Master of Science Thesis in Computer Science
Department of Electrical Engineering, Linköping University, 2019

Improving 3D Point Cloud Segmentation Using Multimodal Fusion of Projected 2D Imagery Data

Linbo He

Master of Science Thesis in Computer Science

Improving 3D Point Cloud Segmentation Using Multimodal Fusion of Projected 2D Imagery Data

Linbo He

LiTH-ISY-EX–19/5190–SE

Supervisor: **Felix Järemo Lawin**
ISY, Linköping University

Examiner: **Michael Felsberg**
ISY, Linköping university

Computer Vision Laboratory
Department of Electrical Engineering
Linköping University
SE-581 83 Linköping, Sweden

Abstract

Semantic segmentation is a key approach to comprehensive image data analysis. It can be applied to analyze 2D images, videos, and even point clouds that contain 3D data points. On the first two problems, CNNs have achieved remarkable progress, but on point cloud segmentation, the results are less satisfactory due to challenges such as limited memory resource and difficulties in 3D point annotation. One of the research studies carried out by the Computer Vision Lab at Linköping University was aiming to ease the semantic segmentation of 3D point cloud. The idea is that by first projecting 3D data points to 2D space and then focusing only on the analysis of 2D images, we can reduce the overall workload for the segmentation process as well as exploit the existing well-developed 2D semantic segmentation techniques. In order to improve the performance of CNNs for 2D semantic segmentation, the study has used input data derived from different modalities. However, how different modalities can be optimally fused is still an open question.

Based on the above-mentioned study, this thesis aims to improve the multistream framework architecture. More concretely, we investigate how different singlestream architectures impact the multistream framework with a given fusion method, and how different fusion methods contribute to the overall performance of a given multistream framework.

As a result, our proposed fusion architecture outperformed all the investigated traditional fusion methods. Along with the best singlestream candidate and few additional training techniques, our final proposed multistream framework obtained a relative gain of 7.3% mIoU compared to the baseline on the semantic3D point cloud test set, increasing the ranking from 12th to 5th position on the benchmark leaderboard.

Acknowledgments

Throughout the journey of this thesis work I have received a great deal of support and assistance from the workplace at Computer Vision Laboratory at ISY. I would like to extend my sincere thanks to my former examiner Dr. Fahad Shahbaz Khan for providing me with a list of exciting thesis proposals and helping me with the construction of the research aim for my thesis together with my supervisor Felix Järemo Lawin, to whom I would also like to express my deepest gratitude for providing useful comments, remarks and valuable advice through the learning process of this master thesis. Furthermore, I am equally grateful to my colleague Goutam Bhat for helping me with extraordinary explanations of terminologies and concepts which I did not quite understand, as well as discussing the constructions of deep learning experiments with me. In addition, I would like to thank my new examiner Michael Felsberg for helping me with the refinement of my thesis report and the arrangement of the thesis defense.

Finally, I would like to thank my parents for their support and encouragement which help me in completion of this thesis work.

Linköping, Mars 2019
Linbo He

Contents

Notation

ABBREVIATIONS

Notation	Explanation
2D	2-dimensional
3D	2-dimensional
RGB	Red Green Blue color model
ANN	Artificial neural network
CNN	Convolutional neural network
FCN	Fully convolutional netowork
DNN	Deep neural network
DCNN	Deep convolutional neural network
Conv layer	Convolutional layer
ReLU	Rectified Linear Unit
DR50	Dilated ResNet50
PSPNet/PSP	Pyramid scene parsing network
PPM	Pyramid Pooling Module
DFN	Discriminative feature network
SN	Smooth network
GAP	Global average pooling
CAB	Channel attention block
RRB	Refinement residual block
LSF	Late sum fusion
LMF	Late max fusion
LCF	Late convolution fusion
OA	Overall accuracy
mIoU	Mean intersection over union
base work	*Deep projective 3d semantic segmentation* [23](which sets the foundation for this thesis)

1 Introduction

1.1 Background

In recent years, the progress in deep learning has advanced in the field of computer vision. In general, there are two reasons behind the success of deep learning. The first one is the explosive growth of data provided from internet, spanning almost all problem domains. This creates huge possibilities for us to find complex domain related patterns and thus produce better future predictions. The second is the remarkable development in computer hardware, especially the GPUs(and RAMs), which allow us to train our deep learning models in reasonable amount of time. As a result, deep learning approaches have, for example, significantly outperformed traditional machine learning methods, which require well designed hand-crafted features [9] for computer vision tasks. Currently, computer vision applications based on deep learning have been successfully applied to areas such as autonomous driving, medical imagery analysis, surveillance, and image search engines to name a few.

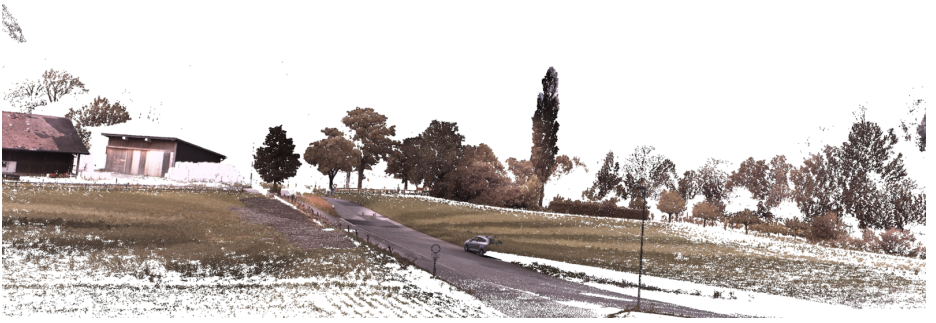
Computer vision includes several sub-domains among which semantic segmentation is a key approach to comprehensive image data analysis. While the most attention are paid towards semantic segmentation of 2D images in recent years, the rapid advancement in 3D acquisition sensors, such as LIDARs and RGB-D cameras, has led to increasing demand for 3D point clouds analysis, which are important for applications such as robotics and scene understanding.

One of the research studies carried out by the Computer Vision Lab at Linköping University was aiming to ease the semantic segmentation of 3D point cloud. As a result, the concept of migrating segmentation problems in 3D domain to 2D space had been found feasible and it was realized by using a series of different techniques according to [23]. The main idea was to reduce the workload of directly processing raw 3D point data for our system, in terms of memory usage and training time. The procedure can be generally decomposed in three sequential steps. The first step is to project 3D data points

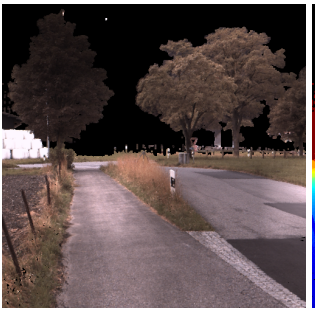
of point clouds into 2D images. The second step is to apply deep learning models on the projected images to classify each pixel and in the final step, the classification scores of the image pixels are projected back to 3D space in order to classify point cloud data.

This thesis is an extension of the work(named as *base work* for later use) mentioned above and focuses on the improvement of its second step where 2D images are classified at pixel level. Indeed, the base work has shown that the point cloud segmentation performance can be improved by enhancing the performance of semantic segmentation models. To do that, extra imagery modalities in addition to the coloured images we normally see are used. The different modalities used in the base work are called *color*, *depth* and *surface normals*. Figure 1.1 illustrates a given 3D scene and one of its many corresponding projected 2D images, shown in different modalities.

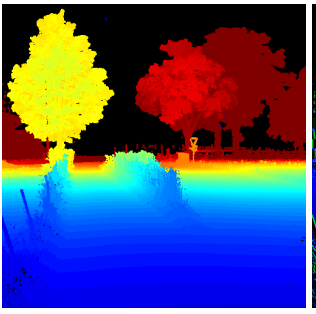
Since the baseline adopted a relatively simple singlestream architecture to train each modality, and used a traditional approach to fuse all the singlestreams, the underlying purpose of this thesis is to improve the overall framework performance by investigating more possibilities for the singlestream architectures as well as the fusion strategies. With the main focus on how to efficiently exploit information from the existing modalities, we build a novel multistream fusion architecture.



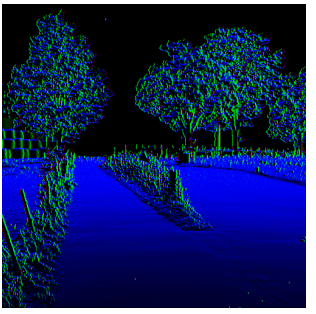
(a) *3D scene*



(b) *Color image*



(c) *Depth image*



(d) *Normals image*

Figure 1.1: *A general view of 3D scene and its projection in 2D space. The projection is one of many 2D images which together patch up the whole 3D scene* ***(a)****. In order to improve the model performance on semantic segmentation tasks, these input images are represented in color, depth, and surface normals modalities, shown in* ***(b)****,* ***(c)****, and* ***(d)****.*

1.2 Motivation

Point cloud data refer to data existing in three dimensional space. They are usually sparse and unstructured in contrast to the regular grid-structured image data. Yet, CNNs have not been successfully applied for point cloud segmentation due to several challenges. Normally, point cloud data are used to be preprocessed with voxelization but it introduces radical increase in memory complexity and loss in spatial resolution. Usually, only small chunks of voxelized data are allowed to be trained on 3D-CNNs because of limited memory, indicating that the models might not be able to utilize the contextual information for target objects. In addition, labeled point cloud data required for 3D-CNNs is scarce because of the difficulties in data annotation, while the annotated 2D datasets are largely accessible on internet. These limitations of point cloud segmentation motivated the base work to find an easier strategy to tackle it, which resulted in a framework that was able to transform the point cloud segmentation tasks into 2D semantic segmentation domain. The framework has achieved an state-of-the-art performance at that time. As a continuation of the base work, it is of interest to investigate how semantic segmentation of 2D images can be further improved since more advanced semantic segmentation architectures have been developed recently. These architectures provide powerful features for analyzing images at pixel level such as the ability to capture contextual and global information to help with the final prediction, which the investigated model of the base work does not have. Further, how different modalities can be optimally fused is still an open question, thus the performance of the investigated multistream fusion framework in the base work can hopefully be improved by investigating more on the fusion strategies.

1.3 Research questions

While the purpose of the base work [23] was mainly to explore the possibility of replacing 3D point cloud model with 2D semantic segmentation model, this thesis focuses on the improvement of the overall performance of what has been built in the base work, by implementing new network architectures and fusion methods. These new implementations lead to the following research questions:

- How much better does the proposed fusion architecture perform compared to other traditional late fusion methods in terms of *mean intersection over union*, under the condition that all singlestream networks are the same?

- How sensitive is the proposed fusion architecture in terms of *mean intersection over union*, when its all singlestream models are replaced with a more powerful one?

- Is it possible to improve the performance of traditional *early fusion* networks in terms of *mean intersection over union* by using an architecture that shares the same concept as the proposed fusion architecture for multistream framework?

1.4 Delimitations

This thesis only focuses on the improvement of semantic segmentation of 2D image data for the base work, while projection methods for transforming data between 2D and 3D space are not considered.

1.5 Thesis outline

The rest of this thesis is organized as follows. Chapter 2 provides background and theory to the field of semantic segmentation. Chapter 3 presents the investigated methods for this thesis and the previous work related to them. Chapter 4 describes the experimental setup, target datasets, and evaluation metrics. Chapter 5 illustrates both the quantitative and qualitative results of the conducted experiments, as well as the analysis. Chapter 6 gives overall conclusions of the experiments.

2

Background

This chapter presents the theoretical background of this thesis. It starts with the basics of neural network and semantic segmentation, then introduces the history of advancement of network architecture, followed by the description of how the particular models investigated in this thesis work. Lastly, several general techniques for improving neural network based models are explained.

2.1 Semantic Segmentation

Semantic segmentation refers to the understanding of an image at pixel level, i.e, we want to assign each pixel with a class, figure 2.1 shows an example image with pixel-wise annotations. In contrast to image classification tasks where an image needs to be classified to a single class, semantic segmentation tasks demands compact pixel-wise predictions from the applied classifiers. Before the breakthrough of deep learning, traditional machine learning(ML) methods were used for semantic segmentation, such as Random Forest [38]. In general, CNN-based deep learning networks currently dominates the computer vision field while traditional ML approaches are incapable to provide comparable results.

The key factor that made the dense predictions possible in deep learning was the realization of the concept of Fully Convolutional Networks(FCN) [39]. Essentially, it replaces fully connected layers at highest level of any image classification neural networks with convolutional layers and add a decoder to restore the spatial resolution in order to output score maps for each pixel. Thus, by simply applying these tricks on any state-of-the-art deep CNNs for image classification, we can acquire corresponding FCNs for semantic segmentation. This thesis uses FCN-based architectures to conduct the experiments.

Figure 2.1: *A semantic segmented image with annotations for different classes.*

2.2 Artificial neural networks

Within a human brain, the visual cortex is responsible to process the visual information received from eyes. In order to make the brain understand the information input, the cortex constructs hierarchical levels of information abstraction by using billions of the basic units, called *Neurons* [14]. Inspired by the human brain recognition process, *Artificial Neural Networks(ANNs)* constructs a hierarchy of artificial neurons that simulate the simplified functionality of organic neurons [27]. Within this hierarchy, different levels are represented by units called *Layers*. The layers can be distinguished between input, hidden and output layers. Hidden layers are located between input layer and output layer, aiming to provide intermediate feature abstractions of the input. Both the number of hidden layers of a network and the number of neurons of a hidden layer can be arbitrary depending on the complexity of a given task. The neurons located in the same layer have the ability to recognize similar features of an input, for instance, low level layers are specified to recognize simple features such as lines and edges and high level layers can recognize faces of cats and dogs for the task of cat-or-dog classification. Note that a layer is denoted as *Fully Connected Layer* when every neuron of this layer is connected to all neurons of next layer. Accordingly, a network consisted of only fully connected layers is named *Fully Connected Network*. A simple example of a fully connected network with two hidden layers is shown in figure 2.2.

In ANN, the neurons are represented as weighted nodes. A weight forms a connection between nodes from two adjacent layers while nodes within same layer share no connections. The magnitude of weight values indicates the importance of input node, and larger values means more importance. For example, in order to predict a cat, the nodes that are associated with typical facial features or body shapes of cats are given with large weights. The final output of a single neuron located in a given intermediate layer is calculated through two steps. The first step is to compute the weighted sum of its inputs, and add a bias term. The second step is to process the result of first step through an *Activation Function* which applies non-linear transformations on the intermediate results of networks in order to make the them learn complex functional mappings between the

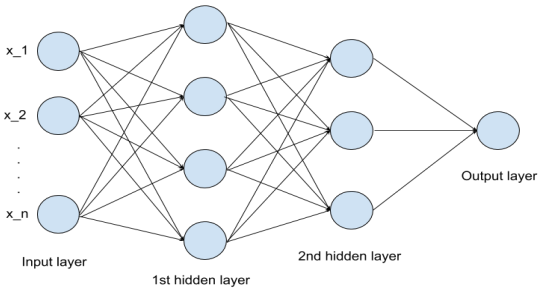


Figure 2.2: *A simple network consisted of two fully connected layers.*

inputs and the corresponding labels. The activation function commonly used is *Rectified Linear Unit*(ReLU), which zeros out negative values of its inputs, defined as:

$$R(x) = max(0, x) \tag{2.1}$$

2.3 Convolutional neural networks

Convolutional neural networks(CNNs) is a special type of feed forward neural networks which has been widely used in computer vision, suitable for image analysis. Using fully connected network to extract and learn particular patterns of images would usually entail large time and memory intensity. Imagine an RGB image with resolution of 1000×1000 pixels is used as input to a fully connected network, only the input layer needs to locate 3 millions$(3 \times 1000 \times 1000)$ nodes already. Then, each node at the first hidden layer would be connecting to 3 millions nodes and if the predefined number of nodes in this layer is set to 1000, the total number of parameters becomes equal to 3 billions which requires 12GB (3 billions $\times$ 4 bytes per floating point) of memory to be stored. Keep in mind that there is only one hidden layer stacked yet. To tackle the challenge as mentioned above, CNNs introduces *Convolutional Layers* as the most important element that is responsible for most of the computational workload. A convolutional(*Conv*) layer uses a set of filters which contain trainable neurons to extract information. Each filter has a small spatial size and maps a local region called *Receptive Field* of input volume into a single value. The mapping covers the width and height dimensions, and always extends through the whole depth dimension of the input volume because the visual appearance of an image is usually represented with several channels. During the forward pass, each filter slides or convolves across the width and height dimensions of the input volume at any position, computing the dot products of the receptive field and the filter, both are represented as three-dimensional matrices(whole depth included). The completion of the process results in a two-dimensional feature map. Figure 2.3 illustrates how the convolutional operation operates.

Through training, each filter learns to recognize certain local patterns of the input volume and uses the corresponding feature map to register what has been extracted. These feature maps are then stacked together across depth dimension and constitute the whole

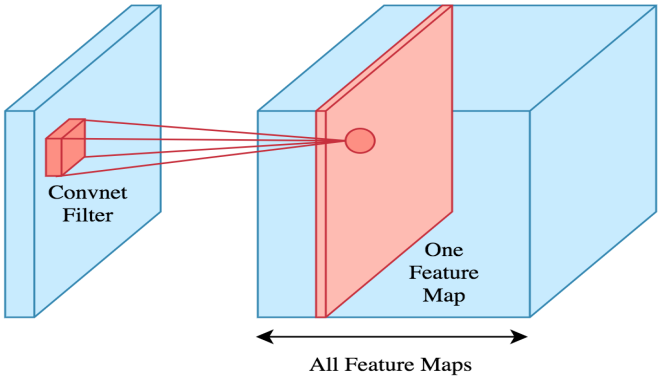


Figure 2.3: *A convolutional filter maps a small local region of an input volume through the whole depth dimension into a single node for one feature map. A Conv layer usually consists of a set of feature maps.*

Conv layer. In order to increase network performance for a given task, it is common to employ more Conv layers.

In general, four hyperparameters need to be defined to regulate the spatial arrangement of output volume before the convolution takes place. These are filter size, number of filters, stride and zero-padding. They are described as following:

filter size The size of a convolutional filter which maps the spatial regions of input volume. It includes both sizes of height and width since the whole depth is mapped by default.

Number of filters This parameter corresponds to the depth of the output volume, i.e. the number of activation maps since one filter generates one activation map. For the same receptive field of input volume, different neurons along the depth dimension of the convolutional layer can be activated if the learned patterns have emerged.

stride The stride value defines how far the filters move from one position to the next. When the stride is 1, the filters are allowed to shift one pixel at a time. A large stride diminishes the overlapped areas as the filters convolve and thus produces an output with downscaled spatial size. Usually, the same stride value is used on both height and width dimensions.

zero-padding This is another factor that effects the spatial size of a convolutional layer. Zero-padding refers to the operation of padding the input volume with zeros along its borders. This hyperparameter is necessary due to the fact that convolution become less as the filters slide towards the borders. In addition, zero-padding allows the filters to produce output with certain spatial size.

In summary, the size of width or height of an activation map can be calculated by the following equation:

$$output\_size = \frac{input\_size - filter\_size + 2 \times zero\_padding}{stride} + 1 \tag{2.2}$$

In addition to convolutional layer, CNNs usually inserts *Pooling Layers* between adjacent convolutional layers. Pooling layers intentionally reduce the spatial size of the feature maps and decrease the number of learnable weights of networks, thus the overall computational complexity is reduced. The pooling operation resembles convolution in the way that every neuron in the pooling layer is associated with a square-sized receptive field of the input volume, but these spatial subregions do not overlap each other. Unlike the Conv layer, a pooling layer is unparameterized, meaning that the pooling operation does not require any learnable weights during training but perform a fixed function on the inputs instead. Furthermore, the same pooling operation is applied on each activation map along the depth dimension of Conv layer so that the depth is preserved. The most common pooling operation is *Max Pooling*, where each neuron in the pooling layer only maps the maximum value of corresponding receptive field. Another variant of pooling operation is *Average Pooling*, where each neuron in the pooling layer uses the mean value of corresponding receptive field. Figure 2.4 shows how these two pooling operations work.

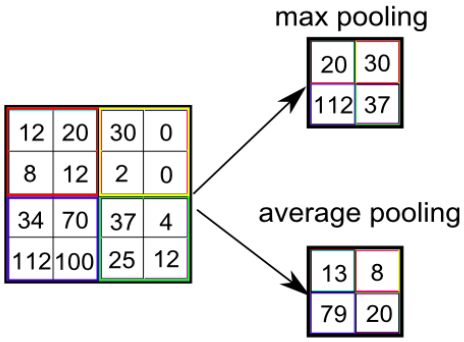


Figure 2.4: *Unparameterized pooling operations. Max pooling of* 2×2 *kernel size maps the maximum value of the corresponding receptive field while average pooling maps the mean value of the receptive field.*

2.4 Deep CNN architectures

Most of the current modern deep convolutional neural networks(DCNNs) are derived from the simple network architecture, named LeNet-5 [25], which made a major breakthrough in the field of object recognition using convolutional networks. Originally, LeNet-5 was used for recognizing digits from one-channel(grayscale) hand-written images of 32×32 resolution. Figure 2.5 depicts the simple structure of LeNet-5 which is consisted of seven layers including input layer. It is a relatively shallow network compared to the modern architectures. The hidden layers starts with two Conv layers of 5×5 kernel size, stride 1 and no zero padding followed by an average pooling layer of 2×2 kernel size respectively. At the end, a set of two consecutive fully connected layers is attached.

