International Journal of Communication Networks and Information Security
ISSN: 2073-607X, 2076-0930
Volume 15 Issue 03 Year 2023

Video Semantic Segmentation Network with Low Latency Based on Deep Learning

Channappa Gowda D V*
Assistant Professor, Department of Information Science and Engineering, the Oxford College of Engineering, Bangalore 560068
Sudee.info@gmail.com
R. Kanagavalli
Professor, Department of Information Science and Engineering, the Oxford College of Engineering, Bangalore 560068
Kanaga.ksr@gmail.com

Article History

Received: 1 November 2023
Revised: 15 November 2023
Accepted: 6 December 2023

Abstract

Recently, new advances in deep learning algorithms have yielded some fascinating results in the field of computer vision technology. As a result, it can now perform activities that formerly required the use of human vision and the brain. Classification, object identification, and semantic segmentation have all seen substantial advancements in deep learning architecture in the last few years. For still images and movies, there has been a major advancement in the field of semantic segmentation. In the field of autonomous driving, effectively segmenting videos semantically remains a complex task, primarily due to the need for high performance, the substantial costs associated with convolutional neural networks (CNNs), and the critical requirement for reduced latency. To address these issues, a specialized machine-learning framework is being developed. This framework utilizes advanced deep learning structures, specifically SegNet and FlowNet 2.0, applied to the CamVid dataset, enabling precise pixel-by-pixel semantic segmentation of video content with maintained low latency. This approach is particularly beneficial for practical applications, leveraging the strengths of both the SegNet and FlowNet architectures. A key feature of this system is its decision-making network, which evaluates each frame to decide if it should be processed by a segmentation network or an optical flow network, based on a calculated confidence score. This decision process is further refined by the introduction of adaptive key frame scheduling, enhancing the efficiency of the system. Performance testing with the ResNet50 SegNet model yielded a mean Intersection over Union (IoU) of 54.27% and an average processing rate of 19.57 frames per second. Moreover, the integration of FlowNet2.0, optimized for GPU operations, improved the frame processing rate to 30.19 fps, with a mean IoU of 47.65%. This improvement was facilitated by a 47.65% utilization rate of the GPU, leading to a notable enhancement in the speed of video semantic segmentation without compromising on quality.

CC License
CC-BY-NC-SA 4.0

Keywords: Semantic Segmentation, Object Detection, Decision Network, CNN

1. Introduction

Over the previous few years, the area of computer vision has attracted the greatest attention from researchers. Many new applications have been developed as a result of the effective implementation of image classification, object recognition, and image segmentation tasks. Face detection, item identification, scene detection, and scene captioning are a few examples of these uses. When it comes to computer vision activities like image and video processing, the emergence of machine learning has a substantial impact on their progress. There are numerous hidden layers in deep learning, which shows that it is a neural network model with a deep structure. An algorithm that mimics the structure of the human brain can recognize patterns that are represented as vectors. To assist the machine in learning features and solve computer vision tasks such as object classification, object recognition, semantic segmentation, and so on, these representations and the related weights are necessary. Mimic the human brain, this makes item recognition easier for machines. Wouldn't it be fantastic if a computer could tell us what we're looking at in the video?

Examples of jobs that focus on the finer elements of an image are image categorization and object detection. After moving from a coarse to a fine degree of inference, making predictions on a pixel-by-pixel basis, or semantic segmentation is inevitable. Semantic segmentation is one of the most popular subjects in computer vision research. It's called semantic segmentation when each pixel in an image or video is assigned a separate category. Use this method for videos as well.

Historically, the exploration of semantic segmentation relied on 2D attributes like color and shape [1], while the use of dense depth maps [2] required extensive manual engineering, leading to high computational demands. However, the emergence of deep learning techniques and advancements in machine learning have simplified the challenges of semantic segmentation and classification, eliminating the need for intensive human input. The application of Convolutional Neural Networks (CNNs) in the semantic segmentation of images has demonstrated superior outcomes [3, 4], offering enhanced analysis and labelling of image features with improved efficiency. CNNs have significantly influenced the field of computer vision, both in elevating the quality of results and in accelerating computational processes.

Sophisticated work in the area of picture semantic segmentation has been carried out in recent years. Despite video being the most widely used multimedia application, it receives the least research and management attention due to its complexity and processing time. With CNN's success in image semantic segmentation and other image processing tasks, fascinating new developments in video semantic segmentation are emerging. A substantial body of research [5]-[7] has concentrated on enhancing the performance of video segmentation. However, there has been relatively limited investigation focused on minimizing latency [8]-[11].

Numerous pivotal projects, such as FCN, PSPNet, U-Net, SegNet, and DeepLab [3], [13]-[16], have made significant contributions to semantic segmentation research. These networks are renowned for their high accuracy in prediction, albeit at the cost of longer processing times per frame. The current research focuses on developing a video semantic segmentation framework suitable for real-time applications, like autonomous driving and indoor navigation. Key to real-time application is achieving low latency, a goal addressed in previous studies like dynamic video segmentation (DVSNet) and deep feature flow (DFF), which involves dual networks [9], [11]. Both DVSNet and DFF incorporate optical flow and per-frame segmentation networks. Crucial components of this research include a decision network, adaptive keyframe update mechanisms, label mapping, and depth inference strategies. This study aims to evaluate the network's performance using an encoder-decoder architecture like ResNet50 SegNet, replacing the DeepLab in the segmentation network, while retaining the same flow network as in [9] and [11]. Experiments were conducted using the CamVid [17] dataset, a driving scenario dataset akin to CityScape [18].

Section 2 provides a literature review of various deep-learning techniques that were used in this work. Section 3 provides a methodology for each network as well as the working procedure of the video segmentation network. Section 5 provides the experiments that were conducted and their outcomes, together with a quantitative and qualitative interpretation of those data, are explained in detail with the results. The last section discusses the conclusion, where we discussed the benefits and drawbacks of using this segmentation strategy, as well as future work.

2. Related Work

Semantic segmentation represents a cutting-edge development in the field of computer vision. Historically, achieving pixel-level classification and masking in images was highly challenging, largely due to the extensive manual computations required. Techniques like dense depth mapping, 3D geometry analysis, and superpixel segmentation were commonly employed in these complex processes. Towards the later part of the 20th century, many image segmentation and classification datasets were painstakingly created using a variety of software tools. However, the advent of deep learning and the implementation of convolutional neural networks have radically transformed this landscape. These advancements have not only streamlined semantic segmentation research but have also greatly simplified the process of data creation for such tasks.

Image segmentation is one of the areas in which deep learning in computer vision has shown to be one of the most successful applications. The work that was done on the building of a fully convolutional network (FCN) [3] is noteworthy and ground-breaking in the field of picture semantic segmentation. In this study, a fully convolutional network was utilized instead of a classification network's fully linked layers to achieve the best results. This work represented a considerable advance. Before the emergence of Fully Convolutional Networks (FCN), semantic segmentation architectures predominantly relied on convolutional networks capped with a fully connected layer at the output [21]-[23]. FCN introduced a significant shift by transforming this final fully connected layer of the classification segment into a convolutional layer. This modification not only reduced the overall number of parameters but also eliminated the restrictions related to the size of the input image. Most importantly, it enabled the preservation of spatial information by discarding the fully connected layer.

O. Ronneberger et al.'s U-Net architecture for medical imaging [14] draws inspiration from the Fully Convolutional Networks (FCN), particularly in its design of expanding and contracting segments. Similar to DeConvNet [24], U-Net utilizes a combination of downsampling and upsampling techniques to retain information that could be lost in the downsampling process. The contracting part of the network is chiefly tasked with generating feature maps, using 3x3 convolutions to effectively extract features during the downsampling phase.On the other hand, deconvolution is utilised during the expanding stage, which results in fewer feature maps but increased spatial dimensions (height, width). U-Net can keep track of pattern information because it copies the feature maps that have been cropped from the contracting party to the expanding part. The final step involves applying a 1x1 convolutional layer's processing to these feature maps to produce segmentation maps and pixel-by-pixel classification of the input image.

