SCSampler: Sampling Salient Clips from Video for Efficient Action Recognition

Bruno Korbar  Du Tran  Lorenzo Torresani
Facebook AI
{bkorbar, trandu, torresani}@fb.com

Abstract

While many action recognition datasets consist of collections of brief, trimmed videos each containing a relevant action, videos in the real-world (e.g., on YouTube) exhibit very different properties: they are often several minutes long, where brief relevant clips are often interleaved with segments of extended duration containing little change. Applying densely an action recognition system to every temporal clip within such videos is prohibitively expensive. Furthermore, as we show in our experiments, this results in suboptimal recognition accuracy as informative predictions from relevant clips are outnumbered by meaningless classification outputs over long uninformative sections of the video.

In this paper we introduce a lightweight “clip-sampling” model that can efficiently identify the most salient temporal clips within a long video. We demonstrate that the computational cost of action recognition on untrimmed videos can be dramatically reduced by invoking recognition only on these most salient clips. Furthermore, we show that this yields significant gains in recognition accuracy compared to analysis of all clips or randomly/uniformly selected clips. On Sports1M, our clip sampling scheme elevates the accuracy of an already state-of-the-art action classifier by 7% and reduces by more than 15 times its computational cost.

1. Introduction

Most modern action recognition models operate by applying a deep CNN over clips of fixed temporal length [40, 6, 44, 50, 11]. Video-level classification is obtained by aggregating the clip-level predictions over the entire video, either in the form of simple averaging or by means of more sophisticated schemes modeling temporal structure [33, 45, 17]. Scoring a clip classifier densely over the entire sequence is a reasonable approach for short videos. However, it becomes computationally impractical for real-world videos that may be up to an hour long, such as some of the sequences in the Sports1M dataset [24]. In addition to the issue of computational cost, long videos often include segments of extended duration that provide irrelevant information for the recognition of the action class. Pooling information from all clips without consideration of their relevance may cause poor video-level classification, as informative clip predictions are outnumbered by uninformative predictions over long unimportant segments.

In this work we propose a simple scheme to address these problems (see Fig. 1 for a high-level illustration of the approach). It consists in training an extremely lightweight network to determine the saliency of a candidate clip. Because the computational cost of this network is more than one order of magnitude lower than the cost of existing 3D CNNs for action recognition [6, 44], it can be evaluated efficiently over all clips of even long videos. We refer to our network as SCSampler (Salient Clip Sampler), as it samples a reduced set of salient clips from the video for analysis by the action classifier. We demonstrate that restricting the costly action classifier to run only on the clips identified as the most salient by SCSampler, yields not only significant savings in runtime but also large improvements in video classification accuracy: on Sports1M our scheme yields a speedup of $15 \times$ and an accuracy gain of 7% over an already state-of-the-art classifier.

Efficiency is a critical requirement in the design of SCSampler. We present two main variants of our sampler. The first operates directly on compressed video [23, 52, 56], thus eliminating the need for costly decoding. The second looks only at the audio channel, which is low-dimensional and can therefore be processed very efficiently. As in recent multimedia work [2, 4, 15, 35], our audio-based sampler exploits the inherent semantic correlation between the audio and the visual elements of a video. We also show that combining our video-based sampler with the audio-based sampler leads to further gains in recognition accuracy.

We propose and evaluate two distinct learning objectives for salient clip sampling. One of them trains the sampler to operate optimally with the given clip classifier, while the second formulation is classifier-independent. We show that, in some settings, the former leads to improved accuracy, while the benefit of the latter is that it can be used without retraining with any clip classifier, making this model a
general and powerful off-the-shelf tool to improve both the runtime and the accuracy of clip-based action classification. Finally, we show that although our sampler is trained over specific action classes in the training set, its benefits extend even to recognition of novel action classes.

