Video Scene Parsing with Predictive Feature Learning: — Supplementary Material —

Xiaojie Jin[1] Xin Li[2] Huaxin Xiao[2] Xiaohui Shen[3] Zhe Lin[3] Jimei Yang[3]
Yunpeng Chen[2] Jian Dong[5] Luoqi Liu[4] Zequn Jie[4] Jiashi Feng[2] Shuicheng Yan[5,2]
[1]NUS Graduate School for Integrative Science and Engineering (NGS), NUS
[2]Department of ECE, NUS [3]Adobe Research [4]Tencent AI Lab [5]360 AI Institute

Abstract

In this supplementary material, we provide more implementation details including the architectures and training settings of baseline models and PEARL. We also present the experimental results and analysis of PEARL on Camvid dataset, as well as more qualitative evaluations of PEARL.

1. Implementation Details

Since the class distribution is extremely unbalanced in video scene parsing, we increase the weight of rare classes during training, similar to [3, 6, 15]. In particular, we adopt the re-weighting strategy in [15]. The weight for the class y is set as $\omega_y = 2^{\lceil \log 10(\eta / f_y) \rceil}$ where f_y is the frequency of class y and η is a dataset-dependent scalar which is defined using the $85\%/15\%$ frequent/rare classes rule.

All of our experiments are carried out on NVIDIA Titan X and Tesla M40 GPUs using Caffe library.

1.1. Network Architectures

To demonstrate that PEARL can be applied with advanced deep architectures, we implement PEARL and baseline models using two state-of-the-art deep architectures: *i.e.* VGG16 and Res101 and compare their performance. For fair comparison, both the frame parsing network and the predictive learning network in PEARL share the same architecture as baseline models except for the input/output layers in the predictive learning network (which takes multiple frames as inputs and outputs RGB frames instead of parsing maps). In the following, we first introduce the architecture details of baseline models (i.e., the following VGG16-baseline and Res101-baseline) and then the differences between PEARL and these baselines.

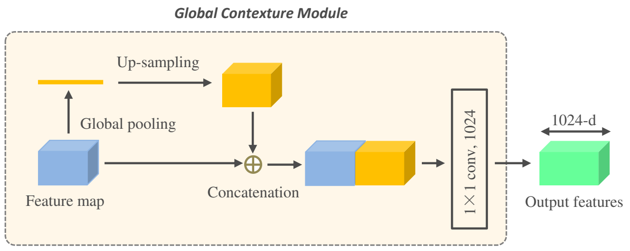


Figure 1: Architecture of the global contexture module which is applied to encode the image global context information as suggested in ParseNet [12]. The output feature map of `fc7`/`conv5_3` layer in VGG16/Res101 architectures in our experiments is passed through such global contextual module to produce a global context augmented feature map by global average pooling, up-sampling and concatenation with `fc7`/`conv3_3` output feature map. A 1×1 convolutional layer is then applied to the concatenated feature map to produce the output features with 1,024 channels.

- **VGG16-baseline** The VGG16-baseline is built upon DeepLab [2] with two modifications. First, to further enhance model's ability for video scene parsing, we add three deconvolutional layers (each followed by ReLU) to up-sample the `fc7` output features of DeepLab. The three deconvolutional layers consist of 4×4 convolutional kernels with striding of size 2 and padding of size 1. The number of kernels are 256, 128 and 64 respectively. Besides, following ParseNet [12], we use the global contexture module for `fc7` features to enhance the model's capability of capturing global context information. As shown in Figure 1, the global contexture module transforms the input feature map to a 1024-channel feature map. In experiments, we find such a module improves the parsing performance of the baseline model as it enlarges the model's receptive field size and utilizes the global information to distinguish local confusing pixels.
- **Res101-baseline** The architecture of our Res101-baseline is illustrated in Figure 2. It is modified from the original Res101 [5] by adapting it to a fully convolutional network, following [14]. Specifically, we replace the average pooling layer and the 1,000-way classification layer with a fully convolutional layer (denoted as $\texttt{conv5\_3}_{\text{cls}}$ in Figure 2) to produce dense parsing maps.