SegNet [15], a specialized fully convolutional network, was devised with a focus on segmenting road scenes. It is characterized by an encoder-decoder structure and incorporates various elements such as convolutional layers, batch normalization, max pooling, the ReLU activation function, and a Softmax classifier. In SegNet, the VGG-16 network, particularly the initial 13 convolutional layers of VGG-16, is employed as the encoder [25]. This network is well renowned for its ability to solve classification challenges. To maintain the high-resolution feature maps at the encoder output, SegNet, which is also a fully convolutional network, removes the completely connected layer that is the topmost layer in the network. The output of the final decoder is passed to the Softmax classifier, which then generates class probabilities.

The DeepLab architecture presents a novel solution to address the issue of spatial resolution loss, a common problem in deep convolutional neural networks (DCNNs) due to repeated max pooling and downsampling in deep convolution layers. This design also tackles challenges associated with varying object scales and imprecise localization. To counteract these issues, DeepLab introduces an innovative approach called "atrous convolution," which involves upsampling the filters prior to convolution. This method substitutes the downsampling stages with filter upsampling in the convolutional layers following the max pooling stages, thereby addressing errors typically introduced during max pooling. This adjustment enables the computation of feature maps at larger sample rates. The architecture then restores the original image dimensions by applying atrous convolution and subsequently performing bilinear interpolation on the feature maps [16], [26].

Within the realm of video analysis, one of the most fundamental aspects of work is known as optical flow. It is essentially a two-layered matrix with the same dimensions as the input frame; the difference between the layers is the offset of each pixel along two distinct axes (i.e., x-axis and yaxis). One of the earliest methods for calculating optical flow is the variational approach, which long stood as a benchmark in the field. Initially, this method was primarily focused on smaller motions, often favoring initial conditions with a zero motion field due to the presence of local minima [29]. Over time, strategies like the Lucas-Kanade method, known as a patch-based approach, gained prominence. This technique involves dividing two frames into patches and then aligning them by minimizing the gradient. Additionally, Brox and Malik [30] introduced a unified approach that combines patch-based methods with variational refinement.

Several algorithms have been developed based on the Brox and Malik method, including PatchMatch [31], FlowField [32], DiscreteFlow [33], and FullFlow [34]. In both DiscreteFlow and FullFlow, Conditional Random Fields (CRF) [27, 28] play a crucial role in processing vector similarities. Another significant development is DeepFlow by Weinzaepfel and colleagues [35], which is influenced by the work of Brox and Malik. DeepFlow merges a variational optical model with a loss function and is based on a deep matching algorithm. This allows for the integration of feature descriptors and matching techniques. Other notable approaches in this domain include EpicFlow and RichFlow [36, 37], which represent unified or combined matching methodologies. Nevertheless, each of these methods is a handcrafted creation [29]. The DeepFlow algorithm is similar to a deep learning method; however, it does not use weights that have been previously learned.

In recent times, the fields of semantic segmentation and computer vision have increasingly adopted deep learning techniques alongside optical flow methods. FlowNet, a notable example, utilizes a single-stack architecture within the conventional CNN framework to compute optical flow. This system processes a pair of images as input and generates optical flow fields as output. FlowNet introduced two distinct designs known as FlowNetS (Simple) and FlowNetC (Correlation). Designed to handle large displacements, FlowNet underwent training on a dataset featuring flying chairs and three-dimensional objects, enabling it to effectively manage complex visual data.

Because image semantic segmentation using deep learning was such a tremendous success, there are now many current studies focusing on video semantic segmentation for applications such as autonomous driving, indoor navigation, augmented reality, and many more. Utilizing image segmentation networks is one of the most straightforward approaches to accomplishing video segmentation. However, they operate on a "per frame" basis, which makes the computation more complicated given that videos contain a greater number of frames than still photos. In addition, picture semantic segmentation does not take into account the temporal links between different frames. Exploiting the temporal relationship that exists between the frames of the video leads to an increase in the accuracy of the prediction [41].

Gadde et al. [41] demonstrated semantic segmentation of video via representation warping. In this strategy, the authors transfer CNN models for image segmentation to video models. They start by introducing a warping module called NetWarp, which warps intermediate CNN representations and mixes them with the current frame. Their network warps internal network representations using nearby optical frames. Their concept builds on PSPNet [13].

The frames are processed in an online manner by the video CNN models that are produced as a result [41]. The NetWarp technique was shown to obtain good performance on semantic segmentation; however, as their feature calculation is performed on a per-frame basis, they are not very effective in terms of minimising the number of computations required. The warping approach does make it possible to increase the speed of the video, but its effectiveness is constrained by video dynamics. This refers to recordings that contain moving objects, in which the entire scene moves relative to the camera, which results in a considerable warping inaccuracy.

Deep feature flow techniques effectively reduce calculation time by 74%. This was accomplished by proposing the use of two distinct networks, a segmentation network and an optical flow network, alongside fixed keyframe scheduling. However, because it uses fixed keyframe scheduling (implying a constant update time between subsequent frames), it lacks both adaptability and customization [11]. The unsupervised learning strategy for video semantic segmentation that was proposed by Zhang et al. [47] introduces a completely conventional adaptive network for semantic segmentation. Whereas, they plan to use labelled instances from the source domain in conjunction with a large number of unlabeled examples from the target domain to reduce the amount of error in their predictions of the target data [47, 48]. On the other hand, the supervised learning strategy [49] is the one that we will use for our study. To achieve low latency for real-time applications, this project aims to create a network architecture similar to DVSNet using a separate segmentation network (ResNet50 SegNet) on a road. In this case, the purpose is to speed up the transfer of information from one location to another.

3. Methodology

This research is motivated by an existing work of dynamic video segmentation network (DVSNet), which segments video frames using both per-frame and optical flow network. The main goal is to achieve low latency for a video segmentation network. Utilizing both of these extremely powerful networks enables us to get the benefits of both networks. The accuracy with which the segmentation network can predict the result of its operations is very high. However, it is well known that Optical Flow networks can predict motion information extremely quickly. This, in conjunction with spatial warping (in which the optical flow output is warped with the segmentation output), helps to make the segmentation process both faster and more performant. However, none of these networks is without its flaws; the segmentation network is sluggish because its processing is done on a per-frame basis, and the flows of the optical flow network can be erroneous at certain periods. Detailed descriptions of network architectures and strategies used in this study are provided in this section. In this study, we use three low-latency networks for video semantic segmentation. But we utilise a segmentation network that is different from SegNet and FlowNet2S as an optical flow network with spatial warping function, SegNet with ResNet50 encoder as a segmentation network. Judgment networks (DN) are added to make the final decision based on the difference between consecutive frames. In order to determine the difference between two frames, DVSNet uses the frame region technique. Contrary to this, we use an adaptive key frame scheduling policy on the full frame instead of the traditional "frame region" approach. To begin, we'll go over the many types of network designs that have been implemented as part of this project.

In computer vision, ResNet (ResNet) is a common architecture that is used frequently. ImageNet has achieved the best score of 96.4 percent accuracy in this design. Adding outputs from a previous layer to the outputs of a stacked layer is done using the skip connection which is as shown in Figure 1. In contrast to LeNet, VGG16, etc., ResNet can train a deeper network without vanishing gradient descent. ImageNet categorization has been enhanced by using skip connections. Semantic segmentation is a strength of theirs.