2. Related work

The problem of selecting relevant frames, clips or segments within a video has been investigated for various applications. For example, video summarization [18, 19, 29, 37, 56, 57, 58] and the automatic production of sport highlights [30, 31] entail creating a much shorter version of the original video by concatenating a small set of snippets corresponding to the most informative or exciting moments. The aim of these systems is to generate a video composite that is pleasing and compelling for the user. Instead the objective of our model is to select a set of segments of fixed duration (i.e., clips) so as to make video-level classification as accurate and as unambiguous as possible.

More closely related to our task is the problem of action localization [22, 39, 38, 54, 61], where the objective is to localize the temporal start and end of each action within a given untrimmed video and to recognize the action class. Action localization is often approached through a two-step mechanism [5, 8, 5, 14, 15, 21, 28, 1], where first an action proposal method identifies candidate action segments, and then a more sophisticated approach validates the class of each candidate and refines its temporal boundaries. Our framework is reminiscent of this two-step solution, as our sampler can be viewed as selecting candidate clips for accurate evaluation by the action classifier. However, several key differences exist between our objective and that of action localization. Our system is aimed at video classification, where the assumption is that each video contains a single action class. Action proposal methods solve the harder problem of finding segments of different lengths and potentially belonging to different classes within the input video. While in action localization the validation model is typically trained using the candidate segments produced by the proposal method, the opposite is true in our scenario: the sampler is learned for a given pretrained clip classifier, which is left unmodified by our approach. Finally, the most fundamental difference is that high efficiency is a critical requirement in the design of our clip sampler. Our sampler must be orders of magnitude faster than the clip classifier to make our approach worthwhile. Conversely, most action proposal or localization methods are based on optical flow [27, 28] or deep action-classifier features [5, 15, 54] that are typically at least as expensive to compute as the output of a clip classifier. For example, the TURN TAP system [14] is one of the fastest existing action proposal methods and yet, its computational cost exceeds by more than one order of magnitude that of our scheme. For 60 seconds of untrimmed video, TURN TAP has a cost of 4128 GFLOPs; running densely our clip classifier (MC3-18 [44]) over the 60 seconds would actually cost less, at 1097 GFLOPs; our sampling scheme lowers the cost down dramatically, to only 168 GFLOPs.

Closer to our intent are methods that remove from consideration uninformative sections of the video. This is typically achieved by means of temporal models that “skip” segments by leveraging past observations to predict which future frames to consider next [55, 10, 53]. Instead of learning to skip, our approach relies on a fast sampling procedure that evaluates all segments in a video and then performs further analysis on the most salient ones.

Our approach belongs to the genre of work that performs video classification by aggregating temporal information from long videos [13, 32, 33, 36, 45, 46, 47, 48, 49, 51, 62]. Our aggregation scheme is very simple, as it merely averages the scores of action classifiers over the selected clips. Yet, we note that the most recent state-of-the-art action classifiers operate precisely under this simple scheme. Examples include Two-Stream Networks [40], I3D [6], R(2+1)D [44], Non-Local Networks [50], SlowFast [11]. While in these prior studies clips are sampled densely or at random, our experiment suggest that our sampling strategy yields significant gains in accuracy over both dense, random, and uniform sampling and it is as fast as random sampling.

3. Technical approach

Our approach consists in extracting a small set of relevant clips from a video by scoring densely each clip with a lightweight saliency model. We refer to this model as the “sampler” since it is used to sample clips from the video. We formally define the task in subsection 3.1, proceed to present two different learning objectives for the sampler in section 3.2, and finally discuss sampler architecture choices and features in subsection 3.3.
3.1. Problem Formulation

