TempNet: Online Semantic Segmentation on Large-scale Point Cloud Series

Yunsong Zhou [1] Hongzi Zhu [1] * Chunqin Li [1] Tiankai Cui [1] Shan Chang [2] * Minyi Guo [1]
[1]Shanghai Jiao Tong University [2]Donghua University
{zhouyunsong,hongzi,supermelcq,cuitiankai,guo-my}@sjtu.edu.cn changshan@dhu.edu.cn

Abstract

Online semantic segmentation on a time series of point cloud frames is an essential task in autonomous driving. Existing models focus on single-frame segmentation, which cannot achieve satisfactory segmentation accuracy and offer unstably flicker among frames. In this paper, we propose a light-weight semantic segmentation framework for large-scale point cloud series, called TempNet, which can improve both the accuracy and the stability of existing semantic segmentation models by combining a novel frame aggregation scheme. To be computational cost-efficient, feature extraction and aggregation are only conducted on a small portion of key frames via a temporal feature aggregation (TFA) network using an attentional pooling mechanism, and such enhanced features are propagated to the intermediate non-key frames. To avoid information loss from non-key frames, a partial feature update (PFU) network is designed to partially update the propagated features with the local features extracted on a non-key frame if a large disparity between the two is quickly assessed. As a result, consistent and information-rich features can be obtained for each frame. We implement TempNet on five state-of-the-art (SOTA) point cloud segmentation models and conduct extensive experiments on the SemanticKITTI dataset. Results demonstrate that TempNet outperforms SOTA competitors by wide margins with little extra computational cost.

1. Introduction

In order to better perceive the driving context, most automated vehicles are equipped with LiDAR sensors to continuously acquire point cloud data. The performance of an online semantic segmentation algorithm for a time series of point cloud frames referred to as *point cloud series*, is essential for an automated vehicle to make correct decisions in real-time. For example, a commercial off-the-shelf (COTS) LiDAR with 128 channels [7] could produce 10 frames per second with each frame containing a large number of around 480,000 points. Due to the discrete and

*Co-corresponding authors

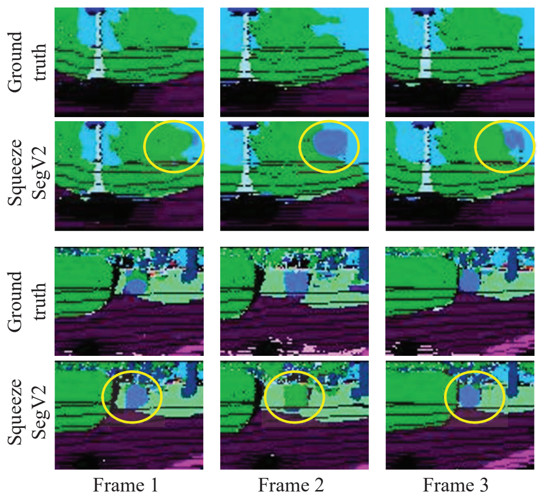

Figure 1. Segmentation examples of using SequeezeSegV2, a SOTA single-frame segmentation method, on two sequences of three consecutive point cloud frames. It can be seen that applying such schemes to point cloud series leads to unstable and inaccurate results, e.g., errors denoted by circles suddenly appear.

sparse spatial distribution nature of point cloud data, semantic segmentation on large-scale point cloud series is a more difficult task compared with semantic segmentation of videos. Given the restricted computational power of an automated vehicle, a practical semantic segmentation method for point cloud series should meet the following two requirements. First, the segmentation results should be accurate so that automated vehicles can make correct driving decisions based on the results. Second, the method should be real-time, which means that any frame of point cloud in the series should be correctly segmented within a certain bounded time. Otherwise, the returned results may become invalid or useless.

In the literature, a number of point cloud semantic segmentation methods [27, 22, 16] have been proposed, which mainly focus on one single and static point cloud frame and may generate inconsistent segmentation results when dealing with consecutive frames of point clouds. For instance, as shown in Figure 1, when using SequeezeSegV2 [31], a state-of-the-art (SOTA) single-frame point cloud segmentation scheme, segmentation errors (illustrated by dashed circles in the figure) may suddenly appear and flicker among a series of consecutive frames. Moreover, these schemes are too computationally expensive to process point cloud series. As a result, there is no existing successful point cloud semantic segmentation scheme, to the best of our knowledge, that can handle the online point-cloud-series semantic segmentation problem.

In this paper, we propose a light-weight point-cloud-series semantic segmentation framework, called *TempNet*, which can improve both the accuracy and the efficiency of existing semantic segmentation models by combining a novel frame aggregation scheme. Inspired by the work [11, 37] in video segmentation tasks, which selectively conducts feature fusion to eliminate flickers, features of previous point cloud frames and those of the current frame are first efficiently obtained and then effectively aggregated to achieve reliable and accurate segmentation results.

There are two main challenges in designing TempNet. First, to obtain features of each point cloud frame for aggregation is of prohibitive computational cost for online point-cloud-series segmentation tasks. To tackle this challenge, full feature extraction and aggregation are only conducted a small portion of frames, referred to as *key frames*, and such enhanced features are directly propagated to intermediate non-key frames. Furthermore, to avoid possible information loss from non-key frames, a *partial feature update* (PFU) network is designed to partially update the propagated features with the features extracted on a non-key frame if a large disparity between the two is quickly assessed. Due to the complexity of real-world scenarios, it is hard to obtain an optimal key frame selection strategy. We take an *adaptive frame scheduling* (AFS) method to dynamically determine the number of key frames according to the disparities assessed in recent non-key frames.

Second, given previous point cloud frames and their features, how to augment the features of the current key frame to improve the stability of segmentation, however, is non-trivial. As to point clouds, when objects are far or surrounded by interfering objects, it is hard to guarantee the robustness of single-frame segmentation schemes. We have the observation, referred to as *local spatial consistency* of point clouds, that the local spatial structure of an object in the point cloud should be consistent between frames although the object might be moving. Leveraging the local spatial consistency of point clouds, we design a *temporal feature aggregation* (TFA) network based on graph attention convolution to effectively aggregate features of consecutive frames. Specifically, TFA prefers to search for neighboring keypoints with similar geometric characteristics and semantic features in the previous frame. Moreover, an attention mechanism is adopted in TFA so that spatially consistent features would be more impactful in the aggregation.

