SLAC: A Sparsely Labeled Dataset for Action Classification and Localization

Hang Zhao[1*], Zhicheng Yan[2*], Heng Wang[2], Lorenzo Torresani[2,3], Antonio Torralba[1]
[1]Massachusetts Institute of Technology, [2]Facebook Applied Machine Learning, [3]Dartmouth College
{hangzhao, torralba}@mit.edu, {zyan3, hengwang, torresani}@fb.com

Abstract

*This paper describes a procedure for the creation of large-scale video datasets for action classification and localization from unconstrained, realistic web data. The scalability of the proposed procedure is demonstrated by building a novel video benchmark, named **SLAC** (**S**parsely **L**abeled **AC**tions), consisting of over 520K untrimmed videos and 1.75M clip annotations spanning 200 action categories. Using our proposed framework, annotating a clip takes merely 8.8 seconds on average. This represents a saving in labeling time of over 95% compared to the traditional procedure of manual trimming and localization of actions. Our approach dramatically reduces the amount of human labeling by automatically identifying hard clips, i.e., clips that contain coherent actions but lead to prediction disagreement between action classifiers. A human annotator can disambiguate whether such a clip truly contains the hypothesized action in a handful of seconds, thus generating labels for highly informative samples at little cost. We show that our large-scale dataset can be used to effectively pre-train action recognition models, significantly improving final metrics on smaller-scale benchmarks after fine-tuning. On Kinetics [14], UCF-101 [30] and HMDB-51 [15], models pre-trained on SLAC outperform baselines trained from scratch, by 2.0%, 20.1% and 35.4% in top-1 accuracy, respectively when RGB input is used. Furthermore, we introduce a simple procedure that leverages the sparse labels in SLAC to pre-train action localization models. On THUMOS14 [12] and ActivityNet-v1.3[2], our localization model improves the mAP of baseline model by 8.6% and 2.5%, respectively.*

1. Introduction

The fields of image categorization and object detection have witnessed parallel outstanding progress in recent years [11, 10]. Arguably, these advances are the result of steady improvements and growths in datasets, which have enabled the successful application of progressively more sophisticated learning models. In image categorization, we moved from the Caltech101 [6] which was introduced in 2004 and contained only 9,146 examples, to the ImageNet dataset [3], which was introduced in 2011 and includes over 1.2M examples. In object detection, we have seen a similar trend in scaling-up training set sizes, despite the additional human annotation cost involved in collecting bounding-box information. Pascal VOC [4] was first released in 2007 and contained only 1,578 examples, while the recently introduced COCO dataset [17] consists of over 200K images and 500K object instance annotations.

In the video domain, the gap in scale between datasets for action classification and those for action localization has been widening. While action classification datasets created a few years ago consisted of a few thousands examples (6849 videos in HMDB51 [15], 13K in UCF101 [30], 3669 in Hollywood2 [18]), recent benchmarks have scaled-up dataset sizes by up to two orders of magnitude (Sports1M [13] contains over 1M videos, and Kinetics [14] has 306K videos). But we have not seen a similar growth in datasets for action localization. THUMOS [12] was created in 2014 and contained 2.7K trimmed videos and 1K untrimmed videos with localization annotations. Today, the largest benchmarks for action localization are only moderately larger. For example, ActivityNet [2] includes 20K videos and 30K annotations, AVA [19] includes 58K clips, and Charade [28] contains 67K temporally localized intervals. In Table 1, we give detailed comparisons between different video datasets.

Why are action localization datasets still so much smaller than those for object detection? And, why are action localization datasets still more than one order of magnitude smaller than those for action categorization? In this paper we put forward the hypothesis that this is primarily due to two reasons. First, gathering temporal annotations on video is very time-consuming. According to our experiments with professional annotators, manually annotating the start and end of action segments in a video takes more than 4 times the length of the video. In order to give accurate temporal annotations, annotators need not only to watch the entire

*Hang and Zhicheng contribute equally to this work.

Datasets	Actions	Videos	Annotations	Source	Localization
UCF101[30]	101	13K	13K	YouTube	No
HMDB51[15]	51	7K	7K	YouTube/Movie	No
Kinetics[14]	400	305K	305K	YouTube	No
Sports1M[13]	487	1.1M	1.1M	YouTube	No
THUMOS 15[12]	101	24K	21K	YouTube	Yes
ActivityNet[2]	200	20K	23K	YouTube	Yes
Charades[28]	157	10K	67K	267 Homes	Yes
AVA[19]	80	214	197K	Movie	Yes
SLAC	200	520K	1.75M	YouTube	Yes

Table 1. Comparisons between SLAC and other video datasets. Note that the annotations from Sports1M are produced automatically by analyzing the text metadata surrounding the videos, and thus inaccurate.

sequence, but also to replay several parts of the video back and forth to find exact boundaries. Second, marking temporal action boundaries is often ambiguous. While object boundaries are defined by their physical extents, the action boundaries are frequently blurry as a result of the smooth continuity of human movements and the poor definition of what constitutes an action.

In this work we propose to recast the temporal annotation task into a form that is more efficient and less ambiguous. The idea is to sample a small number of short clips from each video for annotators to review. An active learning algorithm is used to select one easy clip and several hard clips to label. The annotator is then asked to simply confirm whether such clips truly contain the hypothesized action. Our experiments suggest that providing a binary yes/no answer is easy and fast for annotators. This induces a saving in labeling time of over 95% compared to the traditional procedure of exhaustive review of the entire video for manual marking of action boundaries. The limited amount of human intervention allows us to build large-scale datasets with high-quality consistent annotations. While our procedure gives rise to only a sparse set of annotated clips, we demonstrate that models supervised by such annotations have superior generalization performance for both action classification and localization tasks.

