FINAL YEAR PROJECT

Semantic Video Segmentation

INDIVIDUAL FINAL REPORT

by CAI JINYU
teamed with ZHANG QIPING

Supervised by Dr. Kenneth K.Y. Wong

Department of Computer Science

April, 2019
Abstract

Computer vision plays a major role in the field of artificial intelligence. As a preliminary step of many computer vision tasks, semantic segmentation attracts massive research attention. Current methods lack accuracy, temporal cohesion and computational efficiency, and thus are not sufficient for real time video analysis with high throughput. This project aims to improve the performance of aforementioned aspects by modifying and integrating existing methods. Our work includes exploring per-frame prediction networks and integrating with extra information like optical flow. A new metric for evaluating temporal coherence is defined. And our results show an increased temporal coherence with compromised accuracy.

List of Figures

1 Sample segmentation result visualization ........................................ 1
2 Example data in Cityscapes and sample prediction results .................. 6
3 GRU structure .............................................................................. 8
4 GRFP structure ........................................................................... 9
5 Accel structure ........................................................................... 10
6 Atrous convolution schemes .......................................................... 11
7 Atrous convolution feature comparison .......................................... 12
8 DeepLab Network structure ............................................................ 12
9 Segmentation result of Deeplab V3 ............................................... 13
10 Baseline segmentation result on Cityscapes ................................... 13
11 Baseline segmentation result on consecutive frames ....................... 14
12 PSPNet Structure ........................................................................ 15
13 Netwarp Overview ..................................................................... 16
14 Netwarp Module ......................................................................... 16
15 Netwarp Qualitative Results .......................................................... 16
16 Examples of Optical Flow ............................................................... 17
17 Two architecture of FlowNet .......................................................... 18
18 The architecture of FlowNet 2.0 ..................................................... 19
19 Examples of flow from different methods ....................................... 19
20 Log Loss function ....................................................................... 21
21 Training architecture with FPMR loss ........................................... 22
22 A sample coef function ................................................................ 23
23 The process of inferring ground truth ............................................. 24
24 Sample inferred ground truth of 18-th frame .................................. 25
25 Sample inferred ground truth of 12-th frame .................................. 26
26 Pair training process illustration .................................................... 27
27 Pair training sample result ............................................................. 28
28 Decision Network training methodology ....................................... 30
Abbreviations

GPU ............ Graphic Processing Unit
CPU ............ Central Processing Unit
DNN ............ Deep Neural Network
CNN ............ Convolutional Neural Network
FCN ............ Fully Convolutional Network
RGB ............ Red Green Blue
HOG ............ Histogram of Oriented Gradient
SVM ............ Support Vector Machine
RDF ............ Random Decision Fores
CDF ............ Conditional Random Field
FPMR ............ Flow Prediction Matching Rate
Acknowledgement

I would like to express my most profound gratitude to Dr. Kenneth K.Y. Wong, who proposed this topic and accepted my mate Qiping and me as supervisees, who supplied us with weekly progress feedback and inspiration on this project, who provided us with server for conducting various experiments. His dedication to computer vision research also inspires us on the journey of exploring frontiers of this field.

I also would like to say thanks to my teammate ZHANG Qiping, who worked delicately on this project and shared great insight whenever there is confusion or obstacle in this project.

I also owe gratitude to Dr. Y.K. Choi, who is the second examiner of our interim presentation, where she gave valuable advice to our project and raised questions which provoked our thinking.
Contents

Abstract ii
List of Figures ii
Abbreviations iii
Acknowledgement iv

1 Introduction 1
  1.1 Problem statement .................................................... 1
  1.2 Objectives .............................................................. 2

2 Background 3
  2.1 Theoretical background ................................................. 3
    2.1.1 Computer vision ................................................... 3
    2.1.2 Machine learning .................................................. 3
    2.1.3 Deep neural network .............................................. 3
  2.2 Technical background .................................................. 3
    2.2.1 Machine learning platform .................................... 3
    2.2.2 GPU computation ................................................ 4

3 Literature Review 5
  3.1 System inputs .......................................................... 5
  3.2 Feature extraction ..................................................... 5
  3.3 Modeling and classification .......................................... 5

4 Choice of dataset: Cityscapes 6
  4.1 Overview ............................................................... 6
  4.2 Types of annotation .................................................. 6
  4.3 Complexity ............................................................ 6
  4.4 Diversity ............................................................... 7
  4.5 Popularity ............................................................ 7

5 Current progress and result 8
  5.1 Vast Investigation on GRFP and Accel Network ...................... 8
  5.2 Atrous convolution, an inspiration from DeepLab .................. 11
    5.2.1 Methodology ..................................................... 11
    5.2.2 Result on Cityscapes .......................................... 11
  5.3 Experiments on PSPNet and Netwarp Approach ....................... 15
    5.3.1 Pyramid Scene Parsing Network (PSPNet) ..................... 15
    5.3.2 Netwarp Module integrated with PSPNet ..................... 15
  5.4 FPMR, a new metric for evaluating temporal coherence with optical flow ................................................. 17
    5.4.1 A review on optical flow ....................................... 17
    5.4.2 FlowNet and FlowNet 2.0 overview ............................ 18
    5.4.3 FPMR, a quantitative metric for evaluating temporal coherence ................................................. 20
  5.5 Training network with FPMR ........................................... 21
    5.5.1 FPMR as an additional loss ..................................... 21
    5.5.2 FPMR as a loss coefficient .................................... 23
    5.5.3 Data augmentation via FlowNet ................................ 24
    5.5.4 Pair training with augmented dataset ......................... 27

6 Difficulties and limitations 29
  6.1 Problem with limited GPU resource and computing platform .... 29
6.2 Limitation of frame-by-frame prediction

7 What's next? An unfinished system build

8 Conclusion

References
1 Introduction

Among all kinds of human perceptions, vision is the major source of information. Naturally, computer vision enjoys a great portion of research focus in the field of artificial intelligence. This project is concerned about video semantic segmentation, a preliminary step of computer vision analysis. In this section, we present the problem statement in detail and our objectives of this project.

1.1 Problem statement

As one of the most traditional and basic fields in computer vision research, semantic segmentation intuitively refers to the process of assigning a class label (e.g., road, tree, sky, pedestrian, car, ...) to each pixel of an image. By combining neighbouring pixels with the same label, a segmentation mask is formed (see Figure 1). Great importance is continuously attached to this area since it usually performs as a substantial preprocessing stage in many other computer vision tasks, such as scene parsing and understanding [31]. Generally, semantic segmentation consists of three basic steps: object detection, shape recognition and classification [31], each of which contains space of further efficiency improvements. However, when the problem domain generalizes to video instead of a single image, researchers are confronted with more challenges: in brief, as a concatenation of consecutive frames, video segmentation is a more difficult problem because consistency should be maintained between a certain number of neighborhood images, i.e. the neighboring frames should not contain greatly distinctive classification patterns with each other. This inherent characteristic has contributed to the obstacles in the design of semantic video segmentation algorithms.

Figure 1: Sample segmentation result visualization [31]

According to most reviews and articles, there are several common problems along with the rapid research advancement in semantic video segmentation. A typical problem that has existed since the involvement of machine learning is the lack of large, complete and representative datasets. Although this problem has been attenuated to some degree with the recent development of reference datasets, which has a set of standardized training and testing methods, deficiency remains in terms of their scale and abundance [31]. Yet these existing improvements have already been able to direct a large number of computer vision scientists into the field of semantic video segmentation. Meanwhile, another emerging problem is that, it is quite time-consuming to construct a dataset with pixel-level labeling and high accuracy (low mis-judgement rate by human labeler) simultaneously. This phenomenon inspires researchers to conceive semi-supervised and weakly supervised methods [31]. On the other hand, for some video-level labeling datasets, location detection accuracy has become a dominant problem, which consequently requires the application of other approaches to preserve system performance, leading to greater cost in system design and development for this kind of dataset. This trade-off prevents several graphical models like Markov
Random Field (requiring high labeling accuracy) from being widely used for datasets belonging to this category [31].

As a result of aforementioned constraints, even though it has been several years since semantic video segmentation began taking an important position in machine vision study, variance in the appearance of objects like “angle and direction, size and scale, blurring and reduced quality, camouflage of objects in the environment, objects overlap” [31, p. 2], etc. are still among the main difficulties in this area. Nevertheless, this problem also got solved to some certain extent with the development of deep neural networks (DNNs), which has helped boost system reliability to a great deal in recent research. Also, with the increase of computational power, which leads to more layers in current DNNs, a further enhance in segmentation accuracy could be achieved. With the approaches and tools developed in DNNs, convolutional neural networks (CNNs) accomplished superior performance over other methods in feature extraction, resulting in its important role in semantic video segmentation in view of present researchers. Several derivative models from CNNs, such as fully convolutional network (FCN) have also been introduced, to obtain a large map of the labels by tagging every small part of the image.

By now, although the problems described above have found several solutions that gained great success in semantic video segmentation, there is still a certain distance from this research area to complete maturity. A series of problems still remain in this field of study, for example, the difficulty in segmenting the blurred objects in the video due to the wrong focus of the camera, and the vulnerability encountered while trying to classify an object in a different point of view with that of the training dataset. Considering all these problems in semantic video segmentation, our project aims to compare and modify the previous state-of-art models and develop a new approach that could mitigate their deficiency and obtain a better overall performance.

1.2 Objectives

This project seeks to design an approach to address semantic video segmentation tasks. Ideally, this approach is time-efficient for high throughput videos while maintaining a competitive level of accuracy and temporal cohesion among neighboring frames.

The intermediate goal for this project, is to review and evaluate the state-of-art models and algorithms of semantic video segmentation. Particularly, we are interested in methods that are distinct in neural network design, system architecture and feature processing and claim to have compelling results on certain benchmark datasets. We will reproduce their methods and validate on the same benchmark datasets to see whether certain techniques they used are as critical as claimed. The purpose is to reveal advantages and deficiencies of the existing methods and thus pave way for future experiments to design our own approach.

The ultimate goal for this project, is to design a video semantic segmentation method, whereby the segmentation result is temporally coherent and the segmentation process is cost-efficient. More than the general requirements for spatial correctness and meaningfulness in single 2-D image segmentation tasks, a qualified and ideal method for semantic video segmentation will also avoid oscillating results among nearby frames. Contingent upon the similarity among nearby frames, this method is expected to significantly reduce the computation resource required.

The remaining of this report will provide the theoretical and technical background of this project, helping form a deeper understanding of following sections. And the choice of dataset, Cityscapes, will be justified. Next, current experiment progress will be introduced ,with findings and obstacles presented. The first experiment is aborted due to platform compatibility. And focus will be on the baseline result of the second experiment. Lastly, a future research plan will be discussed based on current status.
2 Background

In this section, we supplement background information regarding some theoretical and technical concepts. These concepts will be referenced in the literature review and current status sections.

2.1 Theoretical background

Three essential concepts regarding this project are introduced in this section: computer vision, machine learning, and deep neural network.

2.1.1 Computer vision

Computer vision consisting of multiple disciplinary fields, studies the way machines obtain high-level understanding from digital images and videos. Video sequences are a major source of data that computer vision researchers need to analyze. Due to this reason, as mentioned in section 1.1, semantic video segmentation naturally becomes an essential preprocessing stage in many computer vision tasks, to classify and identify the significant objects in continuous video frames.

2.1.2 Machine learning

Machine learning is a field that uses statistical techniques to enable computer systems to explore patterns from a large set of data without very explicit task-oriented programming. Our project can be viewed as a pattern finding problem where we let the computer to learn certain patterns that classify and segment some objects from the background. Therefore, machine learning becomes a powerful tool in semantic video segmentation. And since machine learning has the advantage of avoiding strictly static programming, it enables a more comprehensive pattern learning process that trains our model to cope with multiple scenarios.

2.1.3 Deep neural network

Deep neural network emerges from a subfield of machine learning, deep learning. As its name indicates, deep neural network has multiple layers that expands in varied depth. Layers are connected and each represents different level of abstraction of features, making it ideal for feature extraction. It is commonly used in computer vision tasks for its hierarchical feature representation.

2.2 Technical background

2.2.1 Machine learning platform

Machine learning involves considerable amount of computation and therefore it heavily relies on the frameworks that provides computational packages. Several popular frameworks are released by the world’s leading machine learning research teams. Caffe by Berkeley Vision and Learning Center, TensorFlow by Google Brain Team, and PyTorch by Facebook AI Team are among the most popular ones. These deep learning platforms enable fast, flexible manipulation on training process and efficient computation through a distributed training, and ecosystem of tools and libraries, making them ideal tools in this project. Because different platforms are written in different languages, they enjoy some comparative advantages against others. For example, Caffe in C++ is more efficient when doing computation, while PyTorch in Python is more flexible when modifying network architecture and training strategies. Thus, we chose the platform based on the different need of the training process.
2.2.2 GPU computation

Modern GPUs are extremely efficient when processing image for its special advantage of parallel computing. Since both machine learning and the high throughput nature of video sequences require significant computational resource, GPUs rather than CPUs are more suitable in this project.
3 Literature Review

Since the start of this project, substantial work is allocated for a broad literature review on related studies, and they are summarized in this section. According to [31], the latest research on semantic video segmentation mainly focuses on three aspects: (1) Input of Semantic Segmentation Systems; (2) Feature Extraction; (3) Modeling and Classification.

3.1 System inputs

Researchers use different system inputs for semantic video segmentation, [5, 4] make use of binary inputs in their models. On the other hand, systems presented in [22, 37] are established on basis of multi-class inputs. In this case, the quality and accuracy of training mainly depend on the number of existing categories provided in training dataset. Meanwhile, as most researchers have been applying common RGB videos for training [19, 22, 29], there are still a certain number of them that include geographical coordinates to obtain a higher segmentation precision [25, 1].

3.2 Feature extraction

To extract features from the input, [22, 35] applies the method called super-voxels, in which their algorithms firstly try to detect smallest component of a video able to be considered as a 3D structure of images, and then extract the defined features from these detected super-voxels. More traditionally, a majority of research works make use of fundamental hand-craft features, such as pixel color features of video frames [21, 25], histogram of oriented gradient (HOG) [19, 34], appearance-based features (e.g. texture features) [30, 20] and 3D optical flow features [25]. On the other hand, with the recent development of deep learning approaches, application of pre-trained models on CNNs has also become popular method for extraction of automatic features from input dataset [40, 15].

3.3 Modeling and classification

The extant approaches for constructing models and classification methods can mainly be divided into the following categories: a. Unsupervised Methods, such as clustering algorithms [23], graph-based algorithms [12, 38] and random walk algorithms [1, 3]; b. Support Vector Machine (SVM) [14, 20]; c. Random Decision Forest (RDF) [10, 30]; d. Markov Random Field (MRF) [32, 24]; e. Conditional Random Field (CRF) [6, 21]; f. Neural Networks, including traditional neural network [33, 28] and deep neural networks (DNNs, which is further composed of several generalized models like CNNs, RNNs and FCNs [13, 40, 15, 2, 29]). All the models and methods described above consist of numerous subdivisions, which is beyond the scope of literature review in this section.
4 Choice of dataset: Cityscapes

As emphasized in the background section, machine learning is essentially a pattern finding problem among a large volume dataset. Thus, it is of great importance to choose a highly qualified dataset with accurate human-labeled ground truth. In this section, we justify our choice of dataset, Cityscapes, an urban street scene dataset.

4.1 Overview

With the purpose of helping improve visual understanding of complex urban street scenes, Cityscapes, a benchmark suite and large-scale dataset, was introduced to train and test methods for semantic labeling[8]. Cityscapes employed different annotation types with complex object classes, covering diversified scenes and backgrounds, making it a natural choice for our project. Example data is presented in Figure 2, including the image with stereo depth maps, the annotated ground truth and three corresponding sample prediction results. Detailed features will be discussed in the following subsections.

![Figure 2: Example data in Cityscapes and three corresponding prediction results. From left to right: image with stereo depth maps, annotation (ground truth), sample prediction results by three researchers [8]](image)

4.2 Types of annotation

Cityscapes uses three types of annotation, semantic annotation, instance-wise annotation and dense pixel annotation. These types of annotation provide different levels of semantic information in one scene and can be utilized according to various purpose of tasks.

4.3 Complexity

Cityscapes targets at 30 classes of objects, e.g. flat, human, vehicle and etc.. Its class definition is complex and complete, covering major objects in urban street scenes. In this project, we also aim to tackle typical street scene objects so that it can be applied to real-life technology like autonomous driving.
4.4 Diversity

Cityscapes is comprised of street scene video clips from diversified streets from 50 cities during the
daytime of several months(spring, summer, fall) under good or medium weather conditions. This wide
range of contents provides the diversity that is highly desired by any machine learning approach. In
addition, frames in this dataset are manually selected to contain large number of dynamic objects,
varying scene layout and varying background[8], giving higher chance of a more robust model.

4.5 Popularity

Cityscapes is popular among researchers. Almost every published approach regarding semantic seg-
mentation utilizes Cityscapes as benchmark for performance comparison. Metrics and results by other
researchers are also available on its official website. In this project, with the objective of improving
model performance, Cityscapes naturally becomes our choice of dataset.
5 Current progress and result

Following the justification for the choice of dataset, we present our current progress and discuss current result of this project in this section. Two experiments have been carried out so far. We temporarily aborted the first one due to some technique problems and we obtained baseline result from the second experiment.

5.1 Vast Investigation on GRFP and Accel Network

In the first semester of study, we initially made a vast investigation of latest research on semantic video segmentation. We found that although these works vary a lot by their network structure, their main orientation to solve the video segmentation problem could be briefly classified into 2 categories: One focuses on taking the consecutive frames and their inherent similarity as an extra information and do the segmentation based on this prior, so that the output labelled video could achieve greater temporal consistency as a result. The other emphasizes on the discrimination of “key frames” for the purpose of acceleration, so that the network could perform video segmentation with greater efficiency and faster inference time.

Among the works concentrating on the involvement of time-domain information to boost video segmentation accuracy, Nilsson’s work [26] is a great representation containing unique ideas. As is shown in Figure 3, this model introduces a minimal structure called “Gated Recurrent Unit” (GRU) to integrate the temporal transition with the spatial information returned by a CNN to get a more precise segmentation for each frame. With this micro-structure GRU, the frame segmentation of a video can propagate along the time flow. And furthermore, another key design in the model also brings about great difference, as is shown in Figure 4. Instead of simply tracing in the same direction as the video flow, this model converges from both sides of the frame to the middle, thus integrating both “the past” and “the future” information to tackle the time-consistency problem. As a result, video segmentation with this neural network is able to maintain high stability of labelling pattern between consecutive frames. This is also verified by experiment on Cityscapes dataset, where the proposed model with GRU can obtain a class and category IoU of 80.6 and 90.8 respectively, outperforming all comparison baselines included in the paper to a great extent.