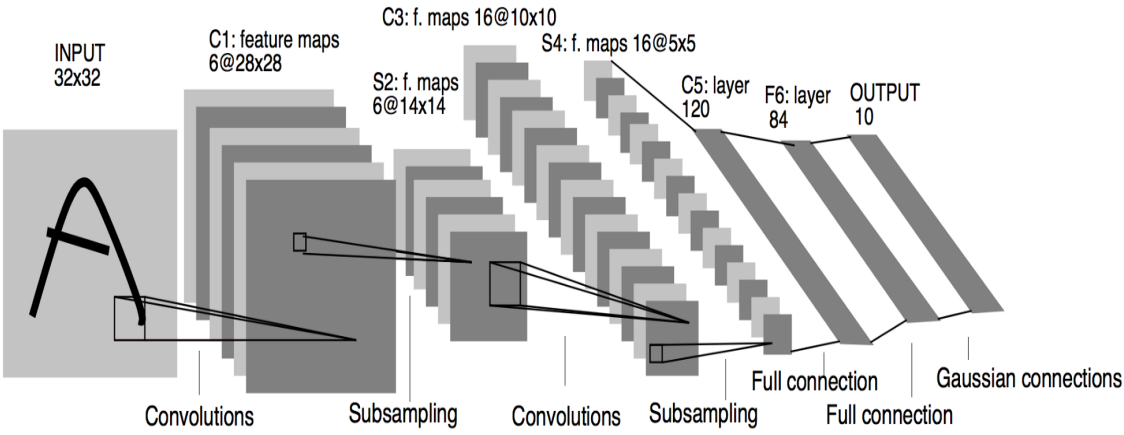


Figure 2.5: *LeNet-5 network consisted of seven layers including the input layer. Image is taken from [25].*

The advent of *AlexNet* [21] paved the way for modern DCNNs. It was the first DCNN that won the image classification challenge ImangeNet ILSVRC2012[1], and significantly outperformed the traditional object recognition approaches. In contrast to LeNet-5, AlexNet has more layers with larger size as shown in figure 2.6. In addition, AlexNet has proved the viability of connecting two Conv layers without inserting a pooling layer in between. After 2012, many nets inspired by the AlexNet have been created, aiming to further improve the performance of AlexNet and continue to tackle the ILSVRC challenge. Three of the most promising nets are VGGNet(second place of ILSVRC2014), GoogLeNet(first place of ILSVRC2014) and ResNet(first place of ILSVRC2015).

GoogLeNet [45] is an DCNN architecture developed by Google team in 2014. It is an improvement of AlexNet with deeper architecture but less parameters. The radical decrease of number parameters is achieved by replacing the first fully connected layer with a pooling layer at the end of the network. Most importantly, GoogLeNet introduced a new type of layer called *Inception Modules*, which combines different scales of an input within the same layer and employs 1×1 Conv layer [29] to perform dimension reduction,

[1] http://www.image-net.org/challenges/LSVRC/2012/

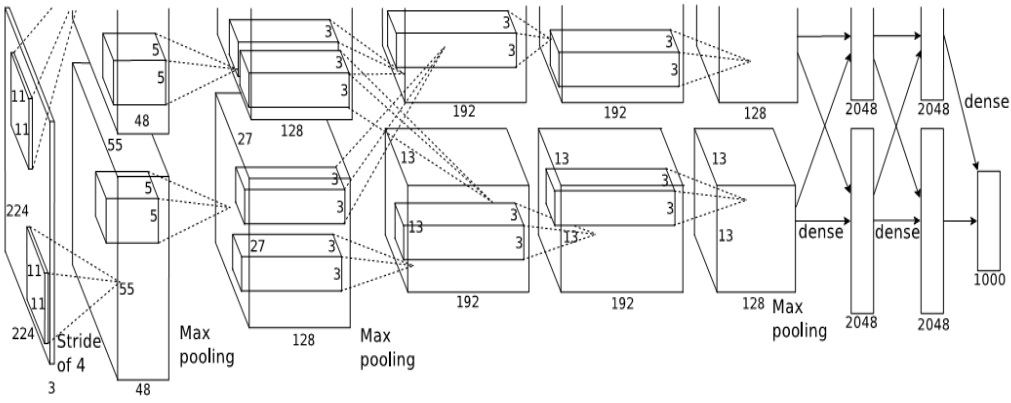


Figure 2.6: *AlexNet architecture overview.it has more layers with larger size compared to LeNet-5. Image is taken from [21].*

see figure 2.7. As a result, the model has the ability to recognize patterns at different scales.

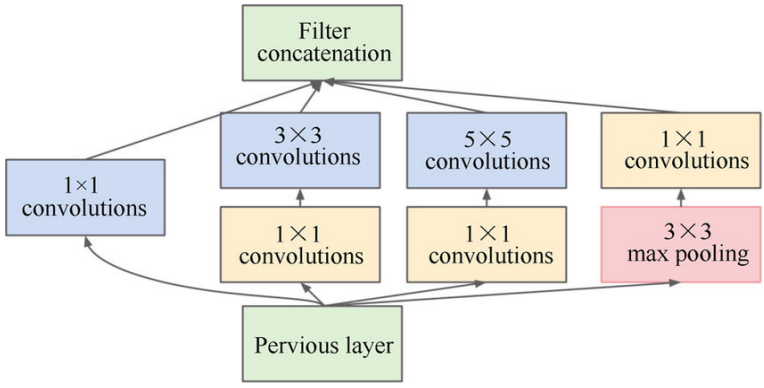


Figure 2.7: *The structure of an inception module of GoogLeNet. The adoption of Conv layers of different kernel sizes within a same layer enhances the model recognition ability for multi-scaled objects. Further, the use of* 1×1 *Conv layers helps with dimension reduction. Image is redrawn from [45].*

In the same year, another state-of-the-art network was released, named *VGGNet* [42]. The main contribution of VGGNet was the proof of that the deeper networks does generally provide better results, which was an arguable topic at that time. There are two versions of VGGNet, named VGG16 and VGG19, containing 16 and 19 weighted layers respectively. All the Conv layers have the same configurations, i.e. 3×3 kernel size, 1 stride and 1 zero padding. Figure 2.8 shows that two or three such Conv layers are put in sequence before a pooling layer in VGG16, which provides advantageous effects on

the network training. For example, three consecutive Conv layers introduces more non-linearities(more ReLUs) and can produce an 7×7 receptive field on the input volume with less parameters compared to use a single 7×7 Conv layer$(3 \times 3 \times 3 = 27 \; vs \; 7 \times 7 = 49)$. A downside of VGGNet is that the most of its parameters are in the last three fully connected layers, making the network hard to train.

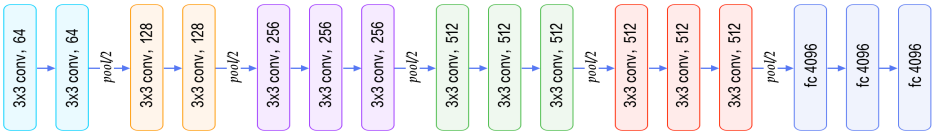


Figure 2.8: *VGG16 architecture overview.*

Kaiming He et al. invented *Deep Residual Network*(*abbr. ResNet*) [19] in 2015. The concepts behind this architecture came from the verification of that the performance does not necessarily get further improved for deeper architectures. In fact, a deep plain network can potentially generate higher training error compared to its shallower counterparts. This is due to the so-called degradation problem [3] which occurs when more layers are added to a network where the accuracy has been saturated. To address it, a special block of Conv layers is introduced, named *Residual Block*. This block applies so-called *shortcut connections* [4][48] to connect two non-consecutive layers. The output of a residual block is the sum of two separate values. The first one is obtained as the input has been through a series of Conv layers, ReLUs and batchnorm layers [20]. The second one is the input itself, which forms the so-called *identity mapping*. The authors hypothesize that it is easier to learn the mapping between an input and an output by learning the difference between them, than directly learning the underlying mapping. Figure 2.9 illustrates how it works.

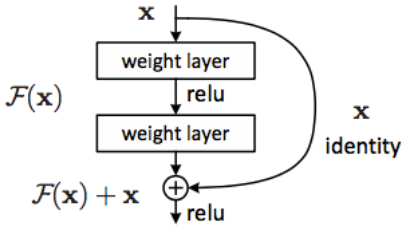


Figure 2.9: *Residual block: learn the difference(residual) $F(x)$ between an input and an output instead of learning the direct underlying mapping. When $F(x)$ is learned, it is then added to the input x to obtain the output. The goal is to improve the robustness of learning and prevent degradation problem. Image is taken from [19].*

A residual network consists of several residual blocks to ensure that the overall performance will not decline as the network gets deeper. Kaiming He et al. has recommended several versions of ResNet and the most common ones used in practice are ResNet34, ResNet50 and ResNet101 where the numbers refer to the number of layers used in respective network. In this thesis, ResNet50 is used as the base architecture for the investigated networks. The building blocks in Resnet50 are called *bottlenecks* which consist of a series of 1×1 Conv layer, 3×3 Conv layer followed by another 1×1 Conv layer. Further, several

bottlenecks of similar configurations constitute a particular block, and ResNet50 consists of four such blocks. Figure 2.10 illustrates the ResNet50 architecture with specifications of number of bottlenecks for each block, and bottleneck configurations.

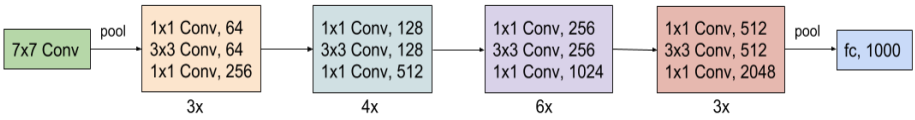


Figure 2.10: *ResNet50 architecture. It starts with an* 7×7 *Conv layer and an* 2×2 *max pooling, followed by four bottlenecks with different configurations, and they are duplicated 3, 4, 6, and 3 times respectively. Lastly, an* 2×2 *average pooling and a fully connected layer are added.*

2.5 Fully convolutional networks

The networks mentioned in the previous section are particularly designed for image classification tasks, where an image needs to go through a successively subsampling procedure until a feature vector with length equal to the number of classes is produced. This mechanism does not provide image analysis at pixel level as required by segmentation tasks. In 2014, Long et al. released the paper of *Fully Convolutional Networks*(FCN) [39] which enables networks for dense predictions. As expressed in the name, FCNs use only Conv layers throughout the whole network without any fully connected layers, figure 2.11) shows the conversion from CNN to FCN. In order to realize this concept, Long et al. have restructured VGG16 network by replacing all of its fully connected layers at the end of network with Conv layers. Note that the last Conv layer should use 1×1 kernel size since it aims to generate the score map with the number of channels equal to the number of classes, so that each pixel can be determined to belong to which class.

Since the spatial resolution of an image becomes smaller due to pooling effect along the encoding procedure during training, it is crucial to restore the original spatial size for the ultimately encoded feature maps. For that, Long et al. applied *deconvolution*, to upsample the downsampled layers, it basically does the opposite as the normal Conv layer. The deconvolutional layers can either serve as a fixed function(bilinear upsampling) or use weights to learn how to upsample.

A single deconvolutional layer can only help to restore the original content for encoded feature maps to certain extent. In the example of fully convolutionalized VGG16, the input image would be downsampled by factor 32 after have been through five 2×2 pooling layers. The restoration of original resolution would result in coarse score maps. Particularly, small objects might disappear completely through a series of pooling layers and might not be restored whatsoever. To solve this issue, Long et al. suggest the use of "skip connections" to fuse the shallower layers and the deeper layers by using element-wise summation. More specifically, as a side branch, the feature maps of shallower layers should use 1×1 convolution to generate score maps with the depth equal to the number of classes, and the feature maps of deeper layers need to be upsampled since the element-wise summation requires equal-sized inputs. This summation allows low level

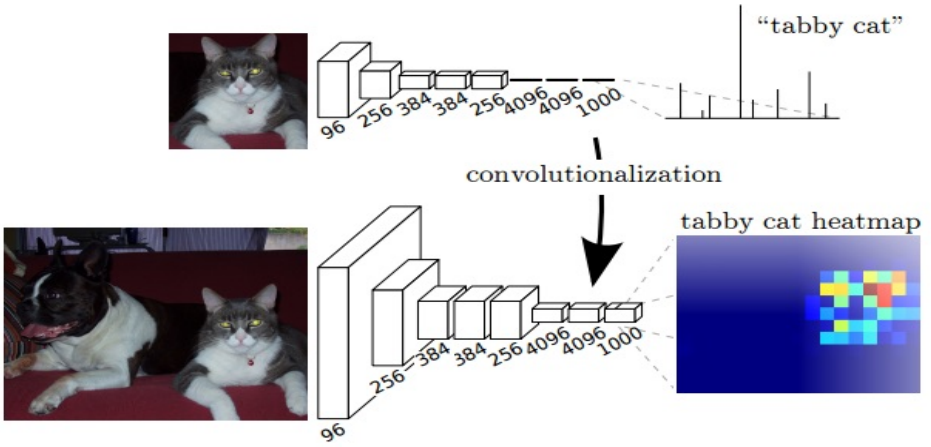


Figure 2.11: *By replacing the fully connected layers with convolutional layers and adding a deconvolutional layer, a network for classification tasks is able to produce a heatmap presenting the classification for each pixel. Image is taken from [39].*

layers with finer local information to be fused with high level layers with finer semantic information. Thus, more robust predictions can be obtained through the learning of layers containing multi-resolution information. Figure 2.12 illustrates the finest version of the VGG16-based FCNs proposed by Long et al., named FCN-8S, which uses skip connections to integrate the last output layers with two middle layers in order to achieve finer predictions. FCN8S was investigated in the base work [23] which this thesis tries to improve upon and would serve as the baseline for the experiments conducted here. More details of the baseline model can be found in section 3.1.

In addition to the enabled dense predictions, FCNs also allow the use of arbitrary-size input images which is not possible with fully connected layers. Considering the high similarities between an FCN and its corresponding CNN, the FCN can still benefit from the pretrained weights used in the CNN model. In this thesis, our experiments are mainly based on the pretrained FCN-based architectures.

2.6 Pyramid scene parsing networks

In 2016, a powerful FCN-based dense prediction framework was introduced by Zhao et al. [57], named *Pyramid scene parsing networks*(*abbr.* PSPNet), which achieved state-of-the-art performance on several segmentation datasets at that time, such as ADE20K [58] and PASCAL VOC 2012 [13]. The concept behind PSPNet comes from the exploration of inability of collecting contextual and global information by previous FCNs. Zhao et al. have specifically pointed out that the learning without taking context into consideration would lead to several failure cases of prediction, such as *mismatched relationship* where an object is incorrectly classified due to its appearance is similar to other class instances, and *inconspicuous classes* where small-scaled objects which have similar appearance to the background can be totally misclassified as the background class. For those issues,

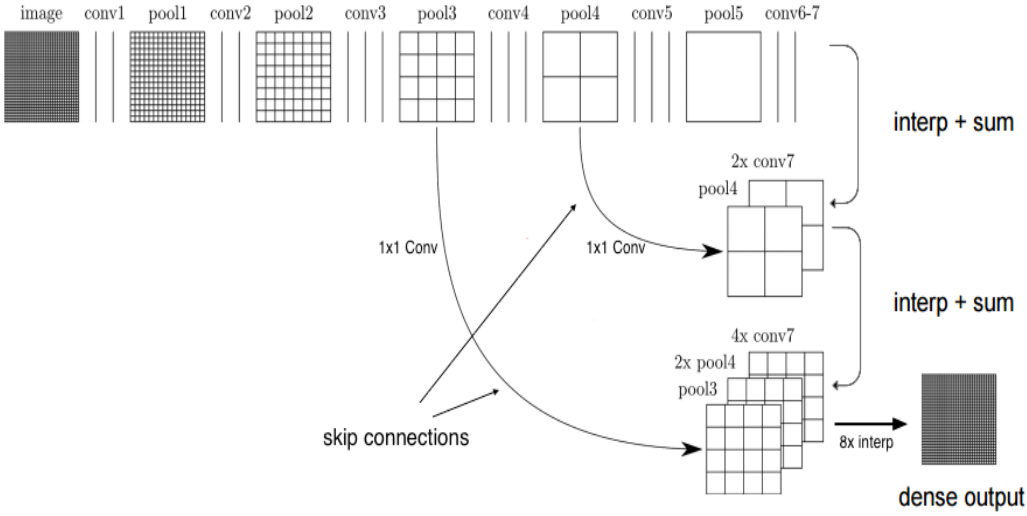


Figure 2.12: *Flow chart of VGG16-based FCN-8S. This approach allows models to take advantage of both the layers with finer local information and the layers with finer semantic consistency for making final decisions. Image is redrawn from [39].*

Zhao et al. proposed a new layer called *Pyramid Pooling Module*(PPM) which is based on the concept of spatial pyramid pooling [18] [24], the goal of PPM is to collect the contextual information for finer predictions by learning features of different scales. The PPM layer is positioned at the end of a encoder of a network, and takes the feature maps from the ultimately encoded layer as inputs. Within PPM, the inputs are first pooled down to 4 different sets of features maps of spatial sizes 1×1, 2×2, 3×3, and 6×6 respectively. Then, each set of feature maps uses 1×1 Conv layer to reduce its depth to $\frac{1}{N}$ of the total number of channels, N denotes the number of levels(in this case N=4). Finally, each set of feature maps is upsampled to the same size as the inputs, so that they can be concatenated with the inputs across the depth dimension. Thus, the PPM as a whole contains rich multi-scale information of the encoded feature maps which would be further learned by an additional 3×3 Conv layer. Figure 2.13 shows how the PPM layer works in general.

In order to ease the training process, PSPNet generates a set of score maps from one of its middle layer as a side branch and add the loss(auxiliary loss) to the final loss. In this way, the optimization of whole network can be divided into two parts where each is simpler to be solved. The auxiliary loss helps to accelerate the learning process while the main branch loss still takes the major responsibilities, a weight of 0.4(according to their ablation study) is assigned to the auxiliary loss to balance its contribution. Note that the auxiliary loss is not used during testing phase. Zhao et al. chose dilated ResNet50 as the base architecture for PSPNet, inspired by [7]. Dilated convolutoinal layer [54] aims to extract information of wider context from input layers with less cost in comparison with multi-layer integration. It uses *dilation rate* to control the distance between the pixels of receptive field as the filters convolve. The normal convolutions use dilation rate 1 which means no gap between the pixels, while the dilation rate of 2 makes a filter cover an

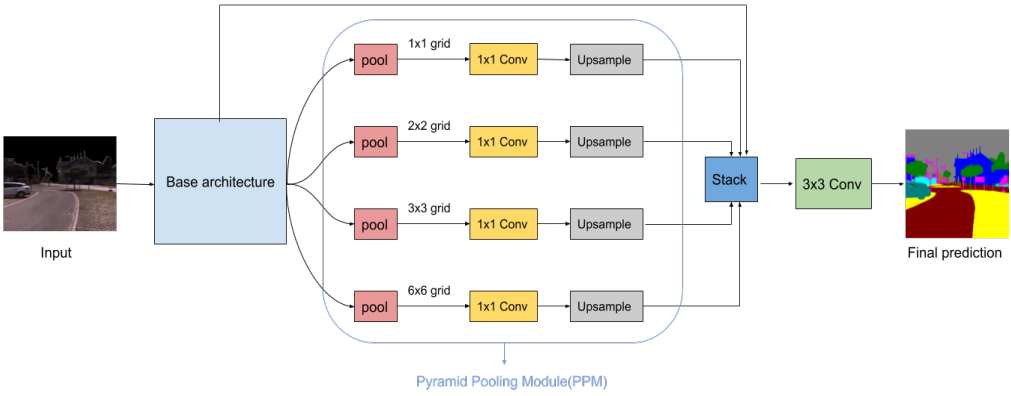


Figure 2.13: *Schematic illustration of PPM layer with arbitrary base architecture as inputs. It integrates both finer and coarser information to make the model more robust against scale change of objects, similar to FCN8S(see figure 2.12).*

enlarged receptive filed with one pixel in gap, as shown in figure 2.14.

