Efficient Temporal Action Localization via Vision-Language Modelling

An Empirical Study on the STALE Model's Efficiency and Generalizability in Resource-constrained Environments

Yunhan Wang

Supervisors: Jan van Gemert, Robert-Jan Bruintjes, Attila Lengyel, Ombretta Strafforello

EEMCS, Delft University of Technology, The Netherlands

A Thesis Submitted to EEMCS Faculty Delft University of Technology,
In Partial Fulfilment of the Requirements
For the Bachelor of Computer Science and Engineering
June 25, 2023

Name of the student: Yunhan Wang
Final project course: CSE3000 Research Project
Thesis committee: Jan van Gemert, Ombretta Strafforello, Attila Lengyel, Robert-Jan Bruintjes, Petr Kellnhofer

An electronic version of this thesis is available at http://repository.tudelft.nl/.

Abstract

Temporal Action Localization (TAL) aims to localize the start and end times of actions in untrimmed videos and classify the corresponding action types. TAL plays an important role in understanding video. Existing TAL approaches heavily rely on deep learning and require large-scale data and expensive training processes. Recent advances in Contrastive Language-Image Pre-Training (CLIP) have brought vision-language modeling into the field of TAL. While current CLIP-based TAL methods have been proven to be effective, their capabilities under data and compute-limited settings are not explored. In this paper, we have investigated the data and compute efficiencies of the CLIP-based STALE model. We evaluate the model performances under data-limited open/close-set scenarios. We find that STALE can demonstrate adequate generalizability using limited data. We experimented with the training time, inference time, GPU utilization, MACs, and memory consumption of STALE by inputting with varying video lengths. We discover an optimal input length for STALE to inference. Using model quantization, we find a significant forward time reduction for STALE on a single CPU. Our findings shed light on the capabilities and limitations of CLIP-based TAL methods under constrained data and compute resources. The insights gained from this research contribute to enhancing the efficiency and applicability of CLIP-based TAL techniques in real-world scenarios. The results provide valuable guidance for future advancements in CLIP-based TAL models and their potential for broader adoption in resource-constrained environments.

1. Introduction

With the rapid growth of video media and advances in deep learning, there has been a significant surge in interest and focus on deep learning-based video understanding. Temporal Action Localization (TAL) is one of the key tasks for video understanding. TAL concerns the detection of when and what actions happen given an untrimmed video. Applications of TAL include video summarization [65], behavior analysis [7], and human-robot interaction [45]. Current TAL methods [46, 52, 55, 67] rely on deep learning and mostly focus on delivering high-performing models. However, decent performance typically requires massive amounts of data and computationally expensive training processes [2]. To tackle this issue with limited resources, it is urgent to reconsider and develop new approaches that are data- and compute-efficient. Moreover, we observe a trend to conduct TAL via vision-language modeling, which is based on Contrastive Language-Image Pre-Training (CLIP) [41].

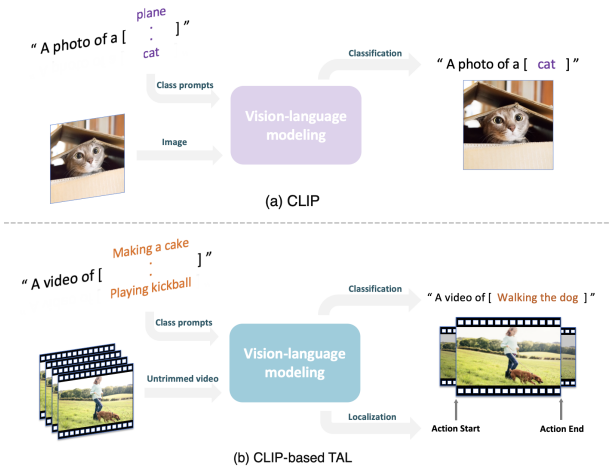

Figure 1. Illustration of the CLIP model [41] (a) and a one-stage pipeline for CLIP-based TAL (b). The classes are treated as textual prompts and are trained to pair with vision instances. In (b), the outputs are the action class and action start/end time for each action instance in each untrimmed video.

Recent research on CLIP has gained wide attention, which demonstrates the effectiveness of pairing natural language signals with vision through contrastive learning [41]. As illustrated in Figure 1a, CLIP takes an image and all possible textual descriptions of this image to output the description that best matches the image. During training, the vision-language module encodes the image and its corresponding caption independently into latent embeddings. Then, CLIP attempts to pair the text embedding with the image embedding via contrastive learning. Benefiting from CLIP's large-scale image-caption data, it has shown the notable capability of "zero-shot" generalization, such that unseen class in unseen input can be predicted.

Subsequently, CLIP-based methods [19, 36] are adopted in video tasks such as TAL and demonstrated remarkable zero-shot capability. As shown in Figure 1b, a one-stage CLIP-based TAL method takes untrimmed video and textual prompt of action class as input and outputs the action class that best describes the action instance as well as action start/end time. Traditional TAL methods encode action classes into one-hot vectors which lose the semantic meaning of these classes. Furthermore, open domain recognition is still a challenging topic, and natural language signals in CLIP-based models have natural advantages to help recognize unseen or more granular actions than the commonly used one-hot encoding for action labels. Therefore, it is worth exploring the full capacity of

CLIP-based TAL methods to generalize on a target domain without being trained or significantly trained on this domain, as well as evaluating their compute efficiency. We hereby raise a research question: *How well do current CLIP-based TAL methods perform and generalize in a limited data and compute setting?*

In this study, we conducted a comprehensive evaluation of the STALE model's data and compute efficiencies. This research makes the following contributions: (1) We find that STALE can demonstrate adequate generalizability using limited data. (2) We discover an optimal input length for STALE to inference, and (3) using model quantization there is significant forward time reduction for STALE on CPU. These findings have positive implications for enhancing the efficiency and practicality of CLIP-based TAL techniques in real-world scenarios. Our results provide guidance for future advancements in CLIP-based TAL models, paving the way for broader adoption in resource-constrained environments.

2. Related Work

Action recognition Action recognition concerns the interpretation of human actions from videos, each video is trimmed to contain frames of one action instance. While deep convolutional neural networks (CNN) outperformed classical methods in diverse image tasks [21, 40], they were yet to show significant improvement in action recognition [20,69]. A Two-Stream Network [50] was proposed to add another optical flow stream to incorporate temporal and spatial information and demonstrated significant improvement in action recognition. This approach then became one of the mainstream model architectures for action recognition [10,13,57]. Another trend was the use of 3D CNN, such as I3D [6], R3D [16], SlowFast [12]. 3D CNN uses an additional dimension of convolutional layers to model temporal information. I3D proposed to inflate 2D ConvNets into 3D in a two-stream network.

