Learning Generalized Feature for Temporal Action Detection: Application for Natural Driving Action Recognition Challenge

Chuong Nguyen† Ngoc Nguyen† Su Huynh† Vinh Nguyen† Son Nguyen
CyberCore AI
{chuong.nguyen, ngoc.nguyen, su.huynh, vinh.nguyen, son.nguyen }@cybercore.co.jp

Abstract

This paper reports our approach for the 2022 AI City Challenge - Naturalistic Driving Action Recognition (Track 3), where the objective is to detect when and what kinds of actions that a driver performs in a long, untrimmed video. Our solution is built upon the single stage ActionFormer detector, in which temporal location and classification are predicted simultaneously for efficiency. The input feature for the detector is extracted offline using our proposed backbone, which we named "ConvNext-Video". However, due to the small size of the dataset, training the model to avoid over-fitting becomes challenging. To address this problem, we focus on training techniques that can improve the generalization of underlying features. Specifically, we utilize two methods: "learning without forgetting" and semi-weak supervised learning on the unlabeled data A2. Finally, we also add a second-stage classifier (SSC) using our ConvNeXt-Video backbone. The SSC Classifer is designed to combine information from multi-clips and multi-view cameras to improve the prediction precision. Our best result achieves 29.1 F1 score on the public test set. Our source code is released at link.

1. Introduction

Temporal Action Detection (TAD) aims to detect the start and end time of an action in an untrimmed video and classify it. In the Track 3 [22] of 2022 AI City Challenge, the objective is to detect inattentive actions of a driver in videos recorded from different cameras. In particular, the dataset includes the training set A1 and validation set A2, each set has 5 drivers, and each driver has 2 videos collected in about 9 minutes from 3 synchronized cameras placed at Dashboard, Right Window and Rear views in the car, as illustrated in Fig. 1. Only A1 has public ground truth labels, while human labeling or semi-supervised labeling on A2 are not allowed. A prediction is true-positive if the start and end time are within 1 second of the ground truth and the category is correctly recognized. It is false-positive if either temporal boundary or category are mismatched (not true-positive), and false-negative if a ground-truth activity that was not correctly identified. The prediction performance is evaluated using F1-score.

Figure 1. Sample frames from Dashboard, Rear View and Right Window respectively.

While most of public datasets contain short activities, typically happening in a period of 4-10 second, videos in the Track 3 dataset have multiple and much longer actions (more than 20 seconds). Hence, this imposes a unique, practical challenge for TAD algorithms. In addition, the small size of the dataset makes training deep-learning

†equal contribution

models prompt to be overfitted.

Therefore, in this work, we focus on extracting discriminating and generalizing features for TAD algorithms. Our contribution is summarized as follows:

- Develop an efficient backbone, that is built upon the ConvNeXt [19] model, to extract feature for video inputs.
- Combine Temporal Sensitive Pretraining [1] with "Learning without Forgetting" [12] technique, and semi-supervised learning to improve the feature generalization.

2. Literature Review

2.1. Action Recognition

Action recognition is one of the most fundamental tasks for video understanding, and there has been extensive studies in this field. For deep-learning approaches, the network architectures can be divided into two main categories: CNN-based approach and Transformer-based approach. CNN-based approaches are dominated by 3D-Convolutional Neural Networks. The early works [6] extend the 2D convolution models into 3D-CNN by inflating the its pretrained weight on ImageNet dataset. To reduce the computation cost, later works [26, 27] factorize the 3D kernel into (2+1)D by mixing spatial and temporal dimensions. Furthermore, Neural Architecture Search [10, 11] are used to find the optimal networks.

Motivated by the success of the transformer in the 2D-image domain, which divides an image into many patches and then applies attention mechanisms on these patches to extract features, recent works [3, 20, 23, 34] have been developed with several self-attention variants and improve performance progressively.

2.2. Temporal Action Detection (TAD)

The framework of TAD can be classified into two-stage and single-stage detectors.

The two-stage approach uses an proposal network to generate candidate video segment, and further classify the proposal or refines their temporal boundary in the later stages. Most works using this approach are focused on action proposal generation, by either detecting action boundary [15, 17, 35] or classifying anchor windows [5, 9]. Recent methods also take advantages of Transformer [28] to increase the long-range temporal connection [24, 37], or by using graph network [29, 31]. In general, the two-stage approach can take the benefits of well-established methods for video recognition in the second stage, achieving higher precision detection but also requires double computation and it is comparatively slower.