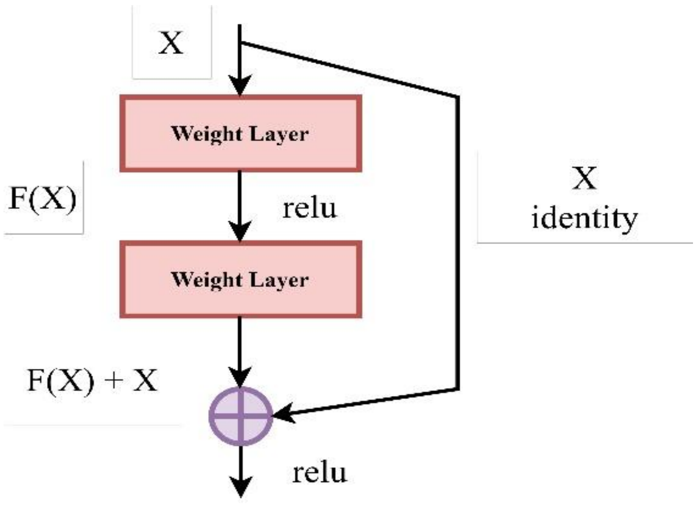

Figure 1. Residual Building Block

It is possible to utilize ResNet as a fixed feature extractor, to fine-tune a neural network, or to train a new dataset because it is a model that has been trained on ImageNet and includes the weights for each feature. It helps with similar issues. Versions of ResNet include ResNet18, ResNet50, and ResNet101. At 18, 50, and 100, ResNet101 is the deepest network. SegNet's encoder backbone will be ResNet50 [15], [19].

$$y = F(x, \{Wi\}) + WsX \qquad (1)$$

Where, Y is the output function, function F(x, {Wi}) represents the residual mapping to be learned, Ws is the weight of identity mapping.

3.1 SegNet

SegNet is characterized by its use of convolution, batch normalization, ReLU activation, and max pooling. Its decoder, as illustrated in Figure 2, includes components like upsampling, deconvolution, and a Softmax classifier. The SegNet encoder employs thirteen convolution layers. In contrast, our research utilizes ResNet50, a more intricate network than VGG16, for both encoding and decoding tasks. For initial training, ResNet50 weights are preloaded from ImageNet. We refer to our combination of the ResNet50 encoder with the SegNet decoder as ReSNet. SegNet, being a fully convolutional network, has its fully connected layer omitted, which simplifies the encoder network by reducing the number of variables it needs to handle.

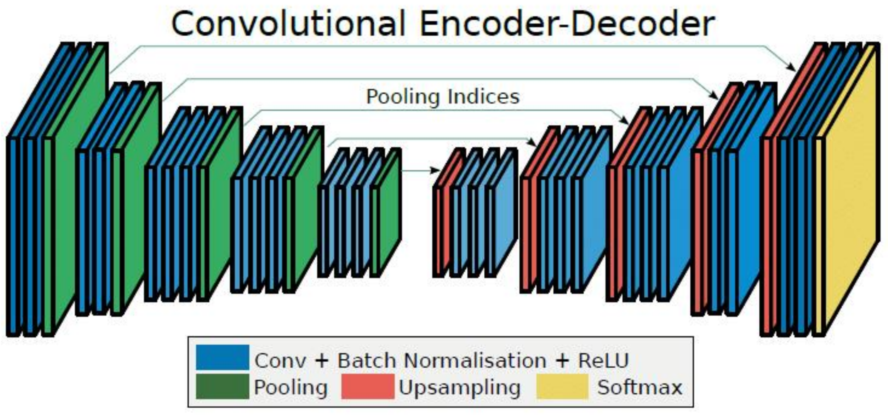
Convolutional Encoder-Decoder
Pooling Indices
Conv + Batch Normalisation + ReLU
Pooling
Upsampling
Softmax

Figure 2. Illustrating the SegNet Architecture

Encoder input is convolution-filtered to build feature maps. These feature maps are batch normalised and ReLU is applied element-wise. Max pooling uses a non-overlapping 2x2 window and stride 2 to ensure translation invariance when an input image frame is slightly shifted. Max-pooling output is subsampled to get a big input image frame context for every feature map pixel. Max-pooling and subsampling can improve translation invariance. This reduces feature map resolution. Such lossy image representations aren't suitable for semantic segmentation, where boundary information is crucial. Storing border information from feature maps can alleviate this problem, but it's impractical. SegNet stores only Max-pooling indices, which contain each window's maximum value. Rather than keeping all border information in feature maps, use this method. Corresponding decoders in the network upsample feature maps using encoder max pooling indices to yield sparse feature maps. At the decoder, filter banks are used to convolve sparse feature maps into dense ones. Softmax classifier calculates the probability for each class using high-dimensional representations from the final decoder. The first encoder's decoder produces multi-channel feature maps, even if the encoder's input has 3 channels (i.e. RGB). Other network decoders' output feature maps are the same size and channel as encoder inputs. SegNet segments frames. The next frames are processed after the preceding frame is finished. Each frame in a 3-minute video can take a lot of time to process. They're sluggish yet good at segmenting output. ResNet50 SegNet is trained on CamVid using ResNet50's ImageNet weights.

3.2 FlowNet

FlowNet is an optical flow algorithm that uses a convolutional neural network. The movement of every pixel in a picture that is caused by relative object movement between frames or by a moving camera is referred to as optical flow. For calculating object motion correction, FlowNet2.0 will be employed. As depicted in Figure 3, FlowNet2.0's architecture is composed of three tiers: Correlation, Simple, and Small Displacement. Each level of this layered network, namely FlowNet-C, FlowNet-S, and FlowNet-SD, is tailored to handle large shifts between objects within an image frame. It's important to note that the datasets utilized by FlowNet-S and FlowNet2S differ in their composition. Flying chair practice takes place without set schedules prior. While Slong went through 1.2 million iterations, Short only went through 600 thousand.

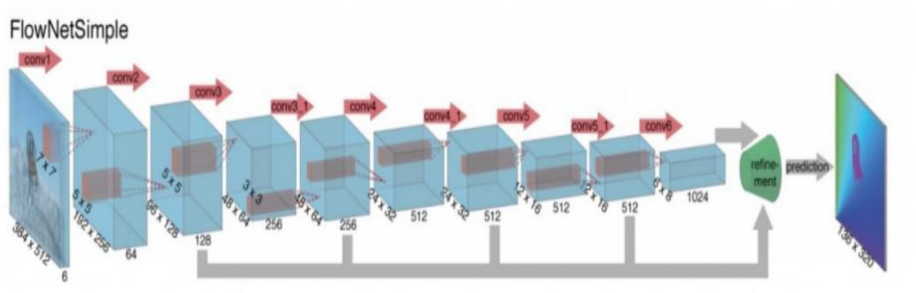

Figure 3. Architecture of FlowNet-Simple

The FlowNet2-Simple architecture, functioning as an encoder-decoder, is designed to create flow fields between two images. It operates by taking two frames, a key frame (Ik) and the current frame (Ii), as inputs, and produces a flow field (F) as the output. FlowNet2S is solely focused on generating flow fields and does not provide segmented outputs. For the final segmentation output, the plan is to integrate the outcomes from the segmentation network with those from the optical flow network. This combination will account for the spatial movements captured in the feature maps at each convolutional layer, tracking changes between the keyframe and its corresponding frame. FlowNet2 offers a significant speed advantage over per-frame segmentation, leading to faster computation times and thus enhanced segmentation speeds.

3.3 Spatial Warping

The integration of extra convolutional layers enables the encoding of both semantic and spatial aspects of an image. Thanks to these spatial maps, the transfer of feature maps through spatial warping becomes more efficient and cost-effective. Flow fields Mi>k can be computed using a flow estimation method, denoted as F, which is applied according to Equation (2):

$$Mi \rightarrow k = F\,(Ik, Ii) \qquad (2)$$

The propagation of feature maps involves expanding them to match the dimensions of the flow fields through bilinear expansion. In this process, an object's position p in the current frame I is shifted to p + δp in the keyframe k.