Recent research has demonstrated that methods based on Vision Transformers [11,30,42,66] achieve significantly better performance in image tasks compared to previous CNN-based methods. Adopting similar ideas from video CNNs to transform 2D transformers to 3D to incorporate temporal information, video transformers [1, 31, 39] have gained popularity in the field of video understanding due to their exceptional performance, establishing them as a mainstream approach. Action recognition models can be used to predict action types from video proposals in solving TAL [19]. In our work for TAL, before localization and classification, we adopt a temporal vision transformer to model the temporal and spatial information of each video.

Temporal action localization Temporal action localization, sometimes known as temporal action detection, aims to localize the start and end time of actions in untrimmed videos and classify the corresponding action types. A video clip can include more than one action instance and background frames without actions. Unlike action recognition, most models for TAL follow a proposal-then-prediction two-stage paradigm [53]. These models first propose frames that can contain actions and attempt to classify the action types in these frames [62].

There are two main approaches to generating proposals: anchor-based and anchor-free. Anchor-based methods [14, 15, 47] generate temporal proposals in videos by distributing dense and multiscale intervals of pre-defined lengths across uniformly distributed temporal locations in the input. Anchor-free methods [29,68] are often based on predicting actionness, startness, and endness scores (probability of action occurring, starting or ending at a temporal position of the video). They are capable to generate proposals with precise boundaries and flexible duration. At the prediction stage, models from action recognition can be incorporated to classify action types from proposals. However, two-stage methods suffer from localization errors propagated to prediction. One-stage framework [4, 28] tackles proposal and prediction simultaneously hence alleviating the propagation of localization errors. Two commonly used datasets to evaluate TAL models are Thumos14 [37] and ActivityNet1.3 [18]. Differing from most TAL works that target at best performance on these datasets, we aim to investigate the model performance under resource-constrained scenarios.

Since 2019, we observe a trend of adopting the classification results from UntrimmedNets [56] for uncertain predictions [26, 27, 36, 63] in ActivityNet. These uncertain predictions normally arise from videos of classes that have relatively low data. UntrimmedNets is a weakly-supervised method that performs well in such videos. To fairly and comprehensively evaluate the performance of our selected model in data-limited settings, we report the results with/without the use of predictions from UntrimmedNets.

Vision-language models Prior research [34] has investigated the relationship between images and words through the use of paired text documents. Recently, CLIP [41] demonstrated outstanding performance of vision-language modeling through large-scale training. Using contrastive learning to pair images with natural language signals, CLIP has shown that paired image-caption data can be leveraged to learn powerful visual representations for zero-shot recognition.

Subsequently, CLIP was widely used as the backbone model for vision tasks that accompany natural semantic meaning, such as image captioning [33], object detection [25], semantic segmentation [24]. Since each action instance contains natural semantic meaning, ActionCLIP [58] was proposed to incorporate CLIP for action recognition and investigate its zero-shot/few-shot capability. Efficient-prompt [19] introduced a lightweight temporal information encoder with a transformer for efficient training and sets baselines for zero-shot action recognition and TAL. Unlike the two-stage framework in Efficient-prompt for TAL, STALE [36] proposed a one-stage CLIP-based method to reduce localization error propagation in two-stage TAL and outperformed Efficient-prompt in zero-shot scenarios. In our work, we aim to explore the efficiency of TAL via vision language modeling, the one-stage CLIP-based STALE is selected to conduct experiments.

Data efficiency Data efficient deep learning has gained increasing attention [3, 22, 59] recently. In video tasks, the self-supervised VideoMAEs [52, 55] were claimed to be data efficient learners with the Masked Autoencoders (MAE) [17] for data augmentation. However, VideoMAEs still require the full dataset to train, and hence are not in "limited data settings". Zero-shot/few-shot video understanding [19, 36, 43, 58] can be seen to be data efficient as they demand few or zero training samples from the target classes to predict. Zero-shot is also known as the open-set scenario. Closed-set scenario refers that all classes in the test set also exist in the training set. Oosterbaan *et al.* demonstrated the performances of predicting a rare class using a classifier trained on different amounts of synthetic samples of that rare class plus the original training data [38]. We perform data efficiency experiments by training on subsets of the original training data in both open-set and closed-set scenarios.

Compute efficiency Common metrics to evaluate a model's compute efficiency are the number of floating-point operations (FLOPs), the number of multiply-accumulate operations (MACs), memory consumption, the number of model parameters, and training/inference time [2]. Hardware differences in used GPUs can lead to different model performances [51]. Approaches to enhance compute efficiency include using mixed precision to store decimal numbers to reduce memory consumption [32], and using quantization to reduce inference time [5]. Quantization in deep learning refers to the technique of approximating a floating-point-based neural network with a neural network that uses low bit-width numbers to alleviate computation. We adopt MACs, GPU memory consumption, and training/inference time to measure the compute efficiency in our experiments due to their representativeness in quantifying a model's compute performance. We measure how well a GPU is utilized during model inference by dividing the experimental MACs/s by the theoretical maximum MACs/s of the GPU. We further study the extent that our selected model's inference time can be reduced with quantization.

3. Methodology

3.1. Temporal Action Localization

To formally define TAL, we first denote V_i as an untrimmed video, each V_i in the training set $D_{\text{train}} = \{(V_i, \Psi_i)\}_{i=1}^{N}$ is labeled with $\Psi_i = \{(a_j, e_j, y_j)\}_{j=1}^{M_i}$, where M_i denotes the number of action instances in V_i, and a_j/a_j represent the action start/end time for the action instance j. The action class of an instance is denoted as y_j. Given D_{train}, TAL attempts to predict each action instance in each video in the corresponding test set D_{test}.

Metrics Mean Average Precision (mAP) is TAL's most commonly adopted evaluation metric. Average Precision (AP) is the average precision of all videos of a class, and mAP is the average precision of all testing videos of each class. Temporal IoU (tIoU) is the ratio of the temporal intersection divided by the union between two temporal intervals in a video. Denote I_p as the predicted temporal interval likely to contain an action of interest and denote I_g as its closest interval of a ground-truth action. $\text{tIoU}(I_p, I_g) = \frac{I_p \cap I_g}{I_p \cup I_g}$.