For action classification, the large-scale of our dataset can be leveraged to pre-train video models. We show that fine-tuning these models on well-established action classification benchmarks (UCF101, HMDB51 and Kinetics) yields significant gains over learning from scratch. On Kinetics, UCF101 and HMDB51, we are able to improve a ResNet-based 3D ConvNet baseline [32] by 2.0%, 20.1% and 35.4%, respectively. We also show that pre-training on SLAC is more beneficial than pre-training on Sports1M or Kinetics. Sports-1M annotations are generated by a tag prediction algorithm, which inevitably injects prominent noise into the dataset. Also, the average length of Sports1M videos is over 5 minutes and the tag predicted action may only happen for a short period of time during the whole video. This poses substantial difficulties in learning good video representations for action classification as further discussed in Section 4. Compared to Kinetics, SLAC contains nearly 6 times more clip annotations (1.75M vs 305K), which may explain the superior generalization performance of deep models trained on our benchmark.

Finally, we demonstrate that the sparse clip annotations in SLAC can also be used to pre-train action localization models, which give dense predictions at each frame. On THUMOS Challenge 2014 and ActivityNet-v1.3 datasets, we are able to improve baseline models by 8.6% and 2.5% in mAP.

The rest of the paper is organized as follows. We review past work on video dataset collection in Section 2. The SLAC dataset and its collection procedure are elaborated in Section 3. We evaluate the merits of the SLAC dataset for action classification in Section 4 and for action localization in Section 5. We conclude our work in Section 6.

2. Related Work

The motivation behind our work is the creation of an annotated video collection that can serve as an instrumental training set for learning *general* spatiotemporal features, i.e., features that can be successfully transferred to other action recognition and detection domains and that can be fine-tuned effectively on smaller benchmarks.

Over the last decade, several video datasets were created for similar reasons. For example, in action recognition, the HMDB51 [15] and the UCF101 [30] datasets were introduced to provide benchmarks with higher variety of actions and more realistic capture conditions compared to precedent datasets, such as KTH [24]. These benchmarks have indeed served our community well, by enabling the hand-design of very effective and robust features for action recognition, such as space-time interest points [16] or dense trajectories [34]. However, these datasets are not large enough to support modern end-to-end training of deep models. The

large-scale Kinetics dataset [14] was recently introduced to fill this gap and it was shown to enable the pre-training of deep models that can be fine-tuned to achieve state-of-the-art performance on smaller action recognition benchmarks, such as HMDB51 and UCF101. Although Kinetics represents a remarkable resource for training deep action classifiers and for learning general video features, it cannot be used for learning action detection models, as it consists of only trimmed videos.

Our objective is to introduce a benchmark that will enable not only the learning of action recognition models but also the pre-training of deep models for action localization in untrimmed videos. Action detection in unconstrained videos is crucial to automatically understand Internet videos, which are typically several minutes long, and include multiple actions or several people interacting. Recently, several datasets for untrimmed video analysis have been presented. THUMOS Challenge 2014 [12] includes 2,765 trimmed training videos on 20 actions. It was subsequently extended into MultiTHUMOS [38] which contains more action classes (65 instead of 20) and more temporal and frame-level annotations. One shortcoming of THUMOS and MultiTHUMOS is that they only include examples of sport activities. Other datasets with fine granularity of classes but focused on narrow domains include MPII Cooking [22] and MPII Cooking 2 [23], which depict human subjects preparing dishes. Models trained on such domain-focused videos may not generalize optimally to recognition of every-day activities. Conversely the Charades dataset [28] was purposefully designed to include more general, daily activities. Since such uneventful activities are rarely shown in videos shared on the Internet, the authors asked people to act out everyday routine activities in their own homes in front of a camera, as previously done in the Activities of Daily Living dataset [20]. ActivityNet-v1.3 [2] includes over 600 hours of untrimmed videos and about 30K temporal annotations of actions spanning many common activities. Finally, the recently introduced AVA dataset [9] provides person-centric annotations, where each action annotations is a spatiotemporal box corresponding to a person performing an atomic action, such as shaking hands or kicking an object.

One common trait of all the aforementioned datasets for action detection is that localization in training videos was obtained through exhaustive manual search of individual actions in each video sequence. To produce accurate temporal localizations the annotator must typically stop, slow down, play back and forth the video multiple times. This renders the process of annotation very time consuming. This has hampered the scaling up in size of benchmarks for action detection. Our approach bypasses manual annotation of temporal extents by automatically identifying the hard clips that are hypothesized to be more informative than easy clips. Furthermore, since labeling on these clips is posed in the form of simple yes/no questions (*e.g.*, is action "kissing" happening in this clip?), annotators can provide labels efficiently and consistently, without the need to stop or replay the video.

Our approach has admittedly two downsides. The first is that our annotation pipeline yields a sparse labeling of clips, where other clips in the video remain unannotated and thus not usable for supervised training. The second disadvantage is that, since manual labels are sought only for the clips deemed relevant by our method, the labeled dataset is inherently biased rather than a uniform sampling of the data source. We demonstrate empirically that the benefits of our approach outweigh these two shortcomings. Specifically, we propose a straightforward procedure for pre-training action models from our sparse localization labels, and show that it improves model performances considerably after fine-tuning on fully-labeled data, compared to learning from scratch or pre-training on other datasets. Furthermore, we show that the dataset bias is small in practice through transfer learning experiments on four different benchmarks for action classification and localization, where our approach yields consistent metric gains.

3. Dataset Collection Pipeline

3.1. The SLAC Dataset at a Glance

The SLAC dataset includes 200 action classes, which are taken from ActivityNet-v1.3 dataset. It has a total of over 520K videos retrieved from YouTube. Each video is strictly shorter than 4 minutes, and the average length is 2.6 minutes. Over 1.75M 2-second clips are sampled by a novel active sampling approach (section 3.6). 755K and 993K clips are annotated as positive and negative samples, respectively, by a team of 70 professional annotators working 6 hours a day. The overall annotation takes effectively 10.5 working days to complete, including 1 hour for training the annotators. We split SLAC into training, validation and testing set with 502K, 6K and 12K videos, respectively. Dataset webpage is hosted at `http://slac.csail.mit.edu`.