Motivated by the success of Single-Stage in object detection, recent works [4, 16, 21, 30] try to adopt the simple architecture of single-stage detector to the action detection. Anchor-Free methods [13, 32], inspired from FCOS object detector [25] are also proposed to further simplify the pipeline. ActionFormer [33] and TadTR [18] are the two concurrent works that adopts the Transformer [28] to the encoder for single-stage detectors. Single-stage detector is simpler since it can predict the action category and temporal boundary concurrently. In addition, it can also serve as the proposal network in any two-stage detectors if higher precision is desired.

3. Method

3.1. Models

The model pipeline is illustrated in Fig. 2. We describe its main components, including the backbone, detector and second-stage classifier, as follows.

3.1.1 Backbone Model

We modify the ConvNeXt [19] backbone from image classification for the task video recognition. The model takes as input a clip $X \in \mathbb{R}^{L \times H \times W \times 3}$ consisting of L RGB frames of size $H \times W$ sampled from the original video.

The ConvNeXt backbone has 4 stages, and we forward L frames to stage 1 and 2 of the network as a regular batch of 2D images, and outputs L features of size $(H/8 \times W/8 \times C)$. Differently, before continuing to the deeper stages, we rearrange the 3D feature of size $(L \times H/8 \times W/8 \times C)$ to a 2D feature of size $(hH/8 \times wW/8 \times C)$, where $hw = L$. For example, for $L = 9$ frames, we set $h = 3$ and $w = 3$, as illustrated in Fig. 3. Intuitively, this casts the problem from video recognition into classifying a "collage image" [2].

To increase the temporal connection, we add a depth-wise convolution of kernel size (h, w) and dilation (H, W) without padding to the ConNeXt block in the deeper stages, as shown in Fig. 4b and name it as ConvNeXt-Vid [2] block.

Concretely, in the ConvNeXt-Vid block, for an input feature of size $hH \times wW \times C$, the depth-wise convolution 7×7 extracts the spatial features $S \in hH \times wW \times C$, and the dilated convolution extracts the temporal feature $T \in H \times W \times C$. The spatial-temporal fusion is then obtained by adding T and S:

$$F = S + \alpha * [T]_{(\times h, \times w)} \tag{1}$$

where $[\cdot]_{(\times h, \times w)}$ denotes repeating the tensor h times vertically and w horizontally, $\alpha \in \mathbb{R}^C$ is the learnable vector to balance between temporal and spatial branch, and $*$ is the element-wise product. This design allows sharing the temporal features T for all frames.

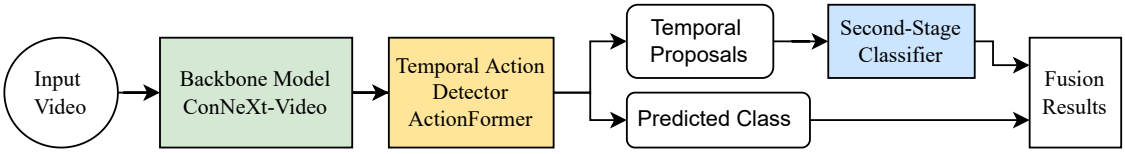


Figure 2. Model Pipeline.

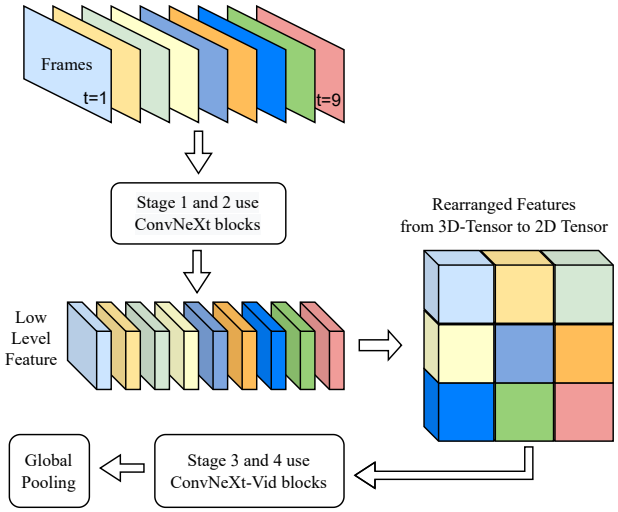
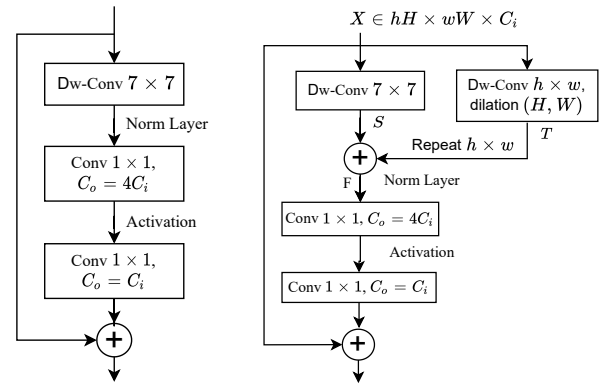


Figure 3. Visualize the pipeline of ConvNeXt-Video backbone. We use an input video with 9 sampled frames as an example.



(a) ConvNeXt block [19] (b) ConvNeXt-Vid block

Figure 4. Modifying ConvNeXt block for Video Recognition

Finally, we use the architecture configuration of ConvNeXt-tiny [19], which has $C = (96, 192, 384, 768)$, $B = (3, 3, 9, 3)$ where C denotes the channel number of the hidden layers in the first stage, and B is the number blocks in each stage. We only replace ConvNeXt by the ConNeXt-Vid blocks in stage 3 and 4 of the model. The final feature is extracted by global average pooling operator.

3.1.2 Temporal Action Detector

We use the ActionFormer [33] as our baseline detector thanks its simple and efficient architecture. ActionFormer has two parts: encoder and decoder. The encoder takes as input a feature $X \in \mathbb{R}^{T \times C}$, where T is the temporal dimension and C is the number feature. The encoder consists several transformer blocks with down-sampling in between to enrich the temporal feature, and output a feature pyramid of different temporal resolution. The decoder includes a classification and regression head, to predict the action category and the temporal boundary. The decoder takes the feature pyramid as input and output the class and start-end time directly without using extra classifier.

3.1.3 Mutli-View Feature Learning

As mentioned in the introduction, the videos are recorded from 3 cameras, thus combining the information from different sources is important. Watching the video and ground truth, we observe that several actions, such as Phone Call or Singing, can be observed easier in the Dashboard view than the others. Therefore, to better utilize the information, we simply concatenate the feature extracted from different views together, assuming that the camera are synchronized. This will triple the input feature dimension for the ActionFormer.

3.1.4 Second-Stage Classifier (SSC)

To further improve the precision of ActionFormer, we also train an extra classifier as the second-stage and use the prediction of transformer as proposals. Since the duration of actions in the dataset is about 20s, we split a segment into 5 mini-clips, each last about 4 seconds, and use the backbone ConvNeXt-Video to extract the feature for each clip.

To better fuse the features extracted from 3 different views and 5 mini-clips from a proposal segment, we rearrange it into a 2D tensor $C \times 3 \times 5$, where C is the number of feature of a single view-single clip. We then pass the feature to sub-network (neck), which include a 2 Convolution of kernel (1,5) and (3,1) to fuse the feature in temporal and views respectively.

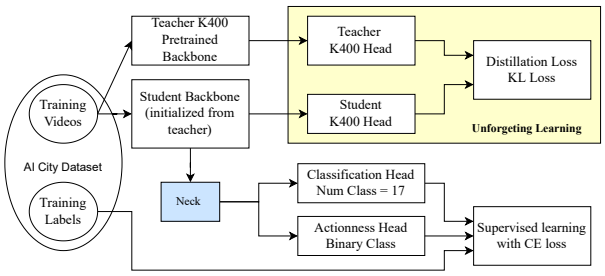


Figure 5. Temporal Sensitive Pretrained without forgetting.

3.1.5 Score Fusion

Since the ActionFormer also returns the class prediction, we test several ways to ensemble the results between the two stages, such as element product or average. However, we found that using only the SSC Classifier yields higher scores in our experiments.

3.2. Training Techniques

3.2.1 Learning without Forgetting

Although training end-to-end both backbone and detector promisingly yields better performance, there are two constraints. First, the dataset must be large enough otherwise the model will be overfitted easily. Second, training with large image size is infeasible due to the huge compute memory required to process the entire video. Therefore, normally the image must be down sampled to small size [14], otherwise the backbone is first pretrained and then features are extracted offline [1]. Hence, we follow the later idea [1] due to the small size of dataset.

To further improve the generalization of the model, we employ the "learning without forgetting" technique [12]. The main idea is illustrated in Fig. 5. Here, the teacher model is pretrained on Kinetics 400 (K400) dataset [6] and frozen during training. The student model is initialized from the teacher, but is augmented new heads for the new task Temporal Sensitive Pretraining (TSP) [1]. The task TSP is trained as normal, such that the model not only learns to classify an action in Classification head, but also detects whether an input clip contains background or action in "Actionness" head. In addition, we use Kullback–Leibler (KL) Loss to distill the output of teacher to the student through its original head, *i.e*. the one that was trained on K400. Note that we don't need the K400 dataset for distillation learning. Since the teacher is frozen, the student learns to mimic what the teacher observes in the new data. Therefore, by using dual supervision where the total loss is:

$$Loss = Loss_{KL} + wLoss_{TSP} \quad (2)$$

we force the backbone learn to extract feature for the new tasks, but refrain it forgetting useful features already learn from the pretrained K400 dataset. This improves the generalization for the transfer learning. In (2), w is the weight to balance the forgetting and new task. We select the default values $w = 50$ but found that $w \in [50 : 200]$ also works.

Furthermore, we add a Neck network to compress the feature in order to decrease the model capacity. The Neck is a single Dilated Convolution with kernel size (h, w) and dilation (H, W) without padding. For an input feature extracted from the Backbone, the Neck reduces its spatial dimension from $(hH \times wW)$ to (H, W) and compresses the channel from $C = 768$ to $C/4 = 192$.

3.2.2 Semi-Weak Supervised Learning

To enrich the dataset, we use the model trained on A1 to generate pseudo labels on the Validation set A2. We select the top-score samples from A2 for training in the second round. However, there are two issues need to consider when using pseudo-labels.

First, since there are possibly false-negative detection, *i.e*. foreground segments that are detected as background. If we use them as background samples to train the network, it will harm the recall ratio. Therefore, we first filter the samples with scores less than 0.05 as negative, greater than 0.3 as positive, and ignored otherwise. For negative samples, we subtract their score from 1 to obtain their background score, and use Soft-NMS to filter the overlaped segments. Finally, we select top-10 segments with highest background score as the negative samples for training, in addition to the foreground samples.

Second, under the restriction that start time and end time must be within 1 second of the ground truth, which is very sensitive and bias to human labeling, the temporal boundary of pseudo labels are not reliable enough to train the network. Therefore, we only use pseudo data to train the classification head of ActionFormer. In practical implementation, we set the regression loss of ActionFormer to 0 if the samples are from pseudo labeled A2.

4. Experiments

4.1. Setup

Datasets: The Track3 - AI City 2022 [22] provides the training set A1 and validation set A2. Each set records driving activity of 5 drivers, and each driver has two videos. Each video lasts about 9 minutes and is recorded at fps 30 from 3 cameras placed at Dashboard, Window and Rear views. For convenience when evaluating the model, we label the dataset A2 and denote it as Local-A2. Our Local-A2 achieves F1 score 46.24, Precision 47.9 and Recall 44.69 on the public evaluation system. This suggests that there are more than 50% mismatches when different people

Training Round	Training Dataset	Pretrained Top-1 Acc.	Detector mAP	Detector F1 score	With SSC F1 score
1	A1	62.5	27.57	22.56	28.84
2	A1 + Pseudo-A2-v1	67.5	31.55	33.03	32.62
3	A1 + Pseudo-A2-v2	75	23.8	-	-

Table 1. Training Progress Summary reported on Local-A2 dataset. With-SSC means combine the Action Former with the Second-Stage Classifier.

label the same dataset. In addition, we also observe that for drivers with ID 65818, 56306, 79336, there are around 1s delay between the Dashboard camera and the other views, indicating that the cameras are not perfectly synchronized. Nevertheless, we use the start-end time in the Dashboard as the synchronized labels for simplicity. We report the accuracy Top-1 and Top-5 on Local A2 set for pretraining the backbone, and the mAP when training ActionFormer during ablation study.

Implementation Details: Our code is implemented using MMAction2 [7] framework when training backbone, and ActionFormer [33] for Temporal Action Detector.

For pretraining the backbone, we use AdamW optimizer [26] with a batch size of 32. We use a cosine decay learning rate scheduler and linear warm-up in the first epoch. The initial learning rate is 1e-3, and decays to the minimal value 5e-6. To regularize the model, we use the stochastic depth with drop path rate 0.5. Because the TSP head is initialized randomly, we multiply the learning rate of the backbone by 0.25 and the Unforgetting head by 0.05 to improve the stability. Since the relative position of camera and drivers is fixed, we crop the right-haft of image in the pre-processing step. For data-augmentation, we don't use image-flipping as common but apply RandAugment [8] and Random Erasing [36]. Unless otherwise mentioned, we sample a clip of 9 frames from each full length video using a random temporal interval in range of 13 to 18, and spatial size of 224×224 during training. When running inference, we fix the temporal interval as 15 frames, thus for the FPS=30, the feature is extracted from a corresponding 4 seconds segment.

For training and testing the detector, we first use the pretrained backbone to extract features. We set the temporal stride $s_T = 8$ frames. For example, for 60 seconds video, we will extract $T = (60 - 4)/(8/30) = 210$ segments, where each segment has 4 second length as aforementioned. We train ActionFormer using AdamW optimizer with weight decay 0.05, cosine decay learning rate schedule and linear warm-up in the first 5 epochs. The initial learning rate is 1e-3. To regularize the model, we set drop-path rate to 0.1, drop-out 0.1, and label smoothing 0.1. We train the model for 50 epochs using batch size equal 1. Other hyper-parameters, including the architecture, are set to the default values.

For the SSC classifier, we use the same training setting as in backbone pretraining. Differently, we train the SSC classifier using multiple segments and multi-view cameras as mentioned in Sec. 3.1.4. Concretely, from each camera view, we randomly sample 5 segments using the ground-truth temporal, thus forming total 3×5 mini-clip input for each action. Due to the memory constraint when training with multi-views input, we can't apply "Learning without Forgetting" in this experiment, but believe the results can be further improved. When inference, we use the proposals from ActionFormer to crop the corresponding video section, and split it to 5 segments for each view. The temporal interval is adaptively selected to cover the whole segment, but not greater than 18 frames.

4.2. Main Results

Our experiment results are summarized in Tab. 1. We conduct three rounds of training with Pseudo labels on A2 dataset, as described belows.

4.2.1 Round 1: Training using A1 dataset

We first train the baseline models on A1 dataset. In this first iteration, we train the TSP model, and achieve 62.5% top-1 and 79% top-2 accuracy on the Local-A2. We use this backbone to extract feature to train the ActionFormer detector. The results are 27.57mAP at Temporal IoU (0.8:0.85:0.9), and F1 score is 22.56. We also train the second-stage classifier (SSC) and achieve 78% top-1 accuracy on the Local-A2. The SSC classifier achieves higher accuracy because we run inference with 5 mini-clips and 3 views, as described in Sec. 3.1.4 while for the TSP we only report the results using single view. When fusing the result of ActionFormer and the SSC classifier, we improve the F1 score to 28.84 on the Local-A2.

4.2.2 Round 2: Training using A1 and Pseudo-A2-v1

We use the pseudo-labels generated in the Round 1, denoted as Pseudo-A2-v1, to retrain the backbone with TSP . After that, we use the feature extracted on A1 to train the detector, but keep the SSC classifier unchanged to avoid bias to dataset A2. In this second iteration, the TSP model achieves 67.5% top-1 and 78.7% top-2 accuracy

and the ActionFormer achieves 31.55 mAP at Temporal IoU (0.8:0.85:0.9) and F1 score 33.03. After fusing the ActionFormer and the SSC classifier, we achieve the F1 score 32.62 on the Local-A2. We continue using the predicted labels as the pseudo labels for the Round 3, denoted as Pseudo-A2-v2.

4.2.3 Round 3: Training using A1 and Pseudo-A2-v2

Repeating the process in Round 2 with the pseudo label Pseudo-A2-v2, we further improve the TSP top-1 to 75%. However, training again the detector yields lower mAP score. We suspect the model is overfitted to the pseudo-A2-v1. Thus, we stop training from here.

4.3. Result Analysis

4.3.1 Proposal Visualization

We visually inspect the quality of temporal proposals, as illustrated Fig. 6. The on-score and off-score at the start-end time are computed by accumulating the confidence score of the proposals that have corresponding start-end time. In general, we observe that the predicted temporal boundary are within 1-3 seconds of the ground truth on the Local-A2 validation set.

4.3.2 Classification error

In this section, we would like to investigate the error due to the classifier, assuming that the temporal proposal is perfect. Figure 7 shows the confusion matrix for the SSC classifier on our Local A2 dataset. We see that class 16 ("Singing with Music") and 17 ("Shaking or Dancing with Music") are the most confused classes. Concretely, along the column 16^{th} in the matrix, in total 22 samples predicted as class 16, only 8 predictions are true-positive, while 6 samples are mismatch with class 17, and 4 samples are mismatched with class 6. In reverse, in the row 17^{th}, only 3 over 10 samples of class 17 are recognized correctly, and 6 of them are wrongly identified as class 16. Another pair of actions are highly confused is class 12 ("Talk to passenger at the Right") and 13 ("Talk to passenger at backseat"), as seen in rows 12^{th} and 13^{th} in the confusion matrix. The results make sense since these classes are also difficult to distinct even for human if using visual cue only. The small size of dataset also hinder the generalized prediction for these classes.

5. Conclusion

In this paper, we propose a solution for Temporal Action Detection, particularly applied for the small-size dataset in the 2022 AI City Challenge - Naturalistic Driving Action Recognition (Track 3). Our method is built upon the newly proposed ConvNeXt-Video backbone for feature extraction, trained with Temporal Sensitive Awareness. To improve the generalization for the model, we adopt the techniques "learning without forgetting" and semi-weak supervised learning. Combined with off-the-shell ActionFormer Detector, we achieve the highest score 31.55 mAP at IoU (0.8:0.85:0.9) and F1 score 32.61 on our Local-A2. On the leader board test set A2, our best F1 score is 29.1.

References

[1] Humam Alwassel, Silvio Giancola, and Bernard Ghanem. TSP: Temporally-sensitive pretraining of video encoders for localization tasks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 3173–3183, 2021. 2, 4

[2] Anonymous. Vidconv: A modernized 2d convnet for efficient video recognition. *under-review*, 2022. 2

[3] Gedas Bertasius, Heng Wang, and Lorenzo Torresani. Is space-time attention all you need for video understanding. *arXiv preprint arXiv:2102.05095*, 2(3):4, 2021. 2

[4] Shyamal Buch, Victor Escorcia, Bernard Ghanem, Li Fei-Fei, and Juan Carlos Niebles. End-to-end, single-stream temporal action detection in untrimmed videos. In *Procedings of the British Machine Vision Conference 2017*. British Machine Vision Association, 2019. 2

[5] Shyamal Buch, Victor Escorcia, Chuanqi Shen, Bernard Ghanem, and Juan Carlos Niebles. Sst: Single-stream temporal action proposals. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 2911–2920, 2017. 2

[6] Joao Carreira and Andrew Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 6299–6308, 2017. 2, 4

[7] MMAction2 Contributors. Openmmlab's next generation video understanding toolbox and benchmark. https://github.com/open-mmlab/mmaction2, 2020. 5

[8] Ekin D Cubuk, Barret Zoph, Jonathon Shlens, and Quoc V Le. Randaugment: Practical automated data augmentation with a reduced search space. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 702–703, 2020. 5

[9] Victor Escorcia, Fabian Caba Heilbron, Juan Carlos Niebles, and Bernard Ghanem. Daps: Deep action proposals for action understanding. In *European conference on computer vision*, pages 768–784. Springer, 2016. 2

[10] Christoph Feichtenhofer. X3d: Expanding architectures for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 203–213, 2020. 2

[11] Dan Kondratyuk, Liangzhe Yuan, Yandong Li, Li Zhang, Mingxing Tan, Matthew Brown, and Boqing Gong. Movinets: Mobile video networks for efficient video recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16020–16030, 2021. 2

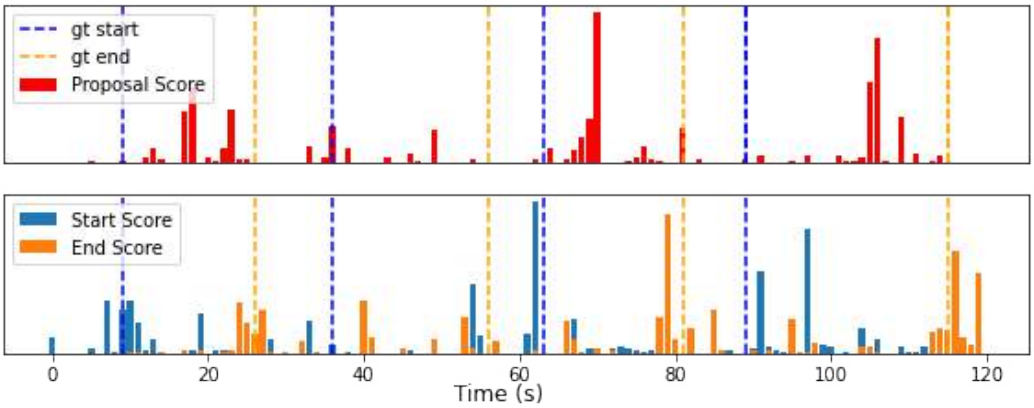


Figure 6. Visualize the start-end and the confidence score of the proposals in the first 120 second for user ID 42271 - 3

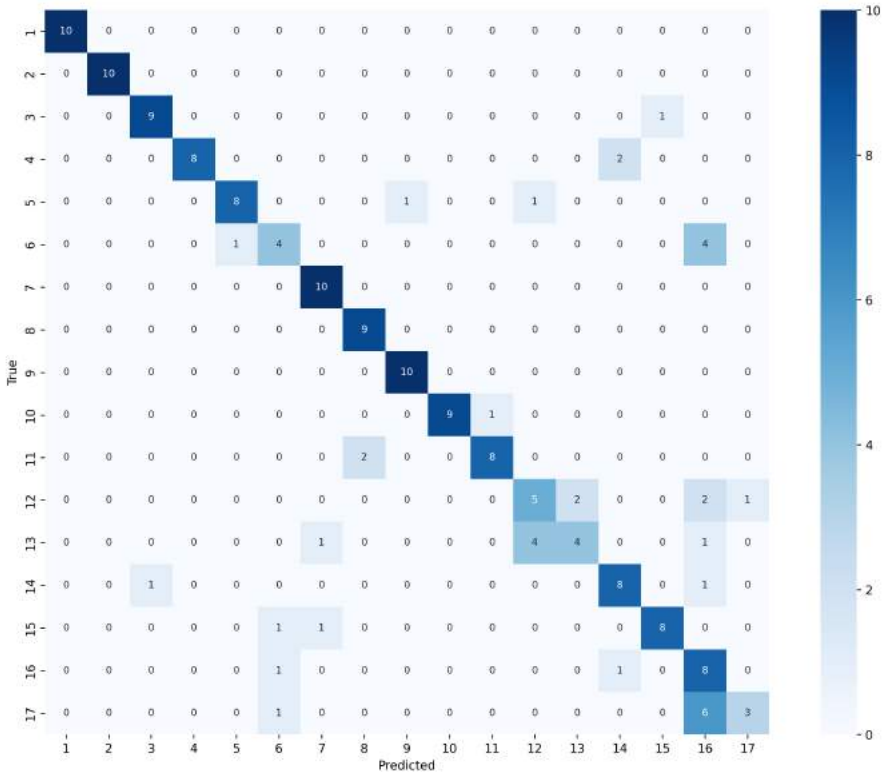


Figure 7. Classification Confusion Matrix. The results is reported using prediction of SSC classifier and the Local A2 ground-truth.

[12] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE transactions on pattern analysis and machine intelligence*, 40(12):2935–2947, 2017. 2, 4

[13] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning salient boundary feature for anchor-free temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 3320–3329, 2021. 2

[14] Chuming Lin, Chengming Xu, Donghao Luo, Yabiao Wang, Ying Tai, Chengjie Wang, Jilin Li, Feiyue Huang, and Yanwei Fu. Learning salient boundary feature for anchor-free temporal action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3320–3329, June 2021. 4

[15] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. Bmn: Boundary-matching network for temporal action proposal generation. In *Proceedings of the IEEE/CVF*

International Conference on Computer Vision, pages 3889–3898, 2019. 2

[16] Tianwei Lin, Xu Zhao, and Zheng Shou. Single shot temporal action detection. In *Proceedings of the 25th ACM international conference on Multimedia*, pages 988–996, 2017. 2

[17] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. Bsn: Boundary sensitive network for temporal action proposal generation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 2

[18] Xiaolong Liu, Qimeng Wang, Yao Hu, Xu Tang, Song Bai, and Xiang Bai. End-to-end temporal action detection with transformer. *arXiv preprint arXiv:2106.10271*, 2021. 2

[19] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022. 2, 3

[20] Ze Liu, Jia Ning, Yue Cao, Yixuan Wei, Zheng Zhang, Stephen Lin, and Han Hu. Video swin transformer. *arXiv preprint arXiv:2106.13230*, 2021. 2

[21] Fuchen Long, Ting Yao, Zhaofan Qiu, Xinmei Tian, Jiebo Luo, and Tao Mei. Gaussian temporal awareness networks for action localization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 344–353, 2019. 2

[22] Mohammed Shaiqur Rahman, Archana Venkatachalapathy, Anuj Sharma, Jiyang Wang, Senem Velipasalar Gursoy, David Anastasiu, and Shuo Wang. Dataset for analyzing various gaze zones and distracted behaviors of a driver, 2022. 1, 4

[23] Gilad Sharir, Asaf Noy, and Lihi Zelnik-Manor. An image is worth 16x16 words, what is a video worth? *arXiv preprint arXiv:2103.13915*, 2021. 2

[24] Jing Tan, Jiaqi Tang, Limin Wang, and Gangshan Wu. Relaxed transformer decoders for direct action proposal generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13526–13535, 2021. 2

[25] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. Fcos: Fully convolutional one-stage object detection. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 9627–9636, 2019. 2

[26] Du Tran, Heng Wang, Lorenzo Torresani, and Matt Feiszli. Video classification with channel-separated convolutional networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5552–5561, 2019. 2

[27] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. A closer look at spatiotemporal convolutions for action recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6450–6459, 2018. 2

[28] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2

[29] Mengmeng Xu, Chen Zhao, David S Rojas, Ali Thabet, and Bernard Ghanem. G-tad: Sub-graph localization for temporal action detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10156–10165, 2020. 2

[30] Le Yang, Houwen Peng, Dingwen Zhang, Jianlong Fu, and Junwei Han. Revisiting anchor mechanisms for temporal action localization. *IEEE Transactions on Image Processing*, 29:8535–8548, 2020. 2

[31] Runhao Zeng, Wenbing Huang, Mingkui Tan, Yu Rong, Peilin Zhao, Junzhou Huang, and Chuang Gan. Graph convolutional networks for temporal action localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 7094–7103, 2019. 2

[32] Runhao Zeng, Haoming Xu, Wenbing Huang, Peihao Chen, Mingkui Tan, and Chuang Gan. Dense regression network for video grounding. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10287–10296, 2020. 2

[33] Chenlin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. *arXiv preprint arXiv:2202.07925*, 2022. 2, 3, 5

[34] Yanyi Zhang, Xinyu Li, Chunhui Liu, Bing Shuai, Yi Zhu, Biagio Brattoli, Hao Chen, Ivan Marsic, and Joseph Tighe. Vidtr: Video transformer without convolutions. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13577–13587, 2021. 2

[35] Peisen Zhao, Lingxi Xie, Chen Ju, Ya Zhang, Yanfeng Wang, and Qi Tian. Bottom-up temporal action localization with mutual regularization. In *European Conference on Computer Vision*, pages 539–555. Springer, 2020. 2

[36] Zhun Zhong, Liang Zheng, Guoliang Kang, Shaozi Li, and Yi Yang. Random erasing data augmentation. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13001–13008, 2020. 5

[37] Zixin Zhu, Wei Tang, Le Wang, Nanning Zheng, and Gang Hua. Enriching local and global contexts for temporal action localization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 13516–13525, 2021. 2