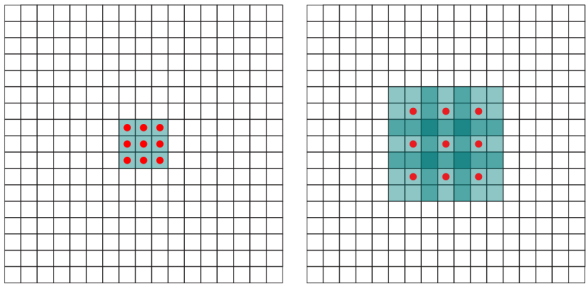


Figure 2.14: *Mapping with dilation.* ***(a)*** *uses a dilation rate of 1 which is for common Conv layers, and* ***(b)*** *adopt dilation rate of 2. Green region depicts the receptive field and red dots are the mapped values. Image is redrawn from [54].*

2.7 Discriminative Feature Network

Discriminative Feature Network(DFN) is an DNN architecture for semantic segmentation applied with new concepts of resolution restoration, the paper [53] was published by Yu et al. in early 2018. The structure of this particular segmentation network lays the foundation for the proposed fusion method(see section 3.3.3) in this thesis. In contrast to common base architectures such as VGG16 or ResNet which produce outputs through an unidirectional forward pass, DFN adopts an external network on top of its base network in order to incorporate information from all the encoded feature maps for the decoding

purposes. This external part is named *Smooth Network*(SN) as shown in figure 2.15, it forms the decoder of DFN as an U-shaped structure [37] [28] which has connections with all the blocks of the base architecture. Yu et al. suggested ResNet as the base achitecture of DFN, but the concept of *Smooth Network* can be applied upon any sort of architecture. The purpose of using smooth network is to address the intra-class inconsistency problem which happens as a network is incapable of discriminating parts of one single object due to different visual appearances and lack of contextual information. As a result, DFN with SN as the most important component, has achieved state-of-the-art performance on many semantic segmentation datasets including PASCAL VOC 2012[2] and Cityscapes [8].

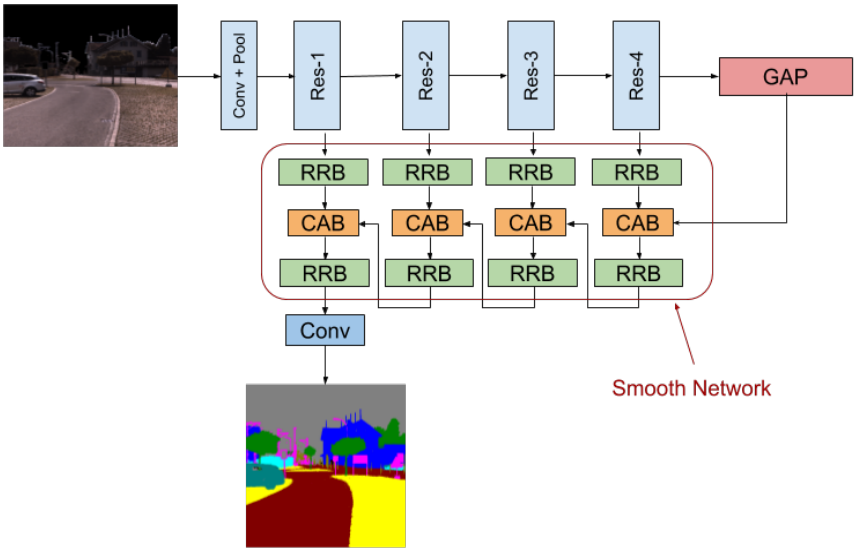


Figure 2.15: *Structure design of Discriminative Feature Network(DFN). It introduces an external network named Smooth Network(SN) for decoding purposes. SN applies Global Average Pooling(GAP) on the encoded feature maps at the beginning, and uses the coarse global information to guide the decoding procedure. How SN works internally is explained below.*

As seen above, the base ResNet contains four major blocks. Each block learns different aspects of an input image and acquires different abilities to recognize semantic and local information of objects.The feature maps of low level blocks encode finer local information but have poor semantic consistency due to small receptive field applied on the input. On the other hand, the feature maps of high level blocks have an improved ability of recognizing objects semantically as the receptive field becomes larger, but the local information becomes harder to capture. In other words, the lower level layers have higher accuracies on spatial predictions while the higher level layers focuses more on semantic consistency. In order to combine the advantages from each block, DFN utilizes

[2]http://www.pascalnetwork.org/challenges/VOC/voc2012/workshop/index.html.

SN to merge feature maps of all the blocks for optimal performance. SN mainly consists of two types of components, *Residual Refinement Block*(RRB) and *Channel Attention Block*(CAB).

Channel Attention Block Attention mechanism helps extract desired features for target tasks. For example, facial expressions can be captured by an attention mechanism [2], and document classification tasks can be addressed with an attention-based hierarchical network [51]. The structure design of CAB is depicted in figure 2.16. An CAB unit takes two inputs, and use both of them twice. The first input from left is stacked with second input from downside across depth dimension, the merged feature maps then go through a series of operations in order to generate weights of range [0, 1] for the feature maps of the first input. Lastly, the weighted first input is merged with the second input again, with element-wise summation.

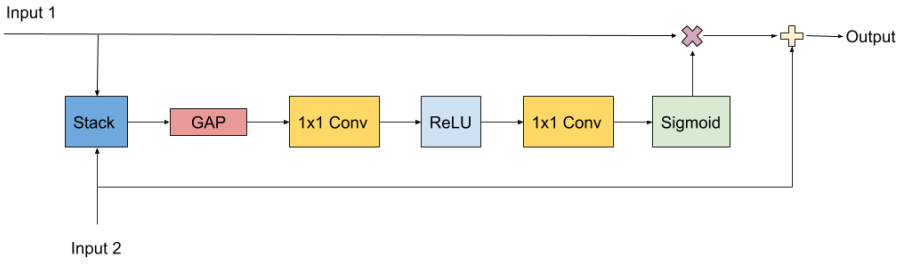


Figure 2.16: *Structure design of Channel Attention Block(CAB). It purposes to utilize high level information(input2) to guide the selection of most effective feature maps of low level blocks(input1). The Sigmoid function is the core operation which assigns weights to feature maps of input1.*

Refinement Residual Block Refinement Residual Block(RRB) enables further learning of feature maps to enhance the recognition ability. As shown in figure 2.17, the first operation within an RRB unit is an 1×1 convolution which integrates information across input depth dimension and produces an output with predefined number of channels. Then, the output goes through a basic residual block(see figure 2.9) with identity mapping. Before the exit, an ReLU is used for additional non-linearities. Note that in [53], Yu el at. proposed 512 as the number of output channels for all the Conv layers in both CAB and RRB units.

With the understanding of how CAB and RRB units function, the SN in figure 2.15 can be easily comprehended. Technically, when the forward pass through ResNet is done and the learned feature maps of each block are saved aside, the smooth network is then activated by taking two inputs from the last block where the first input is the output feature maps and the second one is same as the first input but applied with *global average pooling*(GAP). The first input is processed by an RRB unit for refinement and then is merged with the second input within the CAB in order to select the most useful and ef-

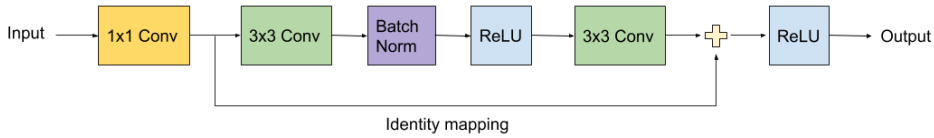


Figure 2.17: *Structure design of Refinement Residual Block(RRB). It aims to refine the inputs by relearning it with the core element, a residual block.*

fective channels from the first input. The usefulness of channels refers to how effective contextual information they can provide for the selection of useful channels of next block. For the second input, it is important to upsample it to the same spatial size as the first input in case they are not equal, so they can be stacked through depth dimension in CAB. Afterwards, another RRB is connected to refine the output of CAB and prepare it as the second input for the CAB of next block. This process is iteratively executed through all four blocks of ResNet in a top-down manner.

2.8 Improved training

In order to improve the robustness of a model against noise and bias, there are mainly two aspects one can consider applying to its training process according to the survey [36]. The first one is to generate more data since DNNs usually have a large amount of parameters which need to be tuned, and it requires a lot of data, one approach is to use data augmentation which varies the existing data in different ways. The second aspect is to improve the model training itself and there are many techniques available to do it. Two common techniques are adopted in this thesis, the first one is transfer learning which make a model take advantage of the knowledge learned by previous models for its own training in order to converge faster and generalize better. The second one is weighted loss which helps mitigate the impact of data imbalance, as seen from table, our target dataset provides highly imbalanced training data

2.8.1 Transfer Learning

Transfer learning refers to the reuse of pretrained weights on new tasks and it has become a popular method to initialize CNNs. Instead of training a network from scratch for a given dataset, it is advantageous to utilize the general features learned by the same model trained on a related task in the way that the network can obtain faster convergence and improved generalization [6]. Normally, the transfer learning works best for a target task which contains data similar to the ones used for pretraining, but transferring features from distant tasks can still be benefical for training purposes [52].

2.8.2 Data Augmentation

The amount of data is one of the key aspects for deep neural networks to achieve desired performance. In general, using more data can improve network performance while too

little data tend to cause overfitting problem [26]. However, the collection of data can be expensive and laborious in practice. To address this issue, *Data Augmentation* provides possibilities to generate more training data by adding variations to the existing data. Note that the labels of augmented data should be altered accordingly.

Data augmentation is particularly suited for image data since it is relatively easy to manipulate the visual appearance at pixel level. Several works have shown an increased performance of neural networks by adopting data augmentation [31] [41]. There are some general augmentation techniques such as flipping, rotation, cropping, scaling and color jittering, suited for many image analysis tasks. The augmentations can be applied either online or offline. Online augmentation occurs during training once an image is loaded by the system but the augmented image is not saved thereafter, while offline augmentation means that the augmented images are generated before the training session starts. Technically, the online approach does not require memory to store the augmented data but leads to slower training while the offline approach functions in opposite way.

2.8.3 Imbalanced data

Imbalanced data typically refers to the training data problem where the classes are not equally distributed. For example, for a binary classification problem where the ratio between the number of training data samples of two classes is 9:1. A classifier that always predicts its input as the first class can still yield a high training accuracy of 90%. The effect of class imbalance has been proven to be adverse on classification performance [5]. This problem is even more visually obvious for segmentation tasks since each pixel represents a class. One solution to alleviate the imbalanced data issue is to assign weights to different classes in the loss function so that an additional loss is imposed when a model is making wrong predictions on minority class during training, in other words, the model is forced to pay more attention to the minority class. There are several strategies for reweighting classes of a target dataset, one of them is called *median frequency balacing*, proposed by [11], using the following formula:

$$\mathrm{weight}(\mathrm{c}) = \frac{\mathrm{median\ frequency}}{\mathrm{frequency}(\mathrm{c})} \tag{2.3}$$

Where $\mathrm{frequency}(\mathrm{c})$ is the total number of pixels of class c of all training images divided by the total number of pixels in images where c is present, and $\mathrm{median\ frequency}$ is the median of these frequencies.

3

Method

This section presents the different network architectures and fusion methods investigated in this thesis, along with the summaries of related work for different types of fusion architectures. It starts with the description of the baseline model, followed by the elaboration of singlestream and multistream networks. Lastly, general techniques for improving network training are covered.

3.1 Baseline Models

For this work, VGG16-based FCN-8S pretrained on ImageNet is selected as the baseline singlestream model. Firstly, it is because that the same model was the investigated single stream in the base work [23] on which this thesis extends, thus using it as baseline we can preserve certain connection and comparability between this thesis and the base work. Secondly, it is a relatively simple model to implement since it adopts homogeneous 3×3 convolutional kernels and 2×2 max pooling throughout the whole network. In order to maintain fair performance comparisons of the investigated networks, we only compare singlesteam networks with singlesteam networks, and the same for multistream frameworks. Therefore, the investigated fusion method, i.e. late sum fusion described in section 3.3.2 in the base work serves as the the baseline for the investigated fusion methods in this thesis.

It is necessary to retrain the baseline model in this thesis since the base work had adopted different configurations, such as training batch size and learning rate schedule. Most importantly, the base work did not use validation set to evaluate performance during training. In this thesis, we reconfigure the baseline model to keep the settings as consistent as possible with the other investigated networks, so that the performance comparisons can provide conclusive insights.

3.2 Singlestream networks

This thesis investigates singlestream networks of dilated ResNet50-based FCN and PSPNet which in turn are used as base architectures for the investigated fusion methods. In fact, the dilated ResNet50 is the backbone of PSPNet, but it is still regarded as a separately investigated singlestream network since one of the research aims is to explore how the selection of singlestream architecture affects our proposed fusion method(see section 3.3.3). How the proposed fusion architecture is applied on different base networks depends on how similar they are. Base networks with high similarities can possibly use the proposed fusion architecture in the exactly same way. Since the whole dilated ResNet50 is a large part of PSPNet, we can apply proposed fusion architecture on them in a similar way, which helps with the analysis of the research aim mentioned above.

All the singlestream networks are more or less initialized with pretrained weights in this thesis. Due to the unavailability of exact pretrained counterparts of dilated ResNet50-based FCN and PSPNet, we use pretrained weights from the common ResNet50. The similar transfer learning strategy has been adpoted in the FCN paper [39], where a VGGNet-based FCN benefits from the pretrained weights(on ImageNet [21]) of the common VGGNet. Figure 3.1 shows the use of pretrained weights for the dilated ResNet50.

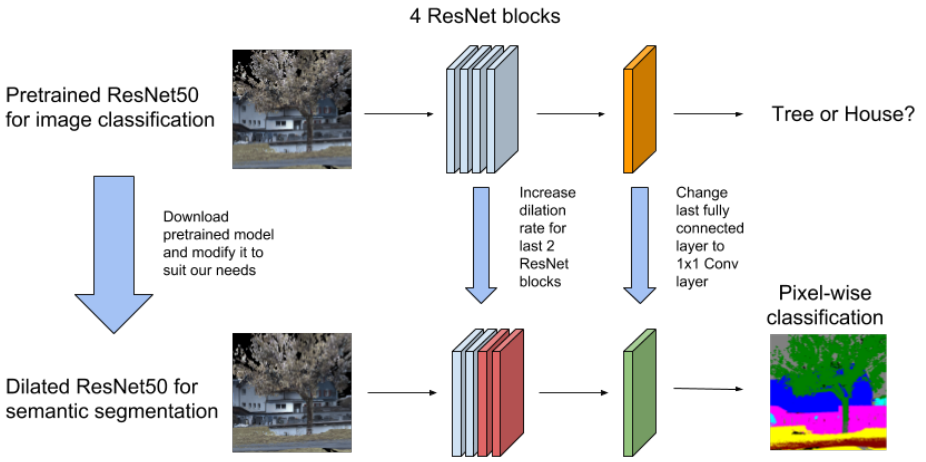


Figure 3.1: *Transferring pretrained weights from ResNet50 aimed for image classification to dilated ResNet50-based FCN for semantic segmentation.*

3.3 Multistream fusion

In this thesis, we use three different traditional fusion methods to combine the three parallel singlestreams representing *color*, *depth*, and *surface normals* respectively. All the input images are of RGB format and all the singlestream networks within a given fusion framework should adopt the same architecture, i.e. dilated ResNet50-based FCN or

PSPNet. This section presents related work for multistream fusion, and describes how the investigated traditional fusion methods and the proposed fusion approach are implemented.

3.3.1 Related work

According to [50], late fusion approaches are in general preferred over early fusion methods because of two main reasons. Firstly, early fusion concatenates feature representations at input stage which can result in high depth dimension, making the learning difficult for the classifier. Second, late fusion provides higher flexibility in modeling since multiple singlestreams can use various architectures and configurations. In addition, the late fusion methods offer more scalability which eases the process to add or remove modalities [1], it is difficult to achieve for the early fusion. Several examples of how others have applied late fusion strategies are listed as follows.

[55] investigated five different fusion variants to combine multiple singlestream networks. These fusion variants refer to element-wise summation, maximum fusion, stack over depth dimension, stack over height dimension and stack over width dimension respectively. All of them can be applied at different layers of each stream followed by a Conv layer to learn the fused representations. In contrast to the different input modalities used in this thesis, they used the same input for each singlestream with different configurations, e.g. one stream may use max pooling throughout the entire network while the other one may use average pooling. Although this type of multistream network represents a model ensemble [10] rather than a mutlimodal framework which this thesis studies, their approaches for integrating multiple singlestreams are still worth investigating for us since the concept of fusion is applicable for any type of multistream network.

[46] and [47] studied multistream fusion by using an external layer called *gating network* in order to balance the contribution of different singlestreams for final predictions. The gating network has to learn certain stacked feature representations from all the singlestreams, so that it is able to assign different weights to their score maps during forward pass. Note that the gating network is learned after all the singlestream networks have been trained separately, and the weighted score maps are then merged through element-wise summation followed by a 3×3 Conv layer for further learning.

In [30], the input modalities represented in text and image of personal information are combined in different ways. Interestingly, in addition to the common fusion method of element-wise summation, the methods of element-wise multiplication and combining both the element-wise summation and multiplication are investigated. The multiplicative fusion method improves the tolerance ability of multistream framework for the mistakes made by the weak modalities and prevents overfitting. As a result, the use of both summation and multiplication fusion methods in a single multistream network achieved the highest performance rates, showing that not only different modalities can be used, different fusion strategies can be combined also.

[32] introduces *RDFNet* which applies a fusion strategy similar to the proposed fusion method(see section 3.3.3) used in this thesis. The main difference is that RDFNet does not have CAB units(see section 2.7) to help with the decoding. In order to learn the merged representations, they use RefineNet [28] units instead. As a result, RDFNet achieved state-of-the-art performance on several multimodal datasets such as [40] and [44].

3.3.2 Late fusion variants

The investigated traditional fusion methods in this thesis are three different late fusion variants, *Late sum fusion*, *Late max fusion*, and *Late fused convolution*. The word *late* of the names indicates that these fusion methods starts the merge at the final score maps of all singlestream networks. Here, they are performed on the multistream frameworks consisited of singlestreams with same architecture, in order to help with the effective performance comparison, but in general, the fusion methods can be applied on any architectures as long as their final feature maps have the same size. In this thesis, we always use all the three input modalities for fusion since both the correlation and independence of different modalities can provide valuable insight for decision-making process [1].

In this thesis, we denote the layers within the three singlestreams $M^i \in R^{H \times W \times D}$ where $i \in (1, 2, 3)$, and H, W, D refer to the height, width, and depth dimensions of the feature maps. A fusion function can be generally denoted as g which is applied upon the three streams: $g(M^1, M^2, M^3) \rightarrow o$ where o is the fused output feature maps. Note that the g is applicable at layers of multistream network where feature maps have the same H, W and D. The following describes the investigated late fusion variants:

Late sum fusion(LSF) At the final score maps of a multistream network, the LSF method, $o^{LSF} = g^{LSF}(M^1, M^2, M^3)$, combines the feature maps from all the three streams by element-wise summing up values, and locations of H, W, D are denoted as *h, w* and *d* respectively.

$$o^{LSF}_{h,w,d} = M^1_{h,w,d} + M^2_{h,w,d} + M^3_{h,w,d} \tag{3.1}$$

where $1 < h < H, 1 < w < W, 1 < d < D$ and $M^1, M^2, M^3, o^{LSF} \in R^{H \times W \times D}$

Late max fusion(LMF) Similar to LSF, the LMF method, $o^{LMF} = g^{LMF}(M^1, M^2, M^3)$, but selects the largest value of corresponding feature maps of all the streams at *h, w* and *d*.

$$o^{LMF}_{h,w,d} = max(M^1_{h,w,d},\ M^2_{h,w,d},\ M^3_{h,w,d}) \tag{3.2}$$

Late convolution fusion(LCF) This is an extension of LSF. It uses the summed score maps at the final Conv layer as the input to a new series of batchnorm layer, ReLU activation function and 1×1 Conv layer. Basically, the intuition is that the LCF inserts new neurons at the end of a multi-stream framework to learn the summed representations of all the singlestreams. Hence, it can be expressed as the equation 3.1 being wrapped in a convolutional function denoted as $conv$.

$$o^{LCF} = conv\big(o^{LSF}\big) \tag{3.3}$$

Prior to any multistream fusion, each singlestream should have been trained on a given modality and its corresponding weights are saved in a separate file. During the training of fusion, the weights of all singlestream networks are loaded back to new model instances and all the singlestreams should be frozen, i.e. unable to update the gradients, because learning multiple singlestreams at same time would require huge memory usage.

Since LSF and LMF do not insert new layers for further learning, it is reasonable to directly apply them on test images to obtain the unbiased performance rates. However, LCF employs additional learnable layers which need to to be trained from scratch though. With frozen singlestreams, this training should be quickly completed within a few number of iterations. The practical implementation of LCF on top of a three-stream framework is illustrated in figure 3.2.