Video classification from clip-level predictions. We assume we are given a pretrained action classifier $f : \mathbb{R}^{F \times 3 \times H \times W} \rightarrow [0, 1]^C$ operating on short, fixed-length clips of $F$ RGB frames with spatial resolution $H \times W$ and producing output classification probabilities over a set of action classes $\{1, \ldots, C\}$. We note that most modern action recognition systems [6, 12, 42, 44] fall under this model and, typically, they constrain the number of frames $F$ to span just a handful of seconds in order to keep memory consumption manageable during training and testing. Given a test video $v \in \mathbb{R}^{T \times 3 \times H \times W}$ of arbitrary length $T$, video-level classification through the clip-classifier $f$ is achieved by first splitting the video $v$ into a set of clips $\{v(i)\}_{i=1}^L$, with each clip $v(i) \in \mathbb{R}^{F \times 3 \times H \times W}$ consisting of $F$ adjacent frames and where $L$ denotes the total number of clips in the video. The splitting is usually done by taking clips every $F$ frames in order to have a set of non-overlapping clips that spans the entirety of the video. A final video-level prediction is then computed by aggregating the individual clip-level predictions. In other words, if we denote with $\text{agg}r$ the aggregation operator, the video-level classifier $\hat{f}$ is obtained as $\hat{f}(v) = \text{agg}r(\{f(v(i))\}_{i=1}^L)$. Most often, the aggregator is a simple pooling operator which averages the individual clip scores (i.e., $f(v) = 1/L \sum_{i=1}^L f(v(i))$) [6, 11, 40, 44, 50] but more sophisticated schemes based on RNNs [34] have also been employed.

Video classification from selected clips. In this paper we are interested in scenarios where the videos $v$ are untrimmed and may be quite long. In such cases, applying the clip classifier $f$ to every clip will result in a very large inference cost. Furthermore, aggregating predictions from the entire video may produce poor action recognition accuracy since in long videos the target action is unlikely to be exhibited in every clip. Thus, our objective is to design a method that can efficiently identify a subset $\hat{S}(v; K)$ of $K$ salient clips in the video (i.e., $\hat{S}(v; K) \in 2^{\{1, \ldots, L\}}$ with $|\hat{S}(v; K)| = K$) and to reduce video-level prediction to be computed from this set of $K$ clip-level predictions as $\hat{f}_{\hat{S}(v; K)}(v) = \text{agg}r(\{f(v(i))\}_{i \in \hat{S}(v; K)})$. (K is hyperparameter studied in our experiments). By constraining the application of the costly classifier $f$ to only $K$ clips, inference will be efficient even on long videos. Furthermore, by making sure that $\hat{S}(v; K)$ includes a sample of the most salient clips in $v$, recognition accuracy may improve as irrelevant or ambiguous clips will be discarded from consideration and will be prevented from polluting the video-level prediction. We note that in this work we address the problem of clip selection for a given pretrained clip classifier $f$, which is left unmodified by our method. This renders our approach useful as a post-training procedure to further improve performance of existing classifiers both in terms of inference speed as well as recognition accuracy.

Our clip sampler. In order to achieve our goal we propose a simple solution that consists in learning a highly efficient clip-level saliency model $s(.)$ that provides for each clip in the video a “saliency score” in $[0, 1]$. Specifically, our saliency model $s(.)$ takes as input clip features $\phi(i) = \phi(v(i)) \in \mathbb{R}^d$ that are fast to compute from the raw clip $v(i)$ and that have low dimensionality ($d$) so that each clip can be analyzed very efficiently. The saliency model $s : \mathbb{R}^d \rightarrow [0, 1]$ is designed to be orders of magnitude faster than $f$, thus enabling the possibility to score $s$ on every single clip of the video to find the $K$ most salient clips without adding any significant overhead. The set $\hat{S}(v; K)$ is then obtained as $\hat{S}(v; K) = \text{topK}\{|s(\phi(i))|_{i=1}^L\}$ where $\text{topK}$ returns the indices of the top-$K$ values in the set. We show that evaluating $f$ on these selected set, i.e., computing $\hat{f}_{\hat{S}(v; K)}(v) = \text{agg}r(\{f(v(i))\}_{i \in \hat{S}(v; K)})$ results in significantly higher accuracy compared to aggregating clip-level prediction over all clips.

In order to learn the sampler $s$, we use a training set $\mathcal{D}$ of untrimmed video examples, each annotated with a label indicating the action performed in the video: $\mathcal{D} = \{(v_1, y_1), \ldots, (v_n, y_n)\}$ with $v_n \in \mathbb{R}^{T_n \times 3 \times H \times W}$ denoting the $n$-th video and $y_n \in \{1, \ldots, C\}$ indicating its action label. In our experiments, we use as training set $\mathcal{D}$ the same set of examples that was used to train the clip classifier $f$. This setup allows us to demonstrate that the gains in recognition accuracy are not due to leveraging additional data but instead are the result of learning to detect the most salient clips for $f$ within each video.