An overview of our data collection pipeline is shown in Figure 1. Next, we elaborate in detail on the collection procedure.

3.2. Challenges in Data Collection

Collecting a large-scale action dataset is challenging due to several reasons. First, compared with negative examples of human action, the number of positive examples is arguably much smaller. If we uniformly sample video/clips to annotate, we may end up with a large number of negative examples which are far less useful than positive examples for training video models. Second, not all negative examples are equally useful. We hypothesize that hard negative

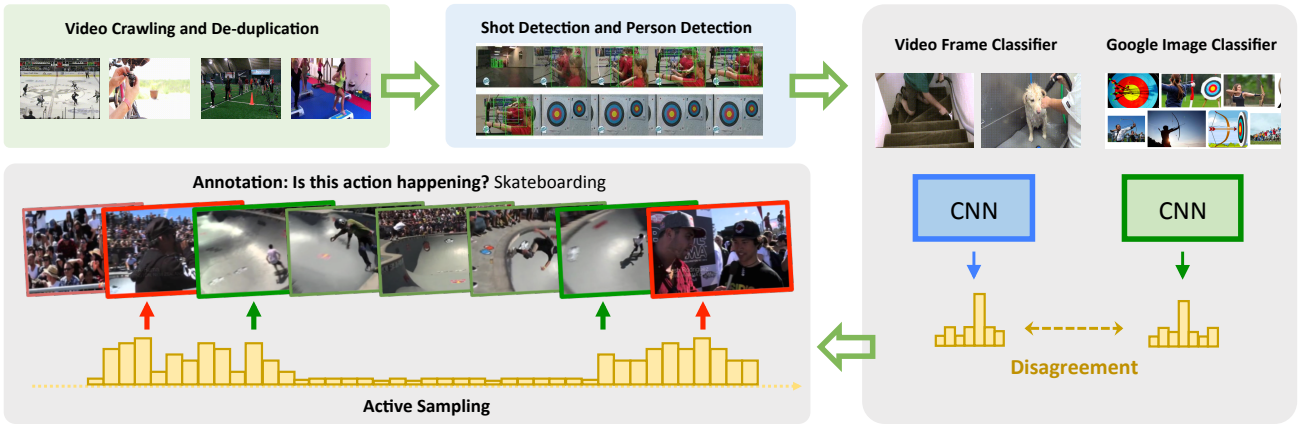

Figure 1. An overview of data collection procedure to obtain Sparsely Labeled Actions.

examples are more valuable than easy negative examples for model learning. We must properly address both challenges to make our data collection procedure cost-effective.

To address the first challenge and sample more positive examples for annotation, we bias our sampling distribution towards videos that are likely to contain actions. This is achieved by leveraging both video search engine (e.g., YouTube) to retrieve videos with actions of interest (Section 3.4), and person detectors to exclude videos that do not contain humans (Section 3.5).

To address the second challenge, we review the different types of negative examples, and their usefulness for training models (Section 3.3).

3.3. Negative Sample Analysis

In general, there are three types of negative samples in human action datasets, with increasing levels of mining difficulty:

No person. Since we are only interested in human actions, negative videos without human presence should be excluded from annotation as much as possible.

No context. Another common type of negative samples is represented by videos that contain people but lack the context of ongoing action. For instance, if the objective is to collect videos of action *Archery*, we should exclude videos if arrows or bows are not present in the videos.

No action. The hardest negative examples are videos that include both people and context, but where the action of interest is not being performed. We identify such hard negative examples by sending them to annotators for validation.

Concrete examples of these prototypical negative samples are shown in Figure 2, where *Archery* is the action to localize. Examples of "No person" can be animations in the video. Examples of "No context" are clips where people are present, but the clips are not related to archery. An example of "No action" is represented by a person who is merely holding a bow and talking, as opposed to performing archery.

Figure 2. An illustration of negative samples on action datasets.

As the first two types of negatives occur more often in videos and are easier to obtain, we use pre-trained models to automatically identify them, and spend most of our annotation budget on the third type.

3.4. Video Retrieval and De-duplication

SLAC	Kinetics	ActivityNet	UCF-101	HMDB-51
171,378	4,029	1,925	660	11

Table 2. De-duplication of SLAC within itself, and against the test/validation sets of prior video datasets. Starting with an initial set of 1.1M videos, we show the number of duplicate videos removed in each step.

Based on the taxonomy of ActivityNet-v1.3 where 200 human actions are defined, we retrieve 1.1M candidate videos (1K to 6K videos per class) from YouTube by using the class label as query to find videos with matching titles and descriptions.

Internal De-duplication. We de-duplicate videos within SLAC, since YouTube may include several copies of the same video, possibly differing in post-processing steps, such as saturation/contrast enhancement or in duration. Specifically, we resize each video to have short edge of size 256 with aspect ratio preserved, and evenly sample

224×224 central crops from frames at 0.2 fps. For each video, we extract *pool5* features of 2048-D dimensions using an ImageNet-pretrained ResNet-50 model for all sampled frames, and compute a mean feature vector. For each action class, we treat its videos as nodes in a graph, and connect two video nodes if their cosine similarity is higher than a threshold γ. After processing all pairs of videos in the same action class, we compute the connected components in the graph, and sample only one video from each connected component [14]. We choose $\gamma = 0.935$ by ensuring through manual inspection that this thresholds removes all duplicate videos in a large subset of video pairs with high similarity scores. Through this de-duplication step, we remove about 171K videos.

External De-duplication. We perform additional de-duplication to ensure that SLAC does not overlap with the validation and testing sets of other video datasets, namely Kinetics, ActivityNet, UCF101 and HMDB51. As the taxonomy between SLAC and other datasets can be different, for each action class A in SLAC we first manually compile a list of similar action classes $\{B_i\}_i$ in the target dataset. For example, we selected actions *swimming backstroke*, *swimming breast stroke*, and *swimming butterfly stroke* in Kinetics as the classes that may contain duplicates of SLAC videos for action *swimming*. We compute the similarity between each SLAC video in class A and each target video of actions $\{B_i\}_i$. We remove a video from SLAC if its similarity score with a video from the target dataset is above the threshold. We sequentially de-duplicate SLAC with respect to Kinetics, ActivityNet, UCF101 and HMDB51, and remove $4,029$, $1,925$, 660 and 11 videos from SLAC.