To declare a prediction as true positive, the predicted class should match the ground truth with a tIoU score above a given threshold. Different thresholds hence affect mAP. A dataset typically pre-defined specific thresholds and Average-mAP, average of the mAP scores under all pre-defined thresholds, is used to compare the performance of different models.

Closed and open-set TAL There are two main scenarios to examine methods for TAL: closed-set and open-set. Let $C_{\text{train}}/C_{\text{test}}$ denote the sets of action classes that exist in $D_{\text{train}}/D_{\text{test}}$. In the closed-set scenario for TAL, the action classes in the training and test sets are identical, such that $C_{\text{train}} = C_{\text{test}}$. While in open-set TAL, the actions classes for training and testing are disjoint, such that $C_{\text{train}} \cap C_{\text{test}} = \emptyset$. Open-set TAL can be referred to as zero-shot TAL: zero samples of target classes are given to train and the goal is to predict each unseen class in each unseen video.

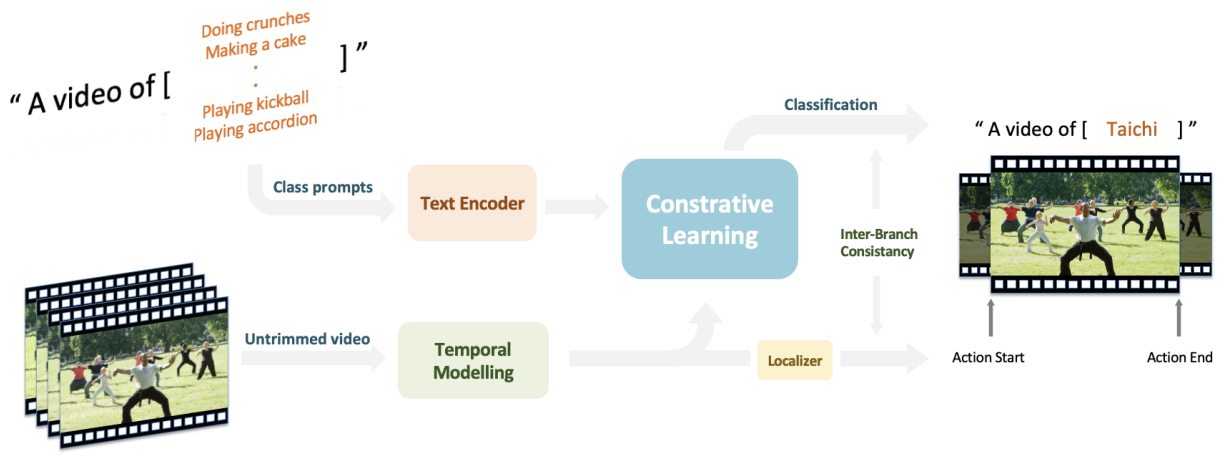

Figure 2. Overview of the STALE model [36]. Action class prompts and videos are encoded and modeled then given to learn their inter-relationships via conservative learning. A localizer is used to localize the action instance. Finally, classification and localization are refined to output consistently using an inter-branch consistency loss.

3.2. Model

Zero-Shot Temporal Action Detection via Vision-Language Prompting (STALE) [36] is selected to conduct data and compute efficiency experiments, which is the most recent and state-of-the-art method on zero-shot TAL. The overview of STALE can be seen in Figure 2. STALE uses a temporal vision transformer [11] to encode videos into video embeddings and a text transformer [54] to encode class prompts into text embeddings. The embedding of a video is divided into snippets and each snippet is masked with a ground truth label. The contrastive learning module receives the text and masked video embeddings and attempts to learn how text embeddings are matched with masked video embeddings via cross-attention [54]. A localizer module is used to learn to localize the action instances in parallel with classification. To ensure the classified action instance is consistent with the localized start/end time, an inter-branch consistency loss is equipped between the classification and the localization branch. At inference time, all possible action classes are given as textual prompts, and STALE predicts the prompt that best matches the given video.

3.3. Data Efficiency

To evaluate STALE in a data-limited closed-set scenario, we uniformly sampled different amounts of the training sets without replacement. The selected amounts in percentage are 10%, 20%, 40%, 60%, 80%. We further restrict that all action classes were represented by at least one sample in the subset. While the zero-shot setting already demonstrated data efficiency: to predict target classes without using any data of target classes, it is interesting to see if the STALE still generalizes in a data-limited open-set scenario. We used the same sampling method for closed-set to sample 50% of training data. We conducted open-set experiments with both 75/25 and 50/50 class splits, that is, only training samples of 75% or 50% selected seen classes were used to train, and evaluate the test samples of the remaining 25% or 50% unseen classes. We use the same closed-set and open-set train splits as provided in [36]. To ensure statistical significance in our results, the experiments were repeated five times to take average results. We did not modify the test set.

Score enhancement We discovered that STALE uses the following technique during post-processing: if STALE's confidence score in a prediction of a video is lower than the confidence score of the weakly-supervised UntrimmedNets [56], then STALE will adopt the prediction of UntrimmedNets. This technique is referred to as score enhancement and we study STALE's performance with and without the use of score enhancement for data efficiency experiments.

3.4. Compute Efficiency

To evaluate STALE's compute efficiency for training, we record the training time and produced Average-mAP in the closed-set scenario. We repeat this experiment five times with the same training configuration and using the same GPU.

To evaluate STALE's compute efficiency for inference, we input video samples of varying lengths to STALE. During the inference of each sample, we record different compute metrics. The video samples are randomly generated PyTorch tensors. We adopt MACs,

Class Split	Enhanced Score	Avg Train Data	ActivityNet v1.3			
			0.5	0.75	0.95	Avg
75% Seen 25% Unseen	with	6575	39.5	21.5	4.2	21.7
		3304	39.1	22.4	3.7	21.7
	w/o	6575	15.7	8.0	1.4	8.4
		3304	15.4	8.5	1.7	8.5
50% Seen 50% Unseen	with	4353	39.1	20.1	4.1	21.1
		2198	38.7	20.3	4.0	21.0
	w/o	4353	3.0	1.9	0.4	1.8
		2198	3.6	1.9	0.5	2.0

Table 1. Comparison of open-set mAP results of STALE with/without the use of score enhancement and trained on all or roughly half of data. Class split refers to the split of the portion of classes for STALE to learn and the remaining portion of classes for STALE to conduct zero-shot prediction. STALE can generalize comparably with roughly 50% of training data.

the number of multiply-accumulate operations, to calculate the computing power consumed by STALE during inference. We use fvcore [44] developed by Facebook Research to derive MACs. We record the inference time to calculate the MACs per second. For the GPU utilization score, we divide the recorded MACs/s by the theoretical MACs/s of the used GPU to demonstrate what percentage of GPU's computing power is utilized to inference a particular input. We also record the memory consumption during inference. We use CUDA's event function to record time and configure it to wait for all kernels in all streams on a CUDA device to complete before calculating inference time. We repeat this experiment five times using the same GPU. To enable the STALE model to process, we change the temporal scale parameter of STALE to be equal to the input video length.