We implement TempNet on five state-of-the-art (SOTA) point cloud segmentation models, i.e., PointNet++ [20], GACNet [29], SequeezeSegV2 [31], DarkNet53Seg [2] and RandLA-Net [5]. We conduct extensive experiments on the SemanticKITTI dataset. Results demonstrate that TempNet outperforms SOTA competitors by wide margins with little extra computational cost. We highlight the main contributions made in this paper as follows: 1) an online point cloud series semantic segmentation framework TempNet is proposed, which is light-weight and easy to implement on existing single-frame segmentation schemes; 2) a temporal feature aggregation network is designed, which utilizes the continuity of motions and attentional pooling to effectively aggregate two point cloud frames in motion; 3) extensive experiments on the real-world SemanticKITTI dataset are conducted and results demonstrate the efficacy of TempNet. The whole suite of codebase will be released.

2. Related Work

2.1. Semantic Segmentation on Point Clouds

Due to the sparsity and disorder of point cloud data, directly applying existing dense calculation methods customized for images to the point cloud semantic segmentation problem would cause a prohibitive computational complexity of $O(n^3)$. Therefore, although a set of methods have been proposed to process point clouds, there are still divergences on how to efficiently utilize point cloud data. Existing models can be mainly divided into three categories, i.e., voxel-based, projection-based, and PointNet-like.

Voxel-based methods. To mitigate the desperate need for huge computational power in conducting 3D convolution, a straightforward way [13, 30, 23, 10] is to partition the point cloud space into a regular grid and apply 3D convolution on each voxel. OctNet [23] and Kd-Net [10] skip the computation of empty voxels and focus on the resolution of information-rich voxels. PointGrid [13] proposes hybrid 3D shape representation to solve the problem of point cloud sparsity and high space consumption. RandLA-Net [5] greatly reduces the computational effort of 3D convolution by rasterizing the local point cloud and reducing the number of uninteresting points. These sparse 3D convolutions can accelerate the convolution operations and share the knowledge base with dense convolutions. However, during the partitioning process, local spatial information between different voxels may be lost, and the computational overhead grows cubically with the resolution of voxels.

Projection-based methods. Projection-based methods are developed from 2D semantic segmentation. DarkNet53Seg [2] and other methods [25, 12, 8] perform projections to transform point clouds into the 2D plane from multiple viewpoints, e.g., front view, top view or spherical view. The projected plane is then processed by image semantic segmentation network. The projection-based methods reach the real-time requirements(SqueezeSegV2

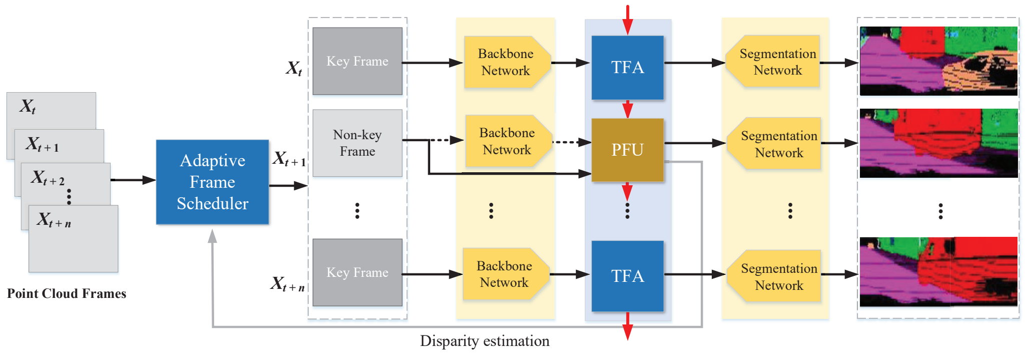

Figure 2. Overview of TempNet architecture. TempNet can be implemented on existing single-frame point cloud segmentation models (denoted as the backbone network and the segmentation network in this figure). In addition, it consists of three components to particularly tackle large-scale point cloud series, i.e., adaptive frame scheduler (AFS), temporal feature aggregation (TFA) network, and partial feature update (PFU) network. The AFS dynamically divides frames into key and non-key frames. For key frames, features are extracted via the backbone network and aggregated with previous features via the TFA network. For non-key frames, features are mainly inherited from previous features except that a large point disparity is detected, in which case the PFU network is used to partially update features. The goal of both PFU and TFA is to keep the features of each frame up-to-date in a cost-efficient way before they are individually segmented.

[31] reaches 13.5ms/per frame) while the final performance of projection-based methods is typically lower than other methods.

PointNet-like methods. PointNet-like methods are the extension of PointNet [19]. First, these methods input the coordinates and RGB features of the original point cloud data directly into the network. Then each point is processed individually using a shared MLP, which makes these methods limited in extracting local spatial relationships. To overcome this deficiency, PointNet++ [20] integrates neighborhoods by sampling and grouping and applies a hierarchical feature learning framework to learn different levels of local-global features. PointCNN [14] and GACNet [29] combines convolution and neighbor weights to fully extract spatial information. KPConv [28] proposes a new point cloud convolution based on kernel point, which achieves state-of-the-art performance in PointNet-like methods.

2.2. Temporal Feature Extraction

Currently, temporal convolutional networks can be applied only to 2D videos to capture temporal features. Meng *et al.* [17] propose QST-CNN for video detection. Sun *et al.* [26] design FstCN to identify human behavior in video. Zhu *et al.* [36] use TORNADO for target detection, or counting the flow of people in video [35, 18]. Yao *et al.* use DMVST-Net to predict traffic conditions [33]; and Li *et al.* [15] and Yu [34] use temporal convolutional networks to estimate traffic flow. Tao Song *et al.* [24] propose a novel method integrated with somatic topological line localization and temporal feature aggregation for detecting multi-scale pedestrians. Fei He *et al.* [21] propose a temporal context enhanced network to exploit temporal context information by temporal aggregation for video object detection, aligning the spatial features from frame to frame. [6] introduces ”tracking-by-detection” into Video Object Segmentation by proposing a new temporal aggregation network and a novel dynamic time-evolving template matching mechanism to achieve significantly improved performance. [4] and [32] design an adaptive feature aggregation method for video object detection to deal with blurring, occluding and out of focus in videos. In summary, all these methods focus on target detection on 2D images, but they cannot be directly applied to 3D point cloud series because of the discontinuity and disorder nature of point clouds. Simply extending the dimension of convolution function from 2D to 4D has a high computational complexity and cannot meet the real-time requirement of online tasks.

