# Weakly Supervised EM Process For Temporal Localization Within Video

Jin Xia*
MVIG group in SJTU
ga.xiajin@gmail.com

Jie Shao
Bytedance AI Lab
shaojie.mail@bytedance.com

Cewu Lu
MVIG group in SJTU
lucewu@sjtu.edu.cn

Changhu Wang
Bytedance AI LAB
wangchanghu@bytedance.com

## Abstract

*In the field of video understanding, fewer segment annotations are available comparing to the video level annotations. This work proposes a weakly supervised EM process for the segment localisation task, which leverages the large amount of existing video labels. Briefly speaking, the basic idea is to perform Expectation-step and Maximization-step to iteratively estimate segment labels and fit models. Our experimental results show the advantage of our method in the task of segment localisation. In the 3rd YouTube-8M video understanding challenge, our final models achieve an MAP score of 81.0% on the public board, which outperforms the baseline method for more than 10% MAP.*

## 1. Introduction

Thanks to the development of digital cameras and smart phones, numerous videos are recorded, uploaded and shared on Internet. The understanding of video content has various applications such as video recommendation, searching and intelligent robots, etc.

A number of large-scale video classification datasets, including Youtube-8m [1], Kinetics-400, Kinetics-600, Kinetics-700 [10, 5] and Moments in Time [15] are introduced to accelerate the study of video understanding algorithms. However, due to the expensive cost of segment annotations, few video segment localisation datasets have been proposed. To boost segment localisation algorithms, The 3rd YouTube-8M video understanding competition [1] introduces the Youtube-8M Segment dataset which has a few human verified segment labels in a small portion of the origin dataset. Participants are encouraged to leverage the large amount of training video labels and design robust segment localisation algorithm having only a small number of segment labels.

*Work done during an internship at Bytedance AI Lab.

In this article, we extend work in [1] and search for models that are robust for video content localisation task. Specifically, we find models such as NextVLAD network [12], Transformer [21] and BiGRU [7] efficient. In addition, to leverage the large-scale video classification dataset, we propose a weakly supervised EM process for the video segment localisation task. By estimating segment label and fitting models iteratively, our models achieve the top score in the 3rd YouTube-8M video understanding challenge.

## 2. Related Work

In this section, we provide a brief review of most recent researches on video classification, segment localisation and weakly supervised method for video understanding.

### 2.1. Video Classification

Video classification task has made considerable progresses in recent years thanks to the evolution of deep learning. C3D [20] uses 3D convolutional nerual networks to extract spatio-temporal features from videos, showing that 3D CNN is a good descriptor for action recognition tasks. Based on 3D CNN, P3D [19], S3D [26], I3D [6] and SlowFast network [8] are proposed to better model the spatio-temporal features. In the mean time, feature aggregation methods [2, 12, 14] are extended to the video classification task.

### 2.2. Segment Localisation

Segment Localisation has attracted attentions in recent years. ActivityNet [4] is introduced to boost segment proposal and temporal localization algorithms. BSN [13] is the state-of-art model for the temporal action proposal task. In addition, LFB [24], Action Transformer [9] and Three Branches [25] are proposed to localize actions in space and time.

### 2.3. Weakly supervised Method

Weakly supervised EM algorithm for text classification is studied in [17]. For the video segment localisation task, various methods are proposed [16, 18, 23, 27] to leverage available video labels.

## 3. YouTube-8M

YouTube-8M is a large-scale labeled video dataset that consists of millions of YouTube video IDs. Video data is pre-computed to audio visual features to fit on a single hard disk and ease the training process. Labels are machine-generated from a diverse vocabulary of 3,800+ visual entities. The goal of the competition is to accelerate research on large-scale video understanding.

### 3.1. YouTube-8M V2

The YouTube-8M V2 dataset, updated in 2018, is a cleaner version of the original YouTube-8M dataset. YouTube-8M V2 includes 6.1 million videos and 3862 classes. The dataset was used for the 2nd YouTube-8M video understanding challenge. YouTube-8M V2 is also one of the datasets used in the 3rd YouTube-8m temporal localisation challenge. Although YouTube-8M V2 contains only video level labels, temporal localisation models could also benefit from it due to its large scale.