Quantization Quantization aims to approximate the same model outcome while replacing model weights with lower precision to reduce computation. An 8-bit integer precision typically leads to an accuracy loss of less than 1% in neural networks [61]. The original STALE model is configured to use 32-bit float as the precision for model weights. We config all linear transformations and convolutional layers in STALE to be quantized using Pytorch's dynamic quantization. We record the time consumption of model forward passes with varying input lengths. Furthermore, since GPUs are not widely considered affordable, we perform this experiment five times with a single CPU with ample memory. We enable only a single thread for inference. We compare the quantization effects across 8-bit integer, 16-bit float, and 32-bit float (baseline) on a single CPU as well as 32-bit float on a single GPU.

4. Experiments

ActivityNet We adopt ActivityNet1.3 to conduct experiments. The original ActivityNet1.3 dataset contains 10,024 videos for training and 5,044 videos for testing, with 200 action classes. To alleviate the massive training time caused by using raw videos to train, we adopted the CLIP pre-processed video features provided by [36]. The videos were processed frame by frame through CLIP [41]. The provided video features correspond to 8840 training videos and 4350 testing videos. The mismatch between the number of samples from our ActivityNet and the original ActivityNet is because many raw videos are no longer downloadable with their links. The commonly used tIoUs for ActivityNet are 0.5, 0.75, and 0.95.

4.1. Data Efficiency in Open-set

Can STALE's zero-shot ability still hold with limited training data? We uniformly sample subsets containing 50% samples of the original training set of ActivityNet. We ensure all classes for the model to "see" are represented at least once in these subsets for zero-shot learning. The full training set contains 6575 and 4353 samples for 75/25 and 50/50 splits respectively. On average in five experiments, 50.2% of training data are used for data-limited 75/25 zero-shot learning and 50.5% for 50/50 zero-shot learning. We further compare results with or without score enhancement.

The experimental results are shown in Table 1. With score enhancement and the full training set, we see a slight drop in average-mAP shifting from 75/25 split to 50/50 split (21.7 $\rightarrow$ 21.1) and a significant drop if we disable score enhancement (8.4 $\rightarrow$ 1.8). This is because the 50/50 split has fewer classes to "see" and more classes to conduct zero-shot prediction.

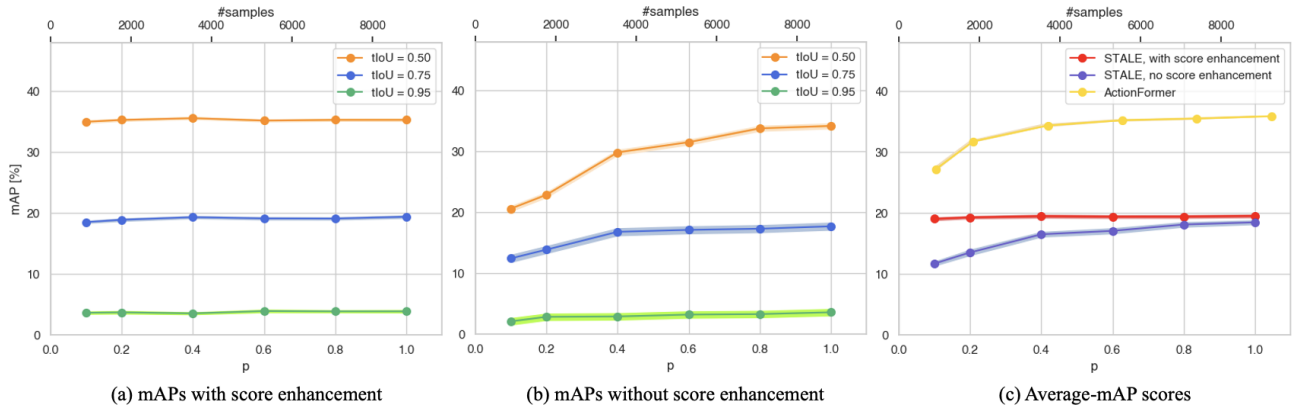

Figure 3. Closed-set mAP scores produced by STALE with different p training data amounts with score enhancement (a), without score enhancement (b), and average-mAPs comparison between STALE with/without score enhancement and ActionFormer [67] (c). We further plot the indicators of the numbers of training samples at the top of each subplot. The ActivityNet dataset used by ActionFormer contains 9251 total training samples, while STALE used 8840 samples. Without score enhancement and $p = 0.8$, STALE can reach closely to the average-mAP produced by STALE when $p = 1.0$.

Within the same split and using the full training set, we observe a huge decrease in all mAP results if we disable score enhancement. This demonstrates that the use of score enhancement helps with STALE's zero-shot performance substantially. Moreover, for both splits, no matter whether we use score enhancement or not, training with roughly half of the original samples does lead to notable changes in mAP. This suggests that STALE's zero-shot ability still holds with even 50% of the training data.

4.2. Data Efficiency in Closed-set

How does STALE perform and generalize in the closed-set scenario as we gradually increase the amount of training data? We uniformly sample 10%, 20%, 40%, 60%, and 80% of training data from the training set. As shown in Figure 3a, with varying amounts of training data, the mAP produced with score enhancement demonstrated a stable tendency under all tIoUs. However, in Figure 3b, if we disable score enhancement, we can observe a clear learning curve with an increasing amount of training data under each tIoU. The tIoU of 0.50 produced the most rapid increase, followed by the tIoU of 0.75. The increasing tendency under the tIoU of 0.95 is much less insignificant.

The overall tendency of performances with/without score enhancement can be reflected in Figure 3c, where the average-mAPs are plotted. We can only observe an overall learning trend of STALE if we disable score enhancement. The average-mAP curve without score enhancement tends to converge to the flat one produced with score enhancement but never surpasses it. The reason why the curves with score enhancement are always flat can be that the STALE's confidence scores are largely bounded by UntrimmedNets' confidence scores [56]. STALE with 80% of training data, that is roughly 7000 samples can produce an average-mAP of less than 0.5 smaller than it trained with the full dataset of 8840 samples.