Where, $\delta p = M_{i\rightarrow k}(P)$. Feature warping is achieved by bilinear interpolation using Equation (3):

$$f_i^c(p) = \sum_q G(q, p + \delta p) f_k^c(q) \qquad (3)$$

In this context, 'q' represents the enumeration of all spatial locations within the feature maps, while 'F' stands for the Feature maps themselves. 'G' denotes the two-dimensional bilinear interpolation kernel, and 'C' refers to the total number of channels present in the feature maps.

$$G(q, p + \delta p) = g(q_x, p_x + \delta p_x).\, g\left(q_y, p_y + \delta p_y\right) \qquad (4)$$

Where, g(a,b) will be equal to max(0, 1-|a-b|).

Due to potential inaccuracies in computed flows, object occlusions, and other variables, spatial warping might not always be precise. To refine the estimation of features, it's essential to adjust their magnitudes using "scale fields" Sik. These scale fields share the same spatial and channel dimensions as the feature maps, ensuring a congruent scaling process. The calculation of the scale field involves applying a scale function to both the keyframe and the current frame, as delineated by the following set of equations.

$$S_{i\rightarrow k} = S(I_k, I_i) \qquad (5)$$

The following equation is used to calculate propagated feature maps:

$$fi = W(f_k, M_{i\rightarrow k}, S_{i\rightarrow k}) \qquad (6)$$

In this scenario, the operation W, as defined in Equation (6), performs feature warping across all regions and channels of the feature maps. This warping is executed prior to the multiplication of these maps with the scale fields.

3.4 Label Mapping

In the process of label mapping, the target labels of FlowNet2S and ResNet50 SegNet are amalgamated. Specifically, FlowNet2S features 19 labels, while ResNet50 SegNet is equipped with 12 labels within the Cityscape dataset. Given that both datasets focus on road scenes and share similar labels, their intersection results in our derived label set. This culminates in a consolidated collection of 11 labels. Following this mapping process, one label from CamVid is categorized as 'Unlabelled', and eight labels from Cityscape are excluded from consideration.

3.5 Decision Network

The Decision Network (DN) is structured with four layers, comprising one convolutional layer and three fully connected layers, topped by the final layer. DN's role is to determine whether to use the flow or segmentation network path for predicting segmentation output. It processes inputs from hidden layers (layers 4 to 6 of the Convolutional layer) and calculates a predicted confidence score based on the difference in ground truth pixels between Oout and Sout (outputs from the segmentation network path). The network uses the flow prediction (F) to estimate Oout. The frame ground truth confidence score is determined as per Equation (7).

$$\sum_{p \epsilon P} \frac{\left(C(O_{out}(p), S_{out}(p))\right)}{P} \qquad (7)$$

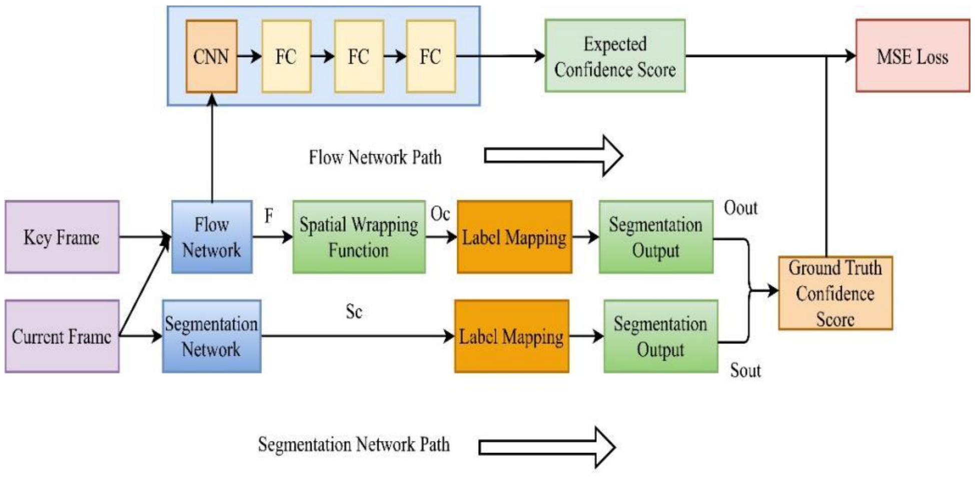

Figure 4. Figure Illustrating Decision Network Architecture

Inference compares a goal threshold 't' to a frame's predicted confidence score to pick a path (flow or segmentation). Figure 4 demonstrates the decision network design with segmentation and flow label mapping.

3.6 Working Procedure of Video Semantic Segmentation Network

This section includes training and evaluation, system working, evaluation measures, assumptions, and restrictions. This paragraph describes low-latency video segmentation networks. Figure 5 displays our semantic segmentation network's response to the decision network. During our network's initial iteration, the current frame is sent to the segmentation network path to generate segmentation output. Next, the difference between key and current frames is determined. The decision network determines each frame's predicted confidence score based on their difference to segment. 't' is a 0-100 inference threshold. The current frame is delivered to the segmentation network if a decision network's expected confidence score is over the threshold. FlowNet2S is unable to create reliable predictions due to the enormous amount of pixels dividing the keyframe from the current frame. The key frame and segmented network paths were changed as a result. Whenever the expected confidence score is smaller than the cutoff, the optical flow network, FlowNet2S, is thought to generate forecasts comparable to those produced by the segmentation network.

Network flow routed. Segmentation allows just current frames. Keyframe and current frame enter flow network path. ResNet50 SegNet segments output when the decision network chooses segmentation. Frames delivered over FlowNet2S are distorted by using the most latest segmented result whenever the decision network selects a flow network path. Flow network scale fields are label-mapped to further classify the 11 classes. These label-mapped scale fields and distorted output produce segmented flow network output. The optical flow network alone cannot segment. Spatial warping helps achieve segmentation on a flow network path comparable to a segmentation network.