3.5. Person detection

In this step we use a person detector to remove the first kind of negative samples: videos that do not contain people. To reduce the computation, shot detection is performed beforehand. We compute the dissimilarity score between color histograms of consecutive frames, and threshold on both the score and its gradient to find shot boundary. From each shot, 2 frames are sampled and a person score is computed. We run the Faster R-CNN [21] trained on the COCO person class to get the maximum score of person being present over the two frames. A threshold of 0.5 on the person score is used to find shots with people (nearly 49% of all video shots). These shots are the initial proposals of human actions used for further processing. To verify the accuracy of this procedure, we test the same person detector on the validation set of ActivityNet-v1.3, and get a recall of $>97\%$. If all shots in a video have scores below the threshold, we remove such video. This removes $193K$ videos.

3.6. Proposal Models and Active Frame Sampling

Between the other two types of negatives, we hypothesize that "No Context" clips are easier to recognize and less useful than "No Action" clips. Unfortunately both types of videos can be returned by YouTube due to the video description matching the class label. We want to bias the distribution of our clip sampling for human annotation towards "No Action" hard negative samples, and away from "No Context" easy negative samples. To achieve this, we trained two image classifiers, which are used to sample clips.

YouTube Frame-based Model. The first model is trained on frames extracted from the top 500 videos retrieved by YouTube for each action class. Only videos frames with person detected are used as positive example for training. This sums to a total of over 600K frames (3K frames per action on average). We also randomly choose frames with low person score as examples of the background class.

Google Image-based Model. The second model is trained on images retrieved from Google Images using the class labels as queries. This strategy yields a total of over 300K images (1.5K per class) after thresholding on person detection. We also use 1000 class labels of ImageNet as keywords to retrieve examples with Google Images and assign these images to a background class.

We use ResNet-50 architecture for both proposal models and train them as 201-way classification, including 200 actions and 1 background class.

We use a measure of consensus between these two classifiers to bias the distribution of sampling clips for annotations towards "No Action" samples. In active learning, this consensus method is known as query-by-committee (QBC) [25]. First, for each video we randomly sample one frame from each shot in the video. We denote with X the set of these frames, representing all the candidate clips for annotation in the video. Next, we want to define a sampling distribution $R(x)$ over $x \in X$, in order to sample the frames that are most informative for annotation. We define the unnormalized sampling probability $R(x)$ as follows:

$$R(x) = \left(P_c^Y(x) + P_c^G(x)\right) * D_{KL}\left(P^Y(x), P^G(x)\right) * \left(\frac{1}{|X|-1} \sum_{x' \in X, x' \neq x} S(f(x), f(x'))\right) \quad (1)$$

where $P^Y(x)$ and $P^G(x)$ are the 201-dimensional predicted probabilities from YouTube frame classifiers Y and Google image classifier G, respectively. c denotes the action label that was used to retrieve the video. Therefore, the first term $(P_c^Y(x) + P_c^G(x))$ represents the consensus of both classifiers on the probability of frame x containing action c. The rationale behind this term is that it can reduce the sampling probability for "No Context" frames. The sec-

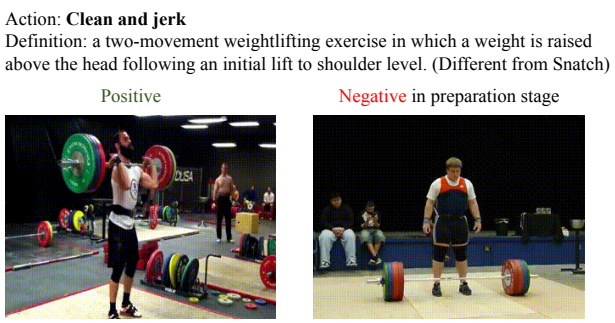

Figure 3. Annotation guideline, including definition of the action class, as well as positive and negative examples.

ond term $D_{KL}(\cdot)$ is the KL divergence, which represents the disagreement of class probabilities between two image classifiers. The third term computes the average similarity between x and all other frames in X. Frame similarity is assessed as the cosine similarity between *conv5* feature activation $f(x)$ extracted from an Imagenet-trained ResNet-50 model. This third term biases our sampling towards representative frames in the video. Once we compute $\{R(x)\}_x$ for all frame candidates in X, we then perform L1 normalization to obtain a proper sampling distribution. We sample at most 8 frames from X (i.e., per video) according to this distribution but only retain samples for which the first term (action probability) is greater than τ, which is chosen as 0.01. In addition, we also sample one extra frame, corresponding to the frame having the highest value of under the first term (consensus). This sample can be viewed as an easy positive example. We select such samples as we hypothesize that a small number of easy positive examples can be useful for model training as well.

After sampling the video frames, we take one second on either side of a frame to convert it into a 2-second clip. We also ensure that such clips do not cross shot boundaries. As a result, a total of 1.75M clips are sampled from 520K videos.

3.7. Video Clip Annotation

The video annotation task is aimed at labeling the 1.75M clips of 2-second duration in a clear and consistent manner.
Annotation guideline We prepare a detailed annotation guideline providing clear action definitions. Both text descriptions and positive/negative clip examples with clarifications are provided to annotators to reduce the ambiguity of action annotation. An illustration of our guideline is shown in Figure 3.
Annotation Tool. Our annotation tool is visualized in Figure 4. Clips are organized into a grid layout, and automatically played. Annotators can quickly mark them as positive/negative samples by clicking on the clips. The border color visualizes the current annotations where positive clips are shown in green and negative clips in red.

Figure 4. Annotation tool. The user can quickly flip the border color of clips to mark positive/negative example by simple clicking. The action being annotated above is *Kayaking*. Due to space constrain, we only show 3 clips here.