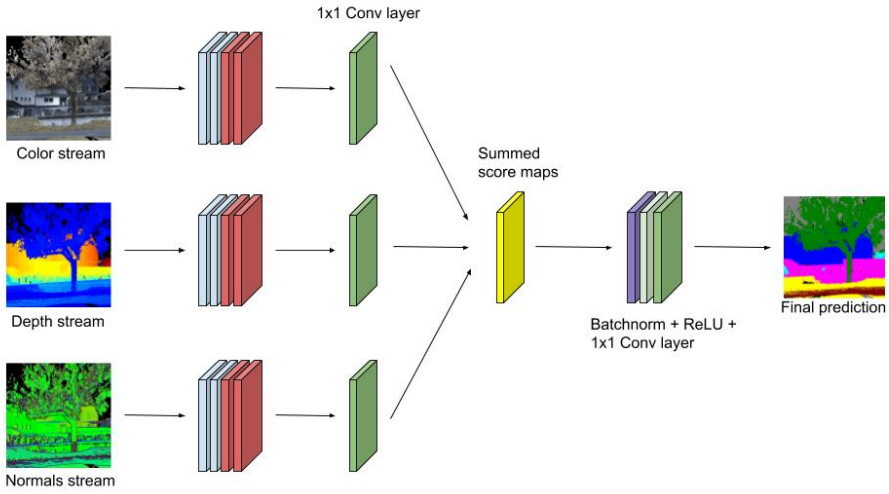


Figure 3.2: *LCF framework consisted of singlestreams with dilated ResNet50. LSF and LMF combine singlestreams in a similar way that final batchnorm, relu and conv layers are removed, and LSF remains the yellow summed score maps while LMF uses max function instead. Same fusion concept can be applied on PSPNet singlestreams.*

3.3.3 Proposed fusion architecture

The proposed fusion approach is greatly inspired by the *Smooth Network* of *Discriminative Feature Network(DFN)* [53] which has been described in detail in section 2.7. Basically, the idea is to use a SN-like decoder for our multistream frameworks in order to better utilize the contextual information and semantic consistency of all the encoded layers for the desired fusion performance. Same as the late fusion variants, all the singlestreams shall be frozen during the training of the components/layers of the proposed one.

We investigate only one version of proposed fusion method on the dilated ResNet50-based multistream framework and three different versions on PSPNet-based frameworks, these experiments are denoted as Dilated_ResNet50_v1, PSPNet_v1, PSPNet_v2 and PSPNet_v3 respectively. Since PSPNet includes a PPM layer, it is of interest to investigate more on how our proposed fusion method can integrate with the PSPNet, or specifically, the pyramid pooling module. The following elaborates on the explanation of each

experiment.

Dilated_ResNet50_v1 Figure 3.3 illustrates the proposed fusion architecture applied upon the dilated ResNet50-based fusion framework, where each of the four major ResNet blocks of a given stream is combined with the corresponding blocks of other streams by stacking across the depth dimension. Note that the final ResNet blocks of all streams are stacked twice with the first stack activating the fusion process. In contrast to the original smooth network which starts from the final ResNet block going through the global average pooling(GAP) directly, we first refine the stacked final ResNet blocks by using an RRB unit before they are downscaled to 1 spatial resolution by the GAP. In fact, all stacked layers are processed by a subsequent RRB unit. This is inspired by the architecture of PSPNet [57] where the stacked layer consisted of PPM and final ResNet block is learned by a 'chunky' Conv layer. Further, we do not use bilinear upsampling operation for the corresponding Res-3 and Res-4 layers since the spatial size of the layers do not change after Res-2. Otherwise, the proposed fusion architecture is the same as the SN of DFN. The goal is to let all the channels from the same blocks of all the singlestrams interact with each other and together contribute to the decoding procedure, and it is achieved by stacking these channels and then filtering out the most useful ones to guide the selection of useful features at the subsequent lower block throughout the whole encoder. In this way, the image restoration loss can be minimized. During the implementation, we need to define the number of output channels for all the Conv layers within SN to enable the fusion. Thus, it is necessary to carry out a small ablation study on this parameter. Detailed experimental configurations can be seen in section 4.2.3.

PSPNet_v1 The first version of the proposed fusion method applied on PSPNet is overall the same as the Dilated_ResNet50_v1(see figure 3.3). Note that the PPM layer needs to be removed during implementation. The idea behind PSPNet_v1 is to investigate how the fusion performance is impacted by the singlestream networks trained for different purposes. For the dilated ResNet50 FCN, the blocks are trained to obtain the optimized final score maps while the same architecture within the PSPNet are trained to provide the optimized contextual information for the PPM layer, which in turn is further learned to produce the final prediction.

PSPNet_v2 The second version is similar to the first one but with PPM layer involved, as seen in figure 3.4. The difference is that instead of starting the fusion by stacking on Res-4 layers, we stack on the Conv layers attached to PPM, i.e. the learned representations of PPM of each singlestream, while other connections remain the same. This version does indeed reduce the memory usage a bit since the Conv layer after PPM produces 512 feature maps while the Res-4 layer produces 2048, therefore the RRB unit prior to GAP uses less parameter. It is of interest to investigate how the PPM layer can help the GAP with more effective global information generation to guide the SN, considering the stacked PPMs contain rich contextual information from different modalities.

PSPNet_v3 In contrast to the second version, the final version removes the GAP operation totally. The new design of PSPNet_v3 is depicted in figure 3.5. Since GAP compresses the spatial size of all feature maps to 1×1 resolution, a lot of valuable contextual information might be lost. Thus, it is of interest to explore the impact of a non-GAP approach for our fusion method. To do so, we directly use the stacked layers of learned representations of the PPMs as the starting point to guide the SN, since they preserve much contextual and global information. Note that during implementation, all PSPNet-based fusion frameworks apply the same number of output channels of the Conv layers within SN because of the highly similar architectures.

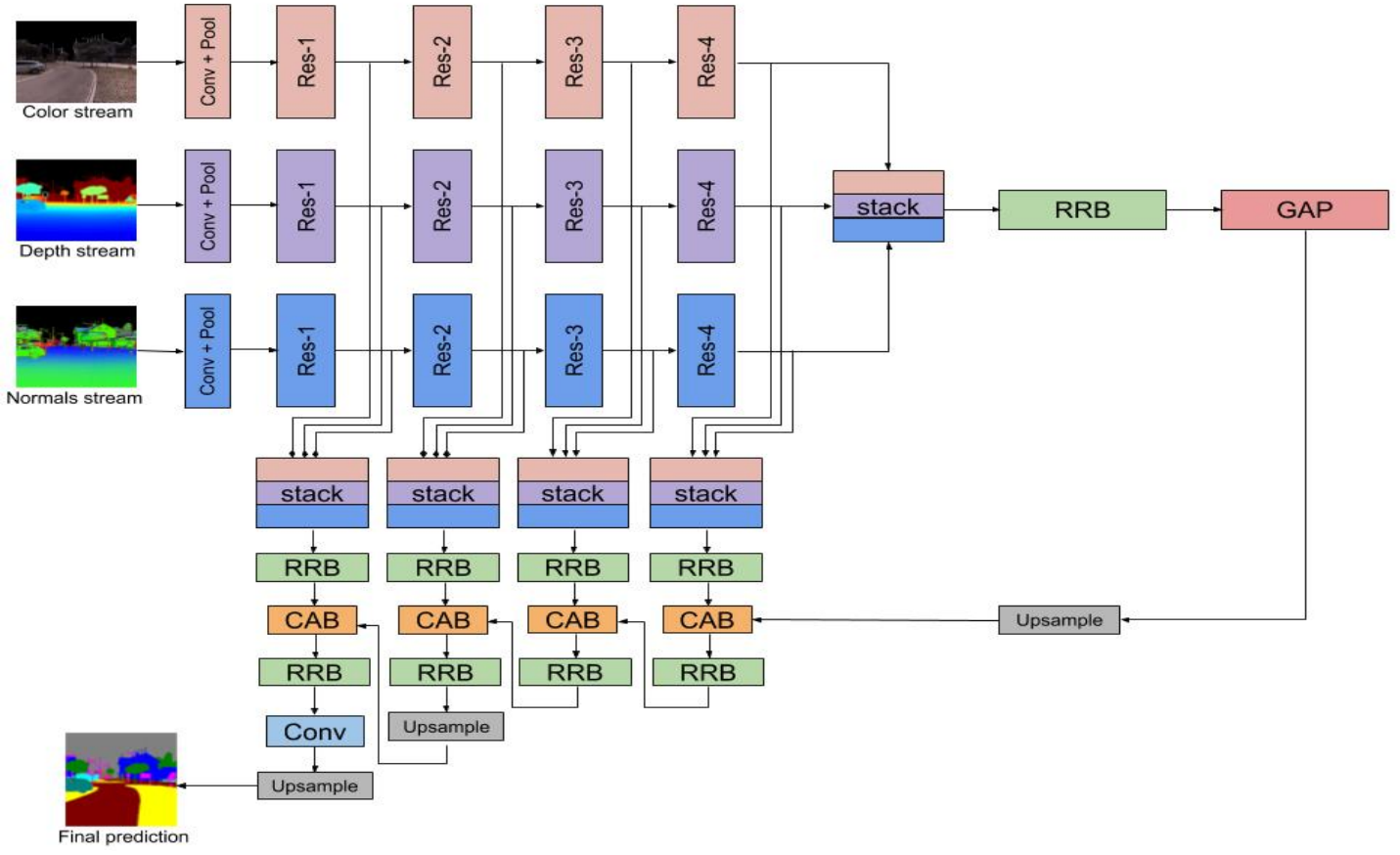


Figure 3.3: *Design structure of Dilated_ResNet50_v1. The SN-like network combines each of the four major ResNet blocks of a given stream with the corresponding blocks of other streams, by stacking across the depth dimension. Res-4 layer of all the singlestreams are stacked twice with the first stack aiming to activate the fusion process by means of global average pooling. All the layers marked by 'stack' need to go through an RRB unit to refine the merged information. Upsampling is applied to enable the merge of layers from different stages. Note the the dilated ResNet50 does not change the spatial size after Res-2 stage. The CAB units are important since they help with the channel selection for the purpose of optimal resolution restoration. This overall architecture is used by PSPNet_v1 as described below.*

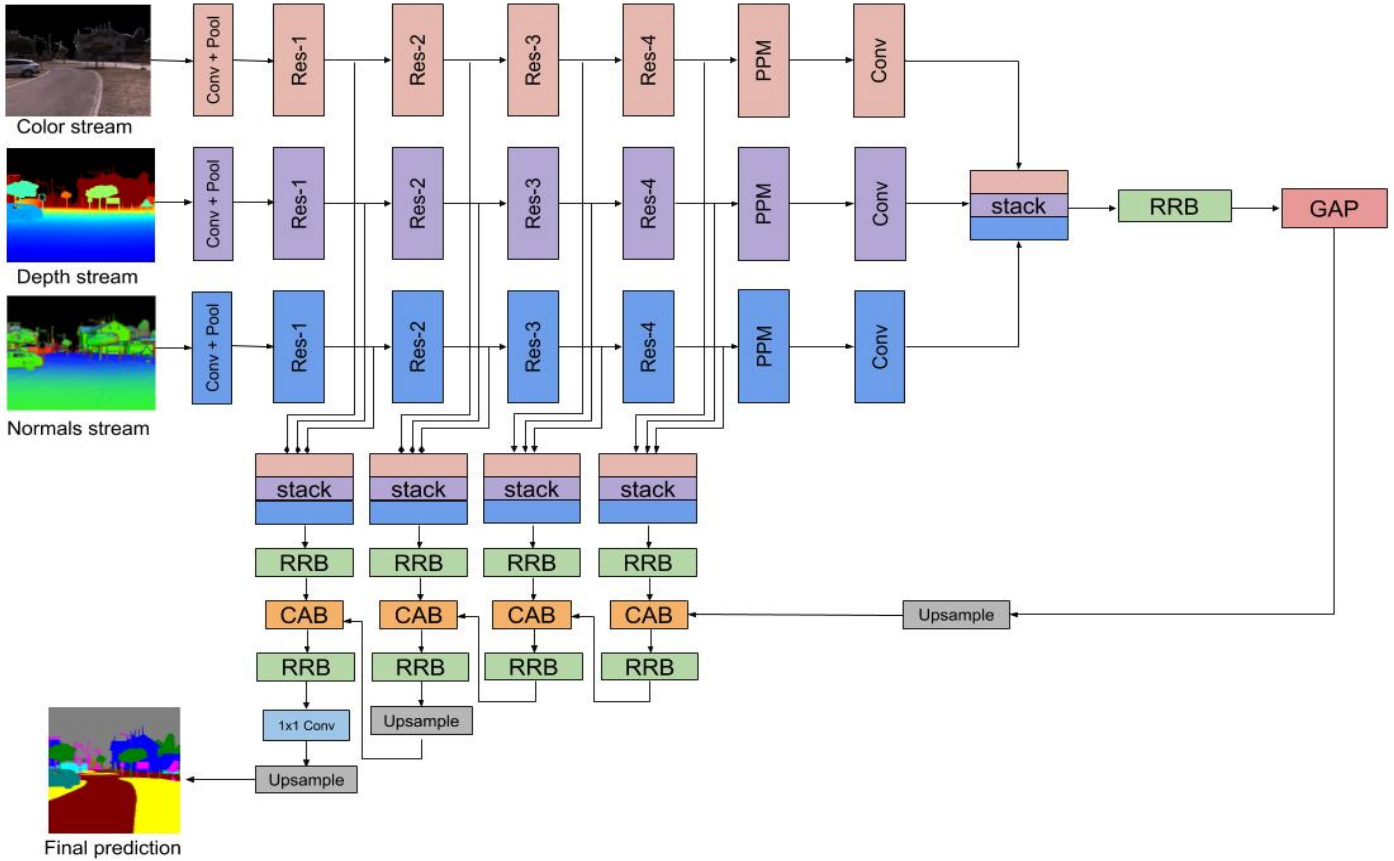


Figure 3.4: *Design structure of PSPNet_v2. The SN-like network combines each of the four major ResNet blocks of a given stream with the corresponding blocks of other streams, by stacking across the depth dimension. The same stacking operation is used to the Conv layer subsequent to PPM in order to enhance the effectiveness of GAP output to start the decoding procedure. All the layers marked by 'stack' need to go through an RRB unit to refine the merged information. Upsampling is applied to enable the merge of layers from different stages. Note the the dilated ResNet50 does not change the spatial size after Res-2 stage. The CAB units are important since they help with the channel selection for the purpose of optimal resolution restoration.*

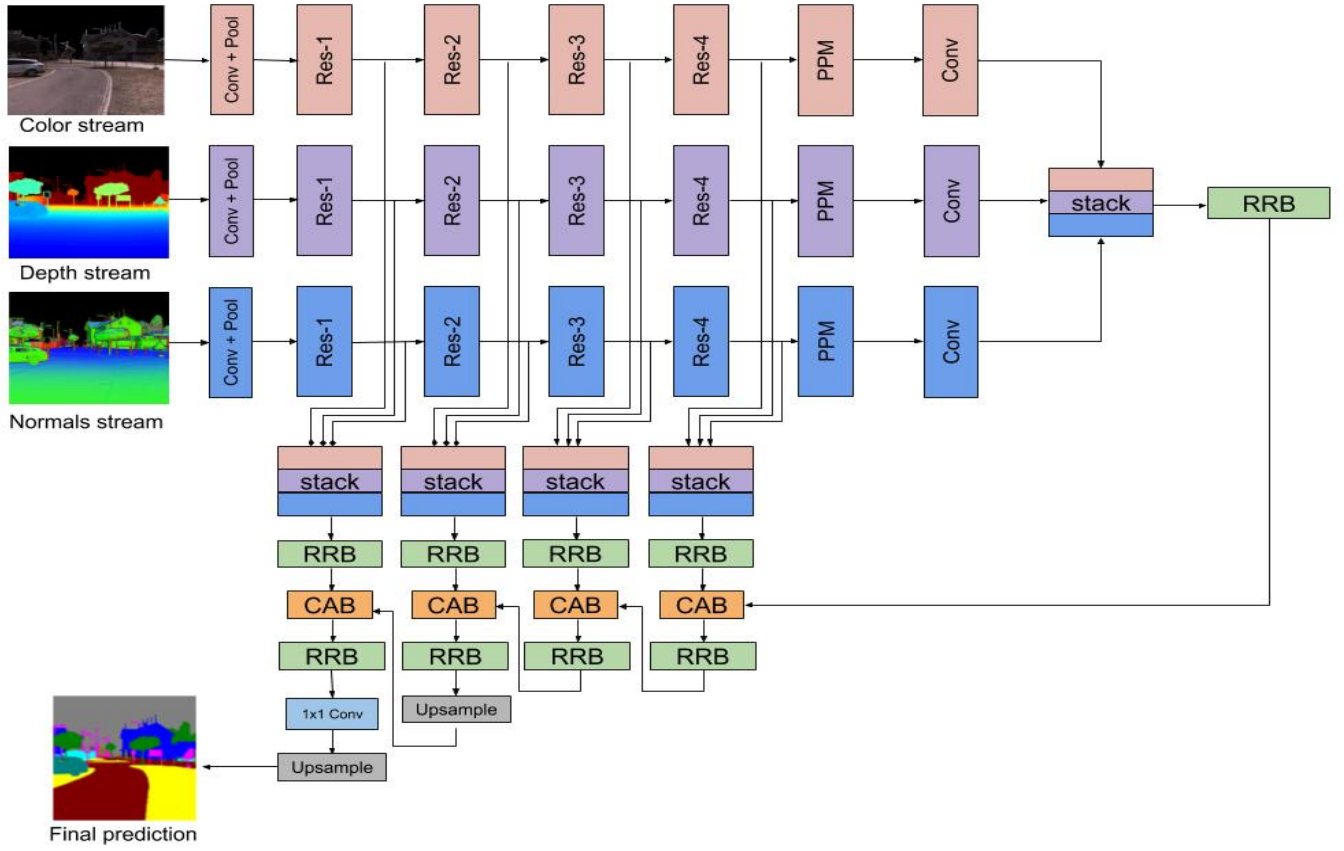


Figure 3.5: *Design structure of PSPNet_v3. The SN-like network combines each of the four major ResNet blocks of a given stream with the corresponding blocks of other streams, by stacking across the depth dimension. The same stacking operation is used to the Conv layer subsequent to PPM in order to enhance the effectiveness of entry point for the decoding procedure. The GAP(see previous figure 3.4) is removed from SN with the purpose to replace the coarse global information with finer contextual information gained from the PPM layer. All the layers marked by 'stack' need to go through an RRB unit to refine the merged information. Upsampling is applied to enable the merge of layers from different stages. Note the the dilated ResNet50 does not change the spatial size after Res-2 stage. The CAB units are important since they help with the channel selection for the purpose of optimal resolution restoration.*

3.4 Early fusion

Early fusion refers to the integration of unimodal feature sets before learning concepts [43]. This operation aims to transform the integrated data into a single feature vector and use it as input to machine learning models. Comparing with the late fusion strategies, early fusion is an exception in the sense that it merges the data at beginning and applies only one singlestream for learning purposes, thus early fusion networks usually introduce relatively low memory and computational complexities. The aim of investigating early fusion experiments is to explore how smooth network(SN) [53] impacts the performance of early fused networks. In this thesis, we combine all the three input modalities(color, depth, surface normals) as a single input to the investigated singlestreams, dilated ResNet50 and PSPNet.

3.4.1 Related work

Even though late fusion methods are often favored as the way to integrate different modalities, the conclusive evidence that indicates late fusion is overall better than early fusion is not found since the model performance is highly problem dependent [34]. In [15], machine learning models are trained to recognize human emotions by combining information from two modalities, facial expression and body gesture. The authors investigated both the early fusion of feature concatenation and the late fusion by multiplication, summation and weighted criteria for the target task. As a result, the early fusion used by different machine learning models performed better than late fusion variants. Similar fusion methods are investigated in [49] for sentinent prediction problem. The input data are extracted from three modalities, i.e. audio, video and text. The early fusion obtained competitive performance rates compared to late fusion of weighted criteria. Interestingly, the fusion methods that were applied on different subsets of the three modalities had achieved higher performance rates than using all the three modalities in some experiments, which showed that not every modality can positively contribute to the final result.

3.4.2 Investigated early fusion network

As usual, we use pretrained weights to initialize the early fusion networks. Since the number of input channels has been changed, it requires manual work to alter the *input channel* parameter of the first Conv layer of the pretrained model in order to adapt the input. Pretrained models are usually trained with RGB images, meaning that the *input channel* of the first Conv layer is 3. To adapt our stacked input, we change it to 9 because of the three modalities. Figure 3.6 illustrates how SN works with an early fused network based on dilated ResNet50. For early fused PSPNet, we only investigate the most effective version of PSPNet_v1 - v3 (see section 3.3.3) with SN. How early fused networks are configured is described in section 4.2.4.

3.5 Improved training

It is of interest to further improve the overall performance of our proposed multistream fusion frameworks for competition purposes on the semantic3D benchmark. Thus, we jointly apply additional three techniques, i.e. improved transfer learning, median frequency balancing and data augmentation(see section 2.8) on the best version of PSPNet_v1 - v3. The improved transfer learning refers to the use of more effective pretrained weights, in this case we replace the weights pretrained on ImageNet with the ones pretrained on ADE20K [59], which is a dataset for semantic segmentation and contains more related classes to our target dataset than ImageNet, such as cars and buildings. During implementation, we first retrain each PSPNet-based singlestream with all of the three techniques, then do the same for the training of the proposed fusion architecture. However, this attempt does not contribute to any of the research aims of this thesis.