### 3.2. YouTube-8M Segments

The YouTube-8M segments dataset is an extension of the YouTube-8M dataset with human-verified segment annotations. The goal is to temporally localize the entities in the videos, i.e., find out when the entities occur. Note that segments are annotated only on a very small portion of YouTube-8M V2 dataset. Precisely, there are 237K segment labels for 1000 classes.

## 4. Models

In this section, we will introduce the models that we use for the segment localisation task. We treat the temporal localization problem as video segment classification problem. We use two kinds of models, the frame level models and sequence models. The frame level models take one segment as input and output predictions for each segment, while the sequence models take the entire video as input and make predictions for every five seconds.

### 4.1. Logistic Model

Logistic Model is one of the frame level models. The logistic model averages features along time axis. Predictions are made using full connected layer and sigmoid activation. The logistic model is a simple but efficient starter model. Our logistic model achieves 72.30% mAP on the public board, which is 10% higher than the official logistic model baseline.

### 4.2. NeXtVLAD

The NeXtVLAD network [12] is another frame level model that we use. It is an extension of the feature aggregation NetVLAD network [2].

Considering a video with $N$ frames and $D$ feature length. NetVLAD with $K$ clusters will encode each frame $x_i$ to $K \times D$ dimensional vector by describing frame $x_i$ using the distance between $x_i$ and cluster $c_j$ and soft assignment for each cluster:

$$\begin{aligned} &v_{ij} = \alpha_j(x_i)(x_i - c_j), \\ &i \in \{1,\ ...,\ N\},\ j \in \{1,\ ...,\ K\} \end{aligned} \tag{1}$$

where $c_j$ is the N dimension cluster anchor point and $\alpha_j(x_i)$ measures soft weight of cluster $c_j$ to $x_i$. The measurement function is modeled using a fully-connected layer with softmax activation:

$$\alpha_j(x_i) = \frac{e^{w_j^T x_i + b_j}}{\sum_{s=1}^{K} e^{w_s^T x_i + b_s}} \tag{2}$$

The video feature is obtained by adding all frame encoded features and concatenating descriptors from each cluster:

$$u = [\sum_i^N v_{i1},\ \sum_i^N v_{i2},\ ..., \sum_i^N v_{iK}]^T \tag{3}$$

In the NeXtVLAD aggregation network, as shown in Figure 1b, the frame feature $x_i$ is first mapping to $\lambda D$ dimensional space by linear transformation, denoted as $\hat{x}_i$. $\hat{x}_i$ is then splitted to $G$ lower-dimensional feature vectors $\{\tilde{x}_i^g | g \in \{1, ..., G\}\}$. The distance between feature vectors and clusters is calculated in the low dimension space:

$$\begin{aligned} &v_{ij}^g = \alpha_g(\hat{x}_i)\alpha_{gj}(\hat{x}_i)(\tilde{x}_{ij}^g - c_j), \\ &i \in \{1,\ ...,\ N\},\ j \in \{1,\ ...,\ K\},\ g \in \{1,\ ...,\ G\} \end{aligned} \tag{4}$$

where the proximity measurement of the decomposed vector $\tilde{x}_i^g$ consists of two parts:

$$\alpha_{gj}(\hat{x}_i) = \frac{e^{w_{gj}^T \hat{x}_i + b_{gj}}}{\sum_{s=1}^{K} e^{w_{gs}^T x_i + b_{gs}}} \tag{5}$$

$$\alpha_g(\hat{x}_i) = \sigma(w_g^T \hat{x}_i + b_g) \tag{6}$$

in which $\sigma(.)$ is a sigmoid function. $\alpha_{gk}(.)$ measures the soft assignment of $\hat{x}_i^g$ to the cluster $j$ while $\alpha_g(\hat{x}_i)$ is the learned attention over groups.

The final video feature is obtained by aggregating the encoded features over time and groups, similar to Eq.3:

$$u = [\sum_{ig} v_{i1}^g,\ \sum_{ig} v_{i2}^g,\ ..., \sum_{ig} v_{iK}^g]^T \tag{7}$$

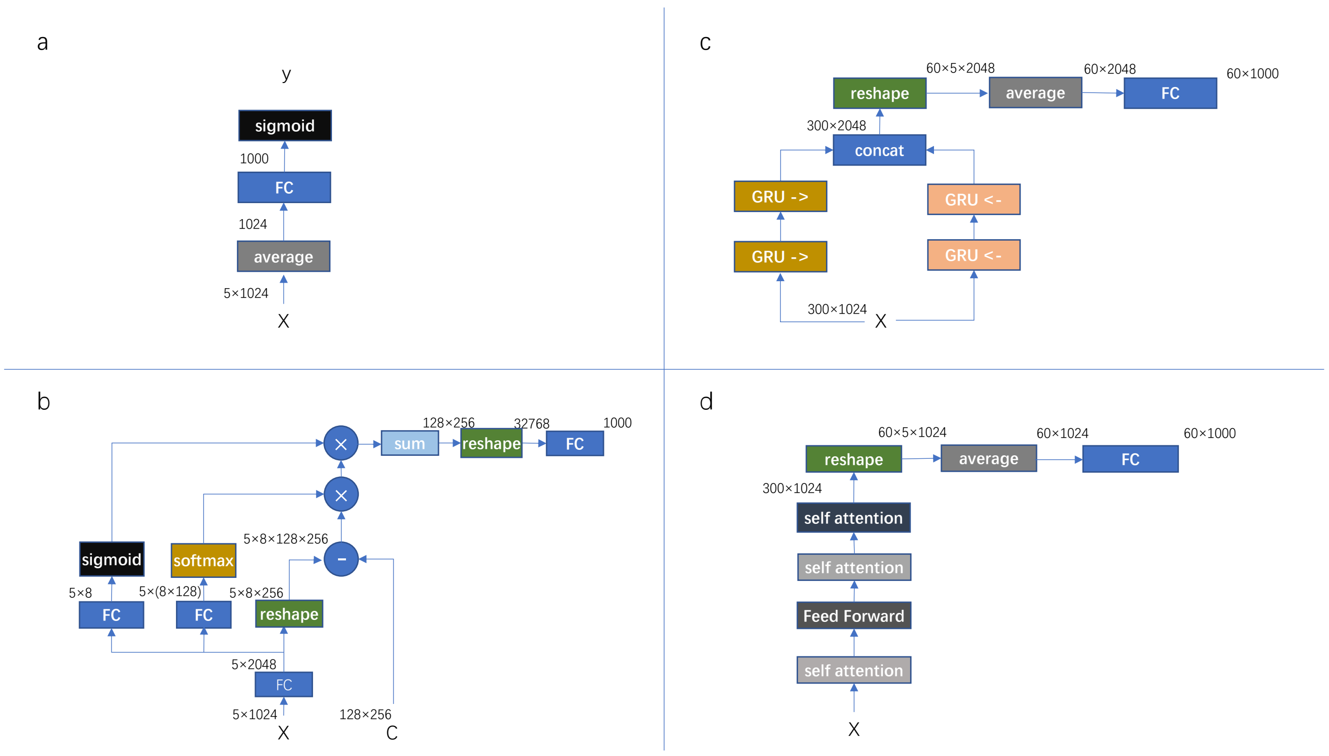

Figure 1. We describe the models that we use for this task. Some normalization layers and broadcast operations are ignored and we use only rgb features for simplicity. Numbers indicate the input/output dimension. For frame level model (a) and (b), the input feature $X$ is of length 5. For sequence model (c) and (d), the input is the entire video and is thus of length 300. (a) **Logistic Model:** the model averages features along time axis. Prediction is made for the averaged feature. (b) **NeXtVLAD:** we set cluster number to 128, cluster feature to 256 and group number to 8. The softmax operation is performed along cluster axis. $C$ is the learnable clusters. $\otimes$ denotes broadcasting element-wise multiplication, and $\ominus$ denotes broadcasting element-wise subtraction. (c) **BiGRU:** the entire video is first passed to forward GRU layers and backward GRU layers. Encoded vectors from forward GRU and backward GRU are then concatenated along the last dimension. Segment vectors are obtained by averaging encoded feature for every five time steps. (d) **Transformer:** the entire video is passed through two Transformer blocks. Average pooling is performed every five time steps to get the segment vectors