	Full annotation	Sparse annotation
Per clip	N/A	8.8 sec
Per video	11.3 min	30.6 sec
Dataset	*est.* 113.2K hr	4,390 hr

Table 3. Cost comparison for full and sparse annotations.

Annotation Quality Control. To ensure a high annotation quality, we first ensure clips from the same action label are labeled by the same annotator. This removes the inter-annotator variance. Second, we personally labeled 10K clips before kicking off the annotation. We refer to this set of clips as the **Golden Set** which is mixed with other clips to be annotated. We use the clips in the Golden Set to compute annotation accuracy per annotator, by comparing the labels provided by us to those given by the annotators. We monitor the accuracy and iteratively give customized clarifications when an annotator has low accuracy. We ensure all annotators have over 90% accuracy. The final average accuracy is 94%.
Annotation Cost. We hire 70 professional annotators working 6 hours per day. The annotation task is finished in 10.5 working days. In the last 2 days, several annotators have already completed the annotations and passed the accuracy check, while others receive more clarifications and re-annotate actions due to low Golden Set accuracy. Through this procedure we gathered on average 4,390 annotator $\times$ hour. The average annotation cost (in time) is 8.8 seconds per clip, and 30.6 seconds per video. To compare the cost with traditional annotation of action localization, we take 20K videos from SLAC and ask the same group of annotators to manually mark start and end of the actions in each video. Statistics are shown in Table 3. The total saving is over 95%. These numbers suggest that collecting localizations manually is unachievable in practice at the scale of our dataset, due to the very long time that this would take and the consequent monetary cost.

4. Action Recognition Benchmark

In this section we present results on action recognition. First, we train 3D ConvNet on SLAC training set and evaluate on SLAC validation and test set. Second, we evaluate the efficacy of SLAC for pre-training models, and compare SLAC with respect to other large scale datasets (such as Kinetics and Sports1M). Meanwhile, we experiment with fusing RGB and optical flow features, and compare with the state-of-the-art in the literature.

4.1. Evaluations on SLAC

Network Architecture. ResNet [11] has emerged as one of the most successful models for image classification. Here we experiment with its 3D version, namely Res3D [32], which extends the convolutional operators from 2D to 3D. The Res3D model adopted in this paper has 34 parametric layers and uses an input of 32 adjacent frames with a fixed resolution of 112×144. We denote it as *Res3D-34*.

Implementation Details. We implement our approach in Caffe2 [1]. For training Res3D-34 model, we use both positive and negative clips in SLAC. There are 200 positive classes, and we assign a single background label to all negative clips. Therefore, Res3D-34 is trained with 201-way softmax loss. SLAC has highly imbalanced positive and negative clips where 57.5% of training clips are negative, and on average each positive class accounts for only 0.215% of total clips. To prevent model training being biased by negative clips, we use weighted sampling to assemble training mini-batches. We empirically use weight 0.02 for negative clips and 1.0 for positive clips. Weights serve as unnormalized multi-nomial distribution from which we sample clips to form mini-batches. Mini-batch size per GPU is set to 16. To speed up the model training on large-scale datasets, we use 16 hosts with 4 Nvidia M40 GPUs per host for distributed training with parameter synchronization. Therefore, the total mini-batch size is 1024. This yields a throughput of about 600 video clips per second. We train on SLAC for 45 epochs, with the first 10 epochs for learning rate warmup [8], and then reduce the learning rate by a factor of 10 every 10 epochs. The base learning rate per GPU is set to 0.005, and the effective learning rate after warmup is 0.32. After training, we evaluate on the SLAC validation set and test set.

Evaluation Metrics. For each clip in validation and test set, we sample a single sequence of 32 frames in the beginning of the clip. Since both validation and test set have highly imbalanced positive and negative clips as in training set, we report both class-agnostic and class-aware clip accuracy. (1) **Class-Agnostic Clip@1**: we count the number of clips where the top predicted label is correct, and divide it by the total number of clips in the validation/test set. (2): **Class-Aware Clip@1**: we first group clips based on groundtruth class label, then compute top-1 accuracy for each class, and finally compute the average of class-wise top-1accuracies.

Results. As shown in Table 4, with RGB input, we achieve 54.3% and 76.5% Class-Agnostic and Class-Aware clip@1 accuracy on validation set. On test set, we obtain 53.7% and 76.2% Class-Agnostic and Class-Aware clip@1 accuracy. We also experiment with using optical flow as input to the video model. We adopt efficient Farneback's approach [5] to quickly compute optical flow. For optical flow, we use the same number of frames and the same resolution as RGB. Results of using optical flow alone are also reported in Table 4, and are about 7% and 12% worse than those with RGB input alone for Class-Agnostic and Class-Aware Clip@1 accuracy. We also experiment with combining RGB and optical flow by late fusion, where the final prediction score is a weighted sum of two separate prediction scores from RGB and optical flow. We empirically set weights for RGB and optical flow to 0.6 and 0.4, where RGB weight is bigger as RGB input alone gives higher accuracy than optical flow input. Late fusion of RGB and optical flow improves the results of RGB input alone by 8.1% and 1.8% for Class-Agnostic and Class-Aware Clip@1 on validation set, and 7.8% and 1.6% for Class-Agnostic and Class-Aware Clip@1 on test set.

4.2. Evaluations on Transfer Learning Tasks

Given the trained Res3D-34 model, we can finetune it on other target datasets. By comparing the performance on target datasets using finetuned models with that from models trained from scratch, we can evaluate the efficacy of SLAC for pre-training large models.

Evaluation Metrics. At test time, we measure performance according to three metrics. (1) **Clip@1**: We sample a single 32-frame clip in the beginning of the video, and report top-1 accuracy. (2) **Video@1**: We evenly sample 10 clips of 32 frames in the video, average the clip predictions and report top-1 accuracy. (3) **Dense prediction w/ multi-crops@1**. We densely sample clips of 32 frames with a stride of 4 frames. For each clip, we also take 5 different crops (4 in the corners, and 1 in the center) as well as their mirrored versions. We average predictions over crops and clips, and report top-1 accuracy.