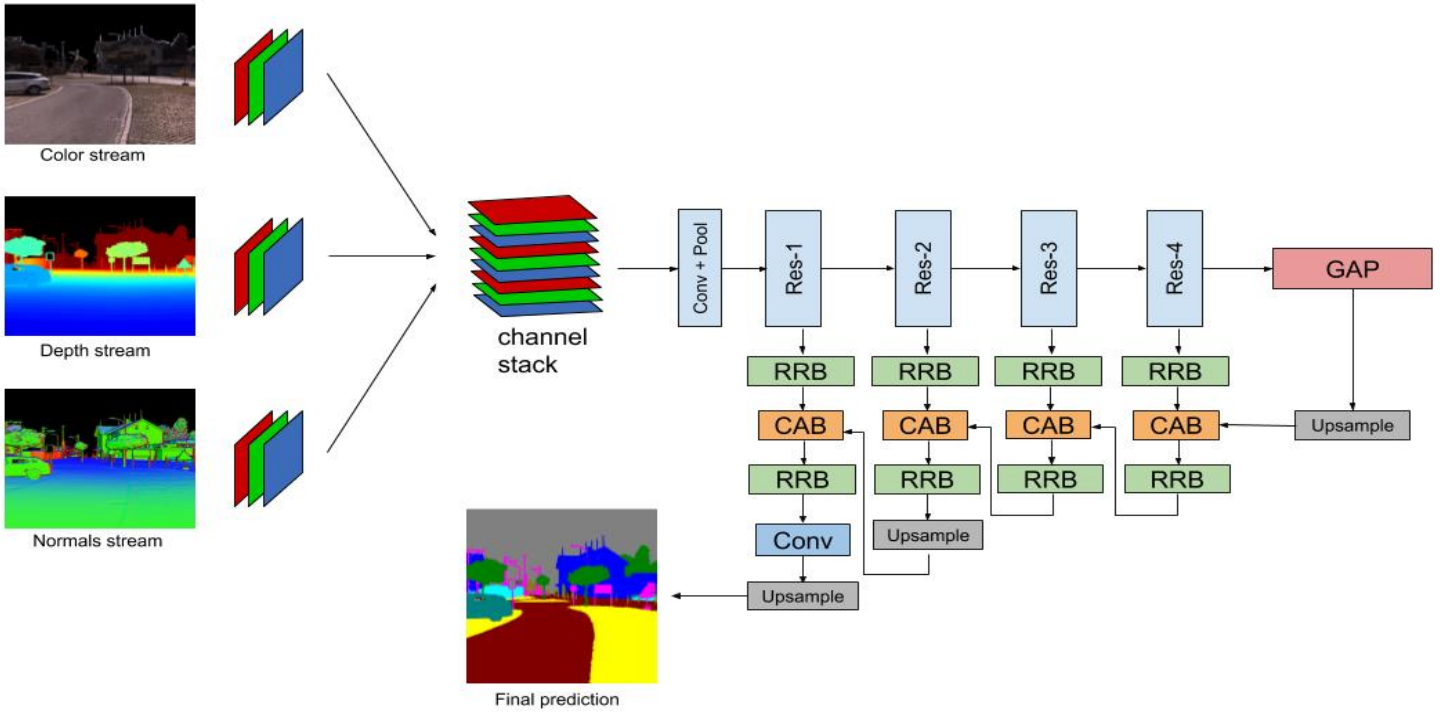


Figure 3.6: *Design structure of early fusion network with SN. At the beginning, RGB channels of all the input images are stacked through depth dimension. The first Conv layer in the pretrained model needs to change the input channel parameter to match the depth of the stacked input images, since models usually are pretrained on three-channel RGB images. During training, SN combines each of the four major ResNet blocks of a given stream with the corresponding blocks of other streams, by stacking across the depth dimension. Res-4 layer would go through GAP to generate global information to start the decoding procedure. Upsampling is applied to enable the merge of layers from different stages. Note the the dilated ResNet50 does not change the spatial size after Res-2 stage. The CAB units are important since they help with the channel selection for the purpose of optimal resolution restoration.*

4

Experiments

This chapter specifies the experimental setup for the investigated networks, and starts with the data description and generation, followed by the presentation of the network configurations. Lastly, the evaluation metrics used for the experiments are discussed.

4.1 Data description

The target dataset used in this thesis is provided from the base work [23]. This dataset contains 2D images which are projected from Semantic3D point clouds [16]. Semantic3D provides a set of large scale point clouds of outdoor environments and the point clouds were acquired by a collection of 30 individual static laser scans including over 4 billion 3D points in total. Each scan generates almost 400 millions points with semantic ground truth annotation at point-level and provides high data resolution and long distance measuring. During the post-processing phase, the colorization was conducted by using a cube map generated from camera images of high quality. The scans were performed in urban and rural scenes, targeting typical European architectures in Central Europe such as town halls and market squares. In order to map the point cloud information into the rendered views, the base work attempted to collect images by rotating the camera 360° around a fixed vertical axis. Each full rotation is consisted of 30 camera views with 12° in between. Since one such camera scan has difficulties to render all objects of varying scales and visibility on 2D images, four camera scans with different pitch angles and translations are used to address the issue and it results in a total of 120 camera views. Note that the images which have more than 10% of pixels with depth less than five meters are removed to preserve a certain amount of contextual information. In addition, images with less than 5% coverage are discarded to maintain a certain level of quality of the target datasets [23].

In addition to the projected color images, the base work generated corresponding images in depth and surface normals modalities in hope of obtaining improved model performance of 2D segmentation. All images have the same size of 500×500 pixels, the

Table 4.1: *An overview of class information of the target 2D image dataset. It presents class name, class index, class description and color annotations. The undefined pixels only appears in input images and do not contribute to the loss calculation.*

class name	class index	description	annotation color
Background	0	sky or shadow of scanning camera.	
man-made terrain	1	mostly pavement.	
natural terrain	2	mostly grass.	
high vegetation	3	trees and large bushes.	
low vegetation	4	flowers or small bushes lower than 2 meters.	
buildings	5	churches, city halls, stations, tenements etc.	
hard scape	6	a clutter class with garden walls, fountains, banks etc.	
scanning artifacts	7	artifacts caused by dynamically moving objects during the recording of the static scan.	
cars	8	cars and trucks.	
undefined	255	ignored pixels.	

value of each pixel lies in the range [0, 255], equal to the possibilities represented by an unsigned 8-bit integer.

4.1.1 Class annotation

Semantic3D uses eight classes for the point cloud benchmark challenge. The classes are: Man-made terrain, Natural terrain, High vegetation, Low vegetation, Buildings, Hard scape, Scanning artefacts, and Cars. In the projected 2D datasets, a 9th class is introduced to represent the unlabelled background in point clouds. The Semantic3D dataset provides both training and test set including 15 and 4 point clouds respectively, each cloud represents either urban or rural scene. These point cloud environments were carefully selected in order to prevent the classifiers from overfitting, for example, point clouds of urban scene contains more man-made objects while point clouds of rural scene contains more natural objects. An overview of class information is listed in table 4.1.

In addition to the eight classes defined in semantic3D, an undefined class denoting points without ground truth is used. These undefined points are very few and are transferred to the 2D datasets through projection. During training of our investigated networks, the undefined pixels are set to *ignored*(an option in pytorch) in the loss function, meaning that they are not able to contribute to the calculation of training loss. Figure 4.1 shows an example of a color image with pixel-wise annotation.

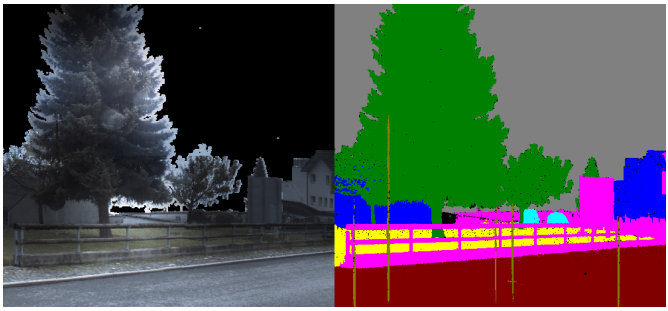


Figure 4.1: *Exanple of a 2D color image with pixel-wise annotations.*

4.1.2 Training and validation set

In our target 2D dataset, the training set contains 3142 images of same size in total with 1044 for each modality. However, Semantic3D does not have a particular validation set but allows the users to manipulate the training set as they want. To have a validation set for our training is important since it helps the model to tune the learnable parameters and prevent overfitting [35]. Unlike the training data, validation samples are not frequently processed of a given model and should not provide the model any knowledge to learn. More technically, the validation set goes through a forward pass to demonstrate the prediction strength of the model after every training epoch, but does not contribute to the loss calculation.

Out of the 15 environments with 1044 images in total, we select 12 with 845 images for training and the rest of 199 images for validation, the class distribution tables for both sets are listed in table 4.2. The selection splits the data of all the three modalities in a proper way that both the training and validation set involve urban and rural scenes. We do not select random samples or pick a certain percentage of images of each environment to form our validation set since the adjacent images are very similar to each other, caused by the overlapping effect of using rotated scanning when producing 2D images. Validating images which are similar to training data gives misleading indications of how well a model generalizes due to overfitting. Figure 4.2 shows an example of four adjacent color images captured by the virtual camera.

Table 4.2: *Class distribution of pixels for manually selected training and validation data of 2D images. Along with table 4.1 to understand the class names.*

Class distribution table(%)										
Class index / Dataset	0	1	2	3	4	5	6	7	8	255
Training set	31.9	11.4	4.5	6.0	3.8	34.6	4.4	1.4	0.9	1.1
Validation set	41.9	11.2	12.2	4.5	2.1	19.9	4.4	2.3	0.6	0.8

(a) view 1 *(b) view 2*

(c) view 3 *(d) view 4*

Figure 4.2: *Four color images captured in sequence by a virtual camera. They can share same objects since the camera rotates with a small number of degrees. Using similar images from a particular scene for training and validation can cause overfitting problem for the model.*

4.1.3 Test set

In order to decrease computational complexity during testing, the reduced test set provided by semantic3D is used. This test set includes four point clouds which are uniformly downsampled with a resolution of 0.01m and does not provide ground truth since semantic3D is an online benchmark dataset. Therefore, the experiments conducted in this thesis are not able to obtain results on projected 2D test images. Instead, the generated score maps of test images are used to annotate the test point clouds by simply summing up

all corresponding projections of each cloud point which in turn is classified according to the largest sum of scores [23]. The 3D test predictions are managed by an automated online submission system on semantic3D. In contrast to the validation set, the test set is used only once, aiming to evaluate the best model configuration which is finetuned by the validation set. The results of 3D test point clouds confirm the actual predictive power of the trained models and enable effective comparison between them [22]. In this thesis, we use the test results of point clouds to compare all the investigated networks, except the ablation study which is based on the validation results.

4.1.4 Data issues

The target dataset used in this thesis in general has three intrinsic properties which might cause problems for our models to learn efficiently. The first one is the ambiguousness in few classes. For example, the high vegetations vs the low vegetations, and the man-made terrain vs natural terrain can share highly similar visual appearances, some examples can be hard for humans to discriminate. Figure 4.3 illustrates an annotated color image of a high vegetation overlapped with a low vegetataion.

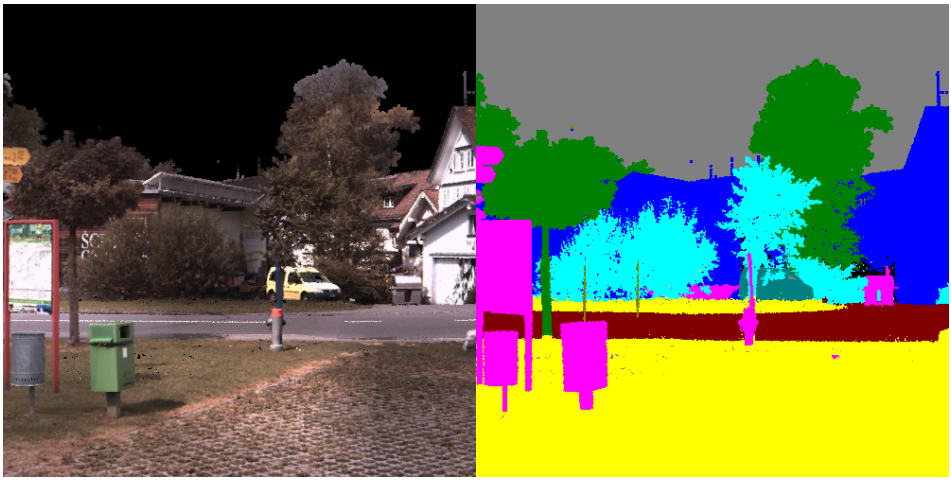


Figure 4.3: *Example of overlapping between high vegetation and low vegetation. They are hard to discriminate even with human eyes.*

The second one refers to the low-quality of the some image samples. Since our synthetic images are projected from semantic3D point clouds, the substantial attributes of point clouds are inherited. For example, the labels of the 3D clouds points that are far away from the camera can be missing sometimes, and are replaced with undefined class instead, this in turn results in unlabelled objects on the projected 2D images, an example is shown in figure 4.4. In addition, some classes are represented with highly irregular shapes or appearances, such as scanning artefacts shown in figure 4.5.

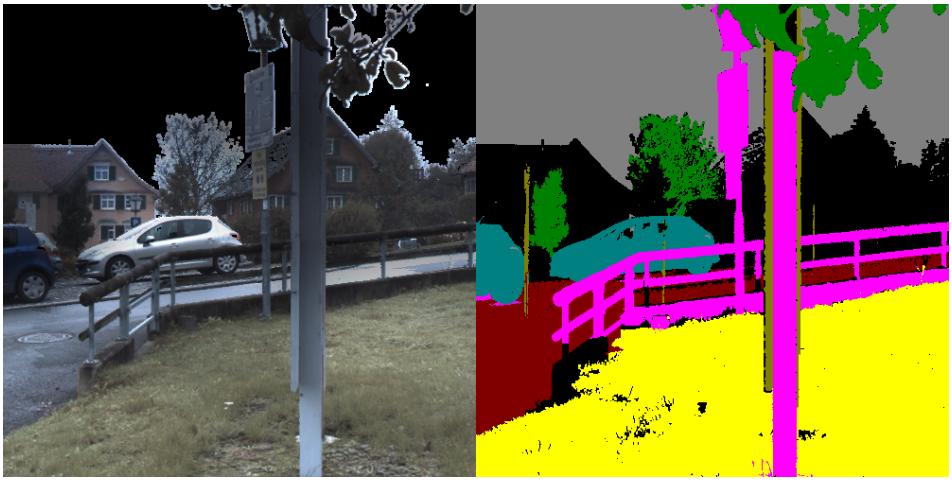


Figure 4.4: *Example image with unlabeled objects. In this case, the buildings that are far away from the camera are marked with undefined class-*

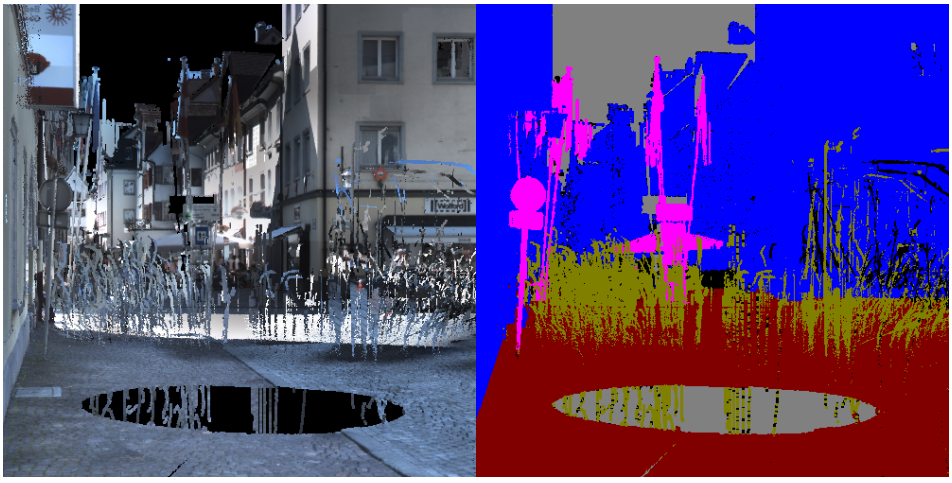
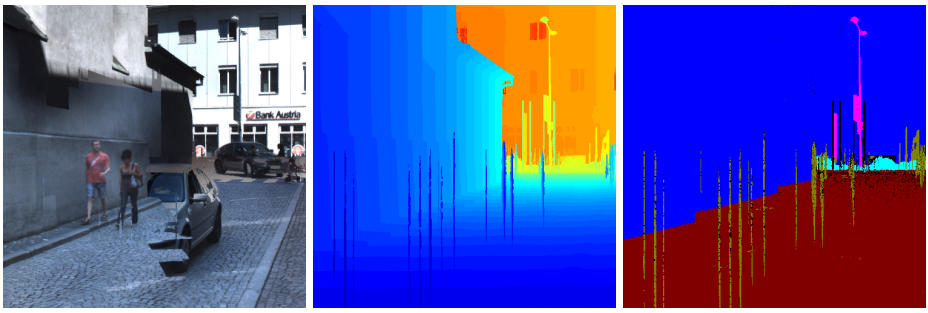


Figure 4.5: *An image with artefacts. These noisy objects can be difficult to learn since they are often represented with irregular shapes.*

The third issue is the inconsistency between the color images and the depth images. For example, pedestrians can appear on color images but not on the corresponding depth images, as shown in figure 4.6. Images of surface normals modality are extracted from depth images so that they are not consistent with color images either. This is because the color and depth information are captured at two separate time points. As a result, the above-mentioned detrimental properties might impose additional challenges on the learning process of our investigated networks along with the strong class imbalance(see table 4.2), but it would be interesting to see how the these issues can be alleviated by using

multimodal information integration.



(a) Color input *(b) Depth input* *(c) Ground truth*

Figure 4.6: *Same image of color and depth modalities can illustrate inconsistent content because the color and depth information are captured at two different time points. In this case, we can see the cars from the color input modality whereas the corresponding image of depth modality does not capture them. This issue only occurs on few samples though.*

4.2 Network configurations

All the experiments in this thesis are carried out using *Pytorch* [33], which is a Python-based scientific computing package provided with GPU support and optimized deep learning algorithms. All training sessions are run on single NVIDIA Tesla K40C GPU of 12GB memory. The input images are preprocessed with normalization which first divides each pixel value of each channel by 255 in order to rescale the values into the range of $[0, 1]$, then subtracts the mean of the corresponding channel of all images from ImageNet on which our singslestreams are pretrained on, followed by the division by the corresponding standard deviation, as a result, the inputs are zero-centered and normalized in the same way as the pretrained models. Since all applied models were pretrained on ImageNet and use a fully connected layer at the end of the network, we need to replace them with a 1×1 Conv layer to enable dense pixel predictions. Note that the singlestreams representing depth and surface normals are initialized with weights pretrained on ImageNet of colored images, this is inspired by [12] which has shown an improved network performance by applying pretrained weights of ImageNet on *depth* singlestream. In this thesis, all non-pretrained Conv layers, such as the modified first layer of early fusion framework and the late part of PSPNet as descibed in Chapter 3, use *He* initialization [17] method.

Inspired by [7] [57], we adopt the so-called "poly" learning rate schedule for all the investigated experiments. This learning rate schedule makes the learning rate decline smoothly over time based on a fixed base learning rate throughout whole training process, the formula can be seen as following:

$$current\_lr = base\_lr \times (1 - \frac{current\_iter}{max\_iter})^{power} \tag{4.1}$$

where *base_lr* varies from different network architectures, *max_iter* depends on the predefined batch size and number of epochs, and $power = 0.9$.

The standard optimization function *Stochastic gradient descent* is applied to all networks to minimize the Cross-Entropy loss function, and the hyperparameters momentum and decay are set to 0.9 and 0.0001 respectively, followed along with [57]. The number of epochs is set to 50 for all the experiments except for the late convolution fusion which has much less learnable parameters. We select the model with the best performance evaluated against the validation set within the 50 epochs to be further evaluated on the point cloud test set to obtain its truly unbiased performance rates. Table 4.3-4.7 specifies all network configurations.

4.2.1 Baseline models

Our baseline singlestream network only uses the color modality and the baseline multistream framework only applies late sum fusion(LSF). VGG16-based FCN8S is used as the base architecture and are pretrained on ImageNet. These selections of singlestream and fusion mthod are derived from the base work [23] which lays the foundation for this thesis. The applied batch size for baseline singlestream is maximized meaning that it is as large as possible without exceeding the limit of GPU memory. The baseline fusion method does not require further training, thus no learning rate or batch size need to be set.