## 4.3. BiGRU

Gated recurrent units (GRU) [7] are gating mechanisms in recurrent neural network. We use GRU in both forward and backward directions (BiGRU) to encode video feature.

In one GRU cell, feature $x_t$ is encoded as following:

$$h_t = (1-z_t)\circ h_{t-1}+z_t\circ\sigma_h(w_h x_t+U_h(r_t\circ h_{t-1})+b_h) \quad (8)$$

where $h_{t-1}$ is the output of GRU cell in the last time step, $w, U, b$ are learnable parameters. $\sigma_h$ is the hyperbolic tangent. The update gate vector $z_t$ and the reset gate vector $r_t$ are represented as:

$$z_t = \sigma_g(w_z x_t + U_z h_{t-1} + b_z) \quad (9)$$

$$r_t = \sigma_g(w_r x_t + U_r h_{t-1} + b_r) \quad (10)$$

in which $\sigma_g$ is the sigmoid activation function.

The output feature for one GRU layer is obtained by concatenating the output vectors of the GRU cell in each time step. As shown in 1c, the entire video is first passed to forward GRU layers and backward GRU layers. Encoded vectors of forward GRU and backward GRU are then concatenated along the feature dimension. Segment vectors are obtained by performing average pooling every five time steps. Final predictions are made based on segment feature.

## 4.4. Transformer

Proposed in [21], Transformer is the working horse in most language tasks. We utilize Transformer to boost content localization performance by leveraging context information.

Set the hidden dimension within Transformer same as feature size $D$. In self-attention layer, the entire video input $X \in R^N \times R^D$ is first encoded to Query $Q$, Key $K$ and Value $V$ by three different linear transformations:

$$Q = Xw_q + b_q,\ Q \in R^N \times R^D \quad (11)$$

$$K = Xw_k + b_k,\ K \in R^N \times R^D \quad (12)$$

$$V = Xw_v + b_v,\ V \in R^N \times R^D \tag{13}$$

The encoded feature of the self-attention layer is calculated as follows:

$$U = \alpha_{softmax}(\frac{QK^T}{\sqrt{D}})V \tag{14}$$

$U$ is then taken into LayerNorm layer [3] and Feed Forward Layer [21] to get the output of the transformer block. As shown in Figure 1d, the entire video is passed through two Transformer blocks. Average pooling is performed every five time steps to get the segment vectors. Final predictions are made based on segment feature.

## 5. EM Process

We will describe our weakly supervised EM algorithm to leverage both video labels and segment labels in this section. In the rest of this section, $F$ denotes the model we use in the process. $\{(X, y)\}$ denotes video feature and label in YouTube-8m V2 dataset. $\{(X_s, y_s)\}$ denotes segment feature and label in YouTube-8m Segments dataset. We split a small portion of the segment dataset as validation dataset $\{(X_{s,val}, y_{s,val})\}$. We note $S$ the number of segments, $C$ the number of classes. $BCE$ denotes the binary cross entropy loss for multi-label tasks. The process is described in Algorithm 1.

### 5.1. Training Loss

Two kinds of losses, namely video loss and segment loss, are used in the EM process.