Oracle sampler. In this work we compare our sampler against an “oracle” $O$ that makes use of the action label $y$ to select the best $K$ clips in the video for classification with $f$. The oracle set is formally defined as $\mathcal{O}(v, y; K) = \text{topK}\{|f_y(v(i))|_{i=1}^L\}$. Note that $\mathcal{O}$ is obtained by looking for the clips that yield the $K$ highest action classification scores for the ground-truth label $y$ under the costly action classifier $f$. In real scenarios the oracle cannot be constructed as it requires knowing the true label and it involves dense application of $f$ over the entire video, which defeats the purpose of the sampler. Nevertheless, in this work we use the oracle to obtain an upper bound on the accuracy of the sampler. Furthermore, we apply the oracle to the training set $\mathcal{D}$ to form pseudo ground-truth data to train our sampler, as discussed in the next subsection.

3.2. Learning Objectives for SCSampler

We consider two choices of learning objective for the sampler and experimentally compare them in 4.2.1.

3.2.1 Training the sampler as an action classifier

A naïve way to approach the learning of the sampler $s$ is to first train a lightweight action classifier $h(\phi(i)) \in [0, 1]^C$ on the training set $\mathcal{D}$ by forming clip examples $(\phi(i), y_n)$ us-
ing the low-dimensional clip features $\phi_n^{(i)} = \phi(v_n^{(i)}) \in \mathbb{R}^d$. Note that this is equivalent to assuming that every clip in the training video contains a manifestation of the target action. Then, given a new untrimmed test video $v$, we can compute the saliency score of a clip in the video as the maximum classification score over the $C$ classes, i.e.,

$$s(\phi^{(i)}) = \max_{c \in \{1, \ldots, C\}} h_c(\phi^{(i)}).$$

The rationale behind this choice is that a salient clip is expected to elicit a strong response by the classifier, while irrelevant or ambiguous clips are likely to cause weak predictions for all classes. We refer to this variant of our loss as AC (Action Classification).

### 3.3.2 Training the sampler as a saliency ranker

One drawback of AC is that the sampler is trained as an action classifier independently from the model $f$ and by assuming that all clips are equally relevant. Instead, ideally we would like the sampler to select clips that are most useful to our given $f$. To achieve this goal we propose to train the sampler to recognize the relative importance of the clips within a video with respect to the classification output of $f$ for the correct action label. To achieve this goal, we define pseudo ground-truth binary labels $z_{n}^{(i,j)}$ for pairs of clips $(i, j)$ from the same video $v_n$:

$$z_n^{(i,j)} = \begin{cases} 
1 & \text{if } f_{yn}(v_n^{(i)}) > f_{yn}(v_n^{(j)}) \\
-1 & \text{otherwise}
\end{cases} \quad (1)$$

We train $s$ by minimizing a ranking loss over these pairs:

$$\ell(\phi_n^{(i)}, \phi_n^{(j)}) = \max \left( -z_{n}^{(i,j)}[s(\phi_n^{(i)}) - s(\phi_n^{(j)}) + \eta], 0 \right) \quad (2)$$

where $\eta$ is a margin hyper-parameter. This loss encourages the sampler to rank higher clips that produce a higher classification score under $f$ for the correct label. We refer to this sampler loss as SAL-RANK (Saliency Ranking).

### 3.3.3 Sampler Architecture

Due to the tight runtime requirements, we restrict our sampler to operate on two types of features that can be computed efficiently from video and that yield a very compact representation to process. The first type of features are obtained directly from the compressed video without the need for decoding. Prior work has shown that features computed from compressed video can even be used for action recognition [52]. We describe in detail these features in subsection 3.3.1. The second type of features are audio features, which are even more compact and faster to compute than the compressed video features. Recent work [2, 3, 4, 15, 26, 35, 60] has shown that the audio channel provides strong cues about the content of the video and this semantic correlation can be leveraged for various applications. In subsection 3.3.2 we discuss how we can exploit the low-dimensional audio modality to find efficiently salient clips in a video.