Table 4.3: *Baseline configurations.*

Baseline models				
Model name	Base arc.	Base lr	Modalities	Batch size
Baseline_single	VGG16(FCN8S)	0.0001	color	5
Baseline_fusion	VGG16(FCN8S)	-	color, depth, normals	-

4.2.2 Singlestream networks

The investigated singlestream networks are dilated ResNet50-based FCN and PSPNet. The total number of singlestream experiments is 6 equal to the number of combinations of 2 networks and 3 modalities. We use the pretrained weights of the original ResNet50 to initialize our models. Due to the lack of pretrained weights on the PPM layer and its subsequent Conv layer in PSPNet, we multiply their 'poly' learning rates by 10 during training in order to force them to learn quicker than the pretrained backbone, inspired by the implementation[1] of [56], and we use the base learning rate of 0.001 for all the singlestreams. To balance the auxiliary loss of PSPNet during training, we multiply it by 0.4(recommended by [57]) before it is added to the mainstream loss. Both dilated ResNet50 and PSPNet use the same batch size of 4 in order to explore the unbiased impact of the PPM layer for our investigated fusion methods. For simplicity purposes, dilated ResNet50 is denoted as DR50, and PSPNet as PSP in the subsequent sections. Table 4.4 provides the information of all the investigated singlestream networks.

[1] https://github.com/zhanghang1989/PyTorch-Encoding

Table 4.4: *Singlestream network configurations.*

Investigated singlestreams		
Experiment name	Base arc.	Modality
DR50_color	Dilated ResNet50	color
DR50_depth	Dilated ResNet50	depth
DR50_normals	Dilated ResNet50	surface normals
PSP_color	PSPNet	color
PSP_depth	PSPNet	depth
PSP_normals	PSPNet	surface normals

4.2.3 Multistream networks

In this thesis, we investigate in total 10 experiments on multistream fusion frameworks with either dilated ResNet50 or PSPNet as the base singlestream architecture, see table 4.6. For the training of fusion layers within the *Late convolution fusion*(LCF) framework and the proposed fusion architectures, we apply the batch size and learning rate as same as the ones used in their singlestream networks, i.e. 4 and 0.001 respectively, whereas *Late sum fusion*(LSF) and *Late max fusion*(LMF) frameworks are directly applied on the test set. Since LCF only needs to learn a single Conv layer, we use fewer epochs to train it.

As mentioned in section 3.3.3, it is necessary to carry out an ablation study on the number-of-output-channels parameter which needs to be defined in all the Conv layers within SN. Thus, we choose the dilated ResNet50 as the base network for this study and evaluate the proposed fusion method on our validation set. The one that gives the best performance would be used for all the other proposed fusion variants due to the similar number of learnable parameters. Initially, we set number-of-output-channels to 512 which is recommended by the DFN paper [53]. Since our dataset is much smaller, it is reasonable to reduce it along the study. The investigated values of number-of-output-channels are shown in table 5.4.

Table 4.5: *Ablation study on number-of-output-channels hyperparameter for our proposed fusion method.*

Ablation study	
number-of-output-channels	Base arc.
512	Dilated ResNet50
256	Dilated ResNet50
128	Dilated ResNet50

Table 4.6: *Multistream frameworks configurations.*

Multistream fusion		
Model name	Base arc.	epochs
DR50_LSF	Dilated ResNet50	-
DR50_LMF	Dilated ResNet50	-
DR50_LCF	Dilated ResNet50	20
DR50_proposed	Dilated ResNet50	50
PSP_LSF	PSPNet	-
PSP_LMF	PSPNet	-
PSP_LCF	PSPNet	20
PSP_proposed_v1	PSPNet	50
PSP_proposed_v2	PSPNet	50
PSP_proposed_v3	PSPNet	50

4.2.4 Early fusion

The early fusion further investigates the effectiveness of the smooth network(SN). Similar to the multistream fusion experiments, we apply SN on the early fused dilated ResNet50 and PSPNet. The overall network configurations are consistent with the ones used for the multistream fusion experiments. Inspired by the FCN paper [39] which tried early fusion on NYUDv2 [40] RGB-D dataset, by first finetuning a pretrained(ImageNet) FCN on the color modality and then training the same model once again on the early fused representations of color and depth modalities, we decide to conduct our experiments in the same way but with early fused representations of color, depth and surface normals modalities. The early fusion-based experiments use the same configurations as the corresponding singlestream networks(see section 4.4).

Table 4.7: *Network configurations with early fusion.*

Investigated early fusion networks.	
Experiment name	Base arc.
DR50_early	Dilated ResNet50
DR50_early_SN	Dilated ResNet50
PSP_early	PSPNet
PSP_early_SN	PSPNet

4.3 Evaluation metrics

All the investigated configurations are evaluated on the test set(see section 4.1.3) in order to build the unbiased basis for comparison purposes. The evaluation metrics used in this thesis are the *overall accuracy*(OA) and the *mean intersection over union*(mIoU). OA gives the mean accuracy of each pixel and mIoU is the average of *intersection over union*

of each class. These metrics are favoured by semantic3D benchmark[2] and used by Long et al. in [39].

OA and mIoU have different conceptions of how the pixel predictions of our models and the corresponding ground truth are compared. In mathematical terms, they both use the confusion matrix N to keep track of the classifications of each class. Let N_{ij} denote the number of pixels that belong to class i but classified as class j, t_i denote the total number of pixels of class i, and let n_{cl} be the total number of classes. OA can then be expressed as:

$$OA = \frac{1}{n_{cl}} \cdot \frac{\sum_i N_{ii}}{t_i} \tag{4.2}$$

and mIoU is given by:

$$mIoU = \frac{1}{n_{cl}} \cdot \frac{\sum_i N_{ii}}{(t_i + \sum_j N_{ji} - N_{ii})} \tag{4.3}$$

mIoU is chosen to be the most important metric to evaluate our models in this thesis while OA is used to provide a general view of model performance, because OA suffers strongly by the issue of class imbalance and can always obtain high accuracy by guessing on the majority class. However, the mIoU can alleviate the problem by penalizing the model performance rates more on the misclassification of minority class. In addition, we show the IoU of each class to carry out an in-depth analysis of the network configurations.

[2] http://www.semantic3d.net/

5
Results

This chapter presents the results of the investigated experiments. Firstly, the quantitative results are shown in table form from which the comparisons of different networks can be established, based on the numerical performance evaluations of both 2D validation images and 3D test point clouds. Secondly, the qualitative results providing the visual comparisons of the networks on manually selected samples are illustrated. Lastly, a general discussion of the results is introduced.

5.1 Quanlitative results

As mentioned in section 4.1, all the networks are evaluated on the validation set of 2D images during training in order to select the best model configurations, which in turn would be used for 3D point clouds reprojection in order to obtain the test results of mIoU and OA from semantic3D. Since this thesis mainly focuses on the task of 2D semantic segmentation, it is important to comprehend how the investigated network configurations behave on 2D data , therefore we present not only the quantitative results of the point cloud test set, but also of the 2D validation set in this section. Further, the effectiveness of the manually selected validation data can be explored by sensing the correlations between the validation and test results. In consideration of the fact that the results are somewhat overfitted towards the validation set, we rank the network performance on the point cloud test data which has not been seen, even though the results involve reprojection algorithm from the base work [23]. For the purpose of simplicity, the validation results are only expressed in mIoU, while the test results can specify addtional IoU of each class and OA. The highest values are marked with bold font in all the tables.

An overview of the results of all the experiments is listed in table 5.1. In general, the multistream frameworks perform better than singlestream networks in quantitative terms of mIoU on both validation and test data. Surprisingly, the third version of the proposed fusion framework(see figure 3.5) based on PSPNet achieved the best performance

rates. Note that the performance measures of multistream frameworks do not always necessarily exceed all of its singlestream networks, for example, the PSP_LSF(61.2%), PSP_LMF(61.6%) and PSP_LCF(60.5%) frameworks have lower mIoU on test set than the singlestream PSP_normals(62.55%).

Table 5.1: *This table presents the results of all the investigated experiments. The performance evaluations of both the validation set of 2D projected images and the test set of 3D point clouds are computed in mIoU(%). In addition, the base architecture, fusion type and modality used in each experiment are listed. For singlestream networks, the fusion type of '-' indicates no fusion method applied, and for multistream frameworks, the modality of 'ALL' means that all the singlestreams representing the color, depth and surface normals respectively, are integrated.*

All experiments					
Model name	mIoU(%)		Base architecture	Fusion type	Modality
	2D val	3D test			
Baseline_color	58.69	53.34	VGG16(FCN8S)	-	color
Baseline_depth	51.27	53.86	VGG16(FCN8S)	-	depth
Baseline_normals	51.94	53.75	VGG16(FCN8S)	-	surface
DR50_color	61.21	53.84	Dilated ResNet50	-	color
DR50_depth	55.41	54.81	Dilated ResNet50	-	depth
DR50_normals	59.58	59.36	Dilated ResNet50	-	surface normals
PSP_color	62.92	56.31	PSPNet	-	color
PSP_depth	55.76	53.55	PSPNet	-	depth
PSP_normals	60.22	62.55	PSPNet	-	surface normals
Baseline_fusion	58.71	57.36	VGG16(FCN8S)	Late fusion	ALL
DR50_LSF	64.30	59.46	Dilated ResNet50	Late fusion	ALL
DR50_LMF	61.14	58.63	Dilated ResNet50	Late fusion	ALL
DR50_LCF	64.59	59.60	Dilated ResNet50	Late fusion	ALL
DR50_proposed	68.00	62.00	Dilated ResNet50	Late fusion	ALL
PSP_LSF	65.08	61.20	PSPNet	Late fusion	ALL
PSP_LMF	62.48	61.16	PSPNet	Late fusion	ALL
PSP_LCF	64.53	60.50	PSPNet	Late fusion	ALL
PSP_proposed_v1	68.39	62.03	PSPNet	Late fusion	ALL
PSP_proposed_v2	68.50	62.45	PSPNet	Late fusion	ALL
PSP_proposed_v3	**68.63**	**63.13**	PSPNet	Late fusion	ALL
DR50_early	65.74	58.04	Dilated ResNet50	Early fusion	ALL
DR50_early_SN	66.76	58.88	Dilated ResNet50	Early fusion	ALL
PSP_early	64.08	61.11	PSPNet	Early fusion	ALL
PSP_early_SN	66.15	61.86	PSPNet	Early fusion	ALL

Singlestream networks

This thesis aims to explore more options of singlestream networks and multistream fusion methods in order to improve the overall performance of the multistream framework investigated by the base work [23]. Table 5.3 shows that both investigated singlestream architectures are generally better than the baseline singlestream (VGG16-based FCN8S) in terms of mIoU. These results are expectable since dilated ResNet50 and PSPNet use deeper architectures and residual learning to be able to learn more high-level feature abstractions of input images and prevent degradation problem. Note that the mIoU is the mainly considered performance measure which pays more attention to minority classes than OA.

Comparing how the networks behave against each individual modality, it is obvious that dilated ResNet50 and PSPNet have the biggest advantage over the baseline singlestream within the surface normals modality, which indicates that the images of surface normals provide more effective and useful information than of color and depth domains. Further, the baseline singlestream performs best at the predictions of man-made and natural terrain in both color and depth but has the lowest performance measures for hardscape and scanning artefacts in all the three modalities. This implies that the FCN8S is good at classifying objects which are frequently appeared in the training data but performs poorly for the classifications of objects that have small-scaled and irregular shapes. In other words, ResNet50 and PSPNet sacrifice their recognition ability of simple objects for a more robust classification of complex objects. In general, PSPNet is the most robust classifier but still can not produce satisfactory results in classifications of classes such as hardscape and scanning artefacts due to the inherent poor data distributions(see table 4.2). Table 5.2 shows the abbreviations for all the classes.

Table 5.2: *Class abbreviations used for all tables in this section.*

Class abbreviations	
Man-made terrain	MT
Natural terrain	NT
High vegetation	HV
Low vegetation	LV
Buildings	BU
Hard scape	HS
Scanning artefacts	SA
Cars	CA

Table 5.3: *Results for all singlestream networks grouped by modality.*

Singlestream networks											
Model name	mIoU(%)		OA	IoU(%) of 3D test							
	2D val	3D test	(%)	MT	NT	HV	LV	BU	HS	SA	CA
Baseline_color	58.69	53.34	88.1	**85.1**	**80.2**	66.0	30.7	90.2	20.5	9.2	44.8
DR50_color	61.21	53.84	87.7	78.7	72.5	72.0	31.1	**91.2**	**26.1**	10.5	**48.6**
PSP_color	**62.92**	**56.31**	**88.3**	79.5	74.3	**77.7**	**42.2**	91.1	21.2	**13.2**	51.3
Baseline_depth	51.27	53.86	**87.5**	**84.4**	**79.9**	69.0	**25.4**	**88.8**	11.7	18.1	53.6
DR50_depth	55.41	**54.81**	85.6	77.8	67.9	**69.5**	23.3	88.0	**19.7**	**31.7**	60.6
PSP_depth	**55.76**	53.55	84.1	81.1	56.5	67.9	24.3	88.3	16.6	31.3	**62.4**
Baseline_normals	51.94	53.75	85.5	81.4	67.5	74.3	21.0	88.6	20.3	23.0	53.9
DR50_normals	59.58	59.36	88.8	83.1	81.0	**77.3**	28.2	89.5	22.1	26.1	67.6
PSP_normals	**60.22**	**62.55**	**90.5**	**87.2**	**84.9**	76.3	**30.2**	**91.6**	**31.0**	**30.1**	**69.1**

Multistream frameworks

Prior to the inspection of how different fusion methods perform, it is necessary to investigate the number of output channels used in all Conv layers within SN, which has been described in section 4.2.3. An ablation study is carried out based on the multistream framework consisted of dilated ResNet50 singlestreams, the results are shown in table 5.4 where the value of 256 gives the highest mIoU on validation set, thus it is used for all the subsequent experiments of the proposed fusion method.

Table 5.4: *Ablation study on our proposed fusion architecture applied on multistream framework of dilated ResNet50.*

Ablation study	
number of channels	mIoU(%) of 2D val
128	67.1
256	**68.0**
512	67.7

For the multistream frameworks, there is no significant difference between them. Table 5.5 exhibits the results of each multistream framework in detail. All the frameworks with the proposed fusion approach have higher performance rates than their counterparts with traditional fusion methods. Further, the baseline fusion framework perform worst among all the investigated fusion architectures. The results(mIoU) between the validation set of 2D images and the test set of 3D point clouds illustrate high correlations, and the validation mIoU is always higher than test mIoU which is reasonable since the test set is totally unseen.

To answer the first research question(How much better does the proposed fusion architecture perform compared to other traditional late fusion methods in terms of mIoU, under the condition that all singlestream networks are the same?) in chapter 1, we rearrange table 5.5 so that only the necessary data is presented. The new table 5.6 is grouped by multistream frameworks with the same base architecture. Note that only the

Table 5.5: *Results of all multistream experiments.*

Multistream frameworks											
Model name	mIoU(%)		OA	IoU(%) of 3D test							
	2D val	3D test	(%)	MT	NT	HV	LV	BU	HS	SA	CA
Baseline_fusion(LSF)	58.71	57.36	89.5	86.5	81.0	76.0	29.4	91.1	15.9	19.3	59.7
DR50_LSF	64.30	59.46	89.9	85.9	81.4	77.2	29.5	91.4	22.5	24.3	63.5
DR50_LMF	61.14	58.63	90.0	86.5	83.1	76.4	30.7	91.3	20.0	18.4	62.6
DR50_LCF	64.59	59.60	89.9	85.9	81.1	77.1	27.2	91.7	27.8	22.3	63.7
DR50_proposed	68.00	62.00	89.2	82.3	72.9	77.3	33.1	**92.9**	**34.4**	37.1	66.0
PSP_LSF	65.08	61.20	90.3	86.8	83.9	76.3	37.7	91.4	22.0	27.0	64.5
PSP_LMF	62.48	61.16	**90.5**	**86.9**	**84.0**	77.8	40.0	91.5	21.3	23.9	63.9
PSP_LCF	64.53	60.50	90.0	86.0	82.5	76.9	35.9	91.3	21.0	26.5	63.9
PSP_proposed_v1	68.39	62.03	88.9	81.9	72.1	76.0	39.0	92.7	32.2	**37.6**	64.7
PSP_proposed_v2	68.50	62.45	89.7	83.2	76.1	78.6	39.4	92.7	31.7	32.4	65.5
PSP_proposed_v3	**68.63**	**63.13**	90.0	83.8	78.0	**78.9**	**43.7**	92.5	25.4	35.0	**67.7**

PSP_proposed_v3 is shown here since it has achieved the highest mIoU on test set compared to other versions. The results show that our proposed fusion method are distinctly better than the other traditional counterparts in the sense that both DR50_proposed and PSP_proposed_v3 dominate the test IoU for 6 of 8 classes. These overall improvements highlight the usefulness of employing RRB and CAB units which integrate information from all the encoder layers for the decoding purposes.

Interestingly, the LMF fusion method performs best on man-made terrain and natural terrain in both groups, which may depend on that these classes are too easy to be recognized of the singlestreams which are tempted to generate very high prediction scores for these classes.

Table 5.6: *Results of all multistream experiments grouped by the same type of base architecture.*

Multistream frameworks											
Model name	mIoU(%)		OA	IoU(%) of 3D test							
	2D val	3D test	(%)	MT	NT	HV	LV	BU	HS	SA	CA
DR50_LSF	64.30	59.46	89.9	85.9	81.4	77.2	29.5	91.4	22.5	24.3	63.5
DR50_LMF	61.14	58.63	**90.0**	**86.5**	**83.1**	76.4	30.7	91.3	20.0	18.4	62.6
DR50_LCF	64.59	59.60	89.9	85.9	81.1	77.1	27.2	91.7	27.8	22.3	63.7
DR50_proposed	**68.00**	**62.00**	89.2	82.3	72.9	**77.3**	**33.1**	**92.9**	**34.4**	**37.1**	**66.0**
PSP_LSF	65.08	61.20	90.3	86.8	83.9	76.3	37.7	91.4	22.0	27.0	64.5
PSP_LMF	62.48	61.16	**90.5**	**86.9**	**84.0**	77.8	40.0	91.5	21.3	23.9	63.9
PSP_LCF	64.53	60.50	90.0	86.0	82.5	76.9	35.9	91.3	21.0	26.5	63.9
PSP_proposed_v3	**68.63**	**63.13**	90.0	83.8	78.0	**78.9**	**43.7**	**92.5**	**25.4**	**35.0**	**67.7**

To answer the second research question(How sensitive is the proposed fusion architecture in terms of mIoU, when its all singlestream models are replaced with a more

powerful one?), another rearranged table 5.7 demonstrates that all the PSPNet-based proposed fusion approaches outperformed the ones based on dilated ResNet50 with slight margin. A thought-provoking finding from this table concerns the results of hardscape predictions of all the frameworks, among which the DR50_proposed achieved the highest IoU(34.4%), a relative increase of 9% compared to the PSP_proposed_v3(25.4%) which obtained the highest mean IoU. The underlying reasons might relate to how the PPM layer of PSPNet is used in our proposed fusion architecture or how the investigated singlestream networks treat the hardscape objects. However, according to the result table 5.3 of singlestreams, it is hard to tell that the dilated ResNet50 have absolute advantages over PSPNet on the classification of hardscapes. On the other hand, the PSP_proposed_v3 fusion framework(see figure 3.5) exploits the stacked PPM layers to start the decoding process whereas the DR50_proposed fusion framework(see figure 3.3) uses the stacked Res-4 layers with global average pooling, and it is known that the PPM layer is able to extract multi-scaled feature information, as described in 2.6. In addition, the hardscape objects usually have irregular shapes and appear in inconsistent environment, as seen from our target dataset. Thus, these facts lead us to conclude that it is not always beneficial to use information-rich feature maps to guide our proposed fusion architecture, since it can be harmful for the framework to recognize class instances which have irregular shapes and are not particularly bound to certain environments. However, the overall framework performance can be increased by integrating more contextual information for decoding purposes, as shown by the results of PSP_proposed_v3.

Table 5.7: *Results of all multistream experiments combined with the proposed fusion architecture.*

Multistream frameworks											
Model name	mIoU(%)		OA	IoU(%) of 3D test							
	2D val	3D test	(%)	MT	NT	HV	LV	BU	HS	SA	CA
DR50_proposed	68.00	62.00	89.2	82.3	72.9	77.3	33.1	**92.9**	**34.4**	37.1	66.0
PSP_proposed_v1	68.39	62.03	88.9	81.9	72.1	76.0	39.0	92.7	32.2	**37.6**	64.7
PSP_proposed_v2	68.50	62.45	89.7	83.2	76.1	78.6	39.4	92.7	31.7	32.4	65.5
PSP_proposed_v3	**68.63**	**63.13**	**90.0**	**83.8**	**78.0**	**78.9**	**43.7**	92.5	25.4	35.0	**67.7**

Early fusion experiments

For the early fusion experiments, the frameworks which have applied the proposed architecture outperformed their traditional counterparts, insignificantly though. Table 5.8 shows the quantitative results to answer the last research question(Is it possible to improve the performance of traditional early fusion in terms of mIoU by using smooth network(SN) on top of it?). The early fusion frameworks generally perform worse than the multistream frameworks, which might indicate that the early fused feature representations involve high extent of noises since all the three input modalities are combined, resulting in that the networks have difficulties to extract useful features. On the other hand, the multistream frameworks separately sort out the noise of each modality first, then combine the respective most effective information to complement each other for the final prediction, but requires higher computational and memory complexities.