The video loss is used when we have only video level labels. The video prediction is made by averaging segment predictions:

$$\hat{y}_{video} = mean(\hat{y}_{seg}),\ \hat{y}_{seg} \in R^S \times R^C \tag{15}$$

The video loss could be represented as:

$$loss_{video}(y, \hat{y}_{video}) = BCE(y, \hat{y}_{video}) \tag{16}$$

The segment loss is used when segment labels are available. Note segment labels could be missing for certain classes and for some segments. The segment loss is thus calculated only for the classes and the segments where segment labels are available:

$$loss_{seg}(y_s, \hat{y}_{seg}) = BCE(y_s, \hat{y}_{seg} \circ \mathbb{1}_{y_s=0\ or\ 1}) \tag{17}$$

where $\mathbb{1}_{y_s=0\ or\ 1}$ is the masking function to mask segments and classes that is not annotated.

### 5.2. Model Initialization

Given the frame level model or sequence model $F$, the model is first pre-trained on dataset $\{(X, y)\}$ using video loss $loss_{video}$. The model is fine-tuned on the dataset $\{(X_s, y_s)\}$ using segment loss $loss_{seg}$.

```
Input:
    Video feature and label in yt8m-v2 dataset, $\{(X, y)\}$;
    Segment feature and label in yt8m seg, $\{(X_s, y_s)\}$;
    A small segment validation set, $\{(X_{s,val}, y_{s,val})\}$;
    Test set in yt8m seg, $\{(X_{s,test})\}$;
Output:
    Segment prediction for test set, $\{(\hat{y}_{s,test})\}$;
1:  Initialize model $F$;
2:  Pre-train $F$ on $\{(X, y)\}$;
3:  Fine-tune $F$ on $\{(X_s, y_s)\}$;
4:  for $i = 1$ to $iterations$ do
5:      E-step: use $F$ to predict segment label on $\{(X)\}$
        with threshold $\delta$, get $\{(X, \hat{y}_{s,1})\}$;
6:      E-step: use $F$ to predict segment label on $\{(X, y)\}$
        with threshold $\delta$ and using assumption that only
        video label could appear in the video, get
        $\{(X, \hat{y}_{s,2})\}$;
7:      M-step: train model $F_1$ on $\{(X, \hat{y}_{s,1})\}$;
8:      M-step: train model $F_2$ on $\{(X, \hat{y}_{s,2})\}$;
9:      Fine-tune $F_1, F_2$ on $\{(X_s, y_s)\}$;
10:     $F \leftarrow max(F_1, F_2)$ based on the performance on
        $\{(X_{s,val}, y_{s,val})\}$;
11: end for
12: Get the final prediction on $\{(\hat{y}_{s,test})\}$ on $\{(X_{s,test})\}$
    using $F$;
13: return $\{(\hat{y}_{s,test})\}$;
```

**Algorithm 1:** EM process for video content localisation

### 5.3. Expectation Step

The Expectation step is to estimate the segment annotations for dataset $\{X\}$. We propose two estimation methods.

The first method assumes that only video labels can appear in the video segments, so the estimated segment labels that are not belong to the corresponding video labels are masked out:

$$\hat{y}_{s,1} = \mathbb{1}_{F(X)>\delta} \circ \mathbb{1}_{y=1} \tag{18}$$

in which $\delta$ is the threshold.

In multi-label classification setting, some labels may be missing from the training set. As mentioned in [22], label correlation can improve the performance of the multi-label classifier. With this in mind, our second expectation method removes the video label prior condition during label estimation:

$$\hat{y}_{s,1} = \mathbb{1}_{F(X)>\delta} \tag{19}$$

### 5.4. Maximization Step

The maximization step optimizes model parameters based on current observed data. After previous expectation step, we have two method annotated datasets $\{(X, \hat{y}_{s,1})\}$ and $\{(X, \hat{y}_{s,2})\}$. Though noisy, these two sets are good choices for model pre-training. We train model $F_1$ and $F_2$

on $\{(X, \hat{y}_{s,1})\}$ and $\{(X, \hat{y}_{s,2})\}$ respectively using segment loss $loss_{seg}$. Since we do not have missing segment label in these two datasets, the masking function in $loss_{seg}$ will always output 1.

Estimated dataset $\{(X, \hat{y}_{s,1})\}$ and $\{(X, \hat{y}_{s,2})\}$ could be noisy and biased. Trained models $F_1$ and $F_2$ are thus probably also biased. We fine-tune $F_1$ and $F_2$ on the human-annotated dataset $\{(X_s, y_s)\}$.

Finally, we evaluate models $F_1$ and $F_2$ on validation set $\{(X_{s,val}, y_{s,val})\}$. The better model is picked for the next EM iteration.

## 6. Experiments

This section provides implementation details and presents our experimental results on the YouTube-8M dataset.

### 6.1. Implementation Details

Our implementation bases on the TensorFlow start code. All of the models are trained using the Adam optimizer [11] on two GPUs. We train one logistic model, two Transformers, one NeXtVLAD model and one BiGRU model. The logistic model and one transformer take parts in the segment label prediction steps. Other models are only pre-trained and fine-tuned in the last EM iteration to boost results. The final prediction is made by averaging outputs of two Transformers, one NeXtVLAD model and one BiGRU. Models are trained for five epochs for both pre-training and fine-tuning. The threshold for positive labels are set to 0.9.

For Transformer models, we train one Transformer using early fusion for rgb and audio feature and the other late fusion. Both of them are set to have two Transformer layers. We set learning rate to 0.00025 and batch size to 512 videos.

For NeXtVLAD network, we apply late fusion for rgb and audio feature. We use similar setting proposed in [12]. We set number of groups to 8, cluster size to 128, hidden dimension to 2048, dropout rate to 0.5. The base learning rate to 0.0004, batch size is set for 4800 segments.

For BiGRU model, we apply early fusion for rgb and audio feature. We use two GRU layers. The base learning rate is set to 0.00025 and batch size is set to 512 videos.

### 6.2. Model Evaluation

We evaluate the performance of our models in each EM iteration. In Table 6.2, the first block is the performance of the logistic model trained purely on YouTube-8M V2 dataset. The second block is the performance of models in the first EM iteration. The third block are models in the second EM iteration. The last block is the final result which aggregates models in the third block. The Transformer early model is pre-trained on the video-label-prior dataset $\{(X, \hat{y}_{s,1})\}$. The Transformer late, NeXtVLAD as well as BiGRU are pre-trained on the label-corrected dataset $\{(X, \hat{y}_{s,2})\}$.

| Model | Private mAP | Public mAP |
|---|---|---|
| Logistic | 0.715 | 0.723 |
| Transformer early | 0.773 | 0.781 |
| Transformer late | 0.770 | 0.773 |
| Transformer early | 0.787 | 0.796 |
| Transformer late | 0.792 | 0.800 |
| NeXtVLAD | 0.763 | 0.771 |
| BiGRU | - | - |
| Ensemble | 0.803 | 0.811 |

Table 1. The mAP scores of submissions during challenge. The first block is the performance of the logistic model trained purely on video label dataset. The second block is the performance of models in 1st EM iteration. The third block are models in 2nd EM iteration. The forth block is the final model which aggregates four models in the third block.

We can observe that models are becoming more and more accurate as EM algorithm performs. Among these different types of models, Transformer achieves the best result. The final ensemble model boost the performances due to the diversity of datasets and the diversity of models.

## 7. Conclusion

In this paper, frame level models and sequence models are studied to support temporal localization task for video understanding. We introduce a weakly supervised EM process which leverages the large amount of video classification dataset. By performing proposed EM process, our model achieves strong performance only using few supervised segment labels. Combining frame level models and sequence models with EM process, our final model achieves one of the top scores in the 3rd YouTube-8M video understanding challenge.

## References

[1] Sami Abu-El-Haija, Nisarg Kothari, Joonseok Lee, Paul Natsev, George Toderici, Balakrishnan Varadarajan, and Sudheendra Vijayanarasimhan. Youtube-8m: A large-scale video classification benchmark. *CoRR*, abs/1609.08675, 2016.

[2] Relja Arandjelovic, Petr Gronát, Akihiko Torii, Tomás Pajdla, and Josef Sivic. Netvlad: CNN architecture for weakly supervised place recognition. *CoRR*, abs/1511.07247, 2015.

[3] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization, 2016.

[4] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceed-*

*ings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015.

[5] João Carreira, Eric Noland, Andras Banki-Horvath, Chloe Hillier, and Andrew Zisserman. A short note about kinetics-600. *CoRR*, abs/1808.01340, 2018.

[6] João Carreira and Andrew Zisserman. Quo vadis, action recognition? A new model and the kinetics dataset. *CoRR*, abs/1705.07750, 2017.

[7] KyungHyun Cho, Bart van Merrienboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder-decoder approaches. *CoRR*, abs/1409.1259, 2014.

[8] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. *CoRR*, abs/1812.03982, 2018.

[9] Rohit Girdhar, João Carreira, Carl Doersch, and Andrew Zisserman. Video action transformer network. *CoRR*, abs/1812.02707, 2018.

[10] Will Kay, João Carreira, Karen Simonyan, Brian Zhang, Chloe Hillier, Sudheendra Vijayanarasimhan, Fabio Viola, Tim Green, Trevor Back, Paul Natsev, Mustafa Suleyman, and Andrew Zisserman. The kinetics human action video dataset. *CoRR*, abs/1705.06950, 2017.

[11] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization, 2014.

[12] Rongcheng Lin, Jing Xiao, and Jianping Fan. Nextvlad: An efficient neural network to aggregate frame-level features for large-scale video classification. *CoRR*, abs/1811.05014, 2018.

[13] Tianwei Lin, Xu Zhao, Haisheng Su, Chongjing Wang, and Ming Yang. BSN: boundary sensitive network for temporal action proposal generation. *CoRR*, abs/1806.02964, 2018.

[14] Antoine Miech, Ivan Laptev, and Josef Sivic. Learnable pooling with context gating for video classification. *CoRR*, abs/1706.06905, 2017.

[15] Mathew Monfort, Alex Andonian, Bolei Zhou, Kandan Ramakrishnan, Sarah Adel Bargal, Tom Yan, Lisa Brown, Quanfu Fan, Dan Gutfruend, Carl Vondrick, et al. Moments in time dataset: one million videos for event understanding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–8, 2019.

[16] Phuc Nguyen, Ting Liu, Gautam Prasad, and Bohyung Han. Weakly supervised action localization by sparse temporal pooling network, 2017.

[17] Kamal Nigam, Andrew McCallum, and Tom Mitchell. Semi-supervised text classification using em. *Semi-Supervised Learning*, pages 33–56, 2006.

[18] Sujoy Paul, Sourya Roy, and Amit K Roy-Chowdhury. W-talc: Weakly-supervised temporal activity localization and classification, 2018.

[19] Zhaofan Qiu, Ting Yao, and Tao Mei. Learning spatio-temporal representation with pseudo-3d residual networks. *CoRR*, abs/1711.10305, 2017.

[20] Du Tran, Lubomir D. Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. C3D: generic features for video analysis. *CoRR*, abs/1412.0767, 2014.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *CoRR*, abs/1706.03762, 2017.

[22] He-Da Wang, Teng Zhang, and Ji Wu. The monkeytyping solution to the youtube-8m video understanding challenge. *CoRR*, abs/1706.05150, 2017.

[23] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection, 2017.

[24] Chao-Yuan Wu, Christoph Feichtenhofer, Haoqi Fan, Kaiming He, Philipp Krähenbühl, and Ross B. Girshick. Long-term feature banks for detailed video understanding. *CoRR*, abs/1812.05038, 2018.

[25] Jin Xia, Jiajun Tang, and Cewu Lu. Three branches: Detecting actions with richer features, 2019.

[26] Saining Xie, Chen Sun, Jonathan Huang, Zhuowen Tu, and Kevin Murphy. Rethinking spatiotemporal feature learning for video understanding. *CoRR*, abs/1712.04851, 2017.

[27] Yuan Yuan, Yueming Lyu, Xi Shen, Ivor W. Tsang, and Dit-Yan Yeung. Marginalized average attentional network for weakly-supervised learning, 2019.