Temporally Distributed Networks for Fast Video Semantic Segmentation

Ping Hu[1], Fabian Caba Heilbron[2], Oliver Wang[2], Zhe Lin[2], Stan Sclaroff[1], Federico Perazzi[2]
[1]Boston University [2]Adobe Research

Abstract

We present TDNet, a temporally distributed network designed for fast and accurate video semantic segmentation. We observe that features extracted from a certain high-level layer of a deep CNN can be approximated by composing features extracted from several shallower sub-networks. Leveraging the inherent temporal continuity in videos, we distribute these sub-networks over sequential frames. Therefore, at each time step, we only need to perform a lightweight computation to extract a sub-features group from a single sub-network. The full features used for segmentation are then recomposed by the application of a novel attention propagation module that compensates for geometry deformation between frames. A grouped knowledge distillation loss is also introduced to further improve the representation power at both full and sub-feature levels. Experiments on Cityscapes, CamVid, and NYUD-v2 demonstrate that our method achieves state-of-the-art accuracy with significantly faster speed and lower latency.

1. Introduction

Video semantic segmentation aims to assign pixel-wise semantic labels to video frames. As an important task for visual understanding, it has attracted more and more attention from the research community [19, 27, 34, 39]. The recent successes in dense labeling tasks [4, 20, 25, 28, 50, 54, 56, 59] have revealed that strong feature representations are critical for accurate segmentation results. However, computing strong features typically require deep networks with high computation cost, thus making it challenging for real-world applications like self-driving cars, robot sensing, and augmented-reality, which require both high accuracy *and* low latency.

The most straightforward strategy for video semantic segmentation is to apply a deep image segmentation model to each frame independently, but this strategy does not leverage temporal information provided in the video dynamic scenes. One solution, is to apply the same model to all frames and add additional layers on top to model temporal context to extract better features [10, 19, 23, 34]. However, such methods do not help improve efficiency as all features must be recomputed at each frame. To reduce redundant computation, a reasonable approach is to apply a strong image segmentation model only at keyframes, and reuse the high-level feature for other frames [18, 27, 31, 58]. However, the spatial misalignment of other frames with respect to the keyframes is challenging to compensate for and often leads to decreased accuracy comparing to the baseline image segmentation models as reported in [18, 27, 31, 58]. Additionally, these methods have different computational loads between keyframes and non-keyframes, which results in high maximum latency and unbalanced occupation of computation resources that may decrease system efficiency.

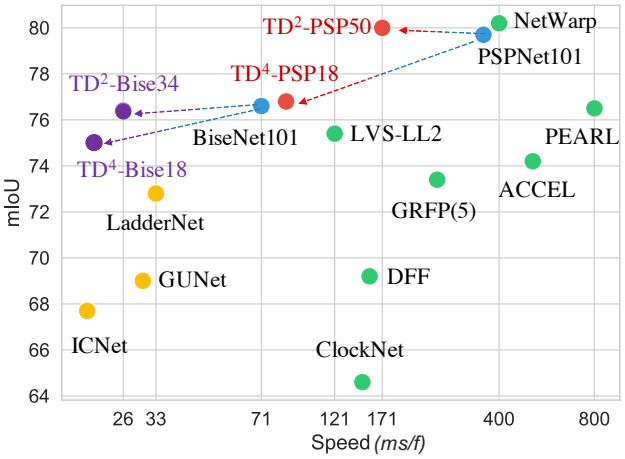


Figure 1. Performance on Cityscapes. Our proposed TDNet variants (denoted as ● and ●) linked to their corresponding deep image segmentation backbones (denoted as ●) with similar number of parameters. Compared with video semantic segmentation methods NetWarp [10], PEARL [19], ACCEL [18], LVS-LLS [27], GRFP [34], ClockNet [39], DFF [58], and real-time segmentation models LadderNet [21], GUNet [32], and ICNet [55], our TDNet achieves a better balance of accuracy and speed.

To address these challenges, we propose a novel deep learning model for high-accuracy and low-latency semantic video segmentation named Temporally Distributed Network (TDNet). Our model is inspired by *Group Convolution* [17, 22], which shows that extracting features with separated filter groups not only allows for model parallelization, but also helps learn *better* representations. Given a

deep image segmentation network like PSPNet [56], we divide the features extracted by the deep model into N (*e.g.* N=2 or 4) groups, and use N distinct shallow sub-networks to approximate each group of feature channels. By forcing each sub-network to cover a separate feature subspace, a strong feature representation can be produced by reassembling the output of these sub-networks. For balanced and efficient computation over time, we let the N sub-networks share the same shallow architecture, which is set to be $\frac{1}{N}$ of the original deep model's size to preserve a similar total model capacity [42, 50, 53].

When segmenting video streams, the N sub-networks are sequentially and circularly assigned to frames over time, such that complementary sub-feature groups are alternatively extracted over time and only one new sub-feature group needs to be computed at each time step. To compensate for spatial misalignment caused by motion across frames, we propose an attention propagation module for reassembling features from different time steps. To further enhance the network's representational power, we also present a grouped distillation loss to transfer knowledge from a full deep model to our distributed feature network at both full and sub-feature group levels. With this new model, we only need to run a light-weight forward propagation at each frame, and can aggregate full features by *reusing* sub-features extracted in previous frames. As shown in Fig 1, our method outperforms state-of-the-art methods while maintaining lower latency. We validate our approach through extensive experiments over multiple benchmarks.

In summary, our contributions include: i) a temporally distributed network architecture and grouped knowledge distillation loss that accelerates state-of-the-art semantic segmentation models for videos with more than 2× lower latency at comparable accuracy; ii) an attention propagation module to efficiently aggregate distributed feature groups over time with robustness to geometry variation across frames; iii) better accuracy and latency than previous state-of-the-art video semantic segmentation methods on three challenging datasets including Cityscapes, Camvid, and NYUD-v2.