Table 5.8: *Results of all experiments based on early fusion.*

Early fusion experiments											
Model name	mIoU(%)		OA	IoU(%) of 3D test							
	2D val	3D test	(%)	MT	NT	HV	LV	BU	HS	SA	CA
DR50_simple_early	65.74	58.04	88.7	**84.9**	**78.5**	**73.8**	28.8	90.6	20.7	26.2	60.8
DR50_proposed_early	**66.76**	**58.88**	88.7	84.7	77.3	73.1	**34.3**	90.6	**21.0**	**26.8**	**63.2**
PSP_simple_early	64.08	61.11	90.9	88.3	85.7	**82.2**	**41.2**	91.4	**17.9**	26.8	55.4
PSP_proposed_early	**66.15**	**61.86**	**91.1**	**89.0**	**86.0**	80.7	40.8	**91.8**	17.0	**29.4**	**60.2**

Experiments with improved training techniques

As mentioned in section 3.5, additional techniques(more effective pretrained weights, median frequecy balancing and data augmentation) would be applied to the best one of the investigated fusion frameworks, in this case the PSP_proposed_v3, aiming to further improve the mIoU/OA on the semantic3D benchmark. Table 5.9 shows the detailed comparisons between networks with common configurations and the improved ones. Note that the model names ended by '*imp*' refer to the networks with improved configurations. In general, all the improved networks have obtained better mIoU on test set except the PSP_normal_imp. Since the additional techniques are jointly applied, the reason why PSP_normal_imp did not achieve improved performance rates is unknown. However, the PSP_proposed_v3_imp outperformed the model named *DeePr3SS*(product of the base work [23]) by almost 7.3% test mIoU, raised our ranking from 12th to 5th place on the semantic3D benchmark leaderboard, shown in figure 5.1.

Table 5.9: *Results of experiments using weighted class, more effective pretrained weights and data augmentation, investigated for benchmark purposes.*

Improved training										
Model name	mIoU(%)	OA	IoU(%) of 3D test							
	3D test	(%)	MT	NT	HV	LV	BU	HS	SA	CA
PSP_color	56.31	**88.3**	79.5	74.3	77.7	42.2	91.1	21.2	**13.2**	51.3
PSP_color_imp	**57.48**	88.2	**81.1**	**80.9**	**78.8**	**44.1**	91.1	**23.3**	8.0	**52.5**
PSP_depth	53.55	84.1	**81.1**	56.5	67.9	24.3	88.3	16.6	**31.3**	62.4
PSP_depth_imp	**57.50**	**86.6**	80.3	**68.2**	**76.2**	**29.2**	**90.9**	**24.8**	21.3	**69.1**
PSP_normals	**62.55**	**90.5**	**87.2**	**84.9**	76.3	30.2	91.6	31.0	**30.1**	**69.1**
PSP_normals_imp	60.75	89.3	85.6	80.9	**78.0**	**32.6**	**92.4**	**35.9**	13.5	67.1
PSP_proposed_v3	63.13	90.0	83.8	78.0	78.9	43.7	92.5	25.4	**35.0**	67.7
PSP_proposed_v3_imp	**65.76**	**91.2**	**86.4**	**82.6**	**82.6**	**48.6**	**93.2**	**33.1**	30.0	**69.6**

Large-Scale Point Cloud Classification Benchmark

home data results submit FAQ people login sign up

reduced-8 results

We use Intersection over Union (IoU) and Overall Accuracy (OA) as metrics. For more details hover the curser over the symbols or click on a classifier. In order to sort the results differently click on a symbol.

	Name	A_IoU	OA	[s]	IoU 1	IoU 2	IoU 3	IoU 4	IoU 5	IoU 6	IoU 7	IoU 8
1	SPGraph	0.732	0.940	3000.00	0.974	0.926	0.879	0.440	0.932	0.310	0.635	0.762
							Large-scale Point cloud segmentation with superpoint graphs, Loic Landrieu and Martin Simonovsky, CVPR2018					
2	KP-FCNN	0.715	0.920	600.00	0.929	0.778	0.899	0.430	0.939	0.273	0.647	0.823
											Anonymous submission	
3	PCSNet	0.712	0.943	1500.00	0.971	0.950	0.879	0.525	0.941	0.388	0.355	0.687
4	GACNet	0.708	0.919	1380.00	0.864	0.777	0.885	0.606	0.942	0.373	0.435	0.778
	Ours	0.658										
											Anonymous submission	
5	MSDeepVoxNet	0.653	0.884	115000.00	0.830	0.672	0.838	0.367	0.924	0.313	0.500	0.782
									Classification of Point Cloud Scenes with Multiscale Voxel Deep Network, Xavier Roynard, Jean-Emmanuel Deschaud and François Goulette			
6	OctFCNNet	0.648	0.894	1200.00	0.943	0.756	0.786	0.342	0.904	0.257	0.478	0.721
											Anonymous submission	
7	RSSP	0.647	0.920	1.00	0.916	0.870	0.870	0.525	0.930	0.158	0.320	0.589
											Anonymous submission	
8	RF_MSSF	0.627	0.903	1643.75	0.876	0.803	0.818	0.364	0.922	0.241	0.426	0.566
							H. Thomas, J. Deschaud, B. Marcotegui, F. Goulette, Y. Le Gall. Semantic Classification of 3D Point Clouds with Multiscale Spherical Neighborhoods. In 3D Vision (3DV), 2018 International Conference on, 2018.					
9	SEGCloud	0.613	0.881	1881.00	0.839	0.660	0.860	0.405	0.911	0.309	0.275	0.643
								L. P. Tchapmi, C. B.Choy, I. Armeni, J. Gwak, S. Savarese, SEGCloud: Semantic Segmentation of 3D Point Clouds, International Conference on 3D Vision (3DV), 2017				
10	OctreeNet_CRF	0.591	0.899	184.84	0.907	0.820	0.824	0.393	0.900	0.109	0.312	0.460
							F. Wang, Y. Zhuang, H. Gu, and H. Hu, OctreeNet: A Novel Sparse 3D Convolutional Neural Network for Real-time 3D Outdoor Scene Analysis, submitted to IEEE Transactions on Automation Science and Engineering.					
11	SnapNet_	0.591	0.886	3600.00	0.820	0.773	0.797	0.229	0.911	0.184	0.373	0.644
								Unstructured point cloud semantic labeling using deep segmentation networks. A. Boulch, B. Le Saux and N. Audebert, Eurographics 3DOR 2017				
12	DeePr3SS	0.585	0.889	0.00	0.856	0.832	0.742	0.324	0.897	0.185	0.251	0.592

Figure 5.1: *semantic3D benchmark leaderboard for the reduced test set.*

5.2 Qualitative results

The qualitative results of different networks are illustrated in 2D images and analyzed in this section. These images represent the dense predictions for the validation data since the ground truth of test data is not available. As analyzed in the quantitative results, the validation and test results are correlated to a great extent, thus the visual results of the validation data should provide justifiable explanations of how these networks perform in

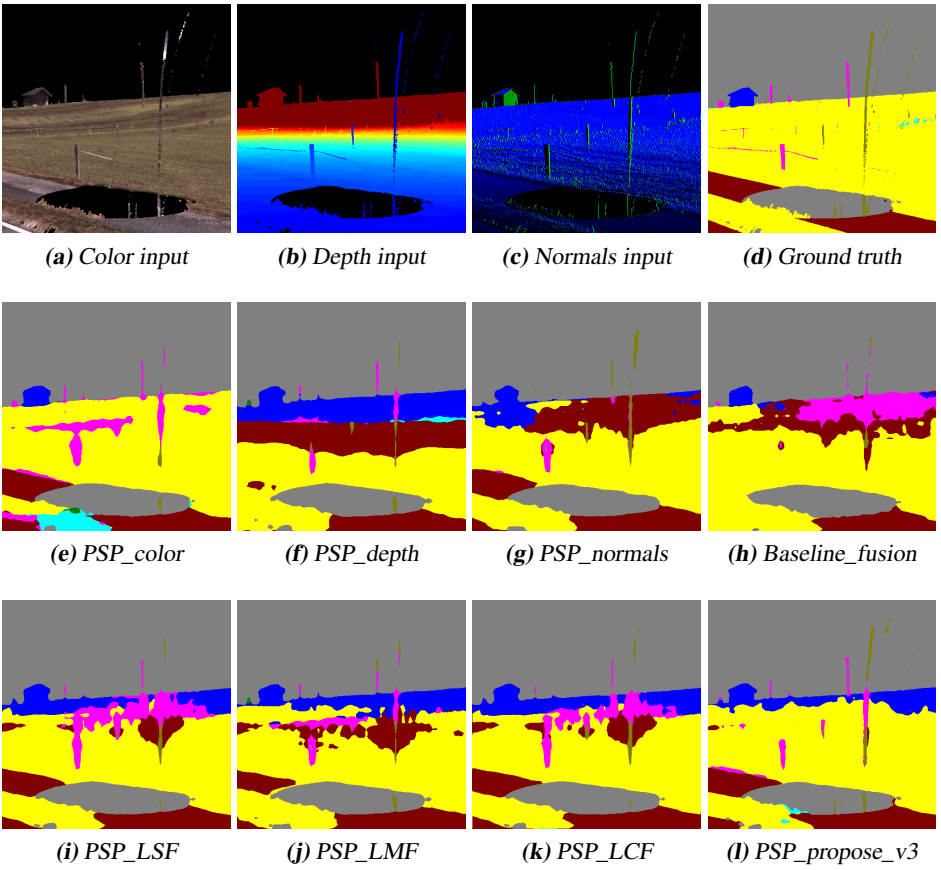
an unseen world. Since this thesis mainly focuses on multistream fusion, it is reasonable to conduct an extensive investigation on the results of the multistream fusion approaches rather than having a general view of all the experiments including singlestream or early fusion networks. Therefore, this section specifically presents the qualitative results of the most effective fusion method, i.e. the PSP_proposed_v3, and analyses them along with the baseline fusion and the other PSPNet-based traditional fusion methods. The qualitative results are manually selected in order to show the most conspicuous differences/similarities between the investigated models. Further, these results respond greatly to the three intrinsic data issues mentioned in section 4.1.4.

5.2.1 Validation set of 2D images

Four specific examples of validation images are listed below, each of them is described and analysed shortly with a title introducing the main aspect. Within the figure of each example, the inputs and the annotated ground truth are given in the first row, the results of the PSPNet-based singlestream and the base fusion networks are given in the second row, lastly, the results of the PSPNet-based multistream frameworks are given in the third row. Note that the baseline fusion method used VGG16-based singlestream networks and it is served as the standard for measuring the PSPNet-based fusion performances, qualitatively.

Ex 1: Diverse multimodal predictions

The first example shows how the different fusion methods manage the situation where all the singlestream networks produces completely different classifications on a certain region on their respective inputs, as seen in figure 5.2. The input images consist of simple objects and the dominant one is the natural terrain that visually extends a long way across the depth dimension. The end part of this object enables the prominent classication divergence between all the singlestreams. The PSP_color **(e)** obtained the most correct predictions on that part while PSP_depth **(f)** and PSP_normals **(g)** classified it as building and man-made terrain respectively. A feasible reason might be that the distant appearance of the terrain in Depth input **(b)** causes problem for the learning process of PSP_depth. In addition, the PSP_normals **(g)** is affected in the same way since Normals input **(c)** is derived from **(b)**. Visually, our PSP_proposed_v3 **(l)** can best distinguish the diverse multimodal predictions compared to the baseline fusion **(h)** and the others **(i) (j) (k)** which are highly influenced by the misclassifications of **(f)** and **(g)**. This example shows that the proposed fusion architecture can effectively select the most useful features from all the singlestreams and best filter out the misleading ones.



(a) Color input *(b) Depth input* *(c) Normals input* *(d) Ground truth*

(e) PSP_color *(f) PSP_depth* *(g) PSP_normals* *(h) Baseline_fusion*

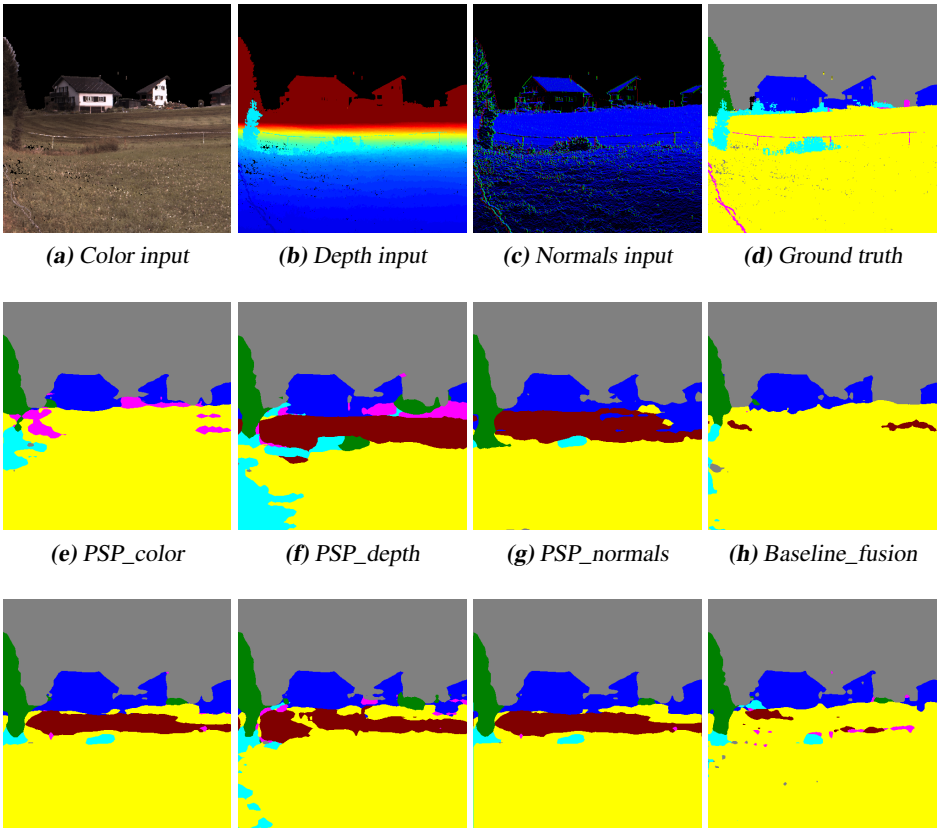
(i) PSP_LSF *(j) PSP_LMF* *(k) PSP_LCF* *(l) PSP_propose_v3*

Figure 5.2: *Example images show how different multistream frameworks manage the highly diverse predictions on a certain part of the natural terrain of the singlestream networks.*

Ex 2: Delusive baseline performance

The second example is similar to the previous one in the way that PSP_proposed_v3 generates the most effective predictions for the natural terrain that is classified inconsistently over separate singlestreams, shown in figure 5.3. Interestingly, the Baseline_fusion **(h)** visually gives the most precise predictions according to the Ground truth **(d)** among all the fusion network predictions. This may raise a question of whether the baseline fusion method is overall better than PSPNet-based multistream fusion frameworks **(i)**, **(j)**, **(k)** and **(l)**. By referencing to the table 5.3 of quantitative results for singlestream networks, both the VGG16-based baseline_color and baseline_depth outperformed the corresponding dilated ResNet50 and PSPNet singlestreams with large margins, in classifications of man-made and natural terrains. However, all the VGG16-based singlestreams have the lowest performance rates on classifying hardscape and scanning artefact objects. This phenomenon might indicate that powerful networks can sacrifice their focuses on the majority classes to improve the recognition abilities of minority classes. To prove that, the

details in **(i)**, **(j)**, **(k)**, and **(l)** showed that more pixels of low vegetation and hardscape were being recognized than **(h)**. In this example, the PSP_proposed_v3 **(l)** gives the best visual result compared the other traditional fusions, as the most misclassifications from **(f)** and **(g)** are removed, but do suffer from recognizing the overlapped objects(natural terrain and low vegetation). In summary, this example does not only demonstrate that one single output result can be delusive, but also the ambiguousness of objects is difficult to manage for all the investigated networks, as described in section 4.1.4.



(a) Color input *(b) Depth input* *(c) Normals input* *(d) Ground truth*

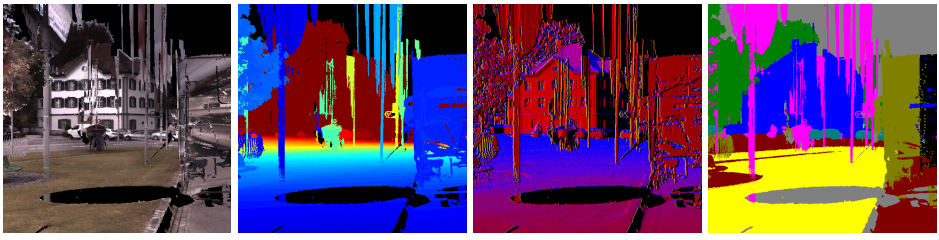
(e) PSP_color *(f) PSP_depth* *(g) PSP_normals* *(h) Baseline_fusion*

(i) PSP_LSF *(j) PSP_LMF* *(k) PSP_LCF* *(l) PSP_propose_v3*

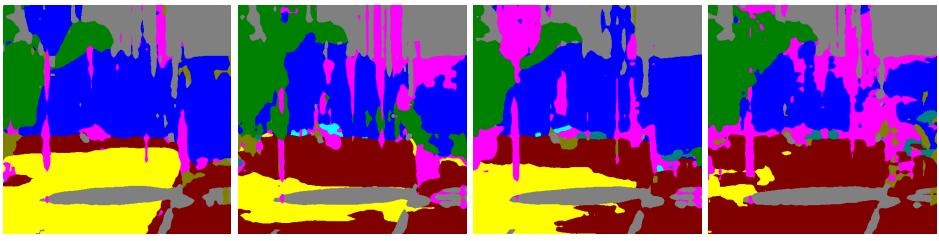
Figure 5.3: *Example images show different networks manage the small-scaled and overlapped objects which have similar appearance. Observe that the base fusion approach classifies most of the confusing parts as the majority class, i.e. natural terrain.*

Ex 3: Miscellaneous objects

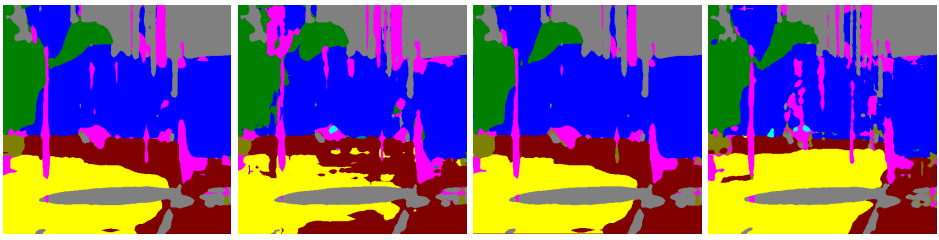
The third example shows a very challenging case for the investigated models to perform semantic segmentation. It involves large amount of objects of irregular shapes which mainly represent hardscapes, scanning artefacts, and even undefined class, as seen in the ground truth **(d)** in figure 5.4. In fact, the objects are perplexing even for human eyes to distinguish, not to mention artificial models. All the presented networks totally missed the scanning artefacts and only classified the hardscapes to a certain degree. Indeed, the results are reasonable considering the tiny amount of the hardscape and scanning artefacts pixels existed in the training data(see class distribution table 4.2). For the hardscape consisted of flag and mast on the left side of **(d)**, the PSP_normals **(g)** among all the singlestreams showed the best qualitative result whereas almost all the PSPNet-based fusion frameworks misclassified it as building, probably influenced by PSP_color **(e)** predicted. This might be due to the strong confidence level on classifying it as building produced by PSP_color and/or the low confidence level on classifying it as hardscape by PSP_normals, resulting in that the fusion frameworks would pay more attention to the one with very high prediction scores. Surprisingly, the baseline fusion **(h)** misclassified almost the entire natural terrain to man-made terrain, which should not be the case since VGG16-based singlestreams are superb at classifying natural and man-made terrain as mentioned previously. One possible explanation could be that the VGG16-based networks are not able to utilize the contextual information for classification, especially when two similar objects of different classes are close to each other, e.g. the natural and man-made terrain. However, the baseline fusion **(h)** is somewhat better at classifying the cars of small scales than the others. These distant cars in the middle of **(d)** are ignored by all the PSPNet-based fusion frameworks, possibly because of the complex surroundings which make the PPM layer of PSPNet provide misleading contextual information for classifying the cars, or because such distant and small-scaled objects are presented infrequently in the training data. Note that the proposed fusion method **(l)** proved again that the misclassifications from **(f)** and **(g)** can be efficiently removed.