Furthermore, we plot a similar average-mAP curve of ActionFormer [67] for data efficiency as reported in [60], where ActionFormer was trained and tested in its original configuration. ActionFormer is a recent TAL model based on a vision transformer architecture. It is noticed that this is not a direct comparison between STALE and ActionFormer since the ActivityNet1.3 they used contains different numbers of training and testing samples. The sampled subsets for data efficiency are not the same for these two evaluations. As demonstrated in Figure 3c, while STALE is the current state-of-the-art in the open-set scenario, there is still a considerable amount of gap from reaching the state-of-the-art in the full-data or data-limited closed-set scenario.

4.3. Compute-Efficient TAL

Configuration In the data-efficiency experiments, we fixed all parameters from the original STALE release [36] throughout our experiments, including random seed. To ensure certain variability in input tensors, for each of the experiments, we choose a different random seed. The random seed is equal to the experiment number (0-4) multiplied by 100. We used the NVIDIA V100S GPUs [8] consistently for our experiments. The reported theoretical MACs/s of the V100 GPU is 8.2 TMACs per second. We configure CUDA to be non-deterministic. We use Intel's Cascade Lake refresh CPU [9], with 8GB memory for the quantiza-

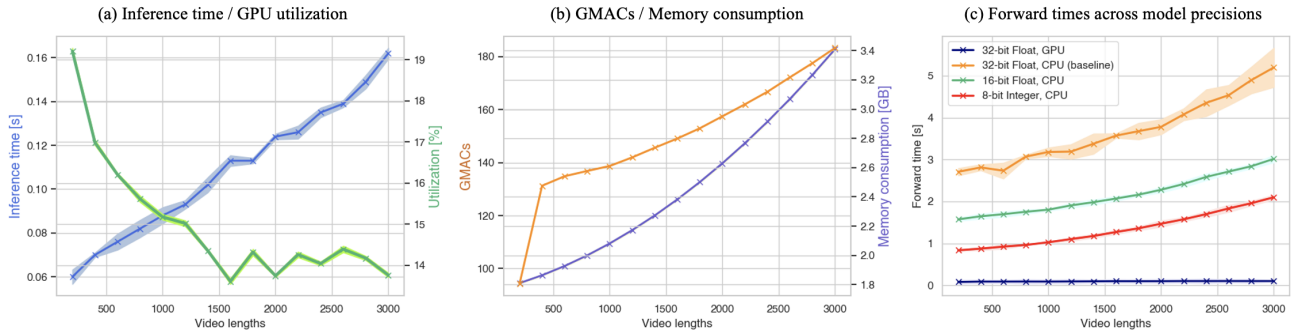

Figure 4. Inference time and GPU utilization (a), GMACs and GPU memory consumption in GB (b) with varying input video lengths. We record these metrics for a full inference cycle for each video. Sub-plot (c) demonstrates the effects of quantization on model forward pass time across different precisions on a CPU. We vary input video length. The forward time on GPU has a linear increasing tendency from 0.080s at length 200 to 0.101s at length 3000. A video length of 200 can lead to the best GPU utilization. Quantizing STALE into 8-bit precision can drastically reduce forward time on the CPU compared to the original 32-bit.

tion experiments. 8GB is ample to run STALE since STALE consumes less than 1GB of memory. All experiments are conducted in the Linux environment on the DelftBlue high-performance computer [9].

Table 2. Comparison of training time, number of model parameters (in millions), average-mAP between ActionFormer (AF) [67] and STALE [36] on ActivityNet.

Model	#params[M]	Time[s]	Avg-mAP
STALE	170.7	400.7	19.4
AF	29.3	1944.9	35.9

Training efficiency We record and compare the training efficiency of STALE with ActionFormer [60]. Both experiments are conducted under the same methodology and training configuration from their original code release. ActivityNet used by STALE contains 8840 training samples, while STALE used 9251 samples. STALE's average-mAP is obtained using score enhancement. As shown in Table 2, while STALE's trainable model parameters are 5.8 times higher than ActionFormer, STALE's training time consumption is only 20% of ActionFormer's. However, ActionFormer outperformed STALE by 16.5 in average-mAP. We hence suggest the use of STALE if short training time is favored. Moreover, STALE/ActionFormer uses video features processed by CLIP [41] or I3D [6] rather than raw videos. They both use the predictions and confidence scores of UntrimmedNets. Thus, the time of pre-processing video features and training UntrimmedNets should also be considered if training on a new dataset.

Inference efficiency How much computing power does STALE consume and utilize for inferencing videos of different lengths? By inputting videos of lengths spanning from 200 to 3000 with 200 increments, we record each video's time, GPU memory, and MACs of an inference cycle and calculate the GPU utilization score. As shown in Figure 4, the inference time and memory consumption increase linearly as we increase video length. The colored area along the curves are standard deviations of those data points. We observe a relatively large variability exhibited in model inference time. In Figure 4b, GMACs also follow a similar smooth increasing trend after increasing video length from 400. It is noticed a significant jump in GMACs changing from a video length of 200 to 400. On the opposite, in Figure 4a, GPU utilization gradually drops as we increase video lengths and converge to around 14% since a video length of 1400. This suggests that to let STALE better utilize the computing power of GPU for inference, it is recommended to input videos of smaller lengths, such as 200. This can produce a GPU utilization score of around 19%.

Quantization on forward time How does model quantization of different precisions affect forward pass time? The original weights of STALE are in 32-bit float. We first record forward times using the 32-bit STALE on a CPU and a GPU. Then, we quantize STALE into 16-bit float and 8-bit integer to record forward times on the CPU. Figure 4c demonstrates the forward time curves across precisions. While the 32-bit STALE on GPU produces an increasing trend in forward time from 0.080s to 0.101s, this curve has yet to show a notable tendency since other curves increase on a greater scale. Quantizing model into lower precisions

should theoretically lead to a faster forward pass. This is evident from those curves on a CPU. 8-bit integer consistently outperforms 16-bit and 32-bit float. 32-bit float on a CPU produces a relatively rugged curve and higher variability. Overall, the quantization of STALE from 32-bit to 8-bit reduced CPU forward pass time by a factor of 2.73. The memory consumption of STALE also decreased by a factor of 2.44. However, we discover that the forward time of 32-bit STALE on GPU can outperform 8-bit STALE on CPU by a factor of 14.15. This is because STALE running on GPU is intensively parallelized through Pytorch and CUDA.