Recently, 4D temporal feature extraction has received great attention, which can be used to deal with point cloud series. MinkowskiNet [3] extends the dimension of convolution function from 2D to 4D. OpenPose [9] uses point cloud series to track posture of human hands in real-time. The idea of PointFlowNet [1] is similar to PointNet, which combines the two features of frame t and $t-1$ to infer the motion of each point. However, the scalability of such schemes is restricted as the number of points in each frame and that of frames in the series increase.

3. Design of TempNet

3.1. Overview

The core idea of TempNet is to utilize the temporal correlation between consecutive point cloud frames to neatly aggregate features so that two goals can be achieved: 1) the features of each frame are enhanced after the aggregation, making segmentation more accurate; 2) features between frames are consistent, eliminating flickers. Specifically, full feature extraction and aggregation are only conducted on a small portion of key frames and such enhanced features are directly propagate to intermediate non-key frames. Meanwhile, the propagated features would be partially updated on a non-key frame if an assessment via a light-weight disparity estimator believes that this non-key frame contains non-negligible information. To this end, as illustrated in Figure 2, TempNet integrates four technical components as follows:

Adaptive frame scheduler (AFS). The AFS is used to determine the selection of key frames and non-key frames. It is hard to obtain an optimal key frame selection strategy that fits in all real-world scenarios. In TempNet, the ratio of key frames is dynamically tuned according to the extent of disparity observed in recent non-key frames.

Static segmentation module. TempNet employs a SOTA static point cloud segmentation scheme as its underlying segmentation core. Such a scheme normally consists of a deep and slow pre-trained backbone network $\mathcal{N}_{fea}$ for feature extraction and a detection network $\mathcal{N}_{det}$ of multiple output branches for semantic segmentation. With $\mathcal{N}_{fea}$, hundreds of thousands of points in one frame are sampled and encoded into a small number of so-called *keypoints* and their corresponding feature vectors, which contain rich spatial information and portray the skeleton of the point cloud. Letting $X_i = \{x_i, x_2, ..., x_N\}$ denote the i-th frame with N points, we have $P_i, H_i = \mathcal{N}_{fea}(X_i)$, where $P_i = \{p_1, p_2, ..., p_n\}$ and $H_i = \{h_1, h_2, ..., h_n\}$ are the sets of locations of n keypoints and the corresponding feature vectors, respectively. With $\mathcal{N}_{det}$, segmentation result can be obtained, i.e., $Y_i = \mathcal{N}_{det}(P_i, H_i)$, where $Y_i = \{y_1, y_2, ..., y_N\}$ is the set of semantic labels for each point in the i-th frame.

Temporal feature aggregation (TFA) network. The TFA network is designed to utilize the temporal correlation between frames to aggregate features on key frames. The neighboring locations and features of two consecutive frames are first measured and then used to calculate attention scores for neighbors. Such attention scores are used as summation weights to aggregate neighboring keypoints sampled from both features so that those *local spatial consistent* keypoints contribute more to the aggregation.

Partial feature update (PFU) network. The PFU network is designed to selectively update inherited features from the previous frame with important local keypoints identified on the current non-key frame if necessary. A spatial consistency estimator is devised to help make a quick update decision at a low computational cost.

Figure 3 illustrated the main idea of TempNet in dealing with large-scale point cloud series. It can be seen that the TempNet takes both the feature consistency and computational cost into account when conducting feature aggregation.

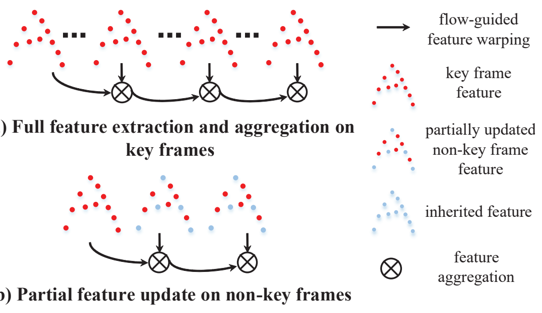
(a) Full feature extraction and aggregation on key frames
(b) Partial feature update on non-key frames
flow-guided feature warping
key frame feature
partially updated non-key frame feature
inherited feature
feature aggregation

Figure 3. An illustration of TempNet in dealing with point cloud series segmentation, where (a) full feature extraction and aggregation are conducted on key frames and (b) partial feature update and aggregation are conducted on non-key frames.

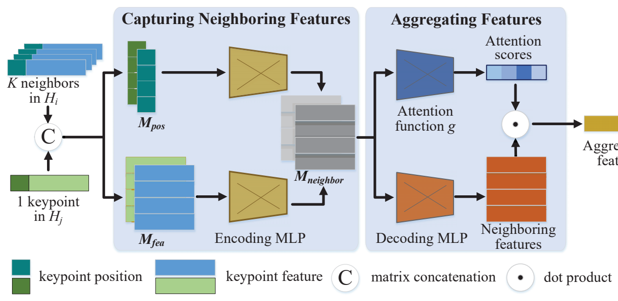

Figure 4. The structure of the TFA module. TFA takes each keypoint in frame j and their corresponding K neighboring keypoints in frame i as input. Local spatial consistency of point cloud data is leveraged to assign attention scores to neighboring keypoints before they are aggregated.