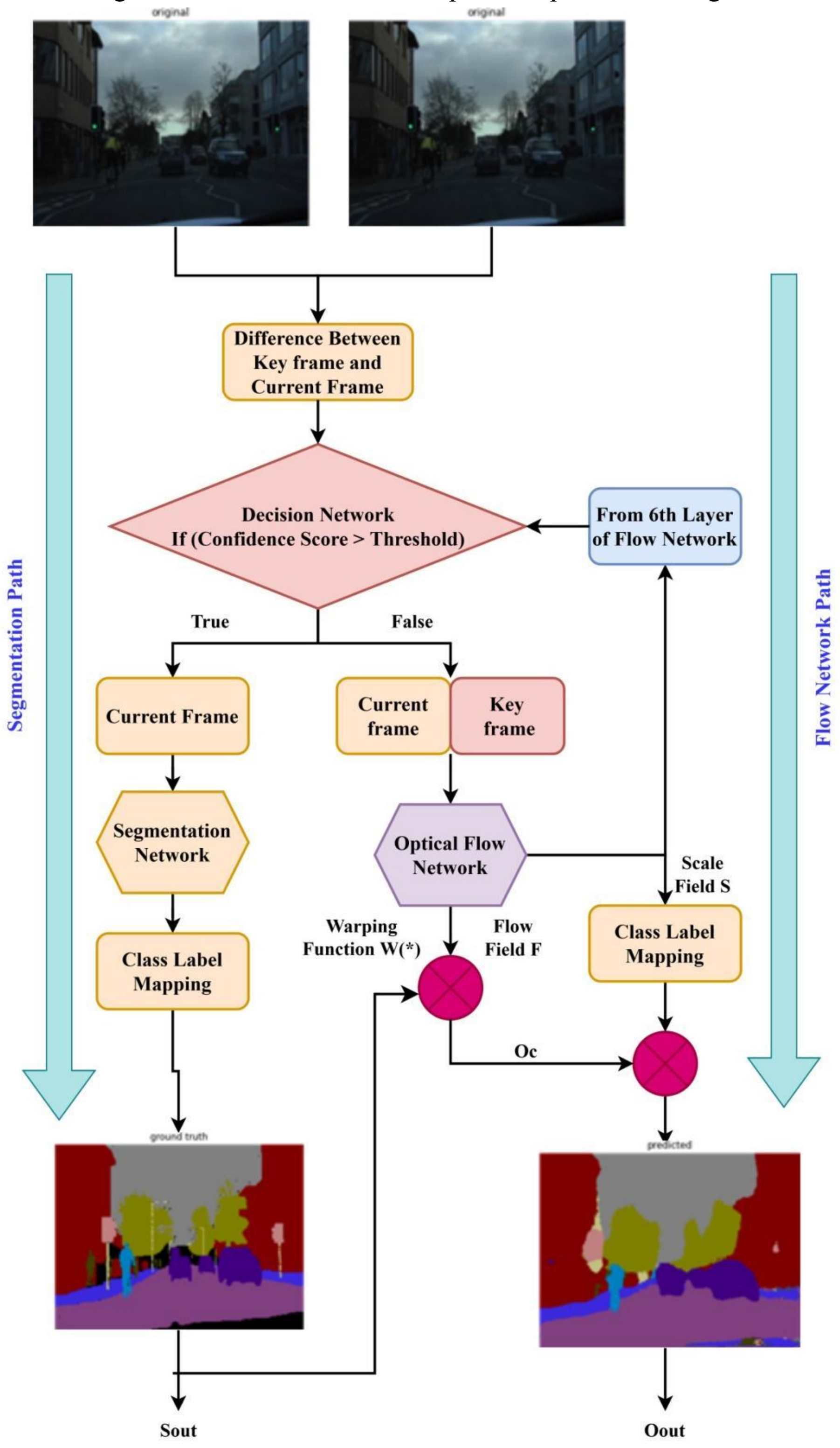

Figure 5. Video Segmentation Network Architecture

First, we infer the segmentation network alone. NVIDIA Tesla T4 GPU and the CPU are used to determine the inference time for a given set of input videos on Google Colab. ResNet50 SegNet's per-frame processing is sluggish, but it produces accurate segmentation. An optical flow network with spatial warping is quick, but not exact. Thus, switching between two networks and using a flow network path most of the time and going through a segmentation network only when frame difference is greater speeds up segmentation and generates efficient segmentation. When a big frame difference is detected between frames, the keyframe is updated. Adaptive keyframe scheduling reduces processing time.

4. Results and Discussion

4.1 Training

A decisions network, a segmented network, as well as an optical flow network including warping, are all included in this research. Training will be provided for segmentation and decision-making networks. For the optical flow network, we'll train the FlowNet2S [38, 39] using the cityscape dataset [18]. As encoders and decoders for the Imagenet database, SegNet has been trained using ResNet50 and then fine-tuned using CamVid [19]. The Ada delta optimizer handles the learning rate in an epoch-adaptive manner. To train ResNet50 SegNet, we used 367 images from CamVid, each at 10-step durations. ResNet50 SegNet employs ResNet50's ImageNet-trained weights. Transfer learning reuses model weights on a similar dataset. Thus, transfer learning is a highly effective strategy because the previously trained model (ResNet50) is well-versed in many labelled categories and contributes to issue performance improvement by fine-tuning our dataset. Therefore, faster training and less generalization error. The segmentation network's output is label-mapped to 11 classes. We discard the "unlabelled" or "Void" class from the CamVid dataset during label mapping. This procedure maintains class labels after merging the segmentation network: FlowNet2S with ResNet50 SegNet. FlowNet2S was then coupled to DVSNet to use its checkpoints and corresponding labels were mapped utilizing scale variables. [9, 10] Applying checkpoints from the baseline FlowNet really wouldn't enhance flows while gliding chairs and 3D-objects data are distinct from road scenes/autonomous driving scenarios because the CamVid dataset [17] is tiny enough to train FlowNet on data samples of road scene. As a result, we use the DVSNet design's FlowNet2S developed on a cityscape sample (1.2 million data sets) [38, 39]. We map FlowNet2S scale field class labels to 11 classes after warping ResNet50 SegNet segmentation output and flow. Now segmentation and flow network class labels match. Multiplying distorted output and label-mapped scale fields predicts flow network path segmentation.

Finally, we train the decision network with 6th-layer flow features. Using flow network features, the decision network is trained as a regression model. Continuous learning should allow the decision network to forecast the expected confidence score for a particular frame. 500 epochs of training are done in 32-epoch batches with a 0.99 decay rate. Continuous learning and comparing predicted confidence score to ground truth confidence score makes the decision network increasingly accurate at predicting a frame's expected confidence score. Thus, segmentation and flow network paths may be switched efficiently. The ground truth confidence score is exclusively utilized for decision network training [9].

We used the CamVid dataset to develop the custom ResNet50 SegNet design, consisting of 701 video sequences having ground-truth labelling for 11 categories. A 30fps video of CamVid was utilized in our research. The first 29 frames are critical, and the current frame is 30. ResNet50 SegNet just accepts the current frame as input, whereas FlowNet2S accepts the keyframe as well.

4.2 Experiments

The programming platform of Google Colab, and the specification of 2.2GHz Intel Core i7, 16GB DDR4 RAM, and an NVIDIA GPU were used throughout the research.

4.3 Experiment 1: Evaluation of Baseline Network

Using cross-entropy loss function and Ada delta optimizer, Resnet50 SegNet was trained at different epochs. When the model stabilizes, accuracy and loss at different epoch rates are recorded. MIoU values at different epoch rates were also tested. For epochs 10 to 50 and 100, MIoU and Classwise MIoU on 233 current frame images were compared. The autonomous driving environment was segmented into smaller groups overall. Result 1 compares and discusses ResNet50 SegNet and SegNet's segmentation predictions together with classwise MIoU scores at different epochs.

4.4 Experiment 2: Evaluation of Video Semantic Segmentation Network

In this experiment, we connect FlowNet2S with ResNet50 SegNet to reduce video segmentation network latency while maintaining quality. After integrating the two networks, the Decision network predicts the confidence score. On a test set of 6959 keyframes and current frames, the combined network (a video segmentation network) is seen. Video Segmentation network (ResNet50 SegNet + FlowNet2S) segmentation prediction is compared with each frame (ResNet50 SegNet) segmentation prediction to evaluate output quality and network speed. Result 2 has a detailed discussion of the results.

4.5 Experiment 3: Validation of a Video Segmentation Model at Various 't' Thresholds

To test the decision network's performance by altering the threshold from 0-100. When 't' is near '100,' the segmentation network path predicts the output. The flow network path for estimated output is more likely to be chosen when 't' is close to zero. Considering this anticipated behavior, we test the system with different threshold settings.

4.6 Experiment 4: Depth Inferences between Objects

The distance of an object from the driver's perspective was depicted. Determine in 2D the separation between the driver and the centre of the targeted object you've selected. In this method, the driver's reference point is the x-center of the image frame, and the target object's centre is its diagonal midway. We only compute distances to cars, pedestrians, cyclists, and poles. Euclidean distance between two points is computed using Equation (8):

$$D_{euclidean} = \sqrt{(x_2 - x_1)2 + (y_2 - y_1)2} \quad (8)$$

The pixel distance is then transformed to metres by applying the equation 9, pixel in metres = 0.0002645833. In self-driving cars, depth information helps adjust the vehicle's speed based on how far or close the item is. Helps avert accidents.