2. Related Work

Image semantic segmentation is an active area of research that has witnessed significant improvements in performance with the success of deep learning [12, 16, 28, 41]. As a pioneer work, the Fully Convolutional Network (FCN) [30] replaced the last fully connected layer for classification with convolutional layers, thus allowing for dense label prediction. Based on this formulation, follow-up methods have been proposed for efficient segmentation [24, 36, 37, 52, 55] or high-quality segmentation [4, 7, 11, 26, 38, 40, 43, 44, 45].

Semantic segmentation has also been widely applied to videos [14, 23, 31, 46], with different approaches employed to balance the trade-off between quality and speed. A number of methods leverage temporal context in a video by repeatedly applying the same deep model to each frame and temporally aggregating features with additional network layers [10, 19, 34]. Although these methods improve accuracy over single frame approaches, they incur additional computation over a per-frame model.

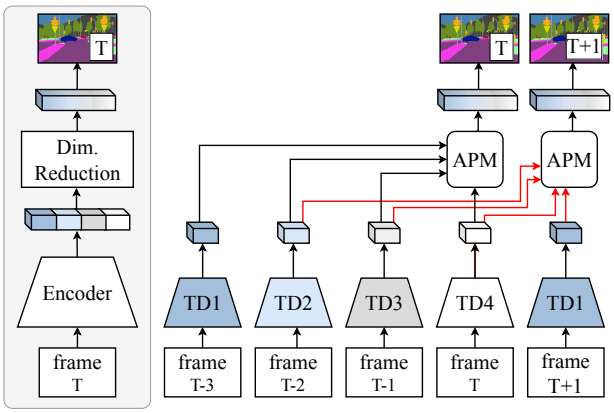
Another group of methods target efficient video segmentation by utilizing temporal continuity to propagate and reuse the high-level features extracted at key frames [18, 27, 39, 58]. The challenge of these methods is how to robustly propagate pixel-level information over time, which might be misaligned due to motion between frames. To address this, Shelhamer *et al*. [39] and Carreira *et al*. [2] directly reuse high-level features extracted from deep layers at a low resolution, which they show are relatively stable over time. Another approach, employed by Zhu *et al*. [58] is to adopt optical flow to warp high-level features at keyframes to non keyframes. Jain *et al*. [18] further updates the flow warped feature maps with shallow features extracted at the current frame. However, using optical flow incurs significant computation cost and can fail with large motion, disocclusions, and non-textured regions. To avoid using optical flow, Li *et al*. [27] instead proposes to use *spatially variant convolution* to adaptively aggregate features within a local window, which however is still limited by motion beyond that of the predefined window. As indicated in [18, 27, 58], though the overall computation is reduced compared to their image segmentation baselines, the accuracy is also decreased. In addition, due to the extraction of high-level features at keyframes, these methods exhibit inconsistency speeds, with the maximum latency equivalent to that of the single-frame deep model. In contrast to these, our approach does not use keyframe features, and substitutes optical-flow with an attention propagation module, which we show improves both efficiency and robustness to motion.

3. Temporally Distributed Network

In this section, we describe the architecture of a Temporally Distributed Network (TDNet), with an overview in Fig 2. In Sec. 3.1 we introduce the main idea of distributing sub-networks to extract feature groups from different temporal frames. In Sec 3.2, we present our attention propagation module designed for effective aggregation of spatially misaligned feature groups.

3.1. Distributed Networks

Inspired by the recent success of *Group Convolution* [17, 22] which show that adopting separate convolutional paths can increase a model's effectiveness by enhancing the sparsity of filter relationships, we propose to divide features from a deep neural network into a group of sub-features and



a) Per-frame Model

b) Temporally Distributed Model (TDNet)

Figure 2. As opposed to applying a single deep model to segment each frame independently (a), in TDNet (b) we distribute feature extraction evenly across sequential frames to reduce redundant computation, and then aggregate them using the Attention Propagation Module (APM), to achieve strong features for accurate segmentation.

approximate them using a set of shallow sub-networks each of which only covers a subspace of the original model's feature representation.

In addition, we observe that the full feature map is large, and dimension reduction (Fig 2(a)) is costly. In PSPNet50 [56], the feature map has 4096 channels and dimension reduction takes about a third of the total computation. To further improve efficiency, based on *block matrix multiplication* [9], we convert the convolutional layer for dimension reduction to the summation of series of convolution operations at the subspace level, which enables us to distribute these subspace-level convolution operations to their respective subnetworks. As a result, the output of the dimension reduction layers is recomposed simply by addition, before being used in the prediction head of the network. Keeping a similar total model size to the original deep model, we show that aggregating multiple shallow network paths can have a similarly strong representational power as the original deep model [42, 48, 50, 53].

In the context of single image segmentation, the advantage of such an approach is that it allows for faster computation by extracting feature paths *in parallel* on multiple devices. However, in the context of segmenting video sequences, we can take advantage of their inherent temporal continuity and distribute the computation along the *temporal* dimension. We apply this distributed feature extraction method to video by applying the sub-networks to sequential frames, and refer to the new architecture as Temporally Distributed Network (TDNet). As shown in Fig 2(b), TDNet avoids redundant sub-features computation by *reusing* the sub-feature groups computed at previous time steps. The full feature representation at each frame is then produced by aggregating previously computed feature groups with the current one.

3.2. Feature Aggregation

A big challenge of aggregating feature groups extracted at different time steps is the spatial misalignment caused by motion between frames. Optical flow-based warping is a popular tool to correct for such changes [10, 18, 34, 58], but it is expensive to compute, prone to errors, and restricted to a single match per pixel. To tackle such challenges, we propose an Attention Propagation Module (APM), which is based on the non-local attention mechanism [47, 49, 57], but extended to deal with spatio-temporal variations for the video semantic segmentation task. We now define how we integrate the APM into TDNet.

As shown in Fig. 3, TDNet is composed of two phases, the *Encoding Phase* and *Segmentation Phase*. The encoding phase extracts alternating sub-feature maps over time. Rather than just generating the ***Value*** feature maps which contain the path-specific sub-feature groups, we also let the sub-networks produce ***Query*** and ***Key*** maps for building correlations between pixels across frames. Formally, the feature path-i produces a sub-feature map $X_i \in \mathcal{R}^{C\times H\times W}$. Then, as in prior work [47], the corresponding encoding module "Encoding-i" converts X_i into a value map $V_i \in \mathcal{R}^{C\times H\times W}$, as well as lower dimensional query and key maps $Q_i \in \mathcal{R}^{\frac{C}{8}\times H\times W}$, $K_i \in \mathcal{R}^{\frac{C}{8}\times H\times W}$ with three 1×1 convolutional layers.

In the segmentation phase, the goal is to produce segmentation results based on the full features recomposed from the outputs of sub-networks from previous frames. Assuming we have m (m=4 in Fig. 3) independent feature paths derived from video frames, and would like to build a full feature representation for frame t by combining the outputs of the previous m-1 frames with the current frame. We achieve this with spatio-temporal attention [35, 49], where we independently compute the ***Affinity*** between pixels of the current frame t and the previous m-1 frames.

$$\boldsymbol{Aff}_p = Softmax(\frac{Q_t K_p^\top}{\sqrt{d_k}}) \tag{1}$$

where p indicates a previous frame and d_k is the dimension of the ***Query*** and ***Key***. Then, the sub-feature maps at the current frame and previous m-1 frames are merged as,

$$V_t' = V_t + \sum_{p=t-m+1}^{t-1} \phi(\boldsymbol{Aff}_p V_p) \tag{2}$$

With this attention mechanism, we effectively capture the non-local correlation between pixels across frames, with time complexity of $\mathcal{O}((m-1)d_k H^2 W^2)$ for the affinity in Eq. 1. However, features for semantic segmentation are high resolution and Eq 2 incurs a high computation cost. To

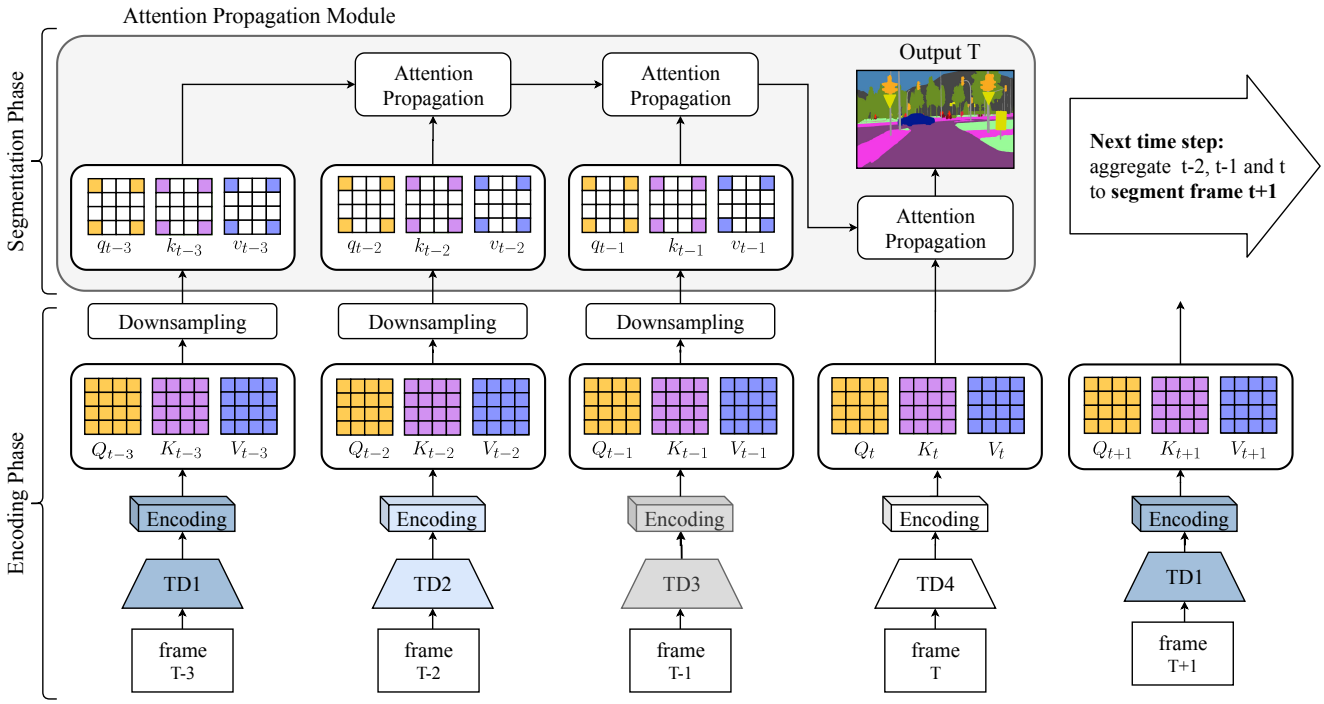


Figure 3. Illustration of TDNet with four sub-networks. Since we circularly distribute sub-networks over sequential frames, any four-frame temporal window will cover a full set of the sub-networks. In order to segment frame *t*, we apply the attention propagation module to propagate and merge sub-feature maps previously extracted from (*t-3*, *t-2*, *t-1*) with the sub-feature map from *t*. For the next frame *t+1*, a full feature representation is aggregated by similarly reusing the sub-features extract at frames (*t-2*, *t-1*, *t*).

improve efficiency, we downsample the attention maps and propagate them over time.

Attention Downsampling. We adopt a simple yet effective strategy, which is to downsample the reference data as indicated by the "Downsampling" module in Fig. 3. Formally, when segmenting a frame T, we apply a spatial pooling operation $\gamma_n(\cdot)$ with stride n to the previous m-1 frames' Query, Key, and Value maps,

$$q_i = \gamma_n(Q_i), \quad k_i = \gamma_n(K_i), \quad v_i = \gamma_n(V_i) \tag{3}$$

With these downsampled maps, the complexity for Eq. 2 decreases to $\mathcal{O}(\frac{(m-1)d_k H^2 W^2}{n^2})$. We conduct experiments and find that n=4 works well to preserve necessary spatial information while greatly decreasing the computational cost (see Sec 5.3).

Attention Propagation. Next, we propose a propagation approach, where instead of computing the attention between the current frame and all previous ones, we restrict computation to neighboring frames, and propagate it through the window. This allows us not only to reduce the number of attention maps we have to compute, but also to also restrict attention computation to subsequent frames, where motion is smaller. Given a time window composed of frames from $t-m+1$ to t together their respective downsampled Query, Key, and Value maps, then for an intermediate frame $p \in (t-m+1, t)$, the attention is propagated as,

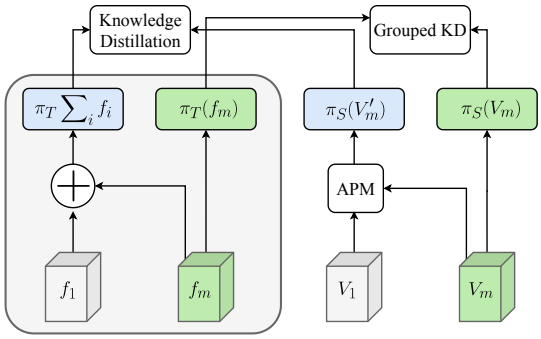
$$v'_p = \phi\left(Softmax(\frac{q_p k_{p-1}^\top}{\sqrt{d_k}})v'_{p-1}\right) + v_p \tag{4}$$

where $v'_{t-m+1} = \gamma_n(V_{t-m+1})$, q, k, and v are the downsampled maps as in Eq. 3, d_k is the number of dimensions for Query and Key, and ϕ_p is a 1×1 convolutional layer. The final feature representation at frame t is then computed as,

$$V'_t = \phi\left(Softmax(\frac{Q_t k_{t-1}^\top}{\sqrt{d_k}})v'_{t-1}\right) + V_t \tag{5}$$

and segmentation maps are generated by: $S_m = \pi_m(V'_m)$, where π_m is the final prediction layer associated with sub-network m.

With this proposed framework, the time complexity is reduced to $\mathcal{O}(\frac{(m-2)\cdot d_k H^2 W^2}{n^4} + \frac{d_k H^2 W^2}{n^2}) \approx \mathcal{O}(\frac{d_k H^2 W^2}{n^2})$. Since the attention is extracted from neighboring frames only, the resulting feature are also more robust to scene motion. We notice that recent work [60] also adopt pooling operation to achieve efficient attention models, but this is in the context of image semantic segmentation, while our model extends this strategy to deal with video data.



a) Teacher b) Student

Figure 4. The knowledge distillation. In the "Overall KD", we align the full outputs between the teacher model (e.g. PSPNet101) and the student model (e.g. out TDNet). In the "Grouped KD", we match the outputs based on only one sub-network to the teacher model's output conditioned on the respective feature subspace.

4. Grouped Knowledge Distillation

During training, we further enhance the complementarity of sub-feature maps in the full feature space by introducing a knowledge distillation [15] strategy, using a strong deep model designed for single images as the teacher network. In addition to transferring knowledge in the full-feature space [13, 15, 29], we propose a grouped knowledge distillation loss to further transfer knowledge at the subspace level in order to make the information extracted from different paths more complementary to one another.

The idea of a grouped distillation loss is illustrated in Fig. 4. We take a deep baseline model like PSPNet101 as the teacher, and take our TDNet with m sub-networks as the student network. The goal is to not only align the output distributions at the whole-model level, but also at a subfeature group level. Based on *block matrix multiplication* [9], we evenly separate the teacher model's feature reduction layer into m independent sub-convolution groups, which output a set of sub-feature groups $\{f_i | i = 1, ..., m\}$. Thus, the original segmentation result is $\pi_T(\sum f)$, and the contribution of the i-th feature group is $\pi_T(f_i)$, given $\pi_T(\cdot)$ being the teacher model's segmentation layer. In TDNet, the target frame's Value map V_m is combined with propagated previous information to be V'_m, thus the full model output is $\pi_S(V'_m)$ and the m-th feature path's contribution is $\pi_S(V_m)$, given $\pi_S(\cdot)$ is the final segmentation layers. Based on these, our final loss function is,

$$\begin{aligned} Loss =& CE(\pi_S(V'_i, gt)) + \alpha \cdot KL(\pi_S(V'_i) || \pi_T(\sum f)) \\ &+ \beta \cdot KL(\pi_S(V_i) || \pi_T(f_i)) \end{aligned} \quad (6)$$

where CE is the cross entropy loss, and KL means the KL-divergence. The first term is the supervised training with ground truth. The second term distills knowledge at the whole-model level. The third term transfers knowledge at feature group level. We set α and β to be 0.5 in our paper.

5. Experiments

We evaluate our method on Cityscapes [5] and Camvid [1] for street views, and NYUDv2 [33] for indoor scenes. On all of these datasets, our method achieves state-of-the-art accuracy with a much faster speed and lower and evenly distributed latency.

5.1. Setup and Implementation

Datasets & Evaluation Metrics. *Cityscapes* [5] contains 2,975/500/1,525 snippets for training/validation/testing. The 20^{th} frame of each snippet is annotated with 19 classes for semantic segmentation. *Camvid* [1] consists of 4 videos with 11-class pixelwise annotations at 1Hz. The annotated frames are grouped into 467/100/233 for training/validation/testing. *NYUDv2* [33] contains 518 indoor videos with 795 training frames and 654 testing frames being rectified and annotated with 40-class semantic labels. Based on these labeled frames, we create rectified video snippets from the raw Kinetic videos, which we will release for testing. Following the practice in previous works [10, 14, 19, 27], we evaluate mean Intersection-over-Union (mIoU) on Cityscapes, and mean accuracy and mIoU on Camvid and NYUDv2.

Models & Baselines. We demonstrate the effectiveness of TDNet on different backbones. We select two state-of-the-art image segmentation models for our experiments: PSPNet [56], and BiseNet* [52]. The latter is a modified/improved version of [52] with the *Spatial Path* being replaced with the output of ResBlock-2, which we found to have higher efficiency and better training convergence. We extend these image models with temporally distributed framework to boost the performance, yielding the models:
TD^2-PSP50, **TD^4-PSP18**: the former consists of two PSPNet-50 [56] backbones with halved output channels as sub-networks, whereas TD^4-PSP18 is made of four PSPNet-18 sub-networks. The model capacity of the temporally distributed models is comparable to the image segmentation network they are based on (PSPNet-101).
TD^2-Bise34, **TD^4-Bise18**. Similarly, we build TD^2-Bise34 with two BiseNet*-34 as sub-networks, and TD^4-Bise18 with four BiseNet*-18 as sub-networks for the real-time applications. Like in PSPNet case, the model capacity of the temporally distributed networks is comparable to the BiseNet*-101.

Speed Measurement & Comparison. All testing experiments are conducted with a batch-size of one on a single Titan Xp in the Pytorch framework. We found that previous methods are implemented with different deep-learning frameworks and evaluated on different types of devices, so for consistent comparisons, we report the speed/latency for

Method	mIoU(%) *val*	mIoU(%) *test*	Speed (ms/f)	Max Latency (ms)
CLK [39]	64.4	-	158	198
DFF [58]	69.2	-	156	575
GRFP(5) [34]	73.6	72.9	255	255
LVS-LLS [27]	75.9	-	119	119
PEARL [19]	76.5	75.2	800	800
LVS [27]	76.8	-	171	380
PSPNet18 [56]	75.5	-	91	91
PSPNet50 [56]	78.1	-	238	238
PSPNet101 [56]	79.7	79.2	360	360
TD^4-PSP18	76.8	-	**85**	**85**
TD^2-PSP50	**79.9**	79.4	178	178

Table 1. Evaluation on the Cityscapes dataset. The "Speed" and "Max Latency" represent the average and maximum per-frame time cost respectively.

these previous methods based on benchmark-based conversions[1] and our reimplementations.

Training & Testing Details. Both our models and baselines are initialized with Imagenet [6] pretrained parameters and then trained to convergence to achieve the best performance. To train TDNet with m subnetworks, each training sample is composed of m consecutive frames and the supervision is the ground truth from the last one. We perform random cropping, random scaling and flipping for data augmentation. Networks are trained by stochastic gradient descent with momentum 0.9 and weight decay 5e-4 for 80k iterations. The learning rate is initialized as 0.01 and decayed by $(1 - \frac{iter}{max-iter})^{0.9}$. During testing, we resize the output to the input's original resolution for evaluation. On datasets like Cityscapes and NYUDv2 which have temporally sparse annotations, we compute the accuracy for all possible orders of sub-networks and average them as final results. We found that different orders of sub-networks achieve very similar mIoU values, which indicates that TDNet is stable with respect to sub-feature paths (see supplementary materials).

5.2. Results

Cityscapes Dataset. We compare our method with the recent state-of-the-art models for semantic video segmentation in Table 1. Compared with LVS [27], TD^4-PSP18, achieves similar performance with only a half the average time cost, and TD^2-PSP50 further improves accuracy by 3 percent in terms of mIoU. Unlike keyframe-based methods like LVS [27], ClockNet [39], DFF [58] that have fluctuating latency between keyframes and non-key frames (e.g. 575ms v.s. 156ms for DFF [58]), our method runs with a balanced computation load over time. With a similar total number of parameters as PSPNet101 [56], TD^2-PSP50 reduces the per-frame time cost by half from 360ms to 178ms while improving accuracy. The sub-networks in TD^2-PSP50 are adapted from PSPNet50, so we also compare their performance, and can see that TD^2-PSP50 outperforms PSPNet50 by 1.8% mIoU with a faster average latency. As shown in the last row, TD^4-PSP18 can further reduce the latency to a quarter, but due to the shallow sub-networks (based on a PSPNet18 model), the performance drops comparing to PSPNet101. However, it still achieves state-of-the-art accuracy and outperforms previous methods by a large gap in terms of latency. Some qualitative results are shown in Fig. 5(a)

[1] http://goo.gl/N6ukTz/, http://goo.gl/BaopYQ/

Method	mIoU(%) *val*	mIoU(%) *test*	Speed (ms/f)
DVSNet [51]	63.2	-	33
ICNet [55]	67.7	69.5	**20**
LadderNet [21]	72.8	-	33
SwiftNet [36]	75.4	-	23
BiseNet*18 [52]	73.8	73.5	20
BiseNet*34 [52]	76.0	-	27
BiseNet*101 [52]	76.5	-	72
TD^4-Bise18	75.0	74.9	21
TD^2-Bise34	**76.4**	-	26

Table 2. Evaluation of high-efficiency approaches on the Cityscapes dataset.

To validate our method's effectiveness for more realistic tasks, we evaluate our real-time models TD^2-Bise34 and TD^4-Bise18 (Table 2). As we can see, TD^2-Bise34 outperforms all the previous real-time methods like ICNet [55], LadderNet [21], and SwiftNet [36] by a large gap, at a comparable, real-time speed. With a similar total model size to BiseNet*101, TD^2-Bise34 achieves better performance while being roughly three times faster. TD^4-Bise18 drops the accuracy but further improves the speed to nearly 50 FPS. Both TD^2-Bise34 and TD^4-Bise18 improve over their single path baselines at a similar time cost, which validates the effectiveness of our TDNet for real-time tasks.

Camvid Dataset. We also report the evaluation of Camvid dataset in Table 3. We can see that TD^2-PSP50 outperforms the previous state-of-the-art method Netwarp [10] by about 9% mIoU while being roughly four times faster. Comparing to the PSPNet101 baselines with a similar model capacity, TD^2-PSP50 reduces about half of the computation cost with comparable accuracy. The four-path version further reduces the latency by half but also decreases the accuracy. This again shows that a proper depth is necessary for feature path, although even so, TD^4-PSP18 still outperforms previous methods with a large gap both in terms of mIoU and speed.

NYUDv2 Dataset. To show that our method is not limited to street-view like scenes, we also reorganize the indoor NYUDepth-v2 dataset to make it suitable for seman-

Method	mIoU(%)	Mean Acc.(%)	Speed(ms/f)
LVS [27]	-	82.9	84
PEARL [19]	-	83.2	300
GRFP(5) [34]	66.1	-	230
ACCEL [18]	66.7	-	132
Netwarp [10]	67.1	-	363
PSPNet18 [56]	71.0	78.7	40
PSPNet50 [56]	74.7	81.5	100
PSPNet101 [56]	76.2	83.6	175
TD^4-PSP18	72.6	80.2	**40**
TD^2-PSP50	**76.0**	**83.4**	90

Table 3. Evaluation on the Camvid dataset.

Method	mIoU(%)	Mean Acc.(%)	Speed(ms/f)
STD2P [14]	40.1	53.8	>100
FCN [30]	34.0	46.1	56
DeepLab [3]	39.4	49.6	78
PSPNet18 [56]	35.9	46.9	19
PSPNet50 [56]	41.8	52.8	47
PSPNet101 [56]	43.2	55.0	72
TD^4-PSP18	37.4	48.1	**19**
TD^2-PSP50	**43.5**	**55.2**	35

Table 4. Evaluation on the NYUDepth dataset.

Overall-KD	Grouped-KD	Cityscapes	NYUDv2
		76.4	36.2
✓		76.5 (+0.1)	36.7 (+0.5)
✓	✓	**76.8 (+0.4)**	**37.4 (+1.2)**

Table 5. The mIoU (%) for different components in our knowledge distillation loss (Eq. 6) for TD^4-PSP18.

tic video segmentation task. As most previous methods for video semantic segmentation do not evaluate on this dataset, we only find one related work to compare against; STD2P [14]. As shown in Table 4, TD^2-PSP50 outperforms STD2P in terms of both accuracy and speed. TD^4-PSP18 achieves a worse accuracy but is more than 5× faster. TD^2-PSP50 again successfully halves the latency but keeps the accuracy of the baseline PSPNet101, and also achieves about 1.6% improvement in mIoU comparing to PSPNet18 without increasing the latency.

5.3. Method Analysis

Grouped Knowledge Distillation. The knowledge distillation based training loss (Eq. 6) consistently helps to improve performance on the three datasets. In order to investigate the effect of different components in the loss, we train TD^4-PSP18 with different settings and show the results in Table 5. The overall knowledge distillation [15] works by providing extra information about intra-class similarity and inter-class diversity. Thereby, it is less effective to improve a fully trained base model on Cityscapes due to the highly-structured contents and relatively fewer categories. However, when combined with our grouped knowledge distillation, the performance can be still boosted with nearly a half percent in terms of mIoU. This shows the effectiveness of our grouped knowledge distillation to provide extra regularization. On the NYUD-v2 dataset which contains more diverse scenes and more categories, our method achieves significant improvements with an 1.2% absolute improvement in mIoU.

Model		n=1	2	4	8	16	32
TD^2-PSP50	mIoU (%)	80.0	80.0	79.9	79.8	79.6	79.1
	latency (ms)	251	205	178	175	170	169
TD^4-PSP18	mIoU (%)	76.9	76.8	76.8	76.5	76.1	75.7
	latency (ms)	268	103	85	81	75	75
TD^4-Bise18	mIoU (%)	75.0	75.0	75.0	74.8	74.7	74.4
	latency (ms)	140	31	21	19	18	18

Table 6. Effect of different downsampling stride n on Cityscapes.

Framework	Single Path Baseline	Shared	Independent
TD^2-PSP50	78.2	78.5	**79.9**
TD^4-PSP18	75.5	75.7	**76.8**

Table 7. Comparisons on Cityscapes for using a shared sub-network or independent sub-networks. The last column shows the baseline model corresponding to TDNet's sub-network.

Attention Propagation Module. Here, we compare our attention propagation module (APM) with other aggregation methods such as: no motion compensation, e.g., just adding feature groups (Add), optical-flow based warping (OFW) and the vanilla Spatio-Temporal Attention (STA) mechanism [35, 49]. As shown in Fig. 6(a), without considering the spatial misalignment (Add) leads to the worst accuracy. Our APM outperforms OFW and STA in both accuracy and latency. In Fig. 6(b), we evaluate our method's robustness to motion between frames by varying the temporal step in input frames sampling. As shown in the figure, APM shows the best robustness, even with a sampling gap of 6 frames where flow based methods fail, our APM drops very slightly in contrast to other methods.

Attention Downsampling. In the downsampling operation used to improve the efficiency of computing attention, we apply spatial max pooling with a stride n. We show the influence of n in Table 6. By increasing n from 1 to 4, the computation is decreased drastically, while the accuracy is fairly stable. This indicates that the downsampling strategy is effective in extracting spatial information in a sparse way. However, while further increasing n to 32, the accuracy decreases due to the information being too sparse.

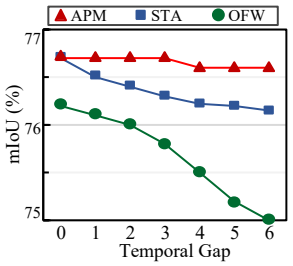
Shared Subnetworks v.s. Independent Subnetworks. When processing a video, the effectiveness of TDNet may come from two aspects: the enlarged representation capacity by distributed subnetworks and the temporal context information provided by neighboring frames. In Table 7, we analyze the contributions of each by using a single subnetwork used for each path, or a group of independent subnetworks. As we can see, aggregating features ex-

(a) Qualitative Results (b) Attention Visualization

Figure 5. Qualitative results of our method on Cityscapes and NYUD-v2 (a), and a visualization of the attention map in our attentive propagation network (b). Given a pixel in frame t (denoted as a green cross), we back-propagate the correlation scores with the affinity matrices, and then visualize the normalized soft weights as heat map over the other frames in the window.

Method	*mIoU (%)*	*Speed (ms/f)*
APM	**76.8**	85
STA	76.5	95
OFW	76.1	97
Add	64.8	**73**

(a) mIoU v.s. Speed



(b) Robustness to temporal variations

Figure 6. TD^4-PSP18 with different temporal aggregation methods on Cityscapes dataset. "APM" denotes our attention propagation module. "STA" represents spatio-temporal attention [35, 49]. "OFW" is the optical-flow [8] based fusion. "Add" means simply adding feature maps.

P1	P2	P3	P4	Cityscapes	NYUDepth-V2
✓	✓	✓	✓	**76.8**	**38.2**
	✓	✓	✓	76.5	38.0
		✓	✓	76.0	37.2
			✓	74.3	34.4

Table 8. Ablation study on TD^4-PSP18 showing how performance decreases with progressively fewer sub-features accumulated.

tracted via a shared single subnetwork can improve the performance of image segmentation baseline, and independent sub-networks can further improve mIoU by 1% without increasing computation cost. This shows that TDNet does not only benefit from the temporal context information but is also effectively enlarging the representation capacity by the temporally distributing distinct subnetworks.

Effect of Sub-networks. As shown in the last part, TDNet benefits from enforcing different sub-networks extract complementary feature groups. Here, we provide detailed ablation studies about the contributions of these sub-networks. Table 8 shows the analysis for TD^4-PSP18, where P4 represents the sub-network at the target frame, and P1∼P3 are the sub-networks applied on the previous frames. As we can see, by removing feature paths from the first frame, the accuracy consistently decreases for both datasets, which proves the effectiveness of feature distribution. To show how these paths are aggregated, in Fig 5(b) we visualize the attention maps of the attention propagation module in TD^4-PSP18. As shown in the figure, given a pixel (denoted as green crosses) in the target frame t, pixels of the corresponding semantic category in the previous frame t-1 are matched. However, in the previous frames t-2 and t-3, *background* pixels are collected. It should be noted that in the attention propagation module, there are layers ϕ (in Eq. 4 and Eq. 5) which process the aggregated features. Thus frames t-2 and t-3 provide contextual information, and frames t-1 and t provide local object information, which are combined together to form strong and robust features for segmentation.

6. Conclusion

We presented a novel temporally distributed network for fast semantic video segmentation. By computing the feature maps across different frames and merging them with a novel attention propagation module, our method retains high accuracy while significantly improving the latency of processing video frames. We show that using a grouped knowledge distillation loss, further boost the performance. TDNet consistently outperforms previous methods in both accuracy and efficiency.

Acknowledgements. We thank Kate Saenko for the useful discussions and suggestions. This work was supported in part by DARPA and NSF, and a gift funding from Adobe Research.

References

[1] Gabriel J Brostow, Jamie Shotton, Julien Fauqueur, and Roberto Cipolla. Segmentation and recognition using structure from motion point clouds. In *ECCV*, 2008. 5

[2] Joao Carreira, Viorica Patraucean, Laurent Mazare, Andrew Zisserman, and Simon Osindero. Massively parallel video networks. In *ECCV*, 2018. 2

[3] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE T-PAMI*, 2017. 7

[4] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 1, 2

[5] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 5

[6] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009. 6

[7] Henghui Ding, Xudong Jiang, Bing Shuai, Ai Qun Liu, and Gang Wang. Context contrasted feature and gated multi-scale aggregation for scene segmentation. In *CVPR*, 2018. 2

[8] Alexey Dosovitskiy, Philipp Fischer, Eddy Ilg, Philip Hausser, Caner Hazirbas, Vladimir Golkov, Patrick Van Der Smagt, Daniel Cremers, and Thomas Brox. Flownet: Learning optical flow with convolutional networks. In *ICCV*, 2015. 8

[9] Howard Whitley Eves. *Elementary matrix theory*. 1980. 3, 5

[10] Raghudeep Gadde, Varun Jampani, and Peter V Gehler. Semantic video cnns through representation warping. In *CVPR*, 2017. 1, 2, 3, 5, 6, 7

[11] Junjun He, Zhongying Deng, and Yu Qiao. Dynamic multi-scale filters for semantic segmentation. In *ICCV*, 2019. 2

[12] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2

[13] Tong He, Chunhua Shen, Zhi Tian, Dong Gong, Changming Sun, and Youliang Yan. Knowledge adaptation for efficient semantic segmentation. In *CVPR*, 2019. 5

[14] Yang He, Wei-Chen Chiu, Margret Keuper, and Mario Fritz. Std2p: Rgbd semantic segmentation using spatio-temporal data-driven pooling. In *CVPR*, 2017. 2, 5, 7

[15] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. 5, 7

[16] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In *CVPR*, 2017. 2

[17] Yani Ioannou, Duncan Robertson, Roberto Cipolla, and Antonio Criminisi. Deep roots: Improving cnn efficiency with hierarchical filter groups. In *CVPR*, 2017. 1, 2

[18] Samvit Jain, Xin Wang, and Joseph E Gonzalez. Accel: A corrective fusion network for efficient semantic segmentation on video. In *CVPR*, 2019. 1, 2, 3, 7

[19] Xiaojie Jin, Xin Li, Huaxin Xiao, Xiaohui Shen, Zhe Lin, Jimei Yang, Yunpeng Chen, Jian Dong, Luoqi Liu, Zequn Jie, et al. Video scene parsing with predictive feature learning. In *ICCV*, 2017. 1, 2, 5, 6, 7

[20] Haijie Tian Yong Li Yongjun Bao Zhiwei Fang and Hanqing Lu Jun Fu, Jing Liu. Dual attention network for scene segmentation. 2019. 1

[21] Ivan Kreso, Sinisa Segvic, and Josip Krapac. Ladder-style densenets for semantic segmentation of large natural images. In *ICCV Workshop*, 2017. 1, 6

[22] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012. 1, 2

[23] Abhijit Kundu, Vibhav Vineet, and Vladlen Koltun. Feature space optimization for semantic video segmentation. In *CVPR*, 2016. 1, 2

[24] Hanchao Li, Pengfei Xiong, Haoqiang Fan, and Jian Sun. Dfanet: Deep feature aggregation for real-time semantic segmentation. In *CVPR*, 2019. 2

[25] Xia Li, Zhisheng Zhong, Jianlong Wu, Yibo Yang, Zhouchen Lin, and Hong Liu. Expectation-maximization attention networks for semantic segmentation. *ICCV*, 2019. 1

[26] Yanwei Li, Xinze Chen, Zheng Zhu, Lingxi Xie, Guan Huang, Dalong Du, and Xingang Wang. Attention-guided unified network for panoptic segmentation. In *CVPR*, 2019. 2

[27] Yule Li, Jianping Shi, and Dahua Lin. Low-latency video semantic segmentation. In *CVPR*, 2018. 1, 2, 5, 6, 7

[28] Chenxi Liu, Liang-Chieh Chen, Florian Schroff, Hartwig Adam, Wei Hua, Alan L Yuille, and Li Fei-Fei. Auto-deeplab: Hierarchical neural architecture search for semantic image segmentation. In *CVPR*, 2019. 1, 2

[29] Yifan Liu, Ke Chen, Chris Liu, Zengchang Qin, Zhenbo Luo, and Jingdong Wang. Structured knowledge distillation for semantic segmentation. In *CVPR*, 2019. 5

[30] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 2, 7

[31] Behrooz Mahasseni, Sinisa Todorovic, and Alan Fern. Budget-aware deep semantic video segmentation. In *CVPR*, 2017. 1, 2

[32] Davide Mazzini. Guided upsampling network for real-time semantic segmentation. *BMVC*, 2018. 1

[33] Pushmeet Kohli Nathan Silberman, Derek Hoiem and Rob Fergus. Indoor segmentation and support inference from rgbd images. In *ECCV*, 2012. 5

[34] David Nilsson and Cristian Sminchisescu. Semantic video segmentation by gated recurrent flow propagation. In *CVPR*, 2018. 1, 2, 3, 6, 7

[35] Seoung Wug Oh, Joon-Young Lee, Ning Xu, and Seon Joo Kim. Video object segmentation using space-time memory networks. *ICCV*, 2019. 3, 7, 8

[36] Marin Orsic, Ivan Kreso, Petra Bevandic, and Sinisa Segvic. In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In *CVPR*, 2019. 2, 6

[37] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint*

arXiv:1606.02147, 2016. 2

[38] Chao Peng, Xiangyu Zhang, Gang Yu, Guiming Luo, and Jian Sun. Large kernel matters–improve semantic segmentation by global convolutional network. In *CVPR*, 2017. 2

[39] Evan Shelhamer, Kate Rakelly, Judy Hoffman, and Trevor Darrell. Clockwork convnets for video semantic segmentation. In *ECCV*, 2016. 1, 2, 6

[40] Bing Shuai, Zhen Zuo, Bing Wang, and Gang Wang. Dag-recurrent neural networks for scene labeling. In *CVPR*, 2016. 2

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *ICLR*, 2015. 2

[42] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *AAAI*, 2017. 2, 3

[43] Towaki Takikawa, David Acuna, Varun Jampani, and Sanja Fidler. Iccv. 2019. 2

[44] Meng Tang, Abdelaziz Djelouah, Federico Perazzi, Yuri Boykov, and Christopher Schroers. Normalized cut loss for weakly-supervised CNN segmentation. In *CVPR*, 2018. 2

[45] Meng Tang, Federico Perazzi, Abdelaziz Djelouah, Ismail Ben Ayed, Christopher Schroers, and Yuri Boykov. On regularized losses for weakly-supervised CNN segmentation. In *ECCV*, 2018. 2

[46] Subarna Tripathi, Serge Belongie, Youngbae Hwang, and Truong Nguyen. Semantic video segmentation: Exploring inference efficiency. In *ISOCC*. IEEE. 2

[47] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NIPS*, 2017. 3

[48] Andreas Veit, Michael J Wilber, and Serge Belongie. Residual networks behave like ensembles of relatively shallow networks. In *NIPS*, 2016. 3

[49] Xiaolong Wang, Ross Girshick, Abhinav Gupta, and Kaiming He. Non-local neural networks. In *CVPR*, 2018. 3, 7, 8

[50] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 2019. 1, 2, 3

[51] Yu-Syuan Xu, Tsu-Jui Fu, Hsuan-Kung Yang, and Chun-Yi Lee. Dynamic video segmentation network. In *CVPR*, 2018. 6

[52] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation. In *ECCV*, 2018. 2, 5, 6

[53] Sergey Zagoruyko and Nikos Komodakis. Wide residual networks. In *BMVC*, 2016. 2, 3

[54] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *CVPR*, 2018. 1

[55] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *ECCV*, 2018. 1, 2, 6

[56] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1, 2, 3, 5, 6, 7

[57] Xizhou Zhu, Dazhi Cheng, Zheng Zhang, Stephen Lin, and Jifeng Dai. An empirical study of spatial attention mechanisms in deep networks. *ICCV*, 2019. 3

[58] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *CVPR*, 2017. 1, 2, 3, 6

[59] Yi Zhu, Karan Sapra, Fitsum A. Reda, Kevin J. Shih, Shawn Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In *CVPR*, 2019. 1

[60] Zhen Zhu, Mengde Xu, Song Bai, Tengteng Huang, and Xiang Bai. Asymmetric non-local neural networks for semantic segmentation. In *ICCV*, 2019. 4