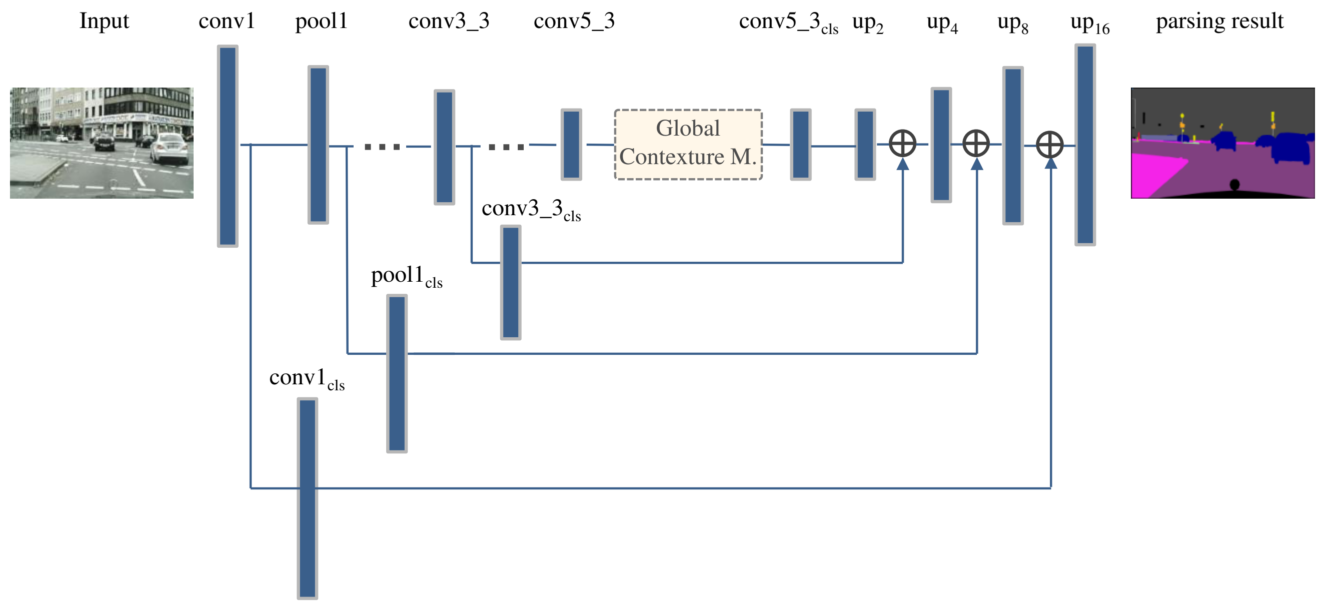


Figure 2: Architecture of Res101-baseline used in our experiments. Built on original Res101 [5], a global contexture module (ref. to Figure 1) is employed to produce the feature map augmented with global context information, which is then fed into a convolutional layer (`conv5_3`$_{cls}$) to produce dense parsing map. The subscript "cls" means the corresponding layer is a convolutional layer which has the same number of filters as the categorical classes, *e.g.* the `conv5_3`$_{cls}$ has 19 filters for Cityscapes dataset. Following FCN [14], skip connections from `conv1`, `pool1`, `conv3_3` are constructed to utilize high frequency information in bottom layers. The `up`$_n$ is an up-sampling layer, of which the output feature map is n times the spatial size of that of `conv5_3`$_{cls}$. The symbol $\oplus$ represents the operation of "summation".

Also, we modify `conv5_1`, `conv5_2` and `conv5_3` to be dilated convolutional layers by setting their dilation size to be 2 to enlarge the size of receptive fields. As a result, the output feature map of `conv5_3` has a stride of 16. Following FCN [14], we utilize high-frequency features learned in bottom layers by adding skip connections from `conv1`, `pool1`, `conv3_3` to corresponding up-sampling to produce parsing maps with the same size as input frames. Similar to VGG16-baseline, we use the global contexture module (ref. to Figure 1) for `conv5_3` features.

In PEARL, the output features of the global contextural module in the predictive learning network are concatenated with those in the frame parsing network. Besides, there are following two differences between the predictive learning network and the baseline model:

- **Bottom Convolutional Layer** Since the predictive learning network takes multiple video frames as input, we adapt the first convolutional layer to a group convolutional layer [7] where the group number is equal to the number of input frames (4 in our experiments). In this way, no extra parameters is added and we can have fair comparison with baseline models.
- **Output Layer** Since in phase I (unsupervised predictive learning) the predictive learning network predicts RGB frames instead of parsing maps, we replace the output layer in baseline model with one having 3 filters. In phase II, the output layers remain the same as baseline models.