4.7 Outcome 1: Results of the Baseline Model

In this analysis, we delve into the outcomes of Experiment 1, with a focus on Figure 6 and 7. These figures illustrate the predicted outputs for ResNet50 SegNet and the Original SegNet respectively. Each figure presents the input image on the left, the predicted output on the right, and the ground truth in the center. In these predictions, distinct colors are assigned to each class, such as purple for cars and red for buildings, mirroring the color coding in the ground truth images. Notably, the predicted output in Figure 7 closely aligns with its corresponding ground truth label and the predictions made by the Original SegNet in Figure 6. However, some classes exhibit less accurate segmentation maps. For instance, the pole segmentation mask in Figure 7 is noticeably less smooth compared to its ground truth counterpart.

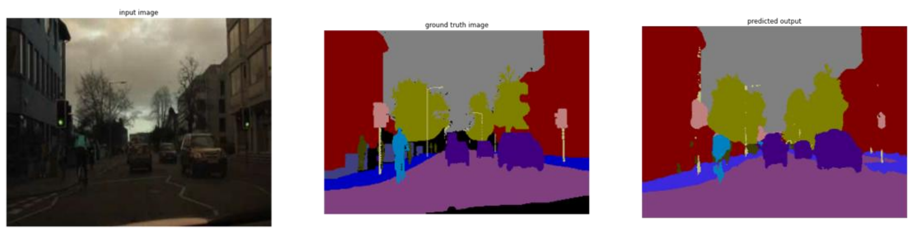

Figure 6. Authorized SegNet implementation

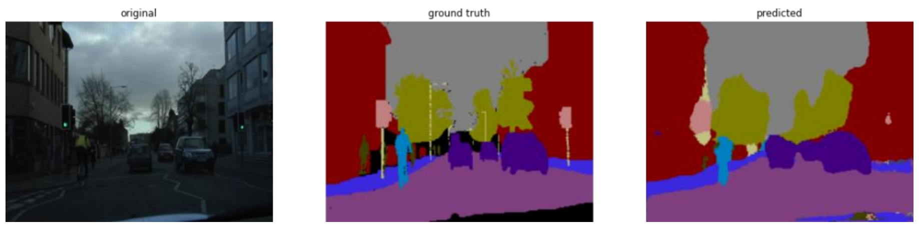

Figure 7. Customised ResNet50_SegNet

Table 1 presents the Intersection over Union (IoU) values for each of the 11 categories. At the 50-epoch mark, the class-specific IoU results on the test dataset were quite impressive, showing improvements over earlier epochs. However, there was an exception for the 'pavement' category, where the Mean IoU (MIoU) saw a 2% reduction. Increasing the number of epochs to 100 led to a decline in class-wise IoU for categories such as buildings, cyclists, fences, pedestrians, and trees, each dropping by more than 1% on the test data.

Table 1. Each Class Label's IoU Classwise

ResNet50_SegNet						
Classes	ClassWiseIoU in % on test set					
	10 epochs	20 epochs	30 epochs	40 epochs	50 epochs	100 epochs
Sky	85.93	86.34	86.60	86.95	86.86	86.81
Building	73.94	74.64	75.04	74.84	75.088	73.55
Pole	3.7	6.44	10.7	10.88	12.49	14.73
Road	88.01	88.7	89.88	89.86	89.28	90.64
Pavement	64.99	66.89	69.68	69.25	67.67	71.76
Tree	69.22	70.22	70.48	70.82	71.09	69.19
Sign symbol	28.77	31.67	31.62	31.70	34.29	32.02
Fence	20.54	25.85	25.23	25.21	27.74	25.98
Car	70.05	72.13	72.74	73.49	73.44	73.89
Pedestrian	25.56	21.19	31.91	32.71	34.11	32.48
Bicyclist	13.1	22.38	27.51	26.19	32.47	37.165

The observed decline in accuracy across several categories with an increase in epochs could be indicative of model overfitting, where the model becomes less adaptable and underperforms on unfamiliar data. Such errors are particularly concerning in autonomous driving applications, emphasizing the need for a model that generalizes well. After thorough evaluation, the 50-epoch mark was identified as optimal for maintaining class generality. In the CamVid test set, ResNet50 SegNet achieved a mean IoU of 54.27% and demonstrated speeds of 1.92 fps and 19.57 fps on CPU and GPU respectively, as detailed in Table 2. The ResNet50, a lighter and moderately deep encoder network, outpaced the speed of DVSNet's deeplab-fast network on GPU. It is hypothesized that deeper networks like ResNet100 or ResNet152 could enhance prediction accuracy. Table 2 also draws a comparison between the original SegNet and ResNet50 SegNet results. The MIoU of our baseline network is roughly 6% different from the SegNet benchmark, suggesting that ResNet50 SegNet matched the segmentation quality of the original. Although it didn't surpass the original method's scores using ResNet50 as an encoder, it achieved comparable results in fewer iterations, close to 100,000.

4.8 Outcome 2: Results of Video Semantic Segmentation Network

In Experiment 2, we observed the impact of integrating the Optical flow network with the baseline model. The resulting predicted segmentation outputs, as shown in Figure 8, demonstrated a decline in quality, appearing somewhat distorted. While the output of ResNet50 SegNet bore resemblance to the model's flow network path, distortions were noticeable in several class predictions. Particularly, classes like poles and fences were only partially predicted. There were instances of false predictions, such as erroneously identifying a sign as a pole, which can be critical in certain contexts. This issue of partial segmentation and distortions could be attributed to the loss of information during the flow generation process.

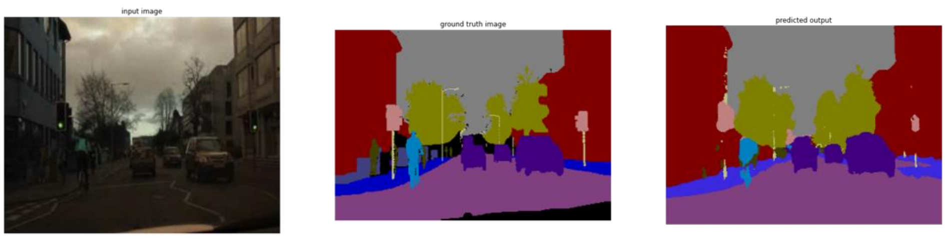

Figure 8. ResNet50_SegNet+FlowNet2S

In the balancing mode, the combined network achieved a Mean IoU (MIoU) of 47% and processing speeds of 16.84 fps and 31.27 fps on CPU and GPU respectively, as noted in Table 2. The integration of the Flow network resulted in a 6% decrease in quality but an increase in processing speed. This reduction in quality by 6% could likely be due to information loss during the flow generation phase. Quality is better preserved near the segmentation network when the Flow network path is selected, thanks to spatial warping. However, as the discrepancy between consecutive frames widens, the spatial warping of segmentation network output with less accurate flows leads to a decrease in both quality and MIoU values.

Table 2. Comparison of SegNet [15] Mean IoU Scores with our Developed Model ResNet50_SegNet (our baseline) and ResNet50_SegNet+ FlowNet2S

Network	Mean IoU	Average fps on CPU	Average fps on GPU
SegNet (original implementation)	60.07%	-	-
ResNet50_SegNet (our baseline)	54.27%	1.92	19.57
ResNet50_SegNet+ FlowNet2S	47.65%	16.84	31.27

The inclusion of FlowNet2S, a rapid optical flow network, enhances the speed of our video segmentation network, surpassing traditional per-frame methods. FlowNet2S is specialized in generating flow data rather than conducting segmentation itself. By spatially warping the flow information from the optical flow network and integrating it with the latest segmented output from the segmentation network path, our system not only accelerates the segmentation process but also maintains the quality of the output.