### 3.3.1 Visual sampler

Wu et al. [52] recently introduced an accurate action recognition model directly trained on compressed video. Modern codecs such as MPEG-4 and H.264 represent video in highly compressed form by storing the information in a set of sparse I-frames, each followed by a sequence of P-frames. An I-frame (IF) represents the RGB-frame in a video just as an image. Each I-frame is followed by 11 P-frames, which encode the 11 subsequent frames in terms of motion displacement (MD), and RGB-residual (RGB-R). MDs capture the frame-to-frame 2D motion while RGB-Rs store the remaining difference in RGB values between adjacent frames after having applied the MD field to rewar the frame. In [52] it was shown that each of these three modalities (IFs, MDs, RGB-Rs) provides useful information for efficient and accurate action recognition in video. Inspired by this prior work, here we train three separate ResNet-18 networks [20] on these three inputs as samplers using the learning objectives outlined in the previous subsection. The first ResNet-18 takes as input an IF of size $H \times W \times 3$. The second is trained on MD frames, which have size $H/16 \times W/16 \times 2$: the 2 channels encode the horizontal and vertical motion displacements at a resolution that is 16 times smaller than the original video. The third ResNet-18 is fed individual RGB-Rs of size $H \times W \times 3$. At test time we average the predictions of these 3 models over all the I-frames and P-frames (MDs and RGB-Rs) within the clip to obtain a final global saliency score for the clip. As an alternative to ResNet-18, we experimented also with a lightweight ShuffleNet architecture [59] of 26 layers. We compare these models in 4.2.2. We do not present results for the large ResNet-152 model that was used in [52], since it adds a cost of 3 GFLOPS per clip which far exceeds the computational budget of our application.

### 3.3.2 Audio sampler

We model our audio sampler after the VGG-like audio networks used in [7, 2, 26]. Specifically, we first extract MEL-spectrograms from audio segments twice as long as the video-clips, but with stride equal to the video-clip length. This stride is chosen to obtain an audio-based saliency score for every video clip used by the action recognizer $f$. However, for the audio sampler we use an observation window twice as long as the video clip since we found this to yield better results. A series of 200 time samples is taken within each audio segment and processed using 40 MEL filters. This yields a descriptor of size $40 \times 200$. This representation is compact and can be analyzed efficiently by the sampler. We treat this descriptor as an image and process it using a VGG network [41] of 18 layers. The details of the architecture are given in the supplementary material.
3.3.3 Combining video and audio saliency

Since audio and video provide correlated but distinct cues, we investigated several schemes for combining the saliency predictions from these two modalities. With AV-convex-score we denote a model that simply combines the audio-based score \( s^A(v(i)) \) and the video-based score \( s^V(v(i)) \) by means of a convex combination \( \alpha s^V(v(i)) + (1 - \alpha)s^A(v(i)) \) where \( \alpha \) is a scalar hyperparameter. The scheme AV-convex-list instead first produces two separate ranked lists by sorting the clips within each video according to the audio sampler and the visual sampler independently. Then the method computes for each clip the weighted average of its ranked position in the two lists according to a convex combination of the two positions. The top-\( K \) clips according to this measure are finally retrieved. The method AV-intersect-list computes an intersection between the top-\( m \) clips of the audio sampler and the top-\( m \) clips of the video sampler. For each video, \( m \) is progressively increased until the intersection yields exactly \( K \) clips. In AV-union-list we form a set of \( K \) clips by selecting \( K' \)-top clips according to the visual sampler (with hyperparameter \( K' \) s.t. \( K' < K \)) and by adding to it a set of \( K - K' \) different clips from the ranked list of the audio sampler. Finally, we also present results for AV-joint-training, where we simply average the audio-based score and the video-based score and then finetune the two networks with respect to this average.