Finally, GoogLeNet employed as the discriminator in phase I is built upon the vanilla GoogLeNet [16] by replacing its last fully connected layer with one having one hidden unit.

1.2. The Architecture of Transform Layer

In our experiments, a "bottleneck" residual block proposed in [5] is used as the transform layer whose architecture is illustrated in Figure 3.

1.3. Training Settings

The training settings of PEARL are listed in Table 1. For fair comparison, the training settings of baseline models are the same as those of phase II in PEARL.

2. Experimental Results and Analysis on Camvid

We further investigate the effectiveness of PEARL on Camvid. Its result and the best results ever reported on this dataset are listed in Table 2. Following [6, 15], loss re-weighting is used on this dataset. One can observe that

Table 1: Training settings of two phases in PEARL when adopting VGG16 and Res101 architectures. "LR" stands for "learning rate". In all experiments, the weight decay and momentum are set to be 0.0001 and 0.9 respectively. Following Deeplab [1], the "poly" learning rate policy is used in all training phases where the value of power equals to 0.9.

Training phase	Batch size	Crop size	Overall epochs	LR-initial	LR-policy
Adopting VGG16 Architecture					
Phase I	8	800	60	1e-6	Poly
Phase II	4	800	30	1e-7	Poly
Adopting Res101 Architecture					
Phase I	8	800	60	1e-3	Poly
Phase II	4	960	30	1e-4	Poly

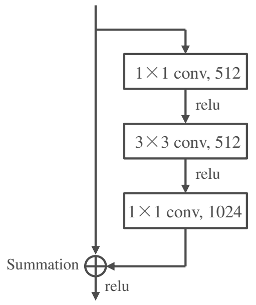


Figure 3: Architecture of the residual block [5] used as the transform layer in PEARL. In the residual block, the shortcut connection performs identity mapping, the output of which is added to the output of the stacked convolutional layers.

PEARL performs the best among all competing methods — improving the PA/CA of the baseline model (Res101-basline) by 1.7%/2.4% respectively, once again demonstrating its strong capability of improving video scene parsing performance. Notably, compared to the optical flow based methods [9] and [13] which utilize CRF to model temporal information, PEARL shows large advantages in performance, verifying its superiority in learning temporal representations for video scene parsing.

Table 2: Comparison with the state-of-the-art on CamVid. Res101 architecture is used in PEARL.

Methods	PA(%)	CA(%)
Res101-baseline (ours)	92.7	80.8
Ladicky *et al.*(*ECCV-10*) [8]	83.8	62.5
SuperParsing(*ECCV-10*) [17]	83.9	62.5
DAG-RNN (*CVPR-16*) [15]	91.6	78.1
MPF-RNN (*AAAI-17*) [6]	92.8	82.3
Liu *et al.* (*ECCV-15*) [13]	82.5	62.5
RTDF (*ECCV-16*) [9]	89.9	80.5
PEARL (ours)	**94.4**	**83.2**

3. Qualitative Evaluation of PEARL

3.1. More Frame Prediction Results from Phase I

Please refer to Figure 4;

3.2. More Video Scene Parsing Results of PEARL

Please refer to Figure 5. Notably, in the challenging predictive parsing task (predicting the parsing map of not observed future frame), PEARL is able to predict temporally smooth and motion aware parsing maps (*e.g.* the *pedestrian*/*car* in Seq.2/Seq.5) by using **only** the preceding RGB frames. This demonstrates PEARL's strong capability of learning about complex video scene dynamics. Benefited from the temporal representations and image local representations learned in PEARL, PEARL is able to produce more accurate and temporally consistent parsing maps than baseline models (as shown in the bottom row in each sequence).

References

[1] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016. 3

[2] 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 *ICLR*, 2015. 1

[3] C. Farabet, C. Couprie, L. Najman, and Y. LeCun. Learning hierarchical features for scene labeling. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 35(8):1915–1929, 2013. 1

[4] G. Ghiasi and C. C. Fowlkes. Laplacian reconstruction and refinement for semantic segmentation. In *ECCV*, 2016.

[5] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015. 1, 2, 3

[6] X. Jin, Y. Chen, J. Feng, Z. Jie, and S. Yan. Multi-path feedback recurrent neural network for scene parsing. In *AAAI*, 2017. 1, 2, 3

[7] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 2

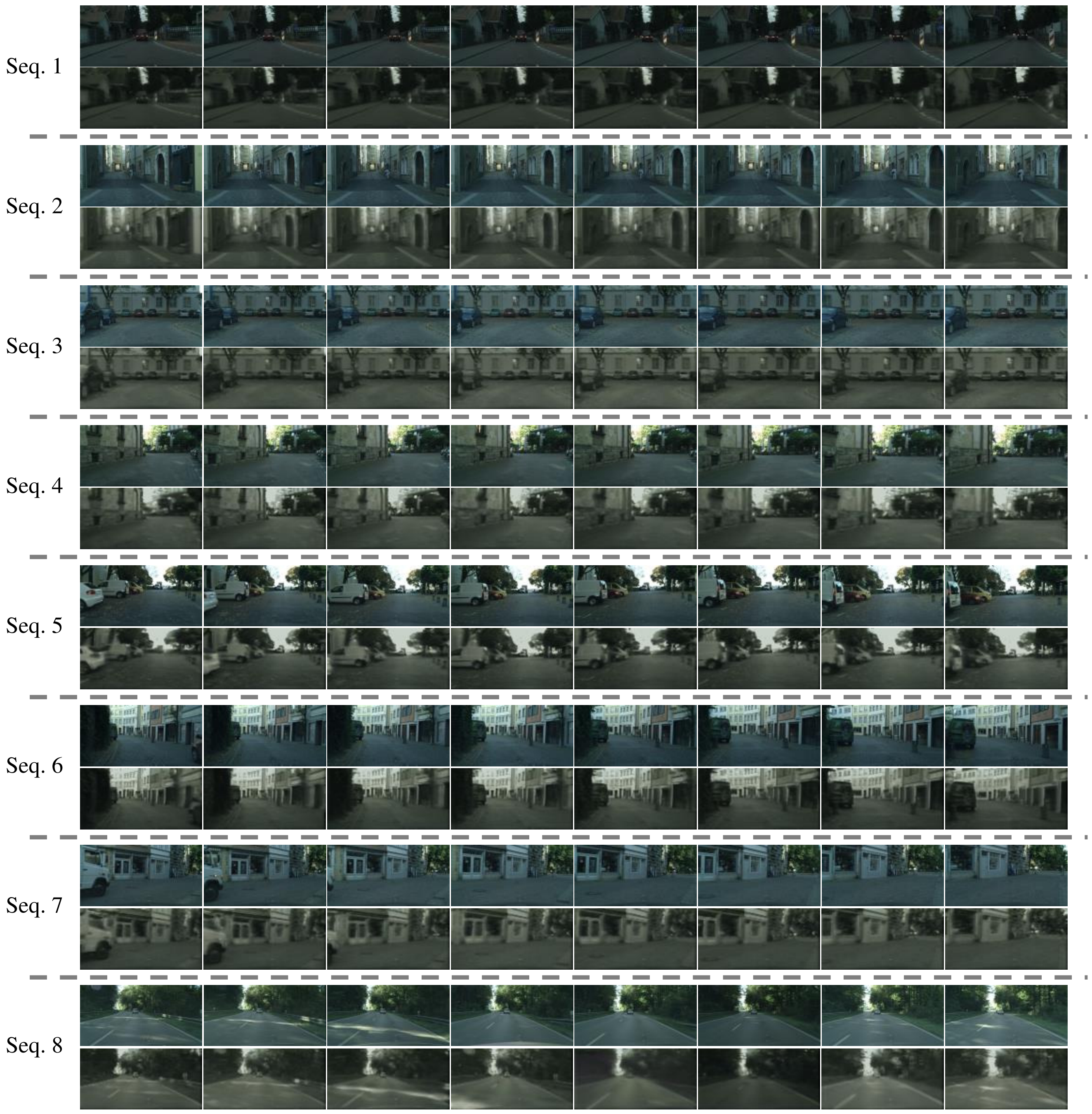
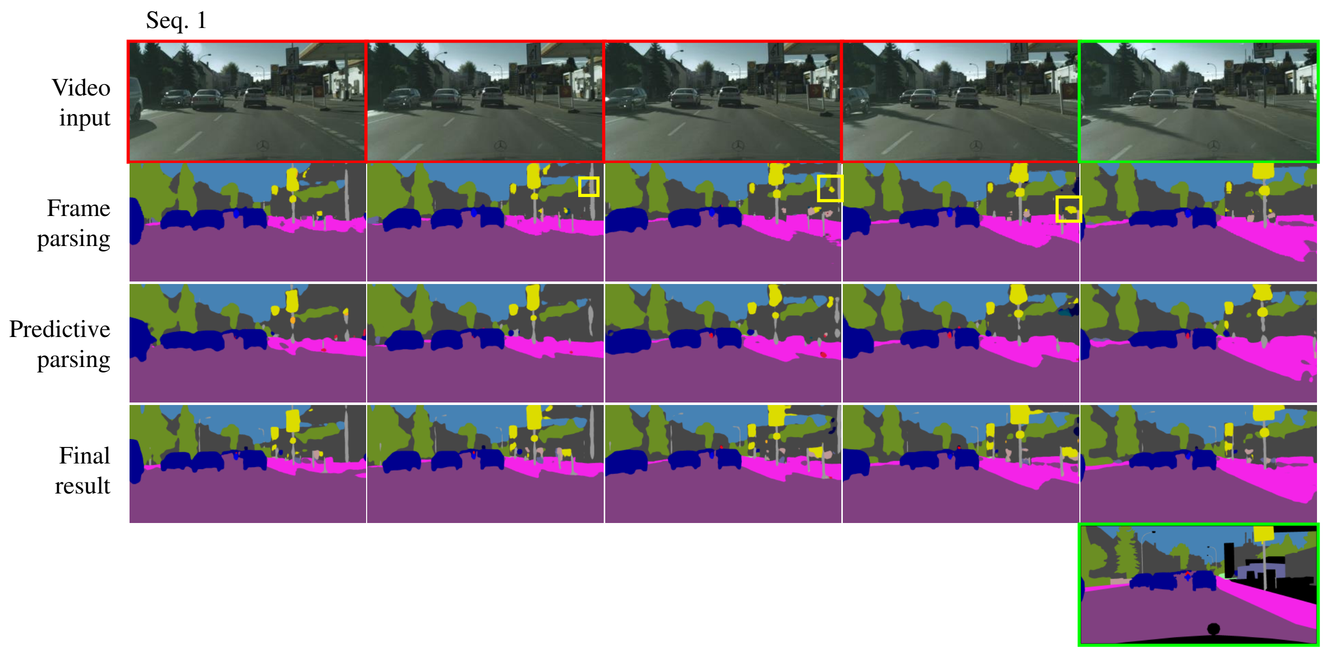


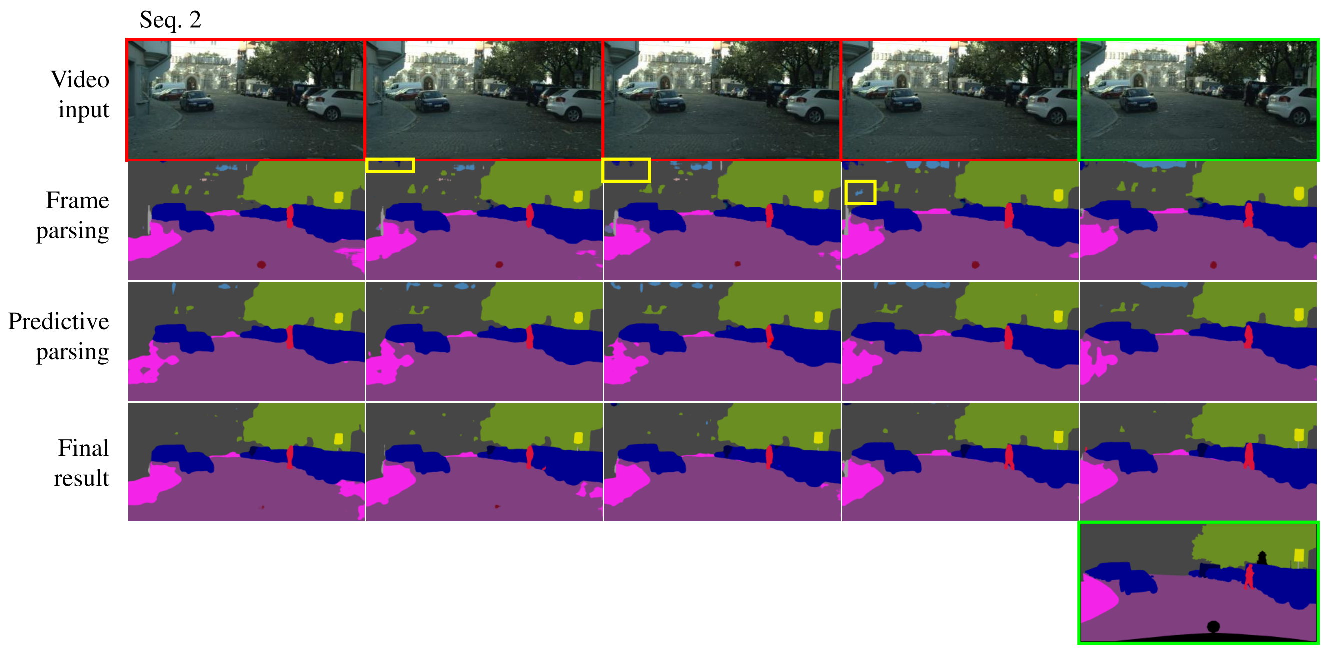
Figure 4: Eight sequences of videos in Cityscapes *val* set with corresponding frame prediction results produced by the predictive learning network in the phase of unsupervised predictive learning. For each sequence, the upper row contains eight contiguous ground truth frames and the bottom row contains corresponding frame predictions. It is observed that the predictive learning network is able to model the structures of objects and stuff as well as the motion information of moving objects in videos. Note that **the inputs for predicting each frame are its preceding RGB frames**, which are not fully presented for brevity. Best viewed in color and zoomed-in pdf.

[8] L. Ladickỳ, P. Sturgess, K. Alahari, C. Russell, and P. H. Torr. What, where and how many? combining object detectors and crfs. In *ECCV*. 2010. 3

[9] P. Lei and S. Todorovic. Recurrent temporal deep field for semantic video labeling. In *ECCV*, 2016. 3

[10] G. Lin, A. Milan, C. Shen, and I. D. Reid. Refinenet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, 2017.





[11] G. Lin, C. Shen, A. v. d. Hengel, and I. Reid. Exploring context with deep structured models for semantic segmentation. *arXiv preprint arXiv:1603.03183*, 2016.

[12] W. Liu, A. Rabinovich, and A. C. Berg. Parsenet: Looking wider to see better. *arXiv preprint arXiv:1506.04579*, 2015. 1

[13] Z. Liu, X. Li, P. Luo, C.-C. Loy, and X. Tang. Semantic image segmentation via deep parsing network. In *ECCV*, 2015. 3

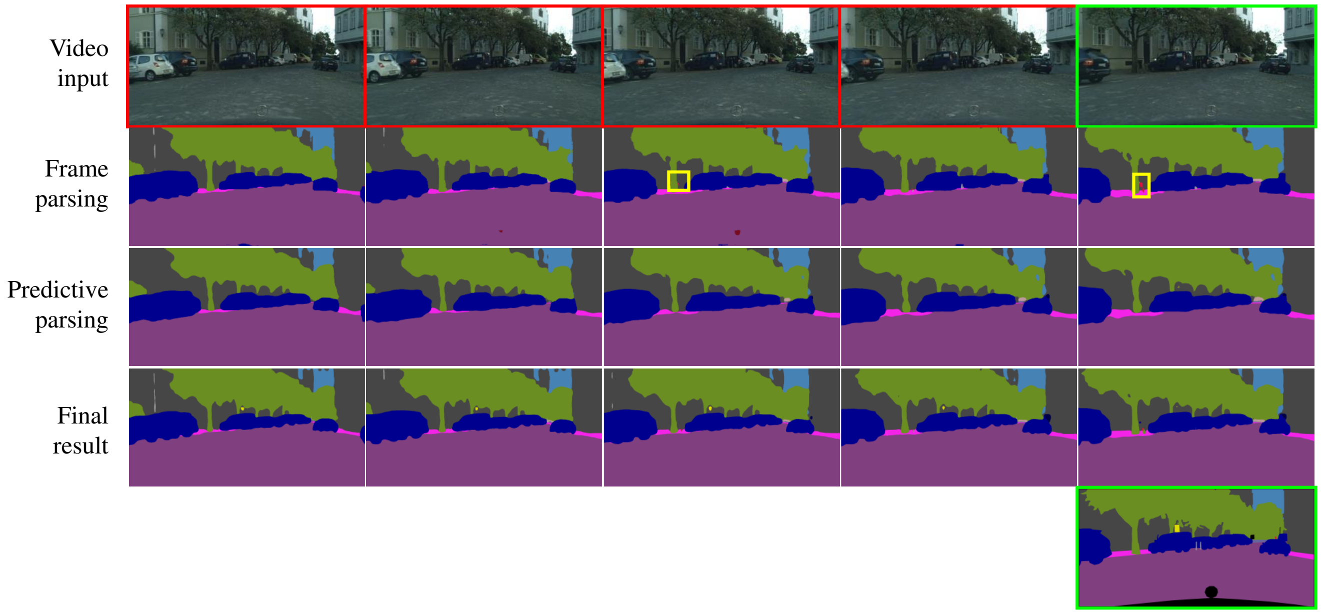
[14] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 2

[15] B. Shuai, Z. Zuo, G. Wang, and B. Wang. Dag-recurrent neural networks for scene labeling. *arXiv preprint arXiv:1509.00552*, 2015. 1, 2, 3

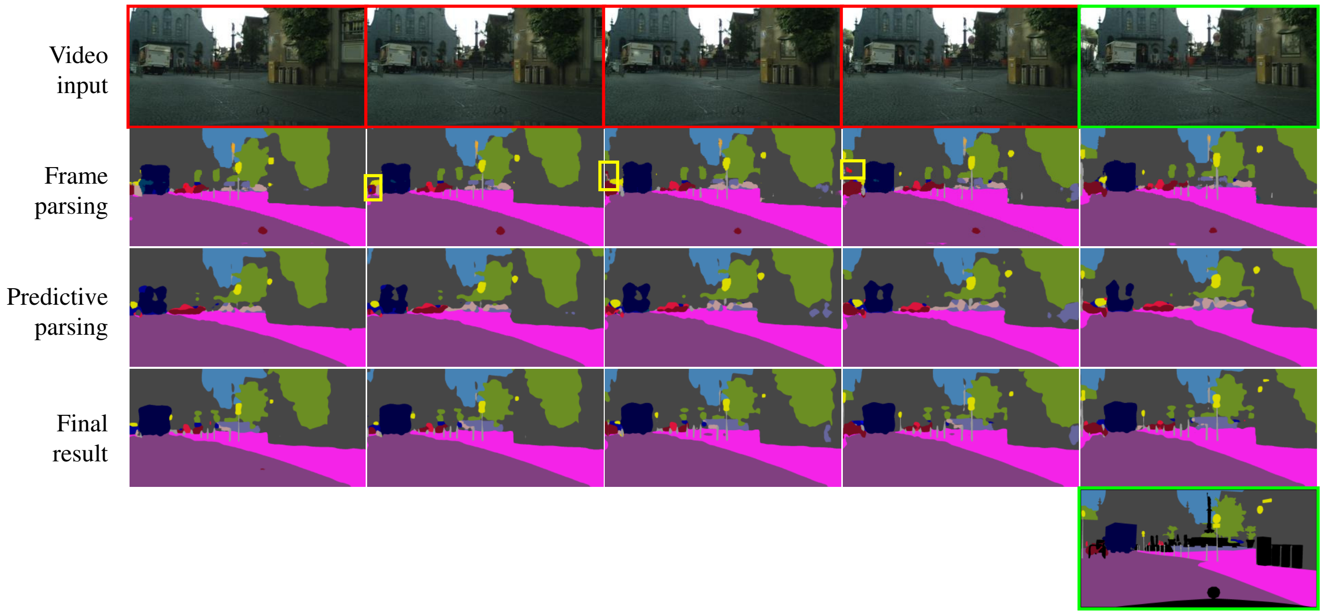
[16] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv:1409.4842*, 2014. 2

[17] J. Tighe and S. Lazebnik. Superparsing: scalable nonparametric image parsing with superpixels. In *ECCV*. 2010. 3

Seq. 3



Seq. 4



[18] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *ICLR*, 2016.

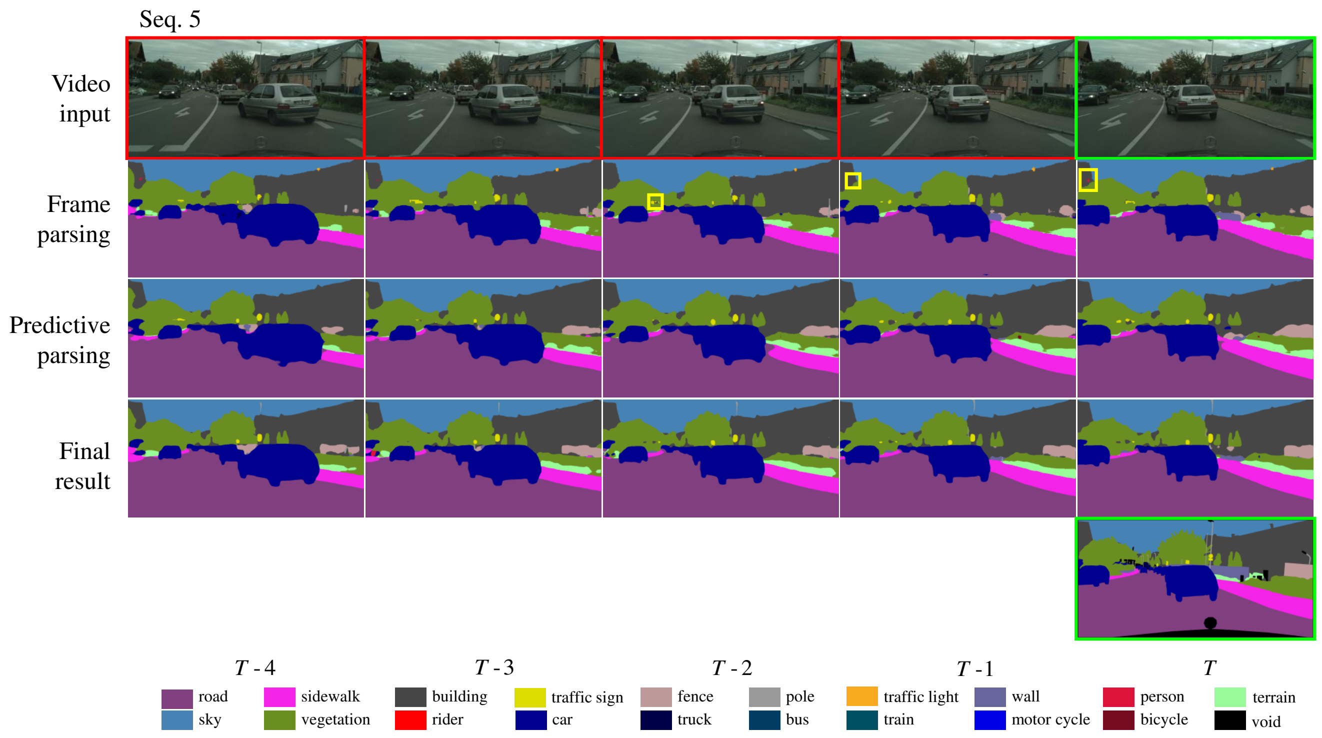


Figure 5: Examples of parsing results of PEARL on Cityscape *val* set. **Top row**: a five-frame sequence. The frame which has ground-truth annotations and its preceding four frames are with green and red boundaries respectively. **Second row**: frame parsing results produced by the VGG16-baseline model which takes a single frame as input. Since it cannot model temporal context, the baseline model produces parsing results with undesired inconsistency across frames as in yellow boxes. **Third row**: predictive parsing results output by PEARL in phase II (predictive learning for video scene parsing). The inconsistent parsing regions in the second row are classified consistently across frames. Note for producing the predictive parsing result for each frame, **the inputs are its preceding RGB frames** which are not fully presented for brevity. **Fourth row**: the final parsing maps produced by PEARL with better accuracy and temporal consistency due to combining the advantages of conventional frame parsing model (the second row) and predictive parsing (the third row). **Bottom row**: the ground truth label map (with green boundary) for the frame T.