![Figure 3: Overview of Spatio-Temporal Transformer Gated Recurrent Unit (STGRU)](image)

For recent works focusing on acceleration based on inference of unimportant frames, Samvit proposed a
network named Accel to solve this problem [18]. The major innovation of their work is the distinction of “key frames”, as is displayed in Figure 5. In Accel, researchers only select a small proportion of frames as key frames, which would experience the same segmentation process in conventional approach, with time-consuming feature network and task subnetwork. A key frame’s feature representation extracted through the full network is cached for the reference of its nearest frames. On the remaining frames, a lightweight feature and task network is applied, and aligned with the cached feature of key frames to retrieve a cheaper segmentation. In the last step, Accel cascades the two score maps and generates a final segmentation result. Consequently, as is shown by experiments done on Cityscapes, the algorithm in Accel dramatically saves computational power by decreasing inference time per frame, correcting warping-related error that compounds on datasets with complex dynamics.
Figure 5: Accel combines a reference branch computing score map on high-detail features from last keyframe, and an update branch to correct the prediction from less detailed features.
5.2 Atrous convolution, an inspiration from DeepLab

While methods mentioned in the previous section are ideal for combating temporal coherence, ground truth labels of neighbor frames are required. Since in CityScapes, only one key frame out of every 30 frames is labeled, we decided to start from segmentation by frame independently. This section introduces the feature extraction methodology we draw inspiration from DeepLab[7]. Current baseline results will be presented with visualization.

5.2.1 Methodology

We choose the method proposed in [7] to extract dense feature with atrous convolution. The scheme of atrous convolution is illustrated in Figure 6. Different from traditional convolution, atrous convolution achieves large receptive field without repeatedly applying max-pooling, which may lose spatial information. Figure 7 illustrates the feature activation using two equivalent convolutions, one being traditional and the other being atrous. It also allows us to modify the density level at which we obtain the feature responses in fully convolutional networks[7].

Atrous convolution layer can be considered as a general convolution layer in that it can be viewed as inserting zeroes between two filter values in the traditional convolutional layer. And traditional convolutional layer is a special case when the rate is equal to 1 in atrous convolutional layer. See Figure 6 for illustration. Thus, the implementation is relatively easy by modifying the traditional convolutional layer.

Sample segmentation result have also been released from the original paper on other datasets. The visualization result (see Figure 9) is smooth and accurate.

With easy implementation and relatively accurate results, we implemented this method in our project on the machine learning platform PyTorch, aforementioned in the background section. A detailed network structure is shown in Figure 8.

5.2.2 Result on Cityscapes

In this project, we trained the proposed model in [7] on Cityscapes dataset. Utilizing one GPU with batch size of 2, after 50 epochs of training, we obtained our first baseline model. Performance of the baseline model was tested on the validation set of Cityscapes. Results are visualized in Figure 10. Our prediction is accurate on distinguishing different instances of objects while some fine details are missed in our prediction compared with the ground truth. For example, in the first rows of prediction, the shape of the human on the bicycle is not as smooth as that of the ground truth, partially due to the
Figure 7: A feature comparison between using atrous convolution and traditional convolution with stride down-sampling and up-sampling.

Figure 8: The encoder-decoder structure for making segmentation prediction using atrous convolution.

... weak lighting condition. And in the second row of prediction, the rods at distant are vague and not as straight as the ground truth.

On the aspect of temporal coherence, which indicates the stability of prediction on consecutive frames of a video clip, we tested it on a frame-by-frame manner. Figure 11 visualizes the per frame prediction of 8 consecutive frames from a video clip. The in-car camera moves forward, giving rise to the relative displacement of the background and other objects. Our baseline model made coherent prediction with no major disappearance or turbulence.
Figure 9: Claimed sample prediction results on Cityscapes proposed by [7]

Figure 10: Prediction results on Cityscapes with our baseline model. From left to right: Original image of street scene, ground truth, our prediction.
Figure 11: Prediction of 8 consecutive frames extracted from a testing video clip of Cityscapes. In time sequence from left to right and top to bottom, the result shows temporal coherence.
5.3 Experiments on PSPNet and Netwarp Approach

From the result of DeepLab, we found that the result wasn’t stable enough along the time domain, while many minimal structures in the surrounding continuously fluctuate across neighborhood frames. To deal with this problem, we further explored and trained another advanced network called PSPNet, which gained far more temporally consistent segmentation result on Cityscapes dataset. Furthermore, we also investigated another network structure called Netwarp, which could theoretically be applied to every CNN including PSPNet for video segmentation tasks. This section briefly introduces the key structures and innovations included in these two approaches, with qualitative and quantitative results of PSPNet and Netwarp-PSPNet displayed to elaborate their efficiency in semantic video segmentation.

5.3.1 Pyramid Scene Parsing Network (PSPNet)

Pyramid Scene Parsing Network (PSPNet) was proposed in 2017, which is originally intended for image segmentation tasks. Although this neural network still executes segmentation on individual frames, surprisingly, the model is still able to achieve a segmentation result with high temporal consistency [39]. And this outcome could be well explained through an observation on its network structure, which is demonstrated in Figure 12. The model first uses CNN to get the feature map of the last convolutional layer (b), then a pyramid parsing module is applied to harvest different sub-region representations, followed by up-sampling and concatenation layers to form the final feature representation, which carries both local and global context information in (c). Finally, the representation is fed into a convolution layer to get the final per-pixel prediction (d). Due to this pyramid parsing module, the network could extract features with different level of scales in the images, so that the labelling of these structures will not differ too much along the time flow. As a result, our segmented video was compared with the labelling of ground truth and the segmentation result of the previous DeepLab version, which has shown visibly more stable outcomes along the time domain than the outcome of DeepLab model whose resulting screenshots were displayed in the previous section.

5.3.2 Netwarp Module integrated with PSPNet

Following PSPNet, another approach called Netwarp was proposed to further boost the performance of the original network. And this method is currently the model with best performance using info from video sequence on Cityscapes dataset [11]. The illustration of Figure 13 and Figure 14 depicts the use of NetWarp modules in three different layers of an image CNN. The video CNN is applied in an online fashion, looking back only one frame. The CNN filter activations for the current frame are modified by the corresponding representations of the previous frame via Netwarp modules. For the structure of this Netwarp module, firstly, optical flow $F_t$ is computed between two video frames at time steps $t$ and $t_1$. Then the Netwarp module transforms the flow ($F_t$) with few convolutional layers, warps the activations of the previous frame and and combines the warped representations with those of the current frame $z_t$. 

Figure 12: Overview of the structure of PSPNet
The resulting representation is then passed onto the remaining CNN layers for semantic segmentation. As a result, Netwarp integrated to Pspnet, called Netwarp-Pspnet, can perform particularly well on the stability of small classes such as poles, traffic lights, and this advantage could also be proved by the IoU given by the experiments in the original paper [11], as is shown in Figure 15. Currently we are trying to implement Netwarp module into Pspnet under Pytorch implementation, and after we finish this testing, we will also consider integrating Netwarp approach into other high-performance image segmentation CNNs, to see if we can get even better result.

Figure 13: Video CNN with NetWarp modules

Figure 14: Illustration of computations in a NetWarp module

Figure 15: Netwarp-PSPNet Results on the Cityscapes test dataset
5.4 FPMR, a new metric for evaluating temporal coherence with optical flow

In the first half of this project, we tackled one of our objectives, improving temporal coherence, only by adopting approaches of per-frame manner. And unfortunately, the only way to evaluate the quality of prediction, in terms of temporal coherence, is qualitative observation, i.e. stacking per-frame predictions into a video and manually evaluate the coherence level by human eyes. This is undoubtedly inaccurate, biased and varies between different people and visualization machines.

In this section, we define a quantitative metric for evaluating temporal coherence, Flow Prediction Matching Rate (FPMR), with the help of optical flow. A review of optical flow is laid out firstly, followed by an overview of the network FlowNet and FlowNet 2.0 which we adopted to estimate optical flow. Then, we formally define Flow Prediction Matching Rate and argue for the reliability of this definition.

5.4.1 A review on optical flow

Optical flow was first introduced by work of Horn and Schunck in [16]. It measures the brightness movement’s velocities which are resulted from the movement of objects or viewers. Thus, it can provide information on the spatial movement of them. An example of optical flow is shown in Figure 16 for elaboration.

Figure 16: Two examples of optical flow from the Flying Chairs dataset. The first three columns are generated image pair and color coded flow field, the last three columns are augmented image pair and corresponding color coded flow field respectively[9]

Although optical flow is not necessarily related to the movement of object, for example, a rotating sphere in 3D space gives zero optical flow [16], in the case of our project, optical flow can estimate a displacement of objects, for example, cars and pedestrians, between two frames captured in the setting of a moving vehicle. And thus, it is potentially an important source of information that can be captured to evaluate and improve temporal coherence.

Recently, researchers focus on large displacements. Matching combiners are also integrated into the variational approach [9]. However, traditional methods solely rely on CPUs for computation, failing to utilize the efficient computational advantages of GPUs. With the introduce of CNNs, Fischer et al. [9] proposed to model estimating optical flow as a supervised learning problem. This innovation made it both efficient and accurate, an ideal quality that we can adopt as part of our project.
5.4.2 FlowNet and FlowNet 2.0 overview

The original FlowNet proposed by Fischer et al. [9] designed an end-to-end flow prediction architectures with deep convolutional neural networks (DCNNs), with two versions being FlowNetSimple and FlowNetCorr. Architectures of the two networks are shown in Figure 17. FlowNetSimple, by its name, simply stacked the two input pictures and put them through the convolutional layers to produce the estimated prediction. FlowNetCorr splitted the two input images into two identical streams for feature map extraction. Then the combining and comparison are carried out in later stages. Both achieved decent error rate on public datasets but not exceeding state-of-art which was still held by traditional methods.

Later, Ilg et al. in [17] improved FlowNet to its 2.0 version. This improvement achieved four times higher accuracy than the original version. FlowNet 2.0 enhanced the performance by exploring the influence of scheduling the presentation of training datasets. Certain schedule of training datasets led to improved results compared to using one dataset in isolation. Secondly, FlowNet 2.0 used a combination of networks to predict the optical flow. The scheme is shown in Figure 18. Several layered networks are responsible for large displacement and one for small displacement. And there is a fusion of the two types of displacement. Together, this whole scheme of networks contributes to a better prediction, but surely at higher computational cost. A visualized comparison between FlowNet 2.0 and other methods is elaborated in Figure 19, together with comparison on computational speed. One can easily observe that the improvement is significant. And thus, in our future work, we adopted FlowNet 2.0 to estimate optical flow.
Figure 18: FlowNet 2.0 adopts a combination of networks to estimate a fusion-ed flow[17]

Figure 19: FlowNet 2.0 performs better in terms of finer detail but the computational speed is slower.[17]
5.4.3 FPMR, a quantitative metric for evaluating temporal coherence

With the background on optical flow, we are ready to introduce the quantitative metric for evaluating temporal coherence.

By definition, temporal coherence measures a model’s stability in predicting two neighboring frames. In a consecutive video sequence, two neighboring frames only present subtle difference. In our dataset, neighboring frames only have 0.06s time gap (each 1.8s video clip contains 30 frames). Thus, ideally, frame-by-frame predictions of the two frames should not vary too much. Disregarding prediction accuracy (true prediction label for each pixel), if one pixel moves from previous frame to current frame at a different coordinate, the two prediction result for the two pixel should remain the same.

Carried with this consensus, we are ready to present our definition of FPMR. FPMR is the ratio of matching pixels out of all valid pixels. A pixel matching means when a previous-frame pixel is warped by optical flow to current frame, its predicted label remains the same. For instance, \( p(x, y) \) is the coordinate of a pixel in frame \( t-1 \), by warping optical flow, we have the corresponding coordinate of the pixel \( p'(x', y') \) in frame \( t \), where \( p' = W(p) \) and \( W \) is the warping function. The pixel is counted as a matching if and only if \( p' \) is still in the range of the image and the label prediction of \( p \) and \( p' \) remains the same. Thus, the definition is,

\[
FPMR = \frac{\sum_{p \in P} M(p, W(p))}{|P|}
\]

where \( P \) is the set of pixels where for every \( p \) in \( P \), \( W(p) \) is in the range of the picture, \(|P|\) is the cardinality of the set, and \( M \) is a function that outputs 1 if the two label at the given coordinates are the same and 0 otherwise.

With this definition, one can observe that, if the optical flow network is almost perfect, i.e. one can almost accurately estimate the movement of a pixel in previous frame, FPMR will clearly indicates the level of temporal coherence of the segmentation network. Note that the range of FPMR is from 0 to 1, where 1 indicates a perfect temporal coherence and 0 indicates no temporal coherence.

After defining FPMR, we experimented on the per-frame DeepLab segmentation network (as in section 5.2.2) and per-frame PSPNet segmentation network (as in section 5.3). We used a video clip of 200 frames in the test set of CityScapes, and used FlowNet 2.0 (as in section 5.4.2) to estimate the optical flow between neighboring frames, i.e. in total 199 different optical flow estimations. In DeepLab segmentation, the average FPMR is 90.5% while in PSPNet segmentation network, the average FPMR is 94.3%.

<table>
<thead>
<tr>
<th>Network</th>
<th>FPMR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab</td>
<td>90.5%</td>
</tr>
<tr>
<td>PSPNet</td>
<td>94.3%</td>
</tr>
</tbody>
</table>

The quantitative results are in accordance with our expectation as watching the prediction video, we can easily see that the PSPNet one is smoother and more steady than the DeepLab one. This qualitative observation adds more credibility to our definition of metric for temporal coherence.
5.5  Training network with FPMR

With the introduction of FPMR, a metric of temporal coherence is set up for examining the quality of a segmentation network. However, when training the segmentation networks, the objective has been limited to output as accurate as ground truth. And since these networks are per-frame nature, they only aim to improve accuracy but not coherence. Thus, a natural next step is to integrate this newly defined metric to our training process.

In this section, we introduce four methods of integrating FPMR into our training process, namely FPMR as an individual loss, FPMR as a loss coefficient, Data augmentation via FlowNet and Pair training with augmented dataset. These methods are not necessarily successful in our experiment and we still present them all for discussion.

5.5.1  FPMR as an additional loss

This first approach that comes to our mind is to convert FPMR as a loss. Since we want our network to produce predictions with FPMR tending to 100%, this loss function should tend to 0 when FPMR is 100% and tend to infinity when FPMR is 0%. This immediately matches the log loss function with true label of 1.

\[
\text{Loss}(\text{FPMR}) = -\log(\text{FPMR})
\]

Figure 20 presents a visualization for this loss function. It is easily noticeable that the magnitude of the loss function with input FPMR of 90% is only \(-\log(0.9) = 0.0458\), which is not comparable to the other part of loss generated from ground truth MSE loss. Typical ground truth loss of an image is at the magnitude of 10,000 to 100,000. Thus, a weight is needed to make FPMR loss not trivial to ground truth loss. We started the experiment with a weight of 1,000,000.

\[
\text{TotalLoss} = \text{GroundTruthLoss} + 1000000 \times \text{FPMRLoss}
\]

![Log Loss when true label = 1](image)

Figure 20: Log Loss function when the true label is 1.

The renovated training network structure is as shown in Figure 21. Instead of one input image in previous training process, we also included a previous frame. Both frames will go through the segmentation network for predictions. And they are also inputted to estimate the optical flow. After that, the key frame prediction will be compared with ground truth, outputting a ground truth loss as usual. The
optical flow and the two predictions together will output the FPMR of this pair of predictions, with the log loss function following. Then, combining (simple adding) the two loss items gives the total loss. Then the loss is back-propagated to update the parameters of the segmentation network.

![Figure 21: The training architecture with the FPMR loss.](image)

The experimental result is unsatisfactory. We tried on different scales of weights multiplying to the FPMR loss from 200,000 to 2,000,000. Only one of them did not diverge from the training process. While even so, the only not-diverging one presented meaningless prediction. Its loss did not significantly descend and the inception result is disordered.

<table>
<thead>
<tr>
<th>Network</th>
<th>FPMR weight</th>
<th>Training result</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab</td>
<td>2,000,000</td>
<td>Loss diverges</td>
</tr>
<tr>
<td>DeepLab</td>
<td>1,000,000</td>
<td>Loss diverges</td>
</tr>
<tr>
<td>DeepLab</td>
<td>500,000</td>
<td>Trivial descent, meaningless prediction</td>
</tr>
<tr>
<td>DeepLab</td>
<td>200,000</td>
<td>Loss diverges</td>
</tr>
</tbody>
</table>

A possible explanation for such behavior is that, log loss function originally takes an input of prediction probability nature. It is a function that transforms the confidence level of prediction to a loss. In our case, the FPMR is not a prediction probability nor a confidence indicator. We simply borrowed it because it satisfies our needed loss function’s extreme points’ expectation. The other interval’s behaviour remains unknown and thus may not suit for the training. In addition, the scale of the weight parameter is not rationally defined. We did it by try-and-error.

Thus, we improved the training by making the weight learn-able. We set an initial value of 1,000,000 and let the network figure out which weight works best. Unfortunately, this training was also not successful at all.
5.5.2 FPMR as a loss coefficient

Although we failed to convert FPMR as a loss itself, we still hope to integrate this metric to the loss layer and back-propagate to update the parameters. This time, we abandon the idea to convert FPMR to a loss, but utilize it as a moderator of the ground truth loss. We transform FPMR to a coefficient, and multiply it to the ground truth loss.

\[ \text{TotalLoss} = \text{coef}(\text{FPMR}) \cdot \text{GroundTruthLoss} \]

The way the coef function works is that, when FPMR tends to 100\%, \( \text{coef}(\text{FPMR}) \) tends to a number less than 1, for example 0.5. When FPMR tens to 0\%, \( \text{coef}(\text{FPMR}) \) tends to a number greater than 1, for example 2.0. When FPMR equals an average temporal coherence like 90\%, the \( \text{coef}(\text{FPMR}) \) should make no changes to the ground truth loss, i.e. it equals 1.0. The coef function can be linear (see Figure 22) or non-linear. For simplicity, we experimented the linear version with \( \text{coef}(0) = 2.0, \text{coef}(0.9) = 1.0 \) and \( \text{coef}(1) = 0.5 \).

\[ \text{Figure 22: A sample coef function with} \quad \text{coef}(0) = 2.0, \text{coef}(0.9) = 1.0 \quad \text{and} \quad \text{coef}(1) = 0.5 \]

Result of this version of training made no better result than treating FPMR as a separate loss layer. Firstly, we doubt that the coef function is not well defined. The special point values are made up by ourselves and linear function may not work well on these segmentation networks. Thus, we improved it again with learn-able point values. We initiated the function with \( \text{coef}(0) = 2.0, \text{coef}(0.9) = 1.0 \) and \( \text{coef}(1) = 0.5 \). But these values are learn-able parameters and we want to let the network decide which values to adopt. Unexpectedly, the network made the parameter \( \text{coef}(1) = 0 \), which means for any different input images, the prediction result remains the same. In this way, FPMR will always be 100\% and the coefficient is 0, contributing to zero loss. However, this totally evaded our original intention. And thus we stopped at this stage of experiment.
5.5.3 Data augmentation via FlowNet

After the unsatisfactory results in section 5.5.1 and 5.5.2, we figured that treating FPMR as a source of loss or coefficient of loss may not necessarily help improve the temporal coherence. However, to improve temporal coherence, it is of no doubt that we need more source of information. In our current training dataset, i.e. CityScapes dataset, only one key frame out of 30 frames of a video clip is labeled with ground truth. Each video clip is 1.8 second long with 30 frames, and the 20-th frame is labeled with ground truth. Thus, the images we used in our training data are independent of each other. Moreover, since we trained the data in a randomly shuffled manner, temporal connection within batches are negligible. This led us to the idea of creating inferred ground truth using optical flow.

The process is simple, for each frame with ground truth, i.e. the 20-th frame, also called key frame, we estimate the optical flow between it and a previous frame. And this optical flow is warped to the ground truth of the 20-th frame, generating an inferred ground truth for the previous frame. A work flow is presented in Figure 23.

![FlowNet Process Diagram](image)

Figure 23: A sample process for inferring the 18-th frame’s ground truth from the 20-th frame

Due to the constraint on the space for storing the data, we were unable to produce inferred ground truth for all other frames. Thus we decided on producing for 2 and 8 frames earlier than the 20-th frame, i.e. we produce the 12-th and 18-th frame’s inferred ground truth. In our experiment, the difference between 19-th and 20-th are too subtle to generate enough optical movement. Thus, we further moved to the 18-th frame. And we also want to explore the relationship of frames with a larger time interval. Since the video clip is 0.06s per frame, we decided to take 8 frames, i.e. around half second before the 20-th frame, resulting at the 12-th frame.
We present some sample inferred ground truth in Figure 24 and Figure 25. Inferred ground truths are noised and of much lower quality than the human-labeled ground truth. It is acceptable as we have no control on the optical flow network, which is not at the scope of this project. Another observation is that, larger time interval difference creates lower quality of inferred ground truth. It is clearly shown in the figures that the inferred ground truth of 12-th frame is of lower quality than that of the 18-th frame.

With these supplementary ground truth, we can double the size of our training data. Keeping the training policy unchanged, we had the following results.

<table>
<thead>
<tr>
<th>Network</th>
<th>Dataset</th>
<th>mIoU</th>
<th>Loss</th>
<th>FPMR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab</td>
<td>CityScapes</td>
<td>0.5568</td>
<td>135,205</td>
<td>90.50%</td>
</tr>
<tr>
<td>DeepLab</td>
<td>CityScapes + 18-th inferred GT</td>
<td>0.5358</td>
<td>149,603</td>
<td>92.45%</td>
</tr>
<tr>
<td>DeepLab</td>
<td>CityScapes + 12-th inferred GT</td>
<td>0.4821</td>
<td>158,861</td>
<td>92.61%</td>
</tr>
</tbody>
</table>

We obtained two more models of DeepLab, which achieved a significant 2% higher FPMR than the original version at the cost of significant drop on the mIoU. The increase in higher FPMR can be accounted for by the use of additional inferred ground truth, which was produced with the help of FlowNet 2.0. And our metric FPMR is also defined with the help of FlowNet 2.0. Thus, we draw a conclusion that such way of augmenting dataset truly improves our model at the right direction. On the other hand, there is a noticeable drop on the accuracy metric mIoU. From the Figure 24 and Figure 25 we can see that the inferred ground truth is much less accurate than the real ground truth. And our evaluation dataset does not contain inferred ground truth. That is why the drop in mIoU is so significant. However, we suspect that, if an improved version of optical flow estimation is provided and the inferred ground truth look more accurate than the current one, we might be able to improve FPMR without compromising mIoU. But again, optical flow estimation is not in the scope of our project and thus we only adopt existed methods for now.

Figure 24: A sample ground truth inferred for the 18-th frame from the 20-th frame. The upper two are from the 20-th and lower two are the 18-th
Figure 25: A sample ground truth inferred for the 12-th frame from the 20-th frame. The upper two are from the 20-th and lower two are the 12-th
5.5.4 Pair training with augmented dataset

In the previous section, we achieved better temporal coherence with the augmented dataset. The sole difference was the training dataset. For every batch, the training policy was to randomly select one picture and use it as a training data. We note that although we supplied inferred ground truth to each key frame, because of the randomly shuffle nature, the pair of training data do not necessarily show up together. We suspect that this weakens the potential to strengthen the temporal coherence.

In the next step, we modified our training input structure. Each time, when a 20-th frame was randomly selected, we also input the corresponding 18-th (or 12-th) frame into the segmentation network. Thus, for each batch, we accumulate two loss as a total loss, which is used for back-propagation. The architecture of the training process is illustrated in Figure 26.

![Figure 26: A pair (18-th and 20-th frame) training process architecture, losses are accumulated for back-propagation](image)

To examine this training procedure, we carried out experiment on DeepLab again. The results are shown in the table below. We also included previous individual training result for reference.

<table>
<thead>
<tr>
<th>Network</th>
<th>Dataset</th>
<th>mIoU</th>
<th>Loss</th>
<th>FPMR</th>
</tr>
</thead>
<tbody>
<tr>
<td>DeepLab</td>
<td>CityScapes</td>
<td>0.5568</td>
<td>135,205</td>
<td>90.50%</td>
</tr>
<tr>
<td>DeepLab</td>
<td>CityScapes + 18-th inferred GT</td>
<td>0.5358</td>
<td>149,603</td>
<td>92.45%</td>
</tr>
<tr>
<td>DeepLab</td>
<td>CityScapes + 12-th inferred GT</td>
<td>0.4821</td>
<td>158,861</td>
<td>92.61%</td>
</tr>
<tr>
<td>DeepLab Pair Training</td>
<td>CityScapes + 18-th inferred GT</td>
<td>0.5030</td>
<td>156,792</td>
<td>92.88%</td>
</tr>
<tr>
<td>DeepLab Pair Training</td>
<td>CityScapes + 12-th inferred GT</td>
<td>0.4521</td>
<td>209,365</td>
<td>93.01%</td>
</tr>
</tbody>
</table>

The results indicate that, with pair training, FPMR is further improved, but not significantly, while mIoU continues to drop. In both 18-th and 12-th augmented dataset, we observed slightly increased FPMRs but sharply dropped mIoUs. When concatenating predictions into videos, although the FPMR increased quantitatively, we cannot observe obvious improvement on the temporal coherence with human
eyes. The is mainly because of the accuracy of prediction dropped, leading to noised prediction which affects the perception for coherence.

We selected some sample predictions by the aforementioned models. They are listed in Figure 27.

Figure 27: Two pair (18-th and 20-th frame) trained models’ sample prediction result
6 Difficulties and limitations

In this section, difficulties and limitations will be identified. As the project proceeds, we had problems on using the training tools and we are also aware of the limitations of our current approach. Details are presented in the following subsections.

6.1 Problem with limited GPU resource and computing platform

Though GPUs are fast at doing graphic computation as mentioned in the background section, it still requires time when dealing with large datasets and large models. [27] waited for 6 days training a model on a single Nvidia GTX Titan X GPU with 12G memory. More experiments use multiple GPUs for one model due to the memory constraint. However, this project shares only four GPUs with another three FYP teams. At most times, there will not be enough GPUs for experiments that require multiple GPUs. Thus, our experiments will mainly rely on single GPU trained models.

Training with one GPU made it hard for us to rebuild previous work with as good quality as the author claims. One example is that when we try to train the network of DeepLab with the same training policy and dataset, our achieved best mIoU is about 0.2 lower than the claimed one. Besides accuracy down dropping, when we apply FlowNet 2.0 in our project, we were only able to compute each frame at the cost of 0.86 second on average. It is not just slower than claimed, but even slower than applying segmentation network to the frame, which takes 0.4 second on average.

Another potential reason that we failed to replicate other work may be the computing platform. We mainly used PyTorch in this project, which is associated with other dependencies. These dependencies update frequently and we were not able to fully replicate the environment the original author adopted (some disclose environment configuration while some not). Thus, the results vary from authors' claim.

6.2 Limitation of frame-by-frame prediction

Though Figure 6 visualizes a relatively coherent prediction on the test dataset based on our baseline model, it is computationally inefficient to make prediction independently on each frame. Although we made attempts to improve the temporal coherence, computational-wise the network is not improved. Given the high throughput nature of videos, such method cost substantial computational resource and may not suit for real-time analysis. Thus, applications like autonomous driving may find it hard to utilize such techniques.

Realizing the computational inefficiency, we made effort to reduce the amount of time needed to predict a sequence of video clip. Originally we thought about only predict a key frame’s label using frame-by-frame segmentation network and using optical flow to propagate the prediction to nearby frames. This method failed to reduce any time needed as estimating optical flow takes even more time in our platform and cannot compete with frame-by-frame prediction in accuracy.
7 What’s next? An unfinished system build

In previous sections, we explored methods to improve segmentation accuracy and temporal coherence. But system-wise, we made little effort to build a whole pipeline that can produce consecutive video prediction. One simple idea, as discussed in section 6.2 is to only predict the key frame and propagate the label with optical flow. When doing the experiment, we found it hard to decide how many frames after the key frame will the next key frame arrive. We set it to be 2 frames and 8 frames for experiments. Results vary and are unrelated to the number of frames we set. Thus, setting a fixed number of frames between two key frames may not be an ideal solution.

To think in another perspective, this problem is equivalent to deciding whether a frame is a key frame. If it is a key frame, then we put it in the segmentation network. If it is not a key frame, we estimate optical flow and propagate the prediction from the key frame. In [36], Xu et al. proposed a dynamic video segmentation architecture. In the architecture, they trained a Decision Network to decide whether a frame is a key frame or not. The Network is trained with ground truth obtained from optical flow. They call the output of the Decision Network confidence score, which in nature is similar to our definition of FPMR. The Decision Network’s training methodology is shown in Figure 28.

![Figure 28: The methodology for training the decision network [36]](image)

This architecture is a great motivation to our project. If more time were given, we would be inspired to integrate this to our project and potentially build a whole pipeline of prediction.
8 Conclusion

This report describes experiment progress, result and analysis of the final year project of semantic video segmentation. Clarifications on this topic were made first by demonstrating a detailed problem statement, objectives of this project, and theoretical and technical concepts related to this project. Related studies are summarized into three aspects: input of semantic segmentation systems, feature extraction and modeling and Classification. Both traditional methods and newly developed methods like machine learning are found in these proposed methods. CityScapes dataset is chosen in this project for its data complexity, diversity and popularity among researchers. The work done includes an investigation on GRFP and Accel network, which utilizes temporal information between frames. Feature extraction by Atrous convolution, proposed by [7] showed satisfactory baseline result and PSPNet showed better result in qualitative evaluation. And an new metric, FPMR, for evaluating temporal coherence is introduced and utilized for training and evaluation purpose. Using FPMR as a loss or loss moderator show no satisfactory result. Using optical flow for data augmentation improves temporal coherence but compromises accuracy. Lastly, a summary of difficulties we identified are laid out and the unfinished system build is suggested for future interest.
References