4. Experiments

In this section we evaluate the proposed sampling procedure on the large-scale Sports1M and Kinetics datasets.

4.1. Large-scale action recognition with SCSampler

4.1.1 Experimental Setup

Action Recognition Networks. Our sampler can be used with any clip-based action classifier \( f \). We demonstrate the general applicability of our approach by evaluating it with six popular 3D CNNs for action recognition. Four of these models are pretrained networks publicly available [9] and described in detail in [44]: they are 18-layer instantiations of ResNet3D (R3D), Mixed Convolutional Network (MC3), and R(2+1)D, with this last network also in a 34-layer configuration. The other two models are our own implementation of I3D-RGB [6] and a ResNet3D of 152 layers leveraging depthwise convolutions (ir-CSN-152) [43]. These networks are among the state-of-the-art on Kinetics and Sports1M. For training procedure, please refer to supplementary material.

Sampler configuration. In this subsection we present results achieved with the best configuration of our sampler architecture, based on the experimental study that we present in section 4.2. The best configuration is a model that combines the saliency scores of an audio sampler and of a video sampler, using the strategy of AV-union-list. The video sampler is based on two ResNet-18 models trained on MD and RGB-R features, respectively, using the action classification loss (AC). The audio sampler is trained with the saliency ranking loss (SAL-RANK). Our sampler \( s(.) \) is optimized with respect to the given clip classifier \( f \). Thus, we train a separate clip sampler for each of the 6 architectures in this evaluation. All results are based on sampling \( K = 10 \) clips from the video, since this is the best hyper-parameter value according to our experiments (see analysis in supplementary material).

Baselines. We compare the action recognition accuracy achieved with our sampler, against three baseline strategies to select \( K = 10 \) clips from the video: Random chooses clips at random, Uniform selects clips uniformly spaced out, while Empirical samples clips from the discrete empirical distribution (i.e., a histogram) of the top \( K = 10 \) Oracle clip locations over the entire training set (the histogram is computed by linearly remapping the temporal extent of each video to be in the interval \([0, 1]\)). Finally, we also include video classification accuracy obtained with Dense which performs “dense” evaluation by averaging the clip-level predictions over all non-overlapping clips in the video.

4.1.2 Evaluation on Sports1M

Our approach is designed to operate on long, real-world videos where it is neither feasible nor beneficial to evaluate every single clip. For these reasons, we choose the Sports1M dataset [24] as a suitable benchmark since its average video length is 5 minutes and 36 seconds, and some of its videos exceed 1 hour. We use the official training/test split. We do not trim the test videos and instead seek the top \( K = 10 \) clips according to our sampler in each video. We stress that our sampling strategy is applied to test videos only. The training videos in Sports1M are also untrimmed. As training on all training clips would be unfeasible, we use the training procedure described in [44] which consists in selecting from each training video 10 random 2-second segments, from which training clips are formed. We reserve to future work the investigation of whether our sampling can be extended to sample training clips from the full videos.

We present the results in Table 1, which includes for each method the video-level classification accuracy as well as the cumulative runtime (in days) to run the inference on the complete test set using 32 NVIDIA P100 GPUs (this includes the time needed for sampling as well as clip-level action classification). The most direct baselines for our evaluation are Random, Uniform and Empirical which use the same number of clips (\( K \)) in each video as SCSampler. It can be seen that compared to these baselines, SCSampler delivers a substantial accuracy gain for all action models, with improvements ranging from 6.0% for R(2+1)D-34 to 9.9% for R(2+1)D-18 with respect to Empirical, which does
Table 1: Video-level classification on Sports1M [24] using $K$ clips selected by our SCSampler, chosen at “Random” or with “Uniform” spacing, by sampling clips according to the “Empirical” distribution computed on the training set, as well as “Dense” evaluation on all clips. Oracle uses the true label of the test video to select clips. Runtime is the total time for evaluation over the entire test set. SCSampler delivers large gains over Dense, Random, Uniform and Empirical while keeping inference efficient. For ir-CSN-152, SCSampler yