Recurrent Flow-Guided Semantic Forecasting

Adam M. Terwilliger, Garrick Brazil, Xiaoming Liu
Department of Computer Science and Engineering
Michigan State University
{adamtwig, brazilga, liuxm}@msu.edu

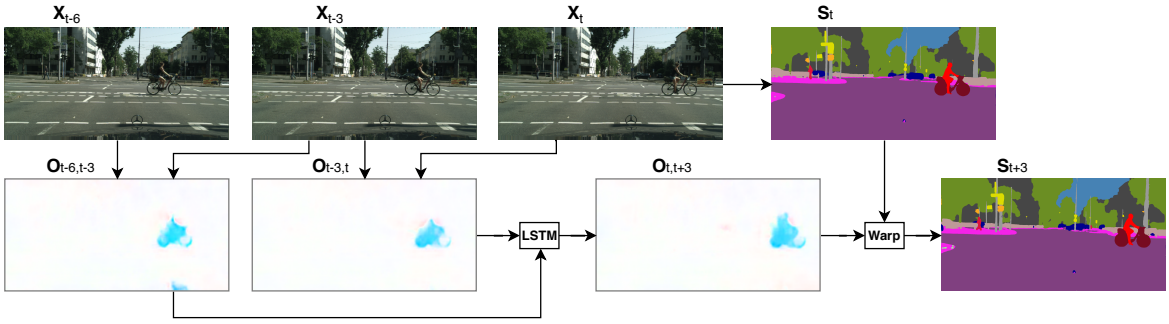

Figure 1: Our proposed approach aggregates past optical flow features using a convolutional LSTM to predict future optical flow, which is used by an learnable warp layer to produce future segmentation.

Abstract

Understanding the world around us and making decisions about the future is a critical component to human intelligence. As autonomous systems continue to develop, their ability to reason about the future will be the key to their success. Semantic anticipation is a relatively under-explored area for which autonomous vehicles could take advantage of (e.g., forecasting pedestrian trajectories). Motivated by the need for real-time prediction in autonomous systems, we propose to decompose the challenging semantic forecasting task into two subtasks: current frame segmentation and future optical flow prediction. Through this decomposition, we built an efficient, effective, low overhead model with three main components: flow prediction network, feature-flow aggregation LSTM, and end-to-end learnable warp layer. Our proposed method achieves state-of-the-art accuracy on short-term and moving objects semantic forecasting while simultaneously reducing model parameters by up to 95% *and increasing efficiency by greater than* $40x$.

1. Introduction

Reasoning about the future is a crucial element to the deployment of real-world computer vision systems. Specifically, garnering semantics about the future of an urban scene is essential for the success of autonomous driving. Future prediction tasks are often framed through video prediction (i.e., given n past RGB frames of a video, infer about m future frames). There is extensive prior work in directly modeling future RGB frames [10, 11, 13, 25, 31, 35]. Although RGB reconstruction may provide insights into representation learning, the task itself is exceedingly complex. On the other hand, real-time autonomous systems could make more intelligent decisions through semantic anticipation (e.g., forecasting pedestrian [3] or vehicle [7]). Recent work has shown that decomposing complex systems into semantically meaningful components can improve performance and enable better long-term planning [40, 41].

Scene understanding through semantic segmentation is one of the most successful tasks in deep learning [26, 27]. Most approaches focus on reasoning about the current frame [2, 5, 30, 50, 51] through convolutional neural networks (CNNs). However, recently a new task was proposed by Luc *et al*. [33] for prediction of future segmentation frames, termed *semantic forcasting*. Luc *et al*. [33] note that directly modeling pixel-level semantics instead of intermediate RGB provides a greater ability to model scene and object dynamics. They further propose a multi-scale auto-regressive CNN as a baseline for the task. Jin *et al*. [23] improve performance even further using a multi-stage CNN to jointly predict segmentation and optical flow.

There are three limitations of prior approaches for semantic forecasting. First, both [23] and [33] use a deep CNN to both learn and apply a warping operation which transforms past segmentation features into the future. However, Jaderberg *et al*. [20] have shown that despite the power of CNNs, they have limited capacity to spatially move and transform data purely through convolution. Hence, this inability extends to optical flow warping, an essential remapping operation, as exemplified in Fig. 1. Therefore, it is generally easier for a CNN to estimate warp parameters than it is to both predict *and* warp through convolution alone. Secondly, Luc *et al*. [33] and Jin *et al*. [23] do not directly account for the inherent temporal dependency of video frames between one another. Rather, both approaches concatenate four past frames as input to a CNN which restricts the network to learn long-term (e.g., > 4 frames) dependencies. Finally, [23] and [33] were not designed with low overhead and efficiency in mind, both of which are essential for real-time autonomous systems.

Our approach addresses each of these limitations with a simple, compact network. By decomposing motion and semantics, we reduce the semantic forecasting task into two subtasks: current frame segmentation and future optical flow prediction. We utilize a learnable warp layer [52, 53] to precisely encode the warp into our structure rather than forcing a CNN to unnaturally learn and apply it at once, hence addressing the first limitation. We additionally utilize the temporal coherency of concurrent video frames through a convolutional long short-term (LSTM) module [18, 42]. We use the LSTM module to aggregate optical flow features over long term time, thus addressing the second limitation. By decomposing the problem into simpler subtasks, we are able to completely remove the CNN previously dedicated to processing concatenated segmentation features from past frames as input. Doing so dramatically reduces the number of parameters of our model and redistributes the bulk of the work to a lightweight flow network. Such processing also efficiently uses single frame segmentation features and avoids redundant re-reprocessing of frames (first-in-first-out concat.). Hence, the decomposition leads to an efficient, low overhead model, and addresses the final limitation.

Our work can be summarized in three main contributions, novel with respect to the semantic forecasting task:

1. A learnable warp layer directly applied to semantic segmentation features,
2. A convolutional LSTM to aggregate optical flow features and estimate future optical flow,
3. A lightweight, modular baseline which greatly reduces the number of model parameters, improves overall efficiency, and achieves state-of-the-art performance on short-term and moving objects semantic forecasting.

2. Related Work

Our work is largely driven by the decomposition of motion and semantics. Our underlying assumption is that the semantic forecasting task can be solved using *perfect* current frame segmentation and a *perfect* optical flow which warps current to future frames. Our underlying motivation for the decomposition is also found in [44], which focuses on the future RGB prediction task. Further, [8, 15, 22] decompose motion and semantics, but focus on improving current frame segmentation, rather than on future frames.

We review prior work with respect to our three main contributions: learnable warp layer, recurrent motion prediction, and lightweight semantic forecasting.

Learnable warp layer. Jaderberg *et al*. [20] provide a foundation for giving CNNs the power to learn transformations which are otherwise exceedingly difficult for convolution and non-linear activation layers to simultaneously learn and apply. Zhu *et al*. [52, 53] utilize a differentiatable spatial warping layer which enables efficient end-to-end training. Ilg *et al*. [19] use a similar warping layer to enable stacks of FlowNet modules which iteratively improve optical flow. Likewise, there are numerous other works which utilize a learnable warp layer [24, 28, 29, 32, 43]. Although [23] warps the RGB and segmentation features, they do not do so using an end-to-end warp layer. As such, the learnable warp layer in our model facilitates strong performance using a single prediction, rather than relying on multiple recursive steps. This influence is emphasized in our state-of-the-art results on short-term forecasting.

Recurrent motion prediction. Numerous works have been published in motion prediction [1, 4, 34, 39, 45, 46, 47, 48, 49], which motivate flow prediction and using a recurrent structure for future anticipation. Villegas *et al*. [44] encode motion as the subtraction between two past frames and utilize a ConvLSTM [18, 42] to better aggregate temporal features. Similar to our work, Patraucean *et al*. [37] utilize an LSTM to predict optical flow that is used to warp segmentation features. However, [37] use a single RGB frame as input in their network, whereas our network utilizes a pair of frames, as seen in Fig. 2. Further, we utilize the power of the FlowCNN to encode strong optical flow features as input to the LSTM, where [37] uses only a single convolutional encoding layer. Lastly, [37] utilize the future RGB frame to produce their segmentation mask, which does not enable their structure for semantic forecasting. As such, our method, to the best of our knowledge, is the first to aggregate optical flow features using a recurrent network and predict future optical flow for semantic forecasting.

Semantic forecasting. Fig. 2 compares the network structures of the two main baselines for semantic forecasting. Luc *et al*. [33] formulates a two-stage network which first obtains segmentation features through the Dilation10 [50]

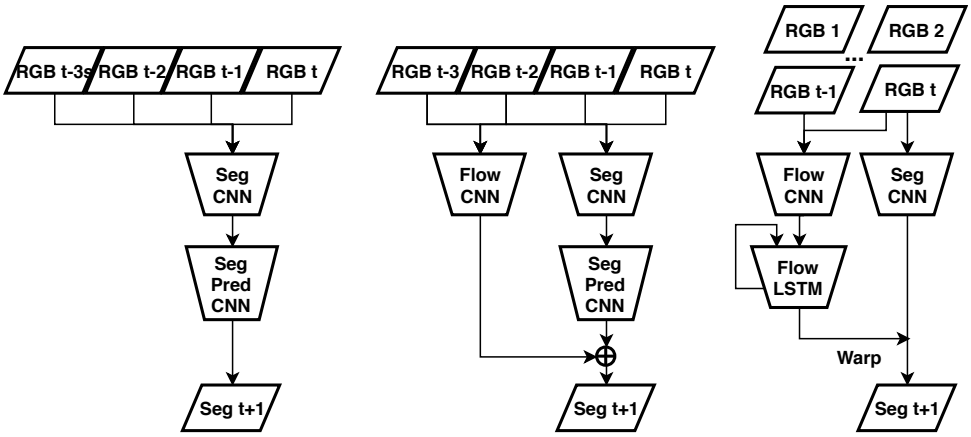

Figure 2: The overall network architectures of Luc *et al*. [33] vs. Jin *et al*. [23] vs. ours (from left to right).

current frame network (SegCNN). The four past segmentation features are then concatenated as input to their SegPredCNN, denoted S_2S, to produce the future segmentation frame. When doing mid-term prediction, S_2S uses an autoregressive approach, passing their previous predictions back into the network. The main differences between S_2S and our proposed method include the removal of the SegPredCNN and addition of the FlowCNN. By directly encoding motion as a flow prediction, our method can achieve better overall accuracy on moving objects than [33].

Jin *et al*. [23] improves upon S_2S through its addition of a FlowCNN and joint training on both flow and segmentation. Note that both [33] and [23] utilize strictly up to 4 past frames as input, while our method uses an LSTM which can leverage any number of past frames in the long-term. Further, [23] utilize an extra convolutional layer to augment the predictions through a residual connection between SegPredCNN and FlowCNN. All three CNNs in [23] are based on Res101-FCN networks, which add numerous layers to ResNet-101 [17]. The overarching motivation for [23] compared to our method is quite different. Jin *et al*. emphasized the benefit of the flow prediction influencing semantic forecasting and vice versa. In contrast, we demonstrate the value of treating these tasks *independently*. Further, there are three main differentiating factors between [23] and ours: warp layer, FlowLSTM, and network size. With the combination of FlowLSTM and warp layer, we are able to forgo the SegPredCNN, thus dramatically reducing our network size, while maintaining high-fidelity segmentation results.

3. Method

Semantic forecasting is defined as the task when given t input frames $\mathbf{X}_{1:t}$, predict the pixel-wise semantic segmentation for future frame $\mathbf{S}_{t+s}$, where s is the future step. In this section, we will present models for both short-term $s = \{1, 3\}$ and mid-term $s = \{9, 10\}$ prediction.

3.1. Short-term Prediction

3.1.1 Model Overview

Our approach is motivated by the notion that semantic forecasting can be decomposed into two sub-tasks: current frame segmentation and future optical flow prediction. Specifically, we propose

$$\mathbf{S}_{t+s} = \mathbf{O}_{t \to t+s}(\mathbf{S}_t), \tag{1}$$

where $\mathbf{S}_t$ is current segmentation and $\mathbf{O}_{t \to t+s}$ is the ground truth optical flow which defines the warp of $\mathbf{S}_t$ into $\mathbf{S}_{t+s}$. With much recent attention focused on current frame segmentation [2, 5, 30, 50, 51], our model is designed to be agnostic to the choice of backbone segmentation network. Thus, we focus on estimating future optical flow $\hat{\mathbf{O}}_{t \to t+s}$. Our method for this estimation contains three components: FlowCNN, FlowLSTM, and warp layer. The FlowCNN is designed to extract high-level optical flow features. The FlowLSTM aggregates flow features over time and produces a future optical flow prediction, while maintaining long-term memory of past frame pairs. The warp layer uses a learnable warp operation which applies the flow prediction directly to the segmentation features produced by the segmentation network. Since the warp layer is learnable, we utilize backpropagation through time (BPTT) for efficient end-to-end training of all three components.

3.1.2 FlowCNN

Although most state-of-the-art optical flow methods are non-deep learning, a critical caveat is that such approaches run on CPU, which can take anywhere between 5 min to an hour for a single frame. Since runtime efficiency is a priority for our work, we utilize FlowNet [14, 19] for optical flow estimation with a CNN. Specifically, we use the FlowNet-c architecture with pre-trained weights from FlowNet2 [19].

Utilizing the pre-trained weights is important because no ground truth optical flow is provided in Cityscapes [9]. The FlowCNN takes a concatenated pair of RGB frames as input and is denoted as:

$$\hat{\mathbf{O}}_{t-s\to t} = f_F(\mathbf{X}_{t-s}, \mathbf{X}_t). \quad (2)$$

Our reasoning for using FlowNet-c is twofold: simple and effective. Firstly, this network has only approximately 5 million parameters, which consumes less than 2 GB of GPU memory at test time and can evaluate in less than 1 second. Secondly, although Jin *et al*. [23] train their FlowCNN from scratch using weak optical flow ground truths, we note that through utilization of FlowNet-c our method demonstrates improved short-term performance with a dramatic reduction in parameters. However, we emphasize that our method is not fixed to only FlowNet-c. Instead it is compatible with any optical flow network, since the flow features can be extracted as input into the FlowLSTM, thus demonstrating the modularity of our method. Since our method is not fixed to a specific SegCNN or FlowCNN architecture, as performance on these individual tasks improve over time, our method can improve with them, demonstrating its potential longevity as a general-purpose baseline for semantic forecasting.

3.1.3 FlowLSTM

One of the main differentiating factors between our proposed method and previous semantic forecasting work is how time is treated. Specifically, both [33] and [23] do not use a recurrent structure to model the inherent temporal dynamics of past frames. Instead, 4 past frames are concatenated thereby ignoring temporal dependency among frames > 4 time-steps away from t and encodes spatial redundancy. In contrast, by utilizing a LSTM module, our method can retain long-term memory as frames are processed over time.

When a pair of input frames is passed through the FlowCNN, the flow features are extracted at a certain level of the refinement layers. We observe similar performance regardless of the level of the refinement layer extracted (predict_conv6–predict_conv2), with one exception. We find it less effective to directly aggregate the flow field predictions compared to aggregating the flow features.

As such, we extract flow features immediately before the final prediction layer of FlowNet-c, which represent 74 channels at 128×256 resolution. We use these features as input to the ConvLSTM with a 3×3 kernel and 1 padding. We then produce the future optical flow prediction using a single convolution layer with 1×1 kernel after the ConvLSTM to reduce the channels from $74 \to 2$ for the flow field. The FlowLSTM can be defined as:

$$\hat{\mathbf{O}}_{t\to t+s} = f_L(\hat{\mathbf{O}}_{1\to s+1}, \hat{\mathbf{O}}_{2\to s+2}, ..., \hat{\mathbf{O}}_{t-s\to t}). \quad (3)$$

3.1.4 Warp Layer

Recall that FlowCNN extracts motion from the input frames and FlowLSTM aggregates motion features over time. Thus, the natural next question is how to use these features for the semantic forecasting task? Both Luc *et al*. [33] and Jin *et al*. [23] utilize a SegPredCNN which attempts to extract motion from the segmentation features *and* apply the motion simultaneously. Jin *et al*. [23] combines the features extracted from the FlowCNN with features extracted from the SegPredCNN through a residual block which learns a weighted summation. By giving a CNN the ability to directly apply the warp operation with a warp layer rather than through convolutional operations, we can forgo the SegPredCNN that both [33] and [23] rely heavily upon. To define the warp layer, we must first define the SegCNN:

$$\hat{\mathbf{S}}_t = f_S(\mathbf{X_t}). \quad (4)$$

Therefore, using equations 2–4, the warp layer can be formally defined as:

$$\hat{\mathbf{S}}_{t+s} = f_W(f_L(f_F(\mathbf{X}_1, \mathbf{X}_{s+1}), ..., f_F(\mathbf{X}_{t-s}, \mathbf{X}_t)), f_S(\mathbf{X_t})). \quad (5)$$

Specifically, the warp layer takes a feature map with an arbitrary number of channels as input. Then using a continuous variant of bilinear interpolation, we compute the warped image by re-mapping w.r.t. the flow vectors. We define all pixels to be zero where the flow points outside of the image.

Finally, we reduce equation 5 as an approximation for the *perfect* warp function seen in equation 1:

$$\hat{\mathbf{S}}_{t+s} = f_W(\hat{\mathbf{O}}_{t\to t+s}, \hat{\mathbf{S}}_t). \quad (6)$$

3.1.5 Loss function

For short-term prediction, the main loss function used is cross-entropy loss with respect to the ground truth segmentation for the 20th frame. We note that due to the recurrent nature of our method, we do not need to rely on weak segmentation L_{ℓ_1} or L_{gdl} gradient difference loss of [33, 23]. Thus our main loss is formulated as:

$$L_{\text{seg}}(\hat{\mathbf{S}}_{\mathbf{t+s}}, \mathbf{S_{t+s}}) = -\sum_{(i,j)\in\mathbf{X_t}} y_{i,j} \log p(\hat{\mathbf{S}}^{i,j}_{t+s}), \quad (7)$$

where $y_{i,j}$ is the ground truth class for the pixel at location (i, j) and $p(\hat{\mathbf{S}}^{i,j}_{t+s})$ is the predicted probability that $\hat{\mathbf{S}}_{t+s}$ at location (i, j) is class y.

3.2. Mid-term Prediction

There are two main types of approach for mid-term prediction: single-step and auto-regressive. [33] and [23] respectively define mid-term prediction as being nine and ten frames forward, approximately 0.5 seconds in the future.

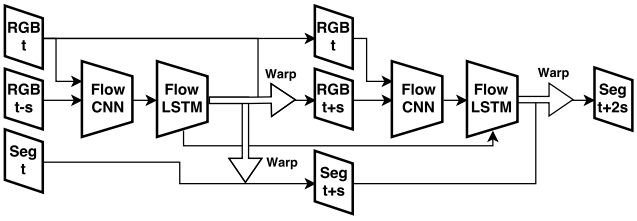

Figure 3: Unrolled network structure for mid-term prediction via auto-regressive approach.

The single-step approach uses past frames $\mathbf{X}_{1:t}$ to predict a single optical flow $\hat{\mathbf{O}}_{t\rightarrow t+m}$, where m is a mid-term jump $(9, 10)$, to warp $\mathbf{S}_t$ to $\mathbf{S}_{t+m}$. In contrast, the auto-regressive approach takes multiple steps to reach $\mathbf{S}_{t+m}$, warping both the segmentation features and the last RGB frame to pass back into the FlowCNN as input for the next step.

For example, consider when $m = 10$. For single-step, our model estimates the optical flow $\hat{\mathbf{O}}_{10\rightarrow 20}$ and warps $\mathbf{S}_{10} \rightarrow \hat{S}_{20}$. For the auto-regressive approach, our model instead estimates the shorter optical flow $\hat{\mathbf{O}}_{10\rightarrow 15}$ and warps $\mathbf{S}_{10} \rightarrow \hat{\mathbf{S}}_{15}$ and $\mathbf{X}_{10} \rightarrow \hat{\mathbf{X}}_{15}$. Then $\mathbf{X}_{10}$ and $\hat{\mathbf{X}}_{15}$ are passed back as input into the FlowCNN to produce the optical flow $\hat{\mathbf{O}}_{15\rightarrow 20}$ which warps $\hat{\mathbf{S}}_{15} \rightarrow \hat{\mathbf{S}}_{20}$.

The benefits of the single-step approach are more efficient training / testing and no error propagation through regressive use its own *noisy* prediction. In contrast, the auto-regressive approach has the benefit of being a single model for any time step, as well as, being able to regress indefinitely into the future. Additionally, most optical flow methods are designed for next frame or short-term flow estimation. Therefore, a single large step may not be as easy to estimate compared to multiple small steps. Further, both approaches can learn non-linear flow due to the presence of the FlowLSTM. For the auto-regressive approach, the unrolled recurrent structure can be visualized in Fig. 3.

3.2.1 Loss function

For mid-term prediction, we utilize not only the cross-entropy loss but also an RGB reconstruction loss. Specifically for the auto-regressive approach, with each small step, we take the ℓ_1 loss between warped $\hat{\mathbf{X}}$ and ground truth $\mathbf{X}$. For more stable training, we use the *smooth* l_1 loss [16], by:

$$L_{\text{rgb}}(\hat{\mathbf{X}}_{\mathbf{t+s}}, \mathbf{X}_{\mathbf{t+s}}) = \begin{cases} 0.5(d_{\ell_1})^2, & |d_{\ell_1}| < 1, \\ |d_{\ell_1}| - 0.5, & \text{otherwise} \end{cases} \tag{8}$$

where d_{ℓ_1} is the ℓ_1 distance as:

$$d_{\ell_1}(\hat{\mathbf{X}}_{\mathbf{t+s}}, \mathbf{X}_{\mathbf{t+s}}) = \sum_{(i,j)\in \mathbf{X}_{\mathbf{t+s}}} \left| \hat{\mathbf{X}}^{i,j}_{\mathbf{t+s}} - {\mathbf{X}_{\mathbf{t+s}}}^{i,j} \right|.$$

The full loss function is

$$L = \lambda_1 L_{\text{seg}} + \lambda_2 L_{\text{rgb}}, \tag{9}$$

where λ_1 and λ_2 are weighting factors treated as hyper-parameters during the training process. We find setting $\lambda_1 = \lambda_2 = 1$ early on during training and then iteratively lowering λ_2 closer to 0 provides optimal results.

4. Experiments

4.1. Dataset and evaluation criteria

Our experiments are focused on the urban street scene dataset Cityscapes [9], which contains $5{,}000$ high-quality images ($1{,}024 \times 2{,}048$) with pixel-wise annotations for 19 semantic classes (pedestrian, car, road, etc.). Further, 19 temporal frames preceding each annotated semantic segmentation frame are available. Our method is able to utilize all 19 previous frames for the $2{,}975$ training sequences.

We report performance of our models on the 500 validation sequences using the standard metric of mean Intersection over Union (IoU) [12]:

$$IoU = \frac{TP}{TP + FP + FN},$$

where TP, FP, and FN represent the numbers of true positive, false positive, and false negative pixels, respectively. We compute IoU with respect to the ground truth segmentation of the 20th frame in each sequence. To further demonstrate the effectiveness of our method on moving objects, we compute the mean IoU on the 8 available foreground classes (*person, rider, car, truck, bus, train, motorcycle, and bicycle*) denoted as IoU-MO.

4.2. Implementation details

In our experiments, the resolution of any input image, $\mathbf{X}$, to either the SegCNN or FlowCNN is set to $512 \times 1{,}024$. The segmentation features, $\hat{\mathbf{S}}$, extracted from the SegCNN are also at $512 \times 1{,}024$, while those extracted from PSP-half are at 64×128. The optical flow prediction, $\hat{\mathbf{O}}$, from FlowNet-c is produced at 128×256. In most experiments, we upsample both $\hat{\mathbf{S}}$ and $\hat{\mathbf{O}}$ to $512 \times 1{,}024$. For training hyperparameters, we trained with SGD using the poly learning rate policy with initial an learning rate of 0.001, power set to 0.9, weight decay set to 0.0005, and momentum set to 0.9. Additionally, we used a batch size of 1 and trained for 50k iterations ($\approx$17 Epochs). All of our experiments were conducted using either a NVIDIA Titan X or a 1080 Ti GPU and using the Caffe [21] framework.

4.3. Baseline comparison

We compare our method to the two main previous works in semantic forecasting on Cityscapes: [33] and [23]. Our comparison is multifaceted: number of model parameters (in millions), model efficiency (in seconds), and model performance (in mean IoU). Additionally, we compare against [33] on the more challenging foreground moving objects

Model	Training (hours)	Testing (sec)	Params (mil)	Training GPUs	Testing Method
S_2S [33]	96 $(8x)$	0.248 $(4.8x)$	$\approx$1.5 $(+300\%)$	1	Single
SP-MD [23]	–	2.182 $(42x)$	$\approx$115 (-95%)	4-8	Sliding
ours	**12**	**0.052**	$\approx$6.0	1	Single

Table 1: Computational complexity analysis with respect to previous work – Luc *et al*. [33] and Jin *et al*. [23]. Models are measured without the SegCNN included (only SegPred). Runtime estimates were calculated by averaging 100 forward passes with each model. Single vs. sliding testing describes using a single forward pass of $512 \times 1,024$ resolution, relative to the costly sliding window approach of eight overlapping 713×713 full resolution crops.

Model	IoU $(t = 3)$	IoU-MO $(t = 3)$	IoU $(t = 9)$	IoU-MO $(t = 9)$
Copy last input [33]	49.4	43.4	36.9	26.8
Warp last input [33]	59.0	54.4	44.3	37.0
S_2S [33]	59.4	55.3	47.8	40.8
ours	**67.1**	**65.1**	**51.5**	**46.3**

Table 2: Comparison of baselines with Luc *et al*. [33] for short-term ($t = 3$) and mid-term ($t = 9$) for all nineteen classes and eight foreground, moving objects (MO) classes.

Model	IoU $(t = 1)$	IoU $(t = 10)$
Copy last input [23]	59.7	41.3
Warp last input [23]	61.3	42.0
SP-MD† [23]	–	52.6
SP-MD [23]	66.1	**53.9**
ours (c)†	73.0	51.8
ours (C) †	**73.2**	52.5

Table 3: Comparison of baselines with Jin *et al*. [23] for short-term ($t = 1$) and mid-term ($t = 10$). † - model contained no recurrent fine-tuning. We compare our models with FlowNet2-c and FlowNet2-C backbones, where C contains $\frac{8}{3}$ more feature channels (24 vs. 64, 48 vs. 128, etc.).

(IoU-MO) baselines. To more directly compare with [33], we utilize their definition of short-term as $t = 3$ and mid-term as $t = 9$. However, [23] redefines short-term as $t = 1$ and mid-term as $t = 10$. Thus, without reproducing each of their methods, we distinctly compare against each definition of short-term and mid-term prediction separately.

4.3.1 Baselines:

Copy last input: [33, 23] use current frame segmentation, $\hat{\mathbf{S}}_\mathbf{t}$, as prediction for future frame segmentation, $\hat{\mathbf{S}}_{\mathbf{t+s}}$.

Warp last input: [23, 33] warp current frame segmentation, $\hat{\mathbf{S}}_\mathbf{t}$, using future optical flow, $\hat{\mathbf{O}}_{\mathbf{t}\rightarrow\mathbf{t+s}}$. [33] estimates future optical flow using FullFlow [6] with the underlying assumption that $\hat{\mathbf{O}}_{\mathbf{t}\rightarrow\mathbf{t+s}} \approx -\hat{\mathbf{O}}_{\mathbf{t}\rightarrow\mathbf{t-s}}$. [23] estimates the optical flow with a FlowCNN (Res101-FCN) trained from scratch using EpicFlow [38] for weak ground truths.

S_2S: Luc *et al*. [33] generate 4 previous frame segmentations: $\hat{\mathbf{S}}_{\mathbf{t-3s}}, \hat{\mathbf{S}}_{\mathbf{t-2s}}, \hat{\mathbf{S}}_{\mathbf{t-s}}, \hat{\mathbf{S}}_\mathbf{t}$ using the current frame network from [50] then concatenate the features as input to the multi-scale CNN from [36] as their SegPredCNN.

SP-MD. Jin *et al*. [23] also generate 4 previous frame segmentations: $\hat{\mathbf{S}}_{\mathbf{t-3s}}$, $\hat{\mathbf{S}}_{\mathbf{t-2s}}$, $\hat{\mathbf{S}}_{\mathbf{t-s}}$, $\hat{\mathbf{S}}_\mathbf{t}$ using their own Res101-FCN network and concatenate as input to their SegPredCNN (Res101-FCN). Simultaneously, their FlowCNN (Res101-FCN) takes in 4 previous RGB frames $\mathbf{X}_{\mathbf{t-3s}}$, $\mathbf{X}_{\mathbf{t-2s}}$, $\mathbf{X}_{\mathbf{t-s}}$,$\mathbf{X}_\mathbf{t}$ to generate future optical flow $\hat{\mathbf{O}}_{\mathbf{t}\rightarrow\mathbf{t+s}}$. Lastly, features from the FlowCNN and SegPredCNN are fused via a residual block to make the final prediction.

4.3.2 Analysis

The computational complexity of our methods relative to [33] and [23] is in Tab. 1. Since our work is motivated by deployment of real-time autonomous systems, we emphasize both low overhead and efficiency. Our approach demonstrates a significant decrease in the number of SegPred parameters relative to [23]. Specifically, we reduce the number of parameters from $\approx 115M$ to $\approx 6.0M$ resulting in a dramatic 95% decrease. This large reduction can be directly linked to our removal of the large SegPredCNN replaced with the lightweight FlowNet-c as our FlowCNN combined with the warp layer. With respect to efficiency, our method sees a $4.8x$ and $42x$ speedup overall relative to [33] and [23], respectively. We infer this speedup is correlated with our need to only process a single past frame's segmentation features and our ability to limit the number of redundant image processing steps.

Although, we increase the number of SegPred parameters from $\approx 1.5M$ to $\approx 6.0M$ relative to [33], we find an appreciable performance increase on short-term, mid-term,

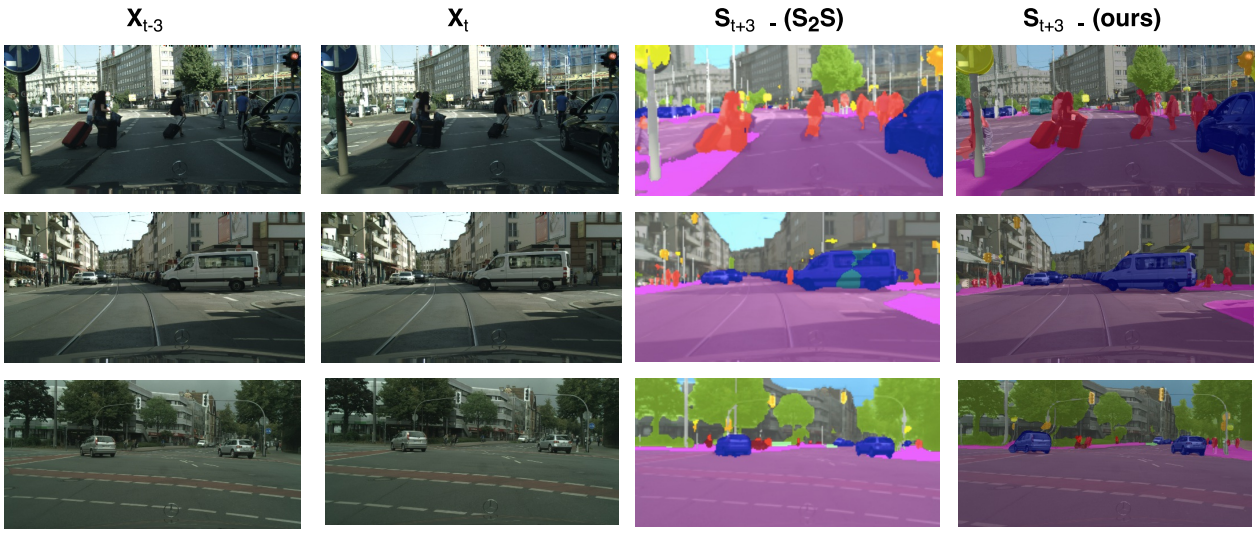

Figure 4: S_2S [33] vs. ours. Qualitative improvements are demonstrated, specifically with respect to moving objects such as pedestrian and vehicle trajectories.

and moving objects prediction (Tab. 2). Specifically, we improve by 7.7% and 3.7% on short-term and mid-term, respectively. On the more challenging moving objects benchmark, we improve on short-term by 9.8% and mid-term by 5.5%. Since we directly encode motion into our model through our FlowCNN and aggregate optical flow features over time with our FlowLSTM, we can infer more accurate trajectories on moving objects, seen in Fig. 4.

Finally, in Tab. 3, we compare with the state-of-the-art method on $t = 1$ and $t = 10$, [23]. For next frame prediction, we demonstrate a new state-of-the-art pushing the margin by 7.1%. For $t = 10$ prediction, our method performs 1.4% worse overall and only 0.1% worse when controlling for recurrent fine-tuning. It should be noted that the model that performs 52.5% achieves roughly a 0.8% boost due to in-painting. More specifically this occurred when the flow field was occluded and predicted a 0 for flow. Thus, we utilized a simple inpainting method which directly copies the current frame segmentation to provide an additional small boost. However, in Fig. 5, we can observe failure cases motivating future work.

In summary, we achieve state-of-the-art performance on short-term ($t = 1, t = 3$) and moving objects ($t = 3, t = 9$) prediction. We further find only $\approx 1\%$ performance degradation on mid-term ($t = 10$) prediction, while reducing the number of model parameters by $\approx 95\%$ and accordingly increasing model efficiency $42x$.

4.4. Ablation studies

In this section, we study the effects of our model from the persepectives of: warp layer, FlowLSTM, time / step size, and auto-regressive vs. single-step.

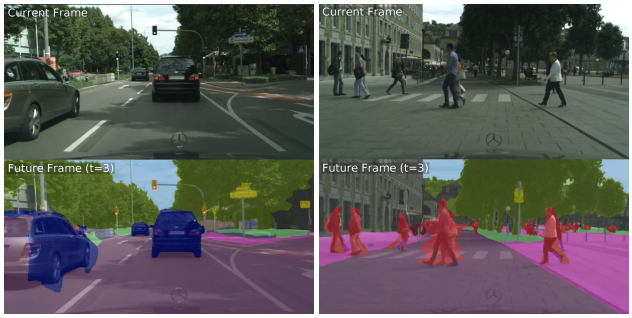

Figure 5: We can visualize two distinct failure cases: occlusion and shape degradation. On the left, we our approach struggles reconstruct when objects enter a scene. On the right, our approach can correctly forecast pedestrians, but doesn't maintain their shape well. Future work including inpainting and object consistency may address these issues.

4.4.1 Impact of warp layer

Our main enabling factor for strong performance on next frame and short-term prediction is the warp layer. We can see in Tab. 4 how significant the degradation performance is when the warp layer is removed from our system. It is such an integral component that our method would not function without it. Thus, substituting convolutional layer and concatenation / element-wise addition operations was proven highly ineffective. To truly ablate the impact of the warp layer, there would likely need to be a SegPredCNN in addition to these concatenation / element-wise addition operators. However, such a change runs contrary to the main motivation for our work being effective with low overhead.

Configuration	IoU ($t = 3$)	IoU ($t = 10$)
Addition	53.8	37.3
Concatenation	54.8	38.6
Warp Layer	**66.0**	**50.0**

Table 4: Impact of warp layer, all configurations use the single-step approach and contain the FlowLSTM.

Configuration	IoU ($t = 3$)	IoU ($t = 10$)
No FlowLSTM	62.3	46.0
FlowLSTM	**66.0**	**50.0**

Table 5: Impact of FlowLSTM, all configurations use the single-step approach and contain the warp layer.

Time	Frames	IoU ($t = 3$)
2	$15, 16$	62.4
4	$13, 14, 15, 16$	67.0
8	$7, 8, ..., 16$	**67.1**

Table 6: Impact of time, all configurations use single-step and contain both the FlowLSTM and warp layer.

4.4.2 Impact of FlowLSTM

One novel aspect of our work relative to others in semantic forecasting is the usage of a LSTM. Specifically, although others utilized recurrent fine-tuning to improve their systems with BPTT; our system is the first, to the best of our knowledge, to contain a memory module. In Tab. 5, the effects of FlowLSTM without recurrent fine-tuning are shown, which demonstrates the value of the memory module within LSTM, for both short- and mid-term prediction.

4.4.3 Impact of time and step size

We can observe in Tabs. 6-7, the benefit of including more past frames and using a small step size for short-term prediction. Smaller step size worked together with the optical flow network trained on next frame flow. Although notice the diminishing returns going further back in time from 4 to 8 past steps. Our method, using the FlowLSTM aggregation, is uniquely able to process frames at a step size of 1 and a predict a jump of 3 into the future. Similarly, our best performing mid-term model processes past frames at step size of 3 to predict a jump of 10. Another aspect of our approach is the lack of redundant processing, as we only process each frame twice, compared to four times with previous methods (using first-in-first-out concat.). Further, our method can process any number of past frames while maintaining its prediction integrity and run-time efficiency.

Step Size	Frames	IoU ($t = 3$)
9	$7, 16$	62.6
3	$7, 10, 13, 16$	66.0
1	$7, 8, ..., 16$	**67.1**

Table 7: Impact of step size, all configurations use the single-step approach and contain both the FlowLSTM and warp layer.

Configuration	IoU ($t = 3$)	IoU ($t = 10$)
Auto-regressive	64.4	48.7
Single-step	**66.0**	**50.0**

Table 8: Impact of auto-regressive vs. single-step methods for both short-term and mid-term prediction, all configurations contain both the FlowLSTM and warp layer.

4.4.4 Impact of auto-regressive vs. single-step

In Tab. 8 we show that both auto-regressive and single-step approaches are roughly equivalent in terms of accuracy, with only a slight degradation in favor of the single-step approach. We conjecture that error propagation with 3 recursive steps is more significant than doing a single large step. The overall pros and cons of single-step vs. auto-regressive depend upon the real-world application deployment. For instance, if efficiency is the highest priority in an autonomous vehicle, limiting the processing of redundant frames with a single-step model would be beneficial. On the other hand, if flexibility to predict any time step in the future is desired, an auto-regressive approach may be preferred.

5. Conclusion

In this paper, we posed semantic forecasting in a new way — decomposing motion and content. Through this decomposition into current frame segmentation and future optical flow prediction, we enabled a more compact model. This model contained three main components: flow prediction network, feature-flow aggregation LSTM, and an end-to-end warp layer. Together these components worked in unison to achieve state-of-the-art performance on short-term and moving objects segmentation prediction. Additionally, our method reduced the number of parameters by up to 95% and demonstrated a speedup beyond $40x$. In turn, our method was designed with low overhead and efficiency in mind, an essential factor for real-world autonomous systems. As such, we proposed a lightweight, modular baseline for recurrent flow-guided semantic forecasting.

References

[1] A. Alahi, K. Goel, V. Ramanathan, A. Robicquet, L. Fei-Fei, and S. Savarese. Social lstm: Human trajectory prediction in crowded spaces. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[2] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *Pattern Analysis and Machine Intelligence (PAMI)*, 2017.

[3] G. Brazil, X. Yin, and X. Liu. Illuminating pedestrians via simultaneous detection & segmentation. In *International Conference on Computer Vision (ICCV)*, 2017.

[4] Y.-W. Chao, J. Yang, B. Price, S. Cohen, and J. Deng. Forecasting human dynamics from static images. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[5] L.-C. 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. *arXiv preprint arXiv:1606.00915*, 2016.

[6] Q. Chen and V. Koltun. Full flow: Optical flow estimation by global optimization over regular grids. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[7] X. Chen, K. Kundu, Z. Zhang, H. Ma, S. Fidler, and R. Urtasun. Monocular 3d object detection for autonomous driving. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[8] J. Cheng, Y.-H. Tsai, S. Wang, and M.-H. Yang. Segflow: Joint learning for video object segmentation and optical flow. In *International Conference on Computer Vision (ICCV)*, 2017.

[9] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The cityscapes dataset for semantic urban scene understanding. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[10] F. Cricri, X. Ni, M. Honkala, E. Aksu, and M. Gabbouj. Video ladder networks. In *NIPS workshop on ML for Spatiotemporal Forecasting*, 2016.

[11] E. L. Denton and V. Birodkar. Unsupervised learning of disentangled representations from video. In *Neural Information Processing Systems (NIPS)*, 2017.

[12] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision (IJCV)*, 2015.

[13] C. Finn, I. Goodfellow, and S. Levine. Unsupervised learning for physical interaction through video prediction. In *Neural Information Processing Systems (NIPS)*, 2016.

[14] P. Fischer, A. Dosovitskiy, E. Ilg, P. Häusser, C. Hazırbaş, V. Golkov, P. Van der Smagt, D. Cremers, and T. Brox. Flownet: Learning optical flow with convolutional networks. In *International Conference on Computer Vision (ICCV)*, 2015.

[15] R. Gadde, V. Jampani, and P. V. Gehler. Semantic video CNNs through representation warping. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[16] R. Girshick. Fast r-CNN. In *International Conference on Computer Vision (ICCV)*, 2015.

[17] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[18] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 1997.

[19] 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 *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[20] M. Jaderberg, K. Simonyan, A. Zisserman, and K. Kavukcuoglu. Spatial transformer networks. In *Neural Information Processing Systems (NIPS)*, 2017.

[21] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. *arXiv preprint arXiv:1408.5093*, 2014.

[22] X. Jin, X. Li, H. Xiao, X. Shen, Z. Lin, J. Yang, Y. Chen, J. Dong, L. Liu, Z. Jie, et al. Video scene parsing with predictive feature learning. In *International Conference on Computer Vision (ICCV)*, 2017.

[23] X. Jin, H. Xiao, X. Shen, J. Yang, Z. Lin, Y. Chen, Z. Jie, J. Feng, and S. Yan. Predicting scene parsing and motion dynamics in the future. In *Neural Information Processing Systems (NIPS)*, 2017.

[24] A. Jourabloo, X. Liu, M. Ye, and L. Ren. Pose-invariant face alignment with a single CNN. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[25] N. Kalchbrenner, A. van den Oord, K. Simonyan, I. Danihelka, O. Vinyals, A. Graves, and K. Kavukcuoglu. Video pixel networks. In *International Conference on Machine Learning (ICML)*, 2017.

[26] Y. LeCun, Y. Bengio, and G. Hinton. Deep learning. *Nature*, 2015.

[27] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 1998.

[28] X. Liang, L. Lee, W. Dai, and E. P. Xing. Dual motion gan for future-flow embedded video prediction. In *International Conference on Computer Vision (ICCV)*, 2017.

[29] Z. Liu, R. Yeh, X. Tang, Y. Liu, and A. Agarwala. Video frame synthesis using deep voxel flow. In *International Conference on Computer Vision (ICCV)*, 2017.

[30] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[31] W. Lotter, G. Kreiman, and D. Cox. Deep predictive coding networks for video prediction and unsupervised learning. In *International Conference on Learning Representations (ICLR)*, 2017.

[32] C. Lu, M. Hirsch, and B. Schölkopf. Flexible spatio-temporal networks for video prediction. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[33] P. Luc, N. Neverova, C. Couprie, J. J. Verbeek, and Y. LeCun. Predicting deeper into the future of semantic segmentation. In *International Conference on Computer Vision (ICCV)*, 2017.

[34] Z. Luo, B. Peng, D.-A. Huang, A. Alahi, and L. Fei-Fei. Unsupervised learning of long-term motion dynamics for videos. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[35] R. Mahjourian, M. Wicke, and A. Angelova. Geometry-based next frame prediction from monocular video. In *Intelligent Vehicles Symposium (IV)*, 2017.

[36] M. Mathieu, C. Couprie, and Y. LeCun. Deep multi-scale video prediction beyond mean square error. In *International Conference on Learning Representations (ICLR)*, 2016.

[37] V. Patraucean, A. Handa, and R. Cipolla. Spatio-temporal video autoencoder with differentiable memory. In *International Conference on Learning Representations (ICLR) Workshop*, 2016.

[38] J. Revaud, P. Weinzaepfel, Z. Harchaoui, and C. Schmid. Epicflow: Edge-preserving interpolation of correspondences for optical flow. In *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[39] S. M. Safdarnejad, X. Liu, and L. Udpa. Robust global motion compensation in presence of predominant foreground. In *British Machine Vision Conference (BMVC)*, 2015.

[40] S. Shalev-Shwartz, N. Ben-Zrihem, A. Cohen, and A. Shashua. Long-term planning by short-term prediction. *arXiv preprint arXiv:1602.01580*, 2016.

[41] S. Shalev-Shwartz and A. Shashua. On the sample complexity of end-to-end training vs. semantic abstraction training. *arXiv preprint arXiv:1604.06915*, 2016.

[42] X. Shi, Z. Chen, H. Wang, D.-Y. Yeung, W.-k. Wong, and W.-c. Woo. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. In *Neural Information Processing Systems (NIPS)*, 2015.

[43] L. Tran and X. Liu. Nonlinear 3D face morphable model. In *Computer Vision and Pattern Recognition (CVPR)*, 2018.

[44] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. In *International Conference on Learning Representations (ICLR)*, 2017.

[45] R. Villegas, J. Yang, Y. Zou, S. Sohn, X. Lin, and H. Lee. Learning to generate long-term future via hierarchical prediction. In *International Conference on Machine Learning (ICML)*, 2017.

[46] C. Vondrick, H. Pirsiavash, and A. Torralba. Anticipating visual representations from unlabeled video. In *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[47] J. Walker, C. Doersch, A. Gupta, and M. Hebert. An uncertain future: Forecasting from static images using variational autoencoders. In *European Conference on Computer Vision (ECCV)*, 2016.

[48] J. Walker, A. Gupta, and M. Hebert. Patch to the future: Unsupervised visual prediction. In *Computer Vision and Pattern Recognition (CVPR)*, 2014.

[49] J. Walker, A. Gupta, and M. Hebert. Dense optical flow prediction from a static image. In *International Conference on Computer Vision (ICCV)*, 2015.

[50] F. Yu and V. Koltun. Multi-scale context aggregation by dilated convolutions. In *International Conference on Learning Representations (ICLR)*, 2016.

[51] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.

[52] X. Zhu, Y. Wang, J. Dai, L. Yuan, and Y. Wei. Flow-guided feature aggregation for video object detection. In *International Conference on Computer Vision (ICCV)*, 2017.

[53] X. Zhu, Y. Xiong, J. Dai, L. Yuan, and Y. Wei. Deep feature flow for video recognition. In *Computer Vision and Pattern Recognition (CVPR)*, 2017.