(a) Color input *(b) Depth input* *(c) Normals input* *(d) Ground truth*



(e) PSP_color *(f) PSP_depth* *(g) PSP_normals* *(h) Baseline_fusion*



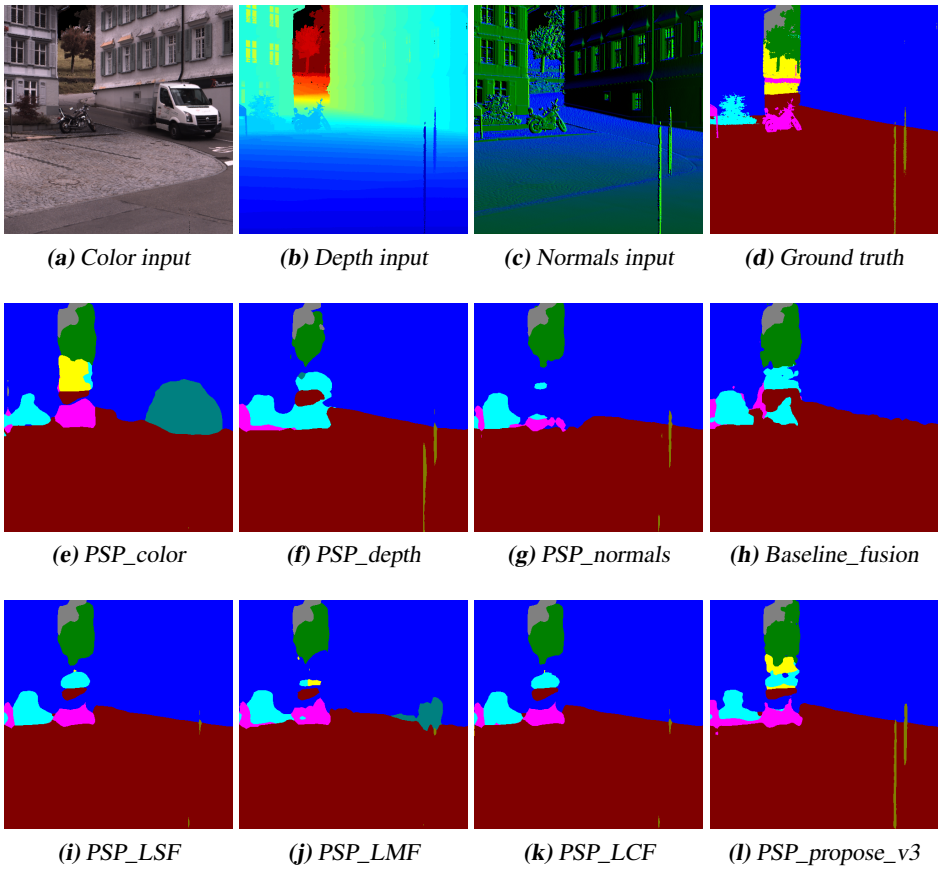
(i) PSP_LSF *(j) PSP_LMF* *(k) PSP_LCF* *(l) PSP_propose_v3*

Figure 5.4: *Example images show how different networks deal with various kinds of objects which can be presented in irregular forms. The scanning artefacts and hardscapes are difficult for all the networks to classify.*

Ex 4: Self-refinement of the proposed fusion architecture

The last example shows a relatively simple urban scene and aims to clarify how the different fusion frameworks manage the details of the distant objects illustrated in ground truth **(d)** in figure 5.5. First of all, the input images reflect the data issue of inconsistency in different input modalities, as mentioned in 4.1.4. Comparing the figures **(a)**, **(b)** and **(c)**, a car object is only visible in **(a)** because the color and depth information were captured at different time points. However, this problem does not cause confusions for the predictions of the fusion frameworks, except for the PSP_LMF **(j)** which is greatly affected by very high prediction scores produced by any of the singlestream networks **(e)**, **(f)**, and **(g)**. Again, all the fusion frameworks seem to suffer from the distant objects, i.e. the natural terrain and the hardscape as seen in the upper middle part of **(d)**. The PSP_color **(e)** is able to correctly classify most of the natural terrain compared to **(f)** and **(d)**, and the PSP_proposed_v3 **(l)** performs better at exploiting what PSP_color has learned than other traditional fusion methods **(i)**, **(j)** and **(k)**. In fact, the motorcycle is the most interesting

object in this example. For the singlestreams, **(e)** knows at least the outline of the motorcycle while **(f)** and **(g)** highly misclassified it. However, all the PSPNet-based fusion frameworks are able to select the most useful information from **(e)**. By inspecting the details of the motorcycle in **(i)**, **(j)**,**(k)** and **(l)**, a distinct difference can be explored between the traditional fusion frameworks and the proposed one. The **(l)** obtains the finest details of the motorcycle which implies that the proposed fusion architecture is not only able to select the most efficient singlestream information, but also can refine the merged information on its own. This is most likely due to the additional RRB and CAB units which focus more on the details of object shapes. Note that for the scanning artefacts appeared in the lower right part of Ground truth **(d)**, only **(l)** can exploit **(f)** and **(g)** to classify it almost perfectly, which is another proof of the attention-to-detail ability acquired by the proposed fusion approach.



(a) *Color input* ***(b)*** *Depth input* ***(c)*** *Normals input* ***(d)*** *Ground truth*

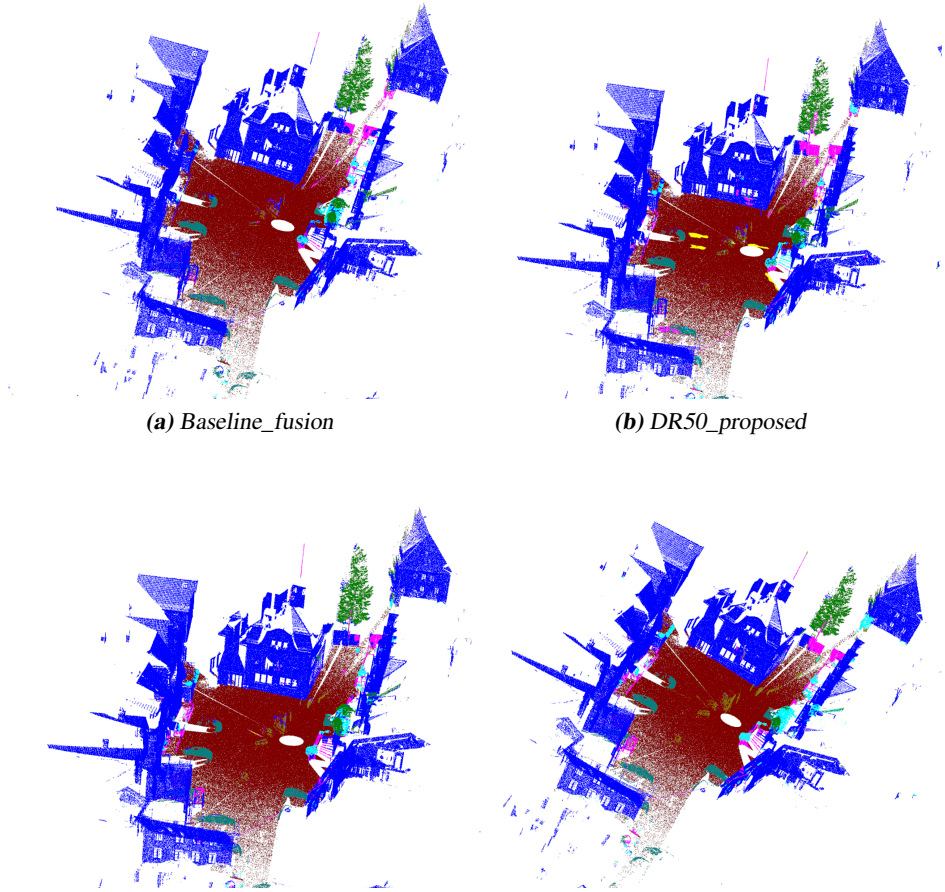
(e) *PSP_color* ***(f)*** *PSP_depth* ***(g)*** *PSP_normals* ***(h)*** *Baseline_fusion*

(i) *PSP_LSF* ***(j)*** *PSP_LMF* ***(k)*** *PSP_LCF* ***(l)*** *PSP_propose_v3*

Figure 5.5: *Example images show how different multistream frameworks utilize their singlestream networks to recognize distant objects. The proposed fusion framework is the most effective one to do that, and to refine the small-scaled objects, in this case the motorcycle specifically.*

5.2.2 Test set of 3D point clouds

Since the ground truth of test data is unavailable, it is difficult to conduct a thorough inspection on the qualitative test results of the different networks. However, it is still viable to sense the minor differences of them by comparing the screenshots taken from 3D test point clouds. Figure 5.6 shows the screenshots on a particular point cloud, generated from the multistream frameworks of interest. How singlestream or early fusion networks perform qualitatively on the test set is not specifically concerned in this thesis though.



(a) *Baseline_fusion*

(b) *DR50_proposed*

(c) *PSP_proposed_v3*

(d) *PSP_proposed_v3_imp*

Figure 5.6: *Screenshots on one of the four 3D test point clouds, generated from the multistream frameworks based on the baseline fusion method (**a**), the proposed fusion architectures with dilated ResNet50 (**b**), common PSPNet (**c**), or improved PSPNet singlestreams (**d**).*

5.3 Discussion

By comparing all the investigated networks quantitatively, it is not difficult to notice that the learning of the fused information across color, depth and surface normals modalities provides more robust results than using any individual modality along. Further, the architectures of the singlestream networks and the way to combine multiple singlestreams do impact the overall performance of multistream frameworks. In this thesis, PSPNet is the most powerful singlestream network that outperformed VGG16-based FCN and dilated ResNet50-based FCN in generalizing unseen test data. Moreover, the third version of our proposed fusion approach outperformed other traditional fusion methods and obtained the highest performance measure among all the experiments. However, the network performance with early fusion could not be improved significantly by adopting the smooth network [53], which introduced the core concept for our proposed multistream fusion architecture.

Through the qualitative comparison between the PSPNet-based proposed fusion method and the other traditional fusion approaches, it is obvious that the proposed one can most effectively utilize the merged information of different modalities and shows the strongest attention to detail ability. Specifically, when multiple singlestream networks generate completely different predictions on certain objects, the proposed fusion framework is able to filter out most of the noises and refine the final output based on its own knowledge repository, gained byRRB and CAB units. On the other hand, the three traditional fusion approaches LSF, LMF, or LCF can be highly influenced by the inconsistent singlestream predictions. Further, all the fusion methods do suffer from the recognition of distant small-scaled objects and irregular shapes which appear infrequently in the training data, but this problem can be more or less alleviated by using techniques such as data augmentation or weighted class. For the baseline fusion framework, the objects such as terrain and building which have regular colors or shapes can be classified decently, but the overlapped or small-scaled objects can be completely misclassified and it is tempted to classify everything as the majority classes, possibly it is because of the lack of ability for utilizing contextual information.

6

Conclusions

In this thesis, different fusion strategies to combine information of multiple modalities are studied. This is done by first investigating the performance of different singlestream networks, i.e. dilated ResNet50 and PSPNet, then applying different fusion methods based on either one of them to explore how a given multistream framework perform under certain conditions. As a result, the third version of our proposed fusion architectures(see figure 3.5) based on PSPNet outperformed the traditional and baseline counterparts by achieving the highest mIoU in both the validation set of 2D images and the test set of 3D point clouds. For the secondary research aim regarding the early fusion improvement, using the fusion architecture similar to the proposed multsitream framework can not significantly improve the performance of the early fused networks. By jointly implementing the additional techniques, i.e. improved pretrained weights, median frequency balancing and data augmentation, on our proposed fusion framework, we were able to obtain a relative improvement of 7.3% mIoU on semantic3D test set compared to the model procuded by the base work [23], increasing the ranking from 12th to 5th place on the benchmark leaderboard at the moment this thesis is written.

Bibliography

[1] Pradeep K Atrey, M Anwar Hossain, Abdulmotaleb El Saddik, and Mohan S Kankanhalli. Multimodal fusion for multimedia analysis: a survey. *Multimedia systems*, 16(6):345–379, 2010. Cited on pages 23 and 24.

[2] Pablo Barros, German I Parisi, Cornelius Weber, and Stefan Wermter. Emotion-modulated attention improves expression recognition: A deep learning model. *Neurocomputing*, 253:104–114, 2017. Cited on page 18.

[3] Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2): 157–166, 1994. Cited on page 12.

[4] Christopher M Bishop et al. *Neural networks for pattern recognition*. Oxford university press, 1995. Cited on page 12.

[5] Mateusz Buda, Atsuto Maki, and Maciej A Mazurowski. A systematic study of the class imbalance problem in convolutional neural networks. *Neural Networks*, 106: 249–259, 2018. Cited on page 20.

[6] Rich Caruana. Learning many related tasks at the same time with backpropagation. In *Advances in neural information processing systems*, pages 657–664, 1995. Cited on page 19.

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE transactions on pattern analysis and machine intelligence*, 40(4):834–848, 2018. Cited on pages 15 and 41.

[8] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016. Cited on page 17.

[9] Navneet Dalal and Bill Triggs. Histograms of oriented gradients for human detection. In *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Com-*

puter Society Conference on, volume 1, pages 886–893. IEEE, 2005. Cited on page 1.

[10] Thomas G Dietterich. Ensemble methods in machine learning. In *International workshop on multiple classifier systems*, pages 1–15. Springer, 2000. Cited on page 23.

[11] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2650–2658, 2015. Cited on page 20.

[12] Andreas Eitel, Jost Tobias Springenberg, Luciano Spinello, Martin Riedmiller, and Wolfram Burgard. Multimodal deep learning for robust rgb-d object recognition. In *Intelligent Robots and Systems (IROS), 2015 IEEE/RSJ International Conference on*, pages 681–687. IEEE, 2015. Cited on page 41.

[13] Mark Everingham, Luc Van Gool, Christopher KI Williams, John Winn, and Andrew Zisserman. The pascal visual object classes (voc) challenge. *International journal of computer vision*, 88(2):303–338, 2010. Cited on page 14.

[14] Kalanit Grill-Spector and Rafael Malach. The human visual cortex. *Annual review of neuroscience*, 27:649–77, 2004. Cited on page 6.

[15] Hatice Gunes and Massimo Piccardi. Affect recognition from face and body: early fusion vs. late fusion. In *2005 IEEE international conference on systems, man and cybernetics*, volume 4, pages 3437–3443. IEEE, 2005. Cited on page 31.

[16] Timo Hackel, Nikolay Savinov, Lubor Ladicky, Jan D Wegner, Konrad Schindler, and Marc Pollefeys. Semantic3d. net: A new large-scale point cloud classification benchmark. *arXiv preprint arXiv:1704.03847*, 2017. Cited on page 35.

[17] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. Cited on page 41.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015. Cited on page 15.

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. Cited on page 12.

[20] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015. Cited on page 12.

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012. Cited on pages 10, 11, and 22.

[22] Max Kuhn and Kjell Johnson. *Applied predictive modeling*, volume 26. Springer, 2013. Cited on page 39.

[23] Felix Järemo Lawin, Martin Danelljan, Patrik Tosteberg, Goutam Bhat, Fahad Shahbaz Khan, and Michael Felsberg. Deep projective 3d semantic segmentation. In *CAIP*, 2017. Cited on pages ix, 1, 3, 14, 21, 35, 39, 42, 47, 49, 53, and 63.

[24] Svetlana Lazebnik, Cordelia Schmid, and Jean Ponce. Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 2169–2178. IEEE, 2006. Cited on page 15.

[25] Yann LeCun, Léon Bottou, Yoshua Bengio, Patrick Haffner, et al. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. Cited on page 10.

[26] Joseph Lemley, Shabab Bazrafkan, and Peter Corcoran. Smart augmentation learning an optimal data augmentation strategy. *IEEE Access*, 5:5858–5869, 2017. Cited on page 20.

[27] Eldon Y Li. Artificial neural networks and their business applications. *Information & Management*, 27(5):303–313, 1994. Cited on page 6.

[28] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5168–5177. IEEE, 2017. Cited on pages 17 and 23.

[29] Min Lin, Qiang Chen, and Shuicheng Yan. Network in network. *arXiv preprint arXiv:1312.4400*, 2013. Cited on page 10.

[30] Kuan Liu, Yanen Li, Ning Xu, and Prem Natarajan. Learn to combine modalities in multimodal deep learning. *arXiv preprint arXiv:1805.11730*, 2018. Cited on page 23.

[31] Geoff Nitschke and Luke Taylor. Improving deep learning with generic data augmentation. 2018. Cited on page 20.

[32] Seong-Jin Park, Ki-Sang Hong, and Seungyong Lee. Rdfnet: Rgb-d multi-level residual feature fusion for indoor semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4980–4989, 2017. Cited on page 23.

[33] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017. Cited on page 41.

[34] Dhanesh Ramachandram and Graham W Taylor. Deep multimodal learning: A survey on recent advances and trends. *IEEE Signal Processing Magazine*, 34(6): 96–108, 2017. Cited on page 31.

[35] Brian D Ripley. *Pattern recognition and neural networks*. Cambridge university press, 2007. Cited on page 37.

[36] Yuji Roh, Geon Heo, and Steven Euijong Whang. A survey on data collection for machine learning: a big data-ai integration perspective. *arXiv preprint arXiv:1811.03402*, 2018. Cited on page 19.

[37] 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. Cited on page 17.

[38] Florian Schroff, Antonio Criminisi, and Andrew Zisserman. Object class segmentation using random forests. In *BMVC*, 2008. Cited on page 5.

[39] Evan Shelhamer, Jonathan Long, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015. Cited on pages 5, 13, 14, 15, 22, 44, and 45.

[40] Nathan Silberman, Derek Hoiem, Pushmeet Kohli, and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *European Conference on Computer Vision*, pages 746–760. Springer, 2012. Cited on pages 23 and 44.

[41] Patrice Y Simard, Dave Steinkraus, and John C Platt. Best practices for convolutional neural networks applied to visual document analysis. In *null*, page 958. IEEE, 2003. Cited on page 20.

[42] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. Cited on page 11.

[43] Cees GM Snoek, Marcel Worring, and Arnold WM Smeulders. Early versus late fusion in semantic video analysis. In *Proceedings of the 13th annual ACM international conference on Multimedia*, pages 399–402. ACM, 2005. Cited on page 31.

[44] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. Sun rgb-d: A rgb-d scene understanding benchmark suite. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 567–576, 2015. Cited on page 23.

[45] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015. Cited on pages 10 and 11.

[46] Abhinav Valada, Ankit Dhall, and Wolfram Burgard. Convoluted mixture of deep experts for robust semantic segmentation. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) Workshop, State Estimation and Terrain Perception for All Terrain Mobile Robots*, 2016. Cited on page 23.

[47] Abhinav Valada, Johan Vertens, Ankit Dhall, and Wolfram Burgard. Adapnet: Adaptive semantic segmentation in adverse environmental conditions. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4644–4651. IEEE, 2017. Cited on page 23.

[48] William N Venables and Brian D Ripley. *Modern applied statistics with S-PLUS*. Springer Science & Business Media, 2013. Cited on page 12.

[49] Jennifer Williams, Ramona Comanescu, Oana Radu, and Leimin Tian. Dnn multimodal fusion techniques for predicting video sentiment. In *Proceedings of Grand Challenge and Workshop on Human Multimodal Language (Challenge-HML)*, pages 64–72, 2018. Cited on page 31.

[50] Lizhong Wu, Sharon L. Oviatt, and Philip R. Cohen. Multimodal integration-a statistical view. *IEEE Transactions on Multimedia*, 1(4):334–341, 1999. Cited on page 23.

[51] Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1480–1489, 2016. Cited on page 18.

[52] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014. Cited on page 19.

[53] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Learning a discriminative feature network for semantic segmentation. *arXiv preprint arXiv:1804.09337*, 2018. Cited on pages 16, 18, 25, 31, 43, and 62.

[54] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. Cited on pages 15 and 16.

[55] Zeng Yu, Tianrui Li, Ning Yu, Xun Gong, Ke Chen, and Yi Pan. Three-stream convolutional networks for video-based person re-identification. *arXiv preprint arXiv:1712.01652*, 2017. Cited on page 23.

[56] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. Cited on page 42.

[57] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *IEEE Conf. on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017. Cited on pages 14, 26, 41, and 42.

[58] Bolei Zhou, Hang Zhao, Xavier Puig, Tete Xiao, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Semantic understanding of scenes through the ade20k dataset. *International Journal of Computer Vision*, pages 1–20, 2016. Cited on page 14.

[59] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, volume 1, page 4. IEEE, 2017. Cited on page 32.