5. Discussion

In this study, we conducted a comprehensive investigation into the data and compute efficiencies of the CLIP-based STALE model. Our evaluation encompassed the model's performance analysis within data-limited scenarios, specifically focusing on both open-set and close-set scenarios. Furthermore, we conducted experiments utilizing varying video lengths to assess the training/inference time, GPU utilization, MACs, and memory consumption of the STALE model. We also accessed the power of model quantization on the forward pass time and model memory consumption.

From our experiments, we conclude the following points. First, the CLIP-based STALE model demonstrates adequate generalizability with limited training data in the open-set scenario. In a closed-set setting, without score enhancement, STALE trained with 7000 samples can produce comparable results to that trained with 8840 samples. With score enhancement, STALE can consistently produce comparable results with only 884 training samples. Second, STALE has a considerable gap in mAP performance compared with recent transformed-based TAL models such as ActionFormer. However, STALE's training time is also significantly shorter, this property can facilitate sectors that prefer shorter training time. Third, during model inference, we discover that STALE can utilize GPU better when the input video length is small. We recommend setting the input video length to smaller than or equal to 200 for fast inference and higher computational efficiency. Lastly, 8-bit model quantization can remarkably reduce the time consumption of forward pass on STALE with a single CPU.

There are limitations exhibited in our research. We did not perform data efficiency experiments on Thumos14 due to the lack of CLIP pre-processed video features. A future supplementary investigation can be re-performing the data efficiency experiments on Thumos14 to support our conclusions. Moreover, we can investigate other techniques to boost model compute efficiency, such as distributed training with multiple GPUs, and multi-threaded inference. We observe another promising direction to add prior knowledge of physics [23] to the model to enhance data efficiency. What and how physics priors can be incorporated to help with TAL is a promising research direction since TAL inherently deals with physical activities.

The outcomes of our study provide valuable insights into both the strengths and limitations of CLIP-based TAL methods when confronted with limited data and computing resources. These findings have significant implications for improving the efficiency and practicality of CLIP-based TAL techniques in real-world settings. Moreover, our results offer valuable guidance for future advancements in CLIP-based TAL models and their potential for wider adoption, particularly in environments with resource constraints.

6. Responsible Research

Our research was independently conducted for pure research purposes, and there are no conflicts of interest to disclose. We have taken numerous measures throughout the research process to ensure the integrity of our findings and adhere to rigorous ethical standards. We evaluate our methodology based on the following primary aspects: ethical implications, integrity, and reproducibility of our research.

6.1. Ethical Implications

Data privacy Since ActivityNet [18] concentrates on human activities, it is necessary to consider the privacy of the human subjects in the data used. The videos in ActivityNet were collected from publicly available sources such as YouTube. The search queries were purely text-based such as "Preparing pasta" and hence contained no information about any target groups. Moreover, during the annotation phase, only information concerning action instances is labeled to minimize the risk of person re-identification. We further ensure data privacy and anonymity by adopting pre-processed video features instead of raw videos in our research.

Data annotation The annotations of ActivityNet were collected by Amazon Mechanical Turk (MTurk) workers [18]. As indicated in [35], (1) the majority of participants do not perceive MTurk as a source of stress, nor do they encounter abusive behavior from requesters. (2) MTurk provides flexibility and benefits that are highly valued by most individuals. Hence, the ethical concerns regarding annotation workers can be largely alleviated. When annotating data through crowdworkers, we strongly encourage employers to ad-

here to these standards for responsible research, according to Silberman *et al*. [48, 49]: (1) Pay workers at least minimum wage. (2) Respond quickly, clearly, concisely, and respectfully to worker questions and feedback. (3) Offering workers legal protections.

Bias Machine learning models can be biased towards some particular features if trained inappropriately. TAL models typically process spatial information hence certain human characteristics can be recorded during training, such as the ethnicity and gender of a person acting. To our best knowledge, there is no study on human characteristics in ActivityNet. Thus, when deploying a TAL model, we advise thoroughly checking if the model is biased toward a particular group. Possible solutions include using a diverse and representative training dataset, using adversarial training [64] which encourages the model to ignore irrelevant attributes such as gender or race when making predictions.

Surveillance One of the potential applications of TAL is to monitor human activities in the background. For example, it can detect whether any worker is sleeping in an office. However, current spatial-temporal action localization methods do not inherently allow human identification. Hence, there is still human justification required to identify who is acting and the surveillance system is yet to be fully automated to target a human subject. When deploying a TAL-based surveillance system, we advise ensuring that individuals are aware of the presence of the surveillance system and its purpose. We also suggest limiting the use of the system to specific authorized purposes, ensuring it is not used for discriminatory practices or infringing on individuals' rights [70].

6.2. Integrity and Reproducibility

We place a strong emphasis on the principles of integrity and reproducibility. To ensure transparency and facilitate reproducibility, we are committed to releasing the complete source code at `https://github.com/yunhanwang1105/Efficient-TAL-vision-language-modeling`, including all configurations used in our experiments.

While we acknowledge that perfect replication of our results may be challenging due to the inherent variability associated with deep learning models and hardware differences, we have taken several measures to minimize randomness and promote consistency. Specifically, we employ fixed random seeds throughout our experiments and incorporate other techniques to reduce variability, such as fixing the training and testing configuration and hardware used. Furthermore, to obtain robust and reliable results that do not stem from random factors, we conduct each experiment at least five times and compute the average performance. By following these rigorous practices, we aim to foster a research environment that encourages transparency and replicability in the field of TAL.

Our research was independently conducted without any external funding. Our motives are purely driven to derive scientific significance in the community. There are no conflicts of interest to disclose.

Acknowledgement

I would like to express my sincere gratitude to my supervisors, Ombretta Strafforello, Attila Lengyel, and Robert-Jan Bruintjes, for their invaluable feedback and support in every meeting, and every message throughout my research project. My critical thinking skills have been largely sharpened. I would also like to thank my responsible professor, Dr. Jan van Gemert, for his precious feedback and guidelines which teach me how to penetrate through the points and conduct insightful research. I would like to express my gratefulness to Sauradip Nag, for open-sourcing STALE. Lastly, I would like to thank all my research project peers for their support and valuable data.

References

[1] Anurag Arnab, Mostafa Dehghani, Georg Heigold, Chen Sun, Mario Lučić, and Cordelia Schmid. Vivit: A video vision transformer. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6836–6846, 2021. 2

[2] Brian R Bartoldson, Bhavya Kailkhura, and Davis Blalock. Compute-efficient deep learning: Algorithmic trends and opportunities. *Journal of Machine Learning Research*, 24:1–77, 2023. 1, 3

[3] Robert-Jan Bruintjes, Attila Lengyel, Marcos Baptista Rios, Osman Semih Kayhan, and Jan van Gemert. Vipriors 1: Visual inductive priors for data-efficient deep learning challenges. *arXiv preprint arXiv:2103.03768*, 2021. 3

[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. *BMVC*, 2019. 2

[5] Zhaowei Cai, Xiaodong He, Jian Sun, and Nuno Vasconcelos. Deep learning with low precision by half-wave gaussian quantization. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5918–5926, 2017. 3

[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, 7

[7] Alexandros Andre Chaaraoui, José Ramón Padilla-López, Francisco Javier Ferrández-Pastor, Mario Nieto-Hidalgo, and Francisco Flórez-Revuelta. A vision-based system for intelligent monitoring: human behaviour analysis and privacy by context. *Sensors*, 14(5):8895–8925, 2014. 1

[8] NVIDIA Corporation. Nvida v100 tensor core gpu. https://images.nvidia.com/content/technologies/volta/pdf/volta-v100-datasheet-update-us-1165301-r5.pdf, 2020. (Accessed on 15/06/2023). 6

[9] Delft High Performance Computing Centre (DHPC). DelftBlue Supercomputer (Phase 1). https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1, 2022. 6, 7

[10] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2625–2634, 2015. 2

[11] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020. 2, 4

[12] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. Slowfast networks for video recognition. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 6202–6211, 2019. 2

[13] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. Convolutional two-stream network fusion for video action recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1933–1941, 2016. 2

[14] Jiyang Gao, Zhenheng Yang, Kan Chen, Chen Sun, and Ram Nevatia. Turn tap: Temporal unit regression network for temporal action proposals. In *Proceedings of the IEEE international conference on computer vision*, pages 3628–3636, 2017. 2

[15] Jiyang Gao, Zhenheng Yang, and Ram Nevatia. Cascaded boundary regression for temporal action detection. *arXiv preprint arXiv:1705.01180*, 2017. 2

[16] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. Can spatiotemporal 3d cnns retrace the history of 2d cnns and imagenet? In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 6546–6555, 2018. 2

[17] Kaiming He, Xinlei Chen, Saining Xie, Yanghao Li, Piotr Dollár, and Ross Girshick. Masked autoencoders are scalable vision learners. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16000–16009, 2022. 3

[18] Fabian Caba Heilbron, Victor Escorcia, Bernard Ghanem, and Juan Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *2015 IEEE conference on computer vision and pattern recognition (CVPR)*, pages 961–970. IEEE, 2015. 2, 8

[19] Chen Ju, Tengda Han, Kunhao Zheng, Ya Zhang, and Weidi Xie. Prompting visual-language models for efficient video understanding. In *European Conference on Computer Vision (ECCV)*, 2022. 1, 2, 3

[20] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014. 2

[21] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Communications of the ACM*, 60(6):84–90, 2017. 2

[22] Attila Lengyel, Robert-Jan Bruintjes, Marcos Baptista Rios, Osman Semih Kayhan, Davide Zambrano, Nergis Tomen, and Jan van Gemert. Vipriors 2: visual inductive priors for data-efficient deep learning challenges. *arXiv preprint arXiv:2201.08625*, 2022. 3

[23] Attila Lengyel, Sourav Garg, Michael Milford, and Jan C van Gemert. Zero-shot day-night domain adaptation with a physics prior. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4399–4409, 2021. 8

[24] Boyi Li, Kilian Q Weinberger, Serge Belongie, Vladlen Koltun, and René Ranftl. Language-driven semantic segmentation. *arXiv preprint arXiv:2201.03546*, 2022. 3

[25] Liunian Harold Li, Pengchuan Zhang, Haotian Zhang, Jianwei Yang, Chunyuan Li, Yiwu Zhong, Lijuan Wang, Lu Yuan, Lei Zhang, Jenq-Neng Hwang, et al. Grounded language-image pre-training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10965–10975, 2022. 3

[26] 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

[27] 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

[28] 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

[29] 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

[30] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 2

[31] Vittorio Mazzia, Simone Angarano, Francesco Salvetti, Federico Angelini, and Marcello Chiaberge. Action transformer: A self-attention model for short-time pose-based human action recognition. *Pattern Recognition*, 124:108487, 2022. 2

[32] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, et al. Mixed precision training. *arXiv preprint arXiv:1710.03740*, 2017. 3

[33] Ron Mokady, Amir Hertz, and Amit H Bermano. Clipcap: Clip prefix for image captioning. *arXiv preprint arXiv:2111.09734*, 2021. 3

[34] Yasuhide Mori, Hironobu Takahashi, and Ryuichi Oka. Image-to-word transformation based on dividing and vector quantizing images with words. In *First international workshop on multimedia intelligent storage and retrieval management*, pages 1–9. Citeseer, 1999. 2

[35] Aaron J Moss, Cheskie Rosenzweig, Jonathan Robinson, Shalom N Jaffe, and Leib Litman. Is it ethical to use mechanical turk for behavioral research? relevant data from a representative survey of mturk participants and wages. *Behavior Research Methods*, pages 1–20, 2023. 8

[36] Sauradip Nag, Xiatian Zhu, Yi-Zhe Song, and Tao Xiang. Zero-shot temporal action detection via vision-language prompting. *arXiv e-prints*, pages arXiv–2207, 2022. 1, 2, 3, 4, 5, 6, 7

[37] University of Central Florida. Thumos challenge. http://crcv.ucf.edu/, 2013slurm-8561742.out. (Accessed on 23/06/2023). 2

[38] Justin Oosterbaan, Robert-Jan Bruintjes, Attila Lengyel, and Jan van Gemert. Data-efficient gan for synthetic samples of rare classes. *TU Delft Bachelor Thesis*, 2021. 3

[39] Chiara Plizzari, Marco Cannici, and Matteo Matteucci. Spatial temporal transformer network for skeleton-based action recognition. In *Pattern Recognition. ICPR International Workshops and Challenges: Virtual Event, January 10–15, 2021, Proceedings, Part III*, pages 694–701. Springer, 2021. 2

[40] Samira Pouyanfar, Saad Sadiq, Yilin Yan, Haiman Tian, Yudong Tao, Maria Presa Reyes, Mei-Ling Shyu, Shu-Ching Chen, and Sundaraja S Iyengar. A survey on deep learning: Algorithms, techniques, and applications. *ACM Computing Surveys (CSUR)*, 51(5):1–36, 2018. 2

[41] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 1, 2, 5, 7

[42] Maithra Raghu, Thomas Unterthiner, Simon Kornblith, Chiyuan Zhang, and Alexey Dosovitskiy. Do vision transformers see like convolutional neural networks? *Advances in Neural Information Processing Systems*, 34:12116–12128, 2021. 2

[43] Hanoona Rasheed, Muhammad Uzair khattak, Muhammad Maaz, Salman Khan, and Fahad Shahbaz Khan. Finetuned clip models are efficient video learners. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 3

[44] Facebook Research. fvcore. https://github.com/facebookresearch/fvcore, 2023. (Accessed on 15/06/2023). 5

[45] Isidoros Rodomagoulakis, Nikolaos Kardaris, Vassilis Pitsikalis, E Mavroudi, Athanasios Katsamanis, Antigoni Tsiami, and Petros Maragos. Multimodal human action recognition in assistive human-robot interaction. In *2016 IEEE international conference on acoustics, speech and signal processing (ICASSP)*, pages 2702–2706. IEEE, 2016. 1

[46] Dingfeng Shi, Yujie Zhong, Qiong Cao, Lin Ma, Jia Li, and Dacheng Tao. Tridet: Temporal action detection with relative boundary modeling. *arXiv preprint arXiv:2303.07347*, 2023. 1

[47] Zheng Shou, Dongang Wang, and Shih-Fu Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1049–1058, 2016. 2

[48] M Six Silberman, Lilly Irani, and Joel Ross. Ethics and tactics of professional crowdwork. *XRDS: Crossroads, The ACM Magazine for Students*, 17(2):39–43, 2010. 9

[49] M Six Silberman, Bill Tomlinson, Rochelle LaPlante, Joel Ross, Lilly Irani, and Andrew Zaldivar. Responsible research with crowds: pay crowdworkers at least minimum wage. *Communications of the ACM*, 61(3):39–41, 2018. 9

[50] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. *Advances in neural information processing systems*, 27, 2014. 2

[51] Zhenheng Tang, Yuxin Wang, Qiang Wang, and Xiaowen Chu. The impact of gpu dvfs on the energy and performance of deep learning: An empirical study. In *Proceedings of the Tenth ACM International Conference on Future Energy Systems*, pages 315–325, 2019. 3

[52] Zhan Tong, Yibing Song, Jue Wang, and Limin Wang. VideoMAE: Masked autoencoders are data-efficient learners for self-supervised video pre-training. In *Advances in Neural Information Processing Systems*, 2022. 1, 3

[53] Elahe Vahdani and Yingli Tian. Deep learning-based action detection in untrimmed videos: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2022. 2

[54] 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. 4

[55] Limin Wang, Bingkun Huang, Zhiyu Zhao, Zhan Tong, Yinan He, Yi Wang, Yali Wang, and Yu Qiao. Videomae v2: Scaling video masked autoencoders with dual masking, 2023. 1, 3

[56] Limin Wang, Yuanjun Xiong, Dahua Lin, and Luc Van Gool. Untrimmednets for weakly supervised action recognition and detection. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 4325–4334, 2017. 2, 4, 6

[57] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. Temporal segment networks: Towards good practices for deep action recognition. In *European conference on computer vision*, pages 20–36. Springer, 2016. 2

[58] Mengmeng Wang, Jiazheng Xing, and Yong Liu. Actionclip: A new paradigm for video action recognition. *arXiv preprint arXiv:2109.08472*, 2021. 3

[59] Ximei Wang, Jinghan Gao, Mingsheng Long, and Jianmin Wang. Self-tuning for data-efficient deep learning. In *International Conference on Machine Learning*, pages 10738–10748. PMLR, 2021. 3

[60] Jan Warchocki. *Benchmarking Data and Computational Efficiency of ActionFormer on Temporal Action Localization Tasks*. Bachelor's thesis, Delft University of Technology, 2023. 6, 7

[61] Hao Wu, Patrick Judd, Xiaojie Zhang, Mikhail Isaev, and Paulius Micikevicius. Integer quantization for deep learning inference: Principles and empirical evaluation. *arXiv preprint arXiv:2004.09602*, 2020. 5

[62] Huifen Xia and Yongzhao Zhan. A survey on temporal action localization. *IEEE Access*, 8:70477–70487, 2020. 2

[63] 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

[64] Jenny Yang, Andrew AS Soltan, David W Eyre, Yang Yang, and David A Clifton. An adversarial training framework for mitigating algorithmic biases in clinical machine learning. *NPJ Digital Medicine*, 6(1):55, 2023. 9

[65] Serena Yeung, Alireza Fathi, and Li Fei-Fei. Videoset: Video summary evaluation through text. *arXiv preprint arXiv:1406.5824*, 2014. 1

[66] Weihao Yu, Mi Luo, Pan Zhou, Chenyang Si, Yichen Zhou, Xinchao Wang, Jiashi Feng, and Shuicheng Yan. Metaformer is actually what you need for vision. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10819–10829, 2022. 2

[67] Chen-Lin Zhang, Jianxin Wu, and Yin Li. Actionformer: Localizing moments of actions with transformers. In *European Conference on Computer Vision*, volume 13664 of *LNCS*, pages 492–510, 2022. 1, 6, 7

[68] Yue Zhao, Yuanjun Xiong, Limin Wang, Zhirong Wu, Xiaoou Tang, and Dahua Lin. Temporal action detection with structured segment networks. In *Proceedings of the IEEE international conference on computer vision*, pages 2914–2923, 2017. 2

[69] Yi Zhu, Xinyu Li, Chunhui Liu, Mohammadreza Zolfaghari, Yuanjun Xiong, Chongruo Wu, Zhi Zhang, Joseph Tighe, R Manmatha, and Mu Li. A comprehensive study of deep video action recognition. *arXiv preprint arXiv:2012.06567*, 2020. 2

[70] Shoshana Zuboff. *The age of surveillance capitalism: The fight for a human future at the new frontier of power: Barack Obama's books of 2019*. Profile books, 2019. 9