Action recognition datasets. We use 4 other datasets for the transfer learning of action recognition experiments. UCF-101, and HMDB-51 are used as target datasets. Kinetics are used as both pre-training dataset and target dataset. Sports-1M is used only as pre-training dataset. We finetune models on UCF-101, HMDB-51 and Kinetics for 110, 50, and 110 epochs. As there are many long videos in Sports-1M, we cut them into shorter clips to better utilize the data and end up with a training set of about 5M samples. For Kinetics, we report the accuracy on the validation set as the annotations of the testing set are not publicly available. For

	Validation Set		Test Set	
Input	Class-Agnostic, Clip@1	Class-Aware, Clip@1	Class-Agnostic, Clip@1	Class-Aware, Clip@1
RGB	54.3	76.5	53.7	76.2
Flow	47.3	64.9	46.3	63.6
RGB + Flow	**62.4**	**78.3**	**61.5**	**77.8**

Table 4. Results on SLAC. We train Res3D-34 model using 201-way softmax loss.

	Target Dataset	UCF-101		HMDB-51		Kinetics	
Input	Training	Clip@1	Video@1	Clip@1	Video@1	Clip@1	Video@1
RGB	w/o pre-training	72.7	75.6	34.0	35.9	57.0	68.2
	w/ pre-training	**93.4**	**95.7**	**67.3**	**71.3**	**58.5**	**70.2**
Flow	w/o pre-training	68.7	80.8	34.0	45.1	42.2	59.2
	w/ pre-training	**87.9**	**93.6**	**60.0**	**69.5**	**45.0**	**62.6**

Table 5. Evaluations on the transfer learning performance of SLAC using different types of input.

Target Dataset	UCF-101		HMDB-51		Kinetics	
Dataset pre-train	Clip@1	Video@1	Clip@1	Video@1	Clip@1	Video@1
w/o pre-training	72.7	75.6	34.0	35.9	57.0	68.2
Kinetics	89.9	92.5	64.9	67.4	57.0	68.2
Sports1M	90.9	92.6	64.1	65.9	57.4	68.4
SLAC	**93.4**	**95.7**	**67.3**	**71.3**	**58.5**	**70.2**

Table 6. Comparisons of SLAC with other datasets for pre-training action recognition models. Test performance is evaluated by finetuning on UCF-101, HMDB-51, Kinetics. All numbers are obtained by training the same model on RGB input.

Method	UCF-101	HMDB-51	Kinetics
Long-term temporal convolutions [33]	92.7	67.2	N/A
ST-Multiplier Net + IDT [7]	94.9	72.2	N/A
Temporal Segment Networks [35]	94.2	69.4	N/A
Two-Stream I3D, Kinetics pre-training [14]	**98.0**	**80.7**	**75.7**
Two-Stream Res3D-34, SLAC pre-training, video@1	96.6	75.6	73.5
Two-Stream Res3D-34, SLAC pre-training, dense prediction, w/ multi-cropping@1	**97.6**	**76.7**	**74.8**

Table 7. Comparing Res3D-34 model pre-trained on SLAC with the state-of-the-art models on UCF101, HMDB51 and Kinetics.

UCF101 and HMDB51, we only use split-1 for training and testing.

Results. As shown in Table 5, with RGB input pre-training on SLAC improves the Clip@1 by 20.7%, 33.3%, and 1.5%, and Video@1 by 20.1%, 35.4%, and 2.0% on UCF-101, HMDB-51 and Kinetics, respectively. When optical flow input is used, pre-training on SLAC improves the Clip@1 by 19.2%, 26.0%, and 2.8%, and Video@1 by 12.8%, 24.4%, and 3.4% on UCF-101, HMDB-51 and Kinetics, respectively.

Comparisons with other pre-training datasets. To put the efficacy of SLAC as a pre-training dataset, we compare it with two other large scale datasets (Kinetics and Sports1M) for transfer learning. Results are summarized in Table 6. For pre-training, Kinetics is better than Sports1M when finetuning models on UCF101 and HMDB51, as the annotations from Sports1M are noisy despite being one order of magnitude more numerous than those of Kinetics. On the other hand, pre-training on Sports1M still improves the accuracy on Kinetics, albeit only slightly. SLAC reports the best accuracy as a pre-training dataset for all three target datasets due to its large-scale and its high-quality annotations. The gain is up to 3.9% for Video@1 when comparing to Kinetics or Sports-1M. In Table 6, SLAC achieves the state-of-the-art performance for transfer learning.

Feature Fusion. The two-stream [29] model has been shown to be successful in action recognition by combing the appearance information from RGB and motion information from optical flow. Table 5 presents the results of transfer learning on SLAC using RGB and optical flow input alone. Overall, we observe similar performance gain on optical flow by pre-training on SLAC as RGB. When

comparing the accuracy between RGB and optical flow with pre-training, RGB is always better than optical flow on all three target datasets. As expected, combining RGB with optical flow using two streams always improves the final accuracy. The improvements in Video@1 metric on UCF-101, HMDB-51 and Kinetics over RGB-only models are 0.9% (95.7% *vs.* 96.6%), 4.3% (71.3% *vs.* 75.6%) and 3.3% (70.2% *vs.* 73.5%), respectively (Table 6 and Table 7).

Comparisons with other methods. Finally, we compare our SLAC-pretrained Res3D-34 model with the state-of-the-art in Table 7. On UCF-101 and HMDB-51, by simply using an off-the-shelf 3D ResNet of 34 layer and leveraging the power of a large-scale well annotated dataset, we outperform three competitive methods [33, 7, 35] by at least 2.7%. On Kinetics validation set, our model performs slightly (0.9%) worse than the two-stream I3D model [14]. There are several factors contributing to such performance gap. First, I3D is a more sophisticated model with parameters inflated from ImageNet pre-trainining. Second, I3D uses the TV-L1 optical flow [39], which is more expensive and accurate than the Farneback optical flow [5] adopted by our model. Third, we train with 32 video frames with size 112×144 while I3D uses 64 frames of size 224×224. We leave empirical explorations of these differences on performance as future work.