Table 3. Gain in fps Using two Network Approaches

Inference on GPU	Segmentation network	(Segmentation network + FlowNet2S)	fps Gain
Our Implementation	19.57 (resnet50_segnet)	31.27	11.7
DVSNet*	5.6 (deepLab-Fast)	19.8	14.2

Our dual-network setup, combining a segmentation network with an optical flow network, leverages the latter's speed to compensate for the former's slower pace. This synergy has led to more significant gains in frames per second (fps) compared to DVSNet [9], as detailed in Table 3. Our system achieved an increase of 11.7 fps, while DVSNet registered a 14.2 fps improvement. Although our network's fps boost to 31.27 was less pronounced compared to DVSNet, the advantage is still notable. The extent of benefit observed is proportional to the segmentation network's speed; since DVSNet's DeepLab-Fast is slower than our Resnet50 Segnet, the gains are somewhat similar.

4.9 Outcome 3: Results collected at various threshold modes

Here we discuss experiment 3's outcomes. Table 4 illustrates MIoU and network speed at different CPU and GPU levels. Whenever the threshold was fixed to 90, which corresponds to slow mode, the segmentation method was selected the majority of the time, and the network achieved a MIoU of 53.69 percent and a framerate of 7.87 and 26.69 on CPU and GPU. Once the threshold was fixed to 80, the segmentation, as well as flow network paths, were selected evenly. The network obtained an MIoU of 47.65%, fps of 16.84, and 31.27 on CPU and GPU. If the threshold were fixed to 65, the flow network path was selected the majority of the time. This enhanced network performance to 19.09 frames per second on CPU and 32.94 frames per second on GPU, with an MIoU of 32.55 percent. Table 5 displays IoU by network speed mode. This table illustrates how a class's efficiency decreased as speed increased.

Table 4. The Segmentation Approach Works at Varying Speed Modes

		CPU	GPU

Speed mode	threshold 't'	speed(fps)	MIoU	speed(fps)	MIoU
Very slow mode	95	1.92	54.27%	19.57	54.27%
Slow mode	90	7.87	53.69%	26.69	53.69%
Balanced mode	80	16.84	47.65%	31.27	47.65%
Fast mode	65	19.09	32.55%	32.94	32.54%
Very fast mode	0	25.45	18.00%	35.88	18.00%

Table 5. IoU by Classwise in % for Different Speed Modes

Classes	**Class wise IoU on Slow mode 't' = 90**	**Class wise IoU on balance mode 't' = 80**	**Class wise IoU on fast mode 't'= 65**
Sky	86.28	83.54	67.63
Building	75.30	72.49	53.16
Pole	6.44	8.51	3.68
Road	88.7	85.61	76.25
Pavement	66.16	62.24	39.15
Tree	70.38	65.59	42.44
Sign symbol	30.95	25.04	12.79
Fence	29.30	25.63	12.25
Car	70.821	54.43	38.79
Pedestrian	28.89	18.59	5.35
Bicyclist	33.53	22.45	6.6

In fast mode, when fps climbed, MIoU decreased. When speed was reduced in slow mode, MIoU rose. Thus, speed and output quality are compromised. The balanced mode combines speed and quality. Real-time, users can adjust the ideal threshold based on interface suggestions. Providing speed and MIoU accuracy flexibility. After a certain point, speed saturates and doesn't grow. Table 4 shows that with a threshold of 0, the maximum GPU speed is 35.88 fps, just 3 fps greater than fast mode.

4.10 Outcome 4: Results of Depth Inference

Figure 9 depicts the distances between bicycles and cars on bounding boxes. The cycle rider is 6.68 metres away from the reference location. The distance between two target cars is 8m and 7.65m. The 7.65m car is getting closer, while the 8.00m car is going away.

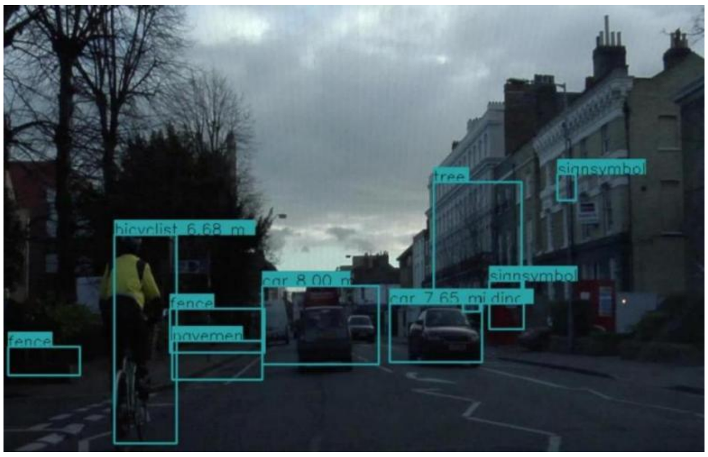

Figure 9. Illustrating Distance between Objects in streets

When moving on a curving route, distance calculations are also erroneous. This may be because the euclidean distance is estimated as a straight line.

5. Conclusion and Future Work

We have developed a video semantic segmentation network focused on achieving low latency. Our ResNet50 SegNet model demonstrated a Mean IoU (MIoU) of 54.27% and managed 1.92 fps on CPU and 19.57 fps on GPU. However, when FlowNet2S with spatial warping was incorporated, the MIoU slightly decreased to 47.65%, but there was a notable increase in processing speeds to 16.84 fps on CPU and 30.19 fps on GPU. This enhancement underscores the effectiveness of integrating an optical flow network to achieve both low latency and comparable segmentation quality. A crucial aspect of our approach was label mapping, enabling the adaptation of models from different datasets to our context. This strategy has allowed us to leverage well-established models on extensive datasets for our purposes. Adding the optical flow network in balanced mode led to an 11.7 fps gain, though this was slightly lower than the improvement observed with DVSNet. As we adjusted the network from slow to balanced mode, there was an increase in speed, accompanied by a 6% reduction in prediction quality for balanced mode and 15% for rapid mode. We also employed distance calculations between objects in a frame from a specific reference point (center of the x-axis), which is critical for assessing proximity to the vehicle and enhancing safety. Our findings suggest that encoder-decoder networks, such as U-Net and FCN with various encoder backbones, are effective in semantic segmentation with low latency, and alternative distance measurements could further improve depth inference, especially on curved roads.

References

[1] J. Shotton, M. Johnson, and R. Cipolla, "Semantic text on forests for image categorization and segmentation," in *26th IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, 2008, doi: 10.1109/CVPR.2008.4587503.

[2] C. Zhang, L. Wang, and R. Yang, "Semantic segmentation of urban scenes using dense depth maps," in *Lecture Notes in Computer Science* , 2010, doi: 10.1007/978-3-642- 15561-1_51.

[3] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *in Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2015, doi: 10.1109/CVPR.2015.7298965.

[4] L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Semantic image segmentation with deep convolutional nets and fully connected CRFs," *in 3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings,* 2015.

[5] R. Gadde, V. Jampani, and P. V. Gehler, "Semantic Video CNNs Through Representation Warping," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, doi: 10.1109/ICCV.2017.477.

[6] S. Jain, X. Wang, and J. E. Gonzalez, "Accel: A corrective fusion network for efficient semantic segmentation on video," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019, doi: 10.1109/CVPR.2019.00907.

[7] V. Nekrasov, H. Chen, C. Shen, and I. Reid, "Architecture Search of Dynamic Cells for Semantic Video Segmentation," *Computer Science*, 2020, doi: 10.1109/wacv45572.2020.9093531.

[8] Y. Li, J. Shi, and D. Lin, "Low-Latency Video Semantic Segmentation," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, doi: 10.1109/CVPR.2018.00628.

[9] Y. S. Xu, T. J. Fu, H. K. Yang, and C. Y. Lee, "Dynamic Video Segmentation Network," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,*2018*,* doi: 10.1109/CVPR.2018.00686.