3.2. Temporal Feature Aggregation

Despite the motion of objects, their local spatial structure should be consistent between frames, which we refer to as *local spatial consistency* of point clouds. For instance, points (and their corresponding features) of a moving car in two point cloud frames should form similar shapes. We utilize local spatial consistency of point clouds to augment point cloud features on key frames so that the ultimate segmentation results could be effectively improved.

Specifically, for two consecutive key frames i and j, the aggregated features for frame j can be calculated as

$$\bar{H}_j = \alpha \cdot \mathcal{G}_{agg}(H_i, H_j) + (1 - \alpha) \cdot H_j, \quad (1)$$

where α is the artificially specified regularization weight; $\mathcal{G}_{agg}$ is the temporal feature aggregation network; H_i and H_j are the sets of feature vectors of frame i and frame j, respectively, obtained with $\mathcal{N}_{fea}$.

In designing $\mathcal{G}_{agg}$, we first search for neighboring keypoints in the previous frame, taking the position of a keypoint in the current frame as the origin. Then, we assign attention scores to those neighboring keypoints so that consistent keypoints get higher weights in the aggregation.

Capturing Neighboring Features in the Previous Frame. As depicted in Figure 4, keypoints of two consecutive frames i and frame j are fed into the temporal feature aggregation network $\mathcal{G}_{agg}$. For each keypoint $p_j \in P_j$, we encode its neighboring features attached with motion information between two frames into a representation matrix $M_{neighbor}(p_j)$.

More specifically, we encode the regularized positions and feature vectors separately. To encode the regularized positions, the neighboring keypoints of p_j in P_i, denoted as $\mathbb{N}_{P_i}(p_j)$, are collected using a KNN algorithm and their relative positions are recorded to construct a matrix:

$$\begin{aligned} M_{pos}(p_j) &= \text{mlp}\left(P_{near} \parallel p_j\right), \\ P_{near} &= \{p_l\}, \text{for } p_l \in \mathbb{N}_{P_i}(p_i), \end{aligned} \tag{2}$$

where the symbol $\parallel$ stands for matrix splicing and expanding vectors into matrices when needed. Similarly, we encode the feature vectors and construct a corresponding matrix of feature vectors of these neighboring keypoints, denoted as $M_{fea}(p_j)$. Ultimately, we concatenate both matrixes to construct the representation matrix,

$$M_{neighbor}(p_j) = \left(M_{pos}(p_j) \parallel M_{fea}(p_j)\right). \tag{3}$$

In this way, both position changes caused by motion and semantic features of previous neighboring keypoints, are captured in $M_{neighbor}(p_j)$, characterizing the relationship between p_j and its neighbors in the previous frame, i.e., $\mathbb{N}_{P_i}(p_j)$.

Aggregating Neighboring Features. As some keypoints in $\mathbb{N}_{P_i}(p_j)$ may not be local spacial consistent with p_j, simply applying the general max/mean pooling for hard integration of neighboring points would lead to inaccurate results. In contrast, we adopt a powerful attentional mechanism to distinguish which neighboring points should have more influence on the current keypoint. We adopt a similar attentional pooling unit as introduced in the work [5]. More specifically, given the matrix $M_{neighbor}(p_j) = \left\{m^1_{p_j}, m^2_{p_j}, \ldots, m^k_{p_j}, \ldots, m^K_{p_j}\right\}$ where K is the size of $\mathbb{N}_{P_i}(p_j)$, we use a shared function $g(\cdot)$ to learn the unique attention score for each neighbor point. Basically, the function consists of a weight-shared *mlp* and *softmax* function, defined as

$$s^k_{p_j} = g\left(m^k_{p_j}, W\right), \tag{4}$$

where W is the learnt weight of the shared *mlp*, $s^k_{p_j}$ is the attention score of the k-th neighbor of keypoint p_j.

Then, the learned attention scores are used as a soft mask, which effectively increase the impact of local-spatial-consistent keypoints in the previous frame. The weighted summation of $m^k_{p_j}$ for $k \in [1, K]$ is as follows:

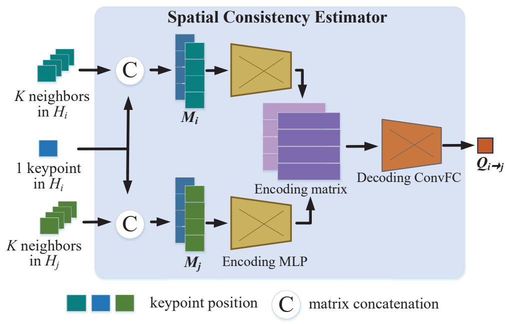

Figure 5. The structure of the spatial consistency estimator used in the PFU module. It takes each keypoint in the previous frame i and the corresponding neighboring keypoints in frame i and the current frame j as input. The estimator judges the geometric similarity of both neighborhoods to derive the consistency estimate $Q_{i\to j}$.

$$\tilde{m}_{p_j} = \sum_{k=1}^{K}\left(m^k_{p_j} \cdot s^k_{p_j}\right). \tag{5}$$

The updated feature vector of p_j can be obtained by $\text{mlp}\left(\tilde{m}_{p_j}\right)$ using the weight-shared *mlp*.

3.3. Partial Feature Updating

As to point cloud series, on one hand, points in consecutive frames share redundant information, which makes reusing some features from previous frames for non-key frames possible and therefore can greatly reduce computational overhead. On the other hand, significant feature changes caused by motion on these non-key frames should also be considered so that no important information is lost.

Instead of using heavy $\mathcal{N}_{fea}$ to calculate P_j and H_j for a non-key frame j, we use a lightweight random sampling algorithm to calculate only P_j. To cost-efficiently quantify whether features H_i passed from the previous frame i are a good approximation of the current frame j, we introduce a *spatial consistency estimator* $Q_{i\to j}$, defined as

$$Q_{i\to j}(p_i) = \mathcal{G}_{cons}\left(p_i, P_i, P_j\right), \tag{6}$$

where $p_i \in P_i$ and $\mathcal{G}_{cons}$ is the spatial coherence measurement network. For each $p_i \in P_i$, $\mathcal{G}_{cons}$ examines the similarity of its local spatial features in frame i and frame j. If $Q_{i\to j}(p_i) \geq \tau$ (τ is a consistency threshold), it indicates that the previous feature h_i should be consistent from frame i to j, so the location p_i and feature h_i can be reused in frame j; otherwise, h_i is abandoned.

We depict the design of $\mathcal{G}_{cons}$ in Figure 5. $\mathcal{G}_{cons}$ takes keypoint locations of two consecutive frames, e.g., P_i and P_j, and $p_i \in P_i$ as input. It should be noted that only the respective neighborhoods of p_i in P_i and P_j are compared, leading to low computational overhead. More specifically, we first construct a local-spatial-information encoding matrix of p_i in P_i as

$$\begin{aligned} M_i(p_i) &= P'_{near} \parallel p_i, \\ P'_{near} &= \{p'_l\} \text{ for } p'_l \in \mathbb{N}_{P_i}(p_j), \end{aligned} \tag{7}$$

where $\mathbb{N}_{P_i}(p_i)$ is the neighboring keypoints of p_i in P_i found with the same KNN algorithm as described in (2). Thus, $M_i(p_i)$ encodes the spatial relationship between p_i and its neighboring keypoints in frame i. Similarly, we construct a local-spatial-information encoding matrix $M_j(p_i)$ with respect to frame j. Given matrices $M_i(p_i)$ and $M_j(p_i)$, we construct *ConvFC* layers to predict the consistency estimator,

$$Q_{i\to j}(p_i) = \mathrm{ConvFC}\left(M_i(p_i) \parallel M_j(p_i)\right). \tag{8}$$

In addition, the focal loss is used to make it easier to learn the $Q_{i\to j}$.

Since only a fraction of keypoints in P_j could get reusable features from the previous frame, we still rely on the static point cloud segmentation backbone $\mathcal{N}_{fea}$ to extract feature vectors for the rest of keypoints in P_j. Nevertheless, as the number of such keypoints is significantly reduced, the computational cost is affordable for online point cloud series segmentation.

3.4. Adaptive frame scheduling

In addition to being used in the partial feature update, the consistency estimates in recent non-key frames are also used to dynamically determine the interval of key frames.

Specifically, to decide whether we need to increase the key frame interval (fewer frames are considered key frames) or decrease the key frame interval (more frames are considered key frames), the ratio of the number of updated keypoints to the total number of keypoints $r_{k\to i}$ is used and calculated as

$$r_{i\to j} = \frac{\sum_{p\in P_i} I(Q_{i\to j}(p_i) \le \tau)}{n_i}, \tag{9}$$

where frame i is the previous frame and n_i is the number of keypoints in frame i. If $r_{i\to j}$ is large, it means that there are a large portion of keypoints that are no longer similar to the last key frame and the number of key frames should be increased; otherwise, the number of key frames should be reduced to save computational cost.

The overall algorithm of TempNet is summarized in the supplementary materials.

4. Performance Evaluation

4.1. Implementation Details

We have implemented TempNet on five SOTA single-frame segmentation models, i.e., PointNet++ [20], GAC-Net [29], SqueezeSegV2 [31], DarkNet53Seg [2] and RandLA-Net [5]. We train and test TempNet and other candidate schemes using the SemanticKITTI [2] dataset, which consists of multiple sequences of 43552 densely annotated LIDAR scans (frames). Each scan is a large-scale point cloud with about 10^5 points distributed in a 3D space of $160\times160\times20$ meters. Raw 3D points contain 3D coordinates without color information. We use sequences indexed from 00 to 07 (16,338 scans), 08 (2,792 scans) and 09-10 (19,130 scans) for training, validation and testing, respectively.

During training, due to memory limitations, two adjacent frames are randomly selected from each small batch. The first frame is set as a key frame and the second frame is set as a non-key frame. During the forward pass, the feature network $\mathcal{N}_{fea}$ acts on P_i to get the feature vector space H_i. Then, the feature consistency network $\mathcal{G}_{cons}$ acts on P_i and P_j to get the feature consistency $Q_{i\to j}$. The partially updated feature vector space $\bar{H}_i$ is computed according to (8) and the current-frame feature vector space of temporal feature aggregation is computed according to (1). Finally, the result of semantic segmentation is obtained by semantic segmentation subnetwork $\mathcal{N}_{det}$ processing. During training, we enforce a probability of 1/3 and 2/3 for $Q_{i\to j} \le \tau$ and $Q_{i\to j} \ge \tau$ respectively, to encourage good performance in both cases of feature propagation and feature recalculation from scratch. For methods that do not use partial feature updates, training is not changed and $Q_{i\to j}$ is simply ignored during inference.

4.2. Methodology

We consider the following candidate point cloud series segmentation methods for comparison: **1) Baseline methods:** the aforementioned five single-frame segmentation schemes are implemented and trained; **2) Dense feature aggregation (DFA) methods:** A naive aggregation scheme where features of all frames are first extracted and then weighted summation is applied to aggregate these features, is applied to single-frame segmentation methods. **3) Direct feature propagation (DFP) methods:** Similar to TempNet except that no partial feature updates are conducted for non-key frames. **4) $*^1$:** it denotes the implementation of TempNet on a specific baseline method with all frames treated as non-key frames. **5) $*^2$:** it denotes the full implementation of TempNet on a specific baseline method where the key and non-key frames are automatically determined by the adaptive frame scheduler;

We take the mean-Intersection-over-Union (mIoU) and accuracy, defined as TP/(TP+FP), of all 19 categories as the standard performance metrics.