5. Action Localization Benchmark

In this section, we present the results of action localization on two popular benchmarks, THUMOS14 and ActivityNet-v1.3.

Framework. For the task of action localization, we adopt the framework from [26]. First, candidate video segments are proposed from the untrimmed videos; second, dense frame-wise labels are predicted for all the proposal segments to localize actions accurately. Note that we only use RGB frames (without optical flow) as input in all experiments of action localization.

Network Architecture. For dense frame-wise action prediction, the Convolutional-De-Convolutional (Conv-Deconv) operators are used in our localization model. Given the input frames of size $32 \times 3 \times 112 \times 112$, the full localization model uses a Res3D-34 trunk, which produces feature maps of size $4 \times 512 \times 7 \times 7$, and 3 successive Conv-Deconv operators, which produce feature maps of size $32 \times 512 \times 1 \times 1$. A 201-way softmax is used to generate predictions over 200 actions and 1 background class. The Conv-Deconv operator, originally proposed in [26], implements convolution (and downsampling) in space and deconvolution (and upsampling) in time in a single step, and it involves more parameters to learn than separate Conv operator and Deconv operator. The number of parameters in Res3D-34 trunk is 63.4M, while the Conv-Deconv operators add extra 20.4M parameters. Details of the localization architecture can be found in *Supplementary Materials*.

Training. When pre-train on SLAC, we randomly take 32 frames from 2-second SLAC clips as training examples, and give all frames a uniform label based on clip annotation. Similar to how we trained recognition models, a single background label is assigned to all negative examples, and sampling weights of positive and negative samples are 1.0 and 0.02, respectively. The pre-trained model is further finetuned on THUMOS14 and ActivityNet-v1.3. For each action annotation, we expand it by 2 seconds at both the beginning and the end of the segment. Then we randomly choose 32 frames as training examples. Frames within the original segment are assigned positive action label, while frames outside of the original segments are assinged background label. The localization models are finetuned using distributed training with 16 hosts of 4 GPU per host for 11 epochs, including 2 warm-up epochs, where learning rate increases linearly from 0.005 to 0.32. Afterwards, learning rate is discounted by $1/10$ every 3 epochs.

Inference. Although localization model can produce frame-level dense predictions, it is non-trivial to design robust algorithm to group frames to form contiguous labeling of action segments based on prediction scores. On the other hand, there are existing segment proposal methods to generate candidate segments with high recall. Therefore, for THUMOS14, we adopt proposals from [27], and apply our dense prediction model for localization; for ActivityNet, we adopt the output from [36] as proposals, and then we follow the method in [26] to refine proposal boundaries. Given a segment proposal, we first extend it by a percentage α on each side ($\alpha = 25\%$), and assign the label with the highest mean prediction over the entire extended segment. Proposals assigned with background label are abandoned. Otherwise, we perform Gaussian density estimation to refine proposal boundaries by computing the mean μ and standard deviation σ of action scores. We scan the prediction score of the predicted action label on each side of the extended proposal, and keep shrinking the boundary until the action score at a frame is higher than $\mu - \beta * \sigma$. β is tuned on a small held-out set of videos taken from training set, and is set to 2.0.

Comparison. We compare the performance of action localization models with and without SLAC pre-training, on the THUMOS14 test set and ActivityNet-v1.3 validation set, in Table 8 and Table 9, respectively. In general, pre-training significantly improves performance of localization models, in terms of mean average precision (mAP). Specifically, the framework is good at improving hits at high IoU thresholds (IoU$>$ 0.6), which can be explained by the accurate action boundary prediction by frame-wise dense labeling. Compared with our localization model based on Res3D-34 and Conv-Deconv operators trained from scratch, the model with the same architecture but pre-trained on SLAC

mAP @ IoU	0.3	0.4	0.5	0.6	0.7	Average
w/o pre-training	33.0	27.1	18.9	10.5	4.0	18.7
pre-train on Sports1M [26]	40.1	29.4	23.3	13.1	7.9	22.8
pre-train on SLAC	**45.0**	**35.8**	**29.2**	**17.2**	**9.5**	**27.3**
R-C3D [37]	44.8	35.6	28.9	N/A	N/A	N/A
SSN [40]	**51.9**	**41.0**	**29.8**	N/A	N/A	N/A

Table 8. Comparison of action localization performance on the THUMOS14 test set with different pre-training datasets, and the state-of-the-art methods.

mAP @ IoU	0.5	0.55	0.60	0.65	0.70	0.75	0.80	0.85	0.90	0.95	Average
before refinement [36]	43.3	39.9	36.7	16.8	8.7	3.9	1.9	0.8	0.2	0.0	15.3
after refinement, w/o pre-training	43.0	**39.5**	**36.0**	**32.0**	**28.6**	24.6	13.5	5.4	1.4	0.2	22.4
after refinement, pre-train on Sports1M [26]	**45.3**	N/A	N/A	N/A	N/A	**26.0**	N/A	N/A	N/A	0.2	23.8
after refinement, pre-train on SLAC	42.9	38.9	34.7	31.0	28.0	24.6	**20.9**	**16.9**	**9.7**	**1.4**	**24.9**

Table 9. Comparison of action localization performance on the ActivityNet-v1.3 validation set before and after refinement with different pre-training datasets.

achieves the absolute gain of 8.6% on THUMOS14 and 2.5% on ActivityNet-v1.3. We also compare with the original Conv-Deconv work pre-trained on Sports-1M [31], our model outperforms it by 4.5% on THUMOS14 and 1.1% on ActivityNet-v1.3. State-of-the-art methods with different frameworks are shown in the last rows of Table 8, our performance beats the competitive R-C3D [37] model, but not SSN [40]. Exploring the best framework for action localization is beyond the focus of this paper, and we will leave it for future work.