[10]E. Shelhamer, K. Rakelly, J. Ho ff man, and T. Darrell, "Clockwork ConvNets for video semantic segmentation," in *Lecture Notes in Computer Science* , 2016, doi: 10.1007/978- 3-319-49409-8_69.

[11]X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei, "Deep feature flow for video recognition," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition,* 2017, doi: 10.1109/CVPR.2017.441.

[12]A. Garcia-Garcia, S. Orts-Escolano, S. Oprea, V. Villena-Martinez, P. Martinez-Gonzal ez, and J. Garcia-Rodriguez, "A survey on deep learning techniques for image and vi deo semantic segmentation," *Applied Soft Computing Journa*l, vol. 70, pp. 41-65, 2018, doi: https://doi.org/10.1016/j.asoc.2018.05.018.

[13]H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," in *Proceedings.30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017, doi: 10.1109/CVPR.2017.660.

[14]O. Ronneberger, P. Fischer, and T. Brox, "U-Net," *MICCAI2015*, 2015, doi: 10.1007/978-3-319-24574-4_28.

[15]V. Badrinarayanan, A. Kendall, and R. Cipolla, "SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation," *IEEE Trans. Pattern Anal. Mach. Intell*, 2017, doi: 10.1109/TPAMI.2016.2644615.

[16]L. C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "DeepLab: Se mantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and F ully Connected CRFs," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018, doi: 10.1109/T PAMI.2017.2699184.

[17]G. J. Brostow, J. Fauqueur and R. Cipolla, "Semantic object classes in video: A high-definition ground truth database," *Pattern Recognit. Lett.*, 2009, doi: 10.1016/j.patrec.20 08.04.005.

[18]M. Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, Bernt Schiele, "The Cityscapes Dataset," in *CVPR Workshop on The Future of Datasets in Vision,* 2015.

[19]Sunil Kumar K N and Dr. Shivashankar, "Compression of PPG Signal through Joint Technique of Auto-encoder and Feature Selection", *International Journal of Health Care Information Systems and Informatics-ESCI Indexed ACM-Digital Library*, vol. 16, no. 4, pp. 1-15, 2021.

[20]Sunil Kumar K N and Dr. Shivashankar, "Bio-signals Compression Using Auto Encoder", *Journal of Electrical and Computer Engineering (Q2 Indexed), Institute of Advanced Engineering and Science publishers,* vol. 11, no. 1, pp. 424, 2021.

[21]Sunil Kumar K N and Dr. Shivashankar, "Security Framework for Physiological Signals using Auto Encoder", *Journal of Advanced Research in Dynamical and Control Systems*–*(Q3 Indexed)*, vol. 12, pp. 583-592, 2020.

[22]K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, doi: 10.1109/CVPR.2016.90.

[23]A. Krizhevsky, I. Sutskever, and H. Geoffrey E., "Imagenet," *Adv. Neural Inf. Process. Syst. 25*, 2012, doi: 10.1109/5.726791.

[24]F. Ning, D. Delhomme, Y. LeCun, F. Piano, L. Bottou, and P. E. Barbano. "Toward automatic phenotyping of developing embryos from videos." *Image Processing, IEEE Transactions on*, vol. 14, no. 9, pp. 1360-1371, 2005.

[25]D. C. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber. "Deep neural networks segment neuronal membranes in electron microscopy images." In *NIPS*, pp. 2852-2860, 2012.

[26]C. Farabet, C. Couprie, L. Najman, and Y. LeCun. "Learning hierarchical features for scene labeling, " *IEEE Transactions on Pattern Analysis and Machine Intelligence,* vol. 35, no. 8, pp. 1915-1929, 2013.

[27]H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, doi: 10.1109/ICCV.2015.178.

[28]K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *3rd International Conference on Learning Representations*, 2015.

[29]E. J. Kirkland and E. J. Kirkland, "Bilinear Interpolation," in *Advanced Computing in Electron Microscopy,* 2010.

[30]X. He, R. S. Zemel, and M. Carreira-Perpindn, "Multiscale conditional random fields for image labeling," *in CVPR*, 2004.

[31] A. Quattoni, M. Collins, and T. Darrell, “Conditional random fields for object recognition,” in *Advances in Neural Information Processing Systems*, 2005.

[32] S. Savian, M. Elahi, and T. Tillo, “Optical Flow Estimation with Deep Learning, a Survey on Recent Advances,” *Pattern Recognition*, vol. 114, p. 107861, 2020.

[33] T. Brox and J. Malik, “Large displacement optical flow: Descriptor matching in variational motion estimation,” *IEEE Trans. Pattern Anal*, 2011, doi: 10.1109/TPAMI.2010.143.

[34] Y. Hu, R. Song, and Y. Li, “Efficient coarse-to-fine patch match for large displacement optical flow,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, doi: 10.1109/CVPR.2016.615.

[35] C. Bailer, B. Taetz, and D. Stricker, “Flow Fields: Dense Correspondence Fields for Highly Accurate Large Displacement Optical Flow Estimation,” *IEEE Trans. Pattern Anal. Mach. Intell*, 2019, doi: 10.1109/TPAMI.2018.2859970.

[36] Moritz Menze, Christian Heipke and Andreas Geiger, “Discrete optimization for optical flow,” In *German Conference on Pattern Recognition*, 2015, pp. 16-28.

[37] Q. Chen and V. Koltun, “Full flow: Optical flow estimation by global optimization over regular grids,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, doi: 10.1109/CVPR.2016.509.

[38] P. Weinzaepfel, J. Revaud, Z. Harchaoui, and C. Schmid, “DeepFlow: Large displacement optical flow with deep matching,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2013, doi: 10.1109/ICCV.2013.175.

[39] Jerome Revaud, Philippe Weinzaepfel, Zaid Harchaoui and Cordelia Schmid, “Epicflow: Edge-preserving interpolation of correspondences for optical flow,” In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 1164-1172.

[40] Y. Hu, Y. Li, and R. Song, “Robust interpolation of correspondences for large displacement optical flow,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017, doi: 10.1109/CVPR.2017.509.

[41] Dosovitskiy A, Fischer P, Ilg E, Häusser P, Hazirbas C, Golkov V, Smagt P V D, C remers D and Brox T, “FlowNet: Learning optical flow with convolutional networks,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, doi: 10.1109/ICCV.2015.316.

[42] E. Ilg, N. Mayer, T. Saikia, M. Keuper, A. Dosovitskiy, and T. Brox, “FlowNet 2.0: Evolution of optical flow estimation with deep networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017, doi: 10.1109/CVPR.2 017.179.

[43] A. Ranjan and M. J. Black, “Optical flow estimation using a spatial pyramid network,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition*, 2017, doi: 10.1109/CVPR.2017.291.

[44] R. Gadde, V. Jampani, and P. V. Gehler, “Semantic Video CNNs Through Representation Warping,” in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, doi: 10.1109/ICCV.2017.477.

[45] S. Jain, X. Wang, and J. E. Gonzalez, “Accel: A corrective fusion network for efficient semantic segmentation on video,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition,* 2019, doi: 10.1109/CVPR.2019.00907.

[46] Y. Li, J. Shi, and D. Lin, “Low-Latency Video Semantic Segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2018, doi: 10.1109/CVPR.2018.00628.

[47] H. Su, V. Jampani, D. Sun, O. Gallo, E. Learned-Miller, and J. Kautz, “Pixel-adaptive convolutional neural networks,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2019, doi: 10.1109/CVPR.2019.01142.

[48] A. Marcu, D. Costea, Vlad Licăreţ, and Marius Leordeanu, “Towards Automatic Annotation for Semantic Segmentation in Drone Videos.,” *arXiv (Cornell University)*, Oct. 2019.

[49] V. Badrinarayanan, F. Galasso, and R. Cipolla, “Label propagation in video sequences,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, doi: 10.1109/CVPR.2010.5540054.