4.3. Performance Comparison

The running time and mIoU scores of each segmentation method are shown in Table 1. It can be seen that $*^2$

[c]Methods	Running time	mIoU(%)	Road	Sidewalk	Parking	Other-ground	Building	Car	Truck	Bicycle	Motorcycle	Other-vehicle	Vegetation	Trunk	Terrain	Person	Bicyclist	Motorcyclist	Fence	Pole	Traffic-sign
[c]PointNet++	328	20.5	72.0	41.8	18.7	5.6	62.3	62.3	0.9	1.9	0.2	0.2	46.5	13.8	30	0.9	1.0	0.0	16.9	5.0	8.9
[c]PointNet++*[2]	375	**26.9**	**80.9**	**53.9**	**20.4**	**15.4**	**68.4**	**73.4**	**5.3**	**2.7**	**6.5**	**2.1**	**59.5**	**19.3**	**38.1**	**3.0**	**8.4**	0.0	**29**	**5.8**	**18.5**
[c]GACNet	508	28.8	85.4	54.3	26.9	4.5	57.4	59.4	3.3	16.0	4.1	3.6	60.0	24.3	53.7	12.9	13.1	0.9	29.0	17.5	24.5
[c]GACNet*[2]	643	**32.8**	**88.8**	**55.2**	**32.1**	**12.3**	**60.2**	**62.1**	**13.2**	**26**	**11.5**	**7.2**	**61.7**	**25.2**	**55.2**	**13.0**	**14.9**	**2.2**	**34.3**	**22.7**	**25.6**
[c]SqueezeSegV2	1236	39.7	88.6	67.6	45.8	17.7	73.7	71.7	13.4	18.5	17.9	14.0	**71.8**	35.8	60.2	20.1	25.1	3.9	41.1	20.2	36.3
[c]SqueezeSegV2*[2]	1457	**44.4**	**90.4**	**68.9**	**57**	**26.4**	**82**	**81.9**	**18.6**	**26.2**	**26.5**	**15.6**	67.6	**48.4**	**60.6**	**21.8**	**33.6**	**4.0**	**52.3**	**22**	**40.0**
[c]DarkNet53Seg	534	49.3	89.6	74.0	34.7	4.0	87.4	90.9	67.0	9.8	18.4	27.3	**87.4**	43.8	**83.7**	44.0	45.3	0.0	**43.8**	**49.6**	36.1
[c]DarkNet53Seg*[2]	601	**52.5**	**90.5**	**75.2**	**37.4**	**4.4**	**87.7**	**92.8**	**67.9**	**12.8**	**26.8**	**40.0**	85.3	**59.5**	75.0	**50.2**	**62.2**	**8.1**	42.9	48.6	**36.7**
[c]RandLA-Net	772	52.4	90.5	**75.3**	60.3	20.4	85.9	92.9	39.1	13.2	26.1	38.2	81.4	61.1	64.9	**48.8**	**47.6**	6.8	55.4	47.8	47.7
[c]RandLA-Net*[2]	829	**55.8**	**90.7**	74.1	**61.8**	**24.4**	**89.8**	**94.1**	**43.8**	**26.1**	**32.2**	**39.2**	**83.9**	**63.7**	**68.7**	48.3	47.4	**9.5**	**60.3**	**51.1**	**50.8**

Table 1. IoU scores on the SemanticKITTI dataset. TempNet generally can improve the performance of the underlying segmentation method in terms of accuracy and efficiency.

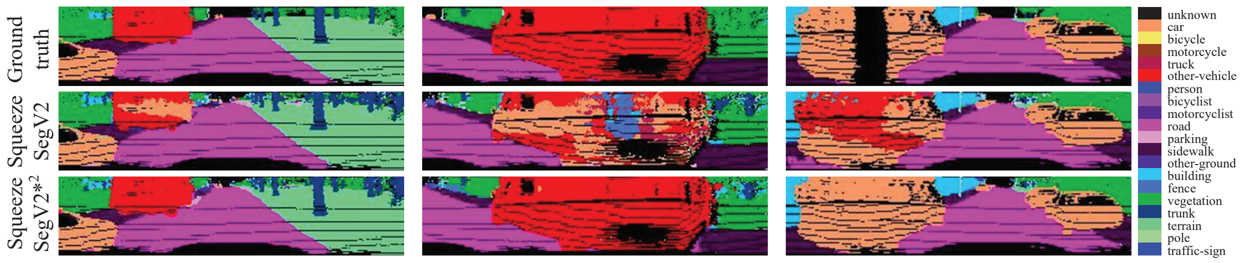

Figure 6. Qualitative example results using SqueezeSegV2 as the underlying segmentation method. Point clouds are visualized after projection. Segmentation accuracy is improved, especially for continuous surfaces that are subordinate to the same moving object.

methods generally can achieve higher scores in most categories than the corresponding **baseline** methods at a moderate computational cost. Particularly, our method can significantly improve the mIoU for categories with more continuous shapes and flatter surfaces, such as car, truck, and building. For example, PointNet++*[2] achieves a 6-point gain on mIoU over the original PointNet++ at an extra 13% computational overhead. The results demonstrate that TempNet can improve accuracy and efficiency for point cloud series segmentation. The qualitative example results are depicted in Figure 6. It can be seen that the overall accuracy is much improved after applying the TempNet framework to the SqueezeSegV2 model, especially for continuous surfaces that are subordinate to the same moving object. Furthermore, flickers between frames as illustrated in Figure 7 are eliminated for the sack of effective feature aggregation.

The performance comparison of applying various aggregation schemes for point cloud series segmentation to SequeezeSegV2 is listed in Table 2. Results on other single-frame methods are similar and omitted due to the page limitation. It can be seen that both TempNet schemes outwit other schemes in terms of accuracy and computational efficiency.

Methods	Online	IoU avg	Acc avg	time(ms)
baseline	-	39.73	86.8	1236
DFA	×	**44.43**	**88.1**	2996
*2	√	**44.43**	87.5	**1475**
DFP	√	38.77	86.3	309
*1	√	**41.89**	**86.9**	**270**

Table 2. Performance comparison of different aggregation schemes applied to SequeezeSegV2.

4.4. Ablation Study

Only results based on SequeezeSegV2 are shown due to page limitation.

Efficiency and accuracy gains. We manually adjust the ratio of key frame and plot the accuracy as a function of the key frame ratio in Figure 8. We obtain record A when all frames are treated as key frames, and record B when almost all frames are non-key frames whereas the single-frame baseline is recorded at C in the figure. Our performance curve is located to the upper right of record c by a large margin. From record C to record A, TempNet can improve the accuracy by 0.7 and the IoU metric by 4.7 with almost similar time overhead compared to the single-frame model. From record C to record B, it achieves more than five times the computational acceleration while maintaining the original performance. This illustrates the superiority of our aggregation scheme in terms of computational efficiency and segmentation accuracy.

Frame scheduling study. We conduct joint experiments on performance for the adaptive frame scheduling algorithm. As we can see in Figure 8, as the number of aggregated frames increases, the performance achieved by the Dense Feature Aggregation algorithm also increases, but

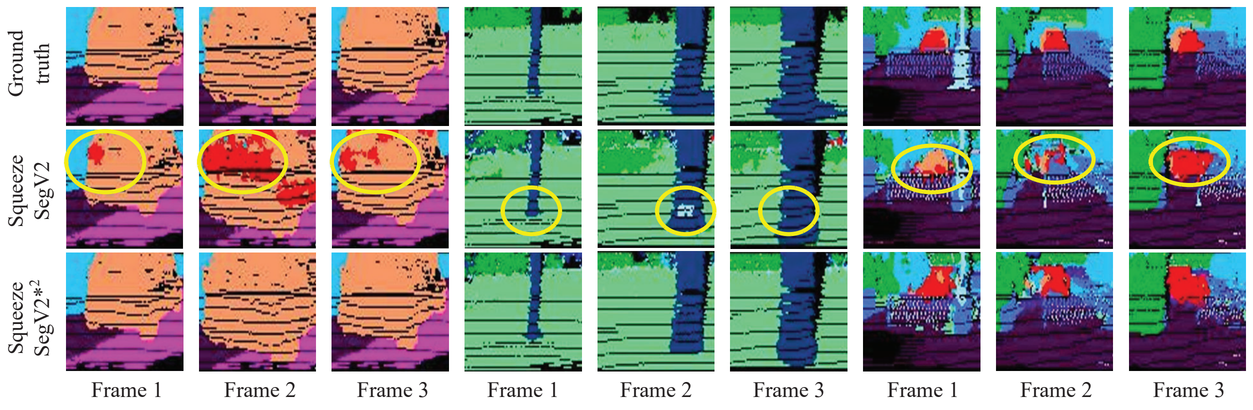

Figure 7. Qualitative example results on consecutive frames using the SqueezeSegV2 as the underlying segmentation method. Flickers between consecutive frames are effectively eliminated.

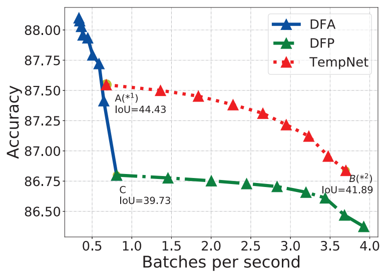

Figure 8. Accuracy as a function of the key frame ratio for different aggregation schemes.

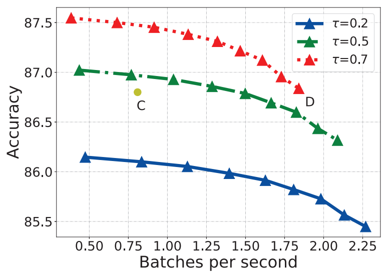

Figure 9. The impact of the consistency threshold τ to the accuracy of TempNet.

with a significant loss in speed. Eventually, performance will reach a bottleneck where it is difficult to continue to rise. The DFP algorithm has a significant improvement in speed as the number of frames increases but at the expense of model accuracy. TempNet is biased to the upper right in the line graph, allowing for a better trade-off between speed and accuracy. In particular, compared to the DFA algorithm, there is a very small decrease in accuracy per record but a large speed improvement has been achieved.

Partial update threshold. We examine the impact of the consistency threshold τ on the performance of TempNet. In this experiment, we vary the key frame ratio and τ and plot the accuracy of TempNet in Figure 9. The threshold τ determines how many keypoint features can be directly inherited in a partial update and discarded points need to have their features recaptured using $\mathcal{N}_{fea}$. It can be seen that when the threshold τ is high, a smaller number of points are directly inherited, and more are directly discarded in the next frame. So it usually leads to higher accuracy but also high computational cost. When the threshold τ is low, more points are considered similar and are inherited directly, which is faster but with a loss in accuracy. From record C to record D, it can be seen that twice the acceleration ratio with nearly the same accuracy is achieved at a threshold of 0.7. In our algorithm design, we are more inclined to get better performance so we generally set the τ value to 0.7.

5. Conclusion

We have proposed a novel framework for online point cloud series semantic segmentation, called TempNet. By combining a novel frame aggregation scheme, our method improves both the accuracy and stability of existing semantic segmentation models. The information between successive frames is aggregated to ensure accuracy through an attention mechanism. By partial update the propagated features with the local features extracted on non-key frames, our model avoids losing information while being computationally efficient. TempNet outperforms SOTA segmentation models on the SemanticKITTI dataset with little extra computational cost.

Acknowledgements

This research was supported in part by National Key RD Program of China (Grant No. 2018YFC1900700) and National Natural Science Foundation of China (Grants No. 61772340, 61972081, 61872240, and 61832006).

References

[1] Aseem Behl, Despoina Paschalidou, Simon Donne, and Andreas Geiger. Pointflownet: Learning representations for rigid motion estimation from point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[2] Jens Behley, Martin Garbade, Andres Milioto, Jan Quenzel, Sven Behnke, Cyrill Stachniss, and Juergen Gall. SemanticKITTI: A dataset for semantic scene understanding of lidar sequences. In *ICCV*, 2019. 2, 6

[3] Christopher Choy, JunYoung Gwak, and Silvio Savarese. 4d spatio-temporal convnets: Minkowski convolutional neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2019. 3

[4] Fei He, Naiyu Gao, Qiaozhe Li, Senyao Du, Xin Zhao, and Kaiqi Huang. Temporal context enhanced feature aggregation for video object detection. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 10941–10948, 2020. 3

[5] Qingyong Hu, Bo Yang, Linhai Xie, Stefano Rosa, Yulan Guo, Zhihua Wang, Niki Trigoni, and Andrew Markham. Randla-net: Efficient semantic segmentation of large-scale point clouds. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11108–11117, 2020. 2, 5, 6

[6] Xuhua Huang, Jiarui Xu, Yu-Wing Tai, and Chi-Keung Tang. Fast video object segmentation with temporal aggregation network and dynamic template matching. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8879–8889, 2020. 3

[7] Velodyne Lidar Inc. Alpha Prime. https://velodynelidar.com/products/alpha-prime/. 1

[8] Chiyu Jiang, Dana Lansigan, Philip Marcus, and Matthias Nießner. Ddsl: Deep differentiable simplex layer for learning geometric signals. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 8769–8778, 2019. 2

[9] Hao Jiang and Quanzeng You. Real-time multiple people hand localization in 4d point clouds, 2019. 3

[10] Roman Klokov and Victor Lempitsky. Escape from cells: Deep kd-networks for the recognition of 3d point cloud models. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 863–872, 2017. 2

[11] Jan Koutnik, Klaus Greff, Faustino Gomez, and Juergen Schmidhuber. A clockwork rnn. *arXiv preprint arXiv:1402.3511*, 2014. 2

[12] Truc Le, Giang Bui, and Ye Duan. A multi-view recurrent neural network for 3d mesh segmentation. *Computers & Graphics*, 66:103–112, 2017. 2

[13] Truc Le and Ye Duan. Pointgrid: A deep network for 3d shape understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018. 2

[14] Yangyan Li, Rui Bu, Mingchao Sun, Wei Wu, Xinhan Di, and Baoquan Chen. Pointcnn: Convolution on x-transformed points. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. 3

[15] Yaguang Li, Rose Yu, Cyrus Shahabi, and Yan Liu. Diffusion convolutional recurrent neural network: Data-driven traffic forecasting. *arXiv preprint arXiv:1707.01926*, 2017. 3

[16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 21–37, Cham, 2016. Springer International Publishing. 1

[17] Bo Meng, XueJun Liu, and Xiaolin Wang. Human action recognition based on quaternion spatial-temporal convolutional neural network and lstm in rgb videos. *Multimedia Tools and Applications*, 77(20):26901–26918, 2018. 3

[18] Yunqi Miao, Jungong Han, Yongsheng Gao, and Baochang Zhang. St-cnn: Spatial-temporal convolutional neural network for crowd counting in videos. *Pattern Recognition Letters*, 125:113–118, 2019. 3

[19] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. PointNet: Deep learning on point sets for 3D classification and segmentation. In *CVPR*, 2017. 3

[20] Charles Ruizhongtai Qi, Li Yi, Hao Su, and Leonidas J Guibas. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In *NeurIPS*, 2017. 2, 3, 6

[21] Yijun Qian, Lijun Yu, Wenhe Liu, Guoliang Kang, and Alexander G Hauptmann. Adaptive feature aggregation for video object detection. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision Workshops*, pages 143–147, 2020. 3

[22] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016. 1

[23] Gernot Riegler, Ali Osman Ulusoy, and Andreas Geiger. Octnet: Learning deep 3d representations at high resolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3577–3586, 2017. 2

[24] Tao Song, Leiyu Sun, Di Xie, Haiming Sun, and Shiliang Pu. Small-scale pedestrian detection based on topological line localization and temporal feature aggregation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 536–551, 2018. 3

[25] Hang Su, Subhransu Maji, Evangelos Kalogerakis, and Erik Learned-Miller. Multi-view convolutional neural networks for 3d shape recognition. In *Proceedings of the IEEE international conference on computer vision*, pages 945–953, 2015. 2

[26] Lin Sun, Kui Jia, Dit-Yan Yeung, and Bertram E Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4597–4605, 2015. 3

[27] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *CoRR*, abs/1409.4842, 2014. 1

[28] H. Thomas, C. R. Qi, J. Deschaud, B. Marcotegui, F. Goulette, and L. Guibas. Kpconv: Flexible and deformable convolution for point clouds. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6410–6419, 2019. 3

[29] Lei Wang, Yuchun Huang, Yaolin Hou, Shenman Zhang, and Jie Shan. Graph attention convolution for point cloud semantic segmentation. In *CVPR*, 2019. 2, 3, 6

[30] Peng-Shuai Wang, Yang Liu, Yu-Xiao Guo, Chun-Yu Sun, and Xin Tong. O-CNN: octree-based convolutional neural networks for 3d shape analysis. *CoRR*, abs/1712.01537, 2017. 2

[31] Bichen Wu, Xuanyu Zhou, Sicheng Zhao, Xiangyu Yue, and Kurt Keutzer. Squeezesegv2: Improved model structure and unsupervised domain adaptation for road-object segmentation from a lidar point cloud. In *ICRA*, 2019. 1, 2, 3, 6

[32] Jialian Wu, Chunluan Zhou, Ming Yang, Qian Zhang, Yuan Li, and Junsong Yuan. Temporal-context enhanced detection of heavily occluded pedestrians. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13430–13439, 2020. 3

[33] Huaxiu Yao, Fei Wu, Jintao Ke, Xianfeng Tang, Yitian Jia, Siyu Lu, Pinghua Gong, Jieping Ye, and Zhenhui Li. Deep multi-view spatial-temporal network for taxi demand prediction. *arXiv preprint arXiv:1802.08714*, 2018. 3

[34] Bing Yu, Haoteng Yin, and Zhanxing Zhu. Spatio-temporal graph convolutional networks: A deep learning framework for traffic forecasting. *arXiv preprint arXiv:1709.04875*, 2017. 3

[35] Shifu Zhou, Wei Shen, Dan Zeng, Mei Fang, Yuanwang Wei, and Zhijiang Zhang. Spatial–temporal convolutional neural networks for anomaly detection and localization in crowded scenes. *Signal Processing: Image Communication*, 47:358–368, 2016. 3

[36] Hongyuan Zhu, Romain Vial, and Shijian Lu. Tornado: A spatio-temporal convolutional regression network for video action proposal. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 5813–5821, 2017. 3

[37] Xizhou Zhu, Yuwen Xiong, Jifeng Dai, Lu Yuan, and Yichen Wei. Deep feature flow for video recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2349–2358, 2017. 2