6. Conclusions

We present a novel and highly efficient data collection procedure to construct large-scale human action video datasets. The proposed pipeline yields a 95% reduction in annotation time. Our resulting dataset, SLAC, includes 1.75M clip annotations spanning over 520K untrimmed videos. We demonstrate that SLAC consistently outperforms Sports-1M and Kinetics as a pretraining dataset for action recognition models. We also show superior transfer learning performance on action localization on THUMOS14 and ActivityNet-v1.3 benchmarks. Dataset, pretrained models, and code will be released upon publication of the paper.

Acknowledgement We thank Du Tran for insightful discussions along the development of this project.

References

[1] Caffe2: A new lightweight, modular, and scalable deep learning framework. `https://caffe2.ai/`.

[2] F. Caba Heilbron, V. Escorcia, B. Ghanem, and J. Carlos Niebles. Activitynet: A large-scale video benchmark for human activity understanding. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 961–970, 2015.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 248–255. IEEE, 2009.

[4] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88(2):303–338, June 2010.

[5] G. Farnebäck. Two-frame motion estimation based on polynomial expansion. *Image analysis*, pages 363–370, 2003.

[6] L. Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.

[7] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal multiplier networks for video action recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4768–4777, 2017.

[8] P. Goyal, P. Dollár, R. Girshick, P. Noordhuis, L. Wesolowski, A. Kyrola, A. Tulloch, Y. Jia, and K. He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

[9] C. Gu, C. Sun, S. Vijayanarasimhan, C. Pantofaru, D. A. Ross, G. Toderici, Y. Li, S. Ricco, R. Sukthankar, C. Schmid, and J. Malik. AVA: A video dataset of spatio-temporally localized atomic visual actions. *CoRR*, abs/1705.08421, 2017.

[10] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask r-cnn. *arXiv preprint arXiv:1703.06870*, 2017.

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[12] Y. Jiang, J. Liu, A. R. Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. Thumos challenge: Action recognition with a large number of classes, 2014.

[13] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

[14] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017.

[15] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. Hmdb: a large video database for human motion recognition. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 2556–2563. IEEE, 2011.

[16] I. Laptev. On space-time interest points. *International journal of computer vision*, 64(2-3):107–123, 2005.

[17] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft coco: Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[18] M. Marszalek, I. Laptev, and C. Schmid. Actions in context. In *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, pages 2929–2936. IEEE, 2009.

[19] C. Pantofaru, C. Sun, C. Gu, C. Schmid, D. Ross, G. Toderici, J. Malik, R. Sukthankar, S. Vijayanarasimhan, S. Ricco, et al. Ava: A video dataset of spatio-temporally localized atomic visual actions. 2017.

[20] H. Pirsiavash and D. Ramanan. Detecting activities of daily living in first-person camera views. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012.

[21] S. Ren, K. He, R. Girshick, and J. Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *Advances in neural information processing systems*, pages 91–99, 2015.

[22] M. Rohrbach, S. Amin, M. Andriluka, and B. Schiele. A database for fine grained activity detection of cooking activities. In *2012 IEEE Conference on Computer Vision and Pattern Recognition, Providence, RI, USA, June 16-21, 2012*, pages 1194–1201, 2012.

[23] M. Rohrbach, A. Rohrbach, M. Regneri, S. Amin, M. Andriluka, M. Pinkal, and B. Schiele. Recognizing fine-grained and composite activities using hand-centric features and script data. *International Journal of Computer Vision*, pages 1–28, 2015.

[24] C. Schuldt, I. Laptev, and B. Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

[25] H. S. Seung, M. Opper, and H. Sompolinsky. Query by committee. In *Proceedings of the fifth annual workshop on Computational learning theory*, pages 287–294. ACM, 1992.

[26] Z. Shou, J. Chan, A. Zareian, K. Miyazawa, and S.-F. Chang. CDC: Convolutional-de-convolutional networks for precise temporal action localization in untrimmed videos. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.

[27] Z. Shou, D. Wang, and S. Chang. Temporal action localization in untrimmed videos via multi-stage cnns. In *2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016*, pages 1049–1058, 2016.

[28] G. A. Sigurdsson, G. Varol, X. Wang, A. Farhadi, I. Laptev, and A. Gupta. Hollywood in homes: Crowdsourcing data collection for activity understanding. In *European Conference on Computer Vision*, 2016.

[29] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in neural information processing systems*, pages 568–576, 2014.

[30] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A dataset of 101 human actions classes from videos in the wild. *arXiv preprint arXiv:1212.0402*, 2012.

[31] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning spatiotemporal features with 3D convolutional networks. In *Proceedings of the IEEE international conference on computer vision*, pages 4489–4497, 2015.

[32] D. Tran, J. Ray, Z. Shou, S.-F. Chang, and M. Paluri. Convnet architecture search for spatiotemporal feature learning. *arXiv preprint arXiv:1708.05038*, 2017.

[33] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[34] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. Action recognition by dense trajectories. In *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, pages 3169–3176. IEEE, 2011.

[35] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Val Gool. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*, 2016.

[36] R. Wang and D. Tao. UTS at ActivityNet 2016. In *Computer Vision and Pattern Recognition Workshop (CVPRW) on ActivityNet Large Scale Activity Recognition Challenge*, 2016.

[37] H. Xu, A. Das, and K. Saenko. R-c3d: Region convolutional 3d network for temporal activity detection. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.

[38] S. Yeung, O. Russakovsky, N. Jin, M. Andriluka, G. Mori, and L. Fei-Fei. Every moment counts: Dense detailed labeling of actions in complex videos. *International Journal of Computer Vision*, pages 1–15, 2015.

[39] C. Zach, T. Pock, and H. Bischof. A duality based approach for realtime TV-L1 optical flow. *Pattern Recognition*, pages 214–223, 2007.

[40] Y. Zhao, Y. Xiong, L. Wang, Z. Wu, X. Tang, and D. Lin. Temporal action detection with structured segment networks. In *ICCV*, 2017.