, 2001.

[65] J-L Starck, Michael Elad, and David L Donoho. Image decomposition via the combination of sparse representations and a variational approach. *IEEE transactions on image processing*, 14(10):1570–1582, 2005.

[66] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017.

[67] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. *Advances in neural information processing systems*, 27, 2014.

[68] Alex Kendall, Vijay Badrinarayanan, and Roberto Cipolla. Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*, 2015.

[69] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv:1904.04514*, 2019.

[70] Jun Fu, Jing Liu, Yuhang Wang, Jin Zhou, Changyong Wang, and Hanqing Lu. Stacked deconvolutional network for semantic segmentation. *IEEE Transactions on Image Processing*, 2019.

[71] Ronghang Hu, Piotr Dollár, Kaiming He, Trevor Darrell, and Ross Girshick. Learning to segment every thing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4233–4241, 2018.

[72] Xide Xia and Brian Kulis. W-net: A deep model for fully unsupervised image segmentation. *arXiv preprint arXiv:1711.08506*, 2017.

[73] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. IEEE, 2016.

[74] Xiaojie Jin, Xin Li, Huaxin Xiao, Xiaohui Shen, Zhe Lin, Jimei Yang, Yunpeng Chen, Jian Dong, Luoqi Liu, Zequn Jie, et al. Video scene parsing with predictive feature learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5580–5588, 2017.

[75] Raghudeep Gadde, Varun Jampani, and Peter V Gehler. Semantic video cnns through representation warping. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4453–4462, 2017.

[76] Xiaojie Jin, Xin Li, Huaxin Xiao, Xiaohui Shen, Zhe Lin, Jimei Yang, Yunpeng Chen, Jian Dong, Luoqi Liu, Zequn Jie, et al. Video scene parsing with predictive feature learning. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5580–5588, 2017.

[77] David Nilsson and Cristian Sminchisescu. Semantic video segmentation by gated recurrent flow propagation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6819–6828, 2018.

[78] Samvit Jain, Xin Wang, and Joseph E Gonzalez. Accel: A corrective fusion network for efficient semantic segmentation on video. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8866–8875, 2019.

[79] Behrooz Mahasseni, Sinisa Todorovic, and Alan Fern. Budget-aware deep semantic video segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1029–1038, 2017.

[80] Angela Dai, Angel X. Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proc. Computer Vision and Pattern Recognition (CVPR), IEEE*, 2017.

[81] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2, 2020.

[82] Koji Minoda, Fabian Schilling, Valentin Wüest, Dario Floreano, and Takehisa Yairi. VIODE: A simulated dataset to address the challenges of visual-inertial odometry in dynamic environments. *IEEE Robotics and Automation Letters*, 6(2):1343–1350, 2021.

[83] Irem Ulku and Erdem Akagündüz. A survey on deep learning-based architectures for semantic segmentation on 2d images. *Applied Artificial Intelligence*, pages 1–45, 2022.

[84] JEREMY JORDAN. Evaluating image segmentation models, 2018.

[85] Kaggle. Kaggle competition.

[86] Ko Ming Hsiao, Geoff West, Svetha Venkatesh, and Mohan Kumar. Temporal data fusion in multisensor systems using dynamic time warping. In *SENSORFUSION 2005: Workshop on Information Fusion and Dissemination in Wireless Sensor Networks*, pages 1–9. IEEE, 2005.

[87] Xiang Li, Jinglu Wang, Xiao Li, and Yan Lu. Hybrid instance-aware temporal fusion for online video instance segmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 1429–1437, 2022.

[88] Miran Heo, Sukjun Hwang, Seoung Wug Oh, Joon-Young Lee, and Seon Joo Kim. Vita: Video instance segmentation via object token association. *arXiv preprint arXiv:2206.04403*, 2022.

[89] De-An Huang, Zhiding Yu, and Anima Anandkumar. Minvis: A minimal video instance segmentation framework without video-based training. *arXiv preprint arXiv:2208.02245*, 2022.

[90] Bowen Cheng, Anwesa Choudhuri, Ishan Misra, Alexander Kirillov, Rohit Girdhar, and Alexander G Schwing. Mask2former for video instance segmentation. *arXiv preprint arXiv:2112.10764*, 2021.

[91] Angela Dai, Angel X Chang, Manolis Savva, Maciej Halber, Thomas Funkhouser, and Matthias Nießner. Scannet: Richly-annotated 3d reconstructions of indoor scenes. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5828–5839, 2017.

[92] Manolis Savva Maciej Halber Thomas Funkhouser Matthias Nießner Angela Dai, Angel X. Chang. Scannet: Richly-annotated 3d reconstructions of indoor scenes, 2018.

[93] Yohann Cabon, Naila Murray, and Martin Humenberger. Virtual kitti 2. *arXiv preprint arXiv:2001.10773*, 2020.

[94] Manolis Savva Maciej Halber Thomas Funkhouser Matthias Nießner Angela Dai, Angel X. Chang. Scannet: Richly-annotated 3d reconstructions of indoor scenes, 2018.

[95] Manoj Kolpe Lingappa. Scannet data processing, 2022.

[96] Claudio Mazzotti, Nicola Sancisi, and Vincenzo Parenti-Castelli. A measure of the distance between two rigid-body poses based on the use of platonic solids. In *Symposium on Robot Design, Dynamics and Control*, pages 81–89. Springer, 2016.

[97] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. The MIT Press, 11 2005.

[98] Christopher KI Williams and Carl Edward Rasmussen. *Gaussian processes for machine learning*, volume 2. MIT press Cambridge, MA, 2006.

[99] Xingjian Shi, Zhourong Chen, Hao Wang, Dit-Yan Yeung, Wai-Kin Wong, and Wang-chun Woo. Convolutional lstm network: A machine learning approach for precipitation nowcasting. *Advances in neural information processing systems*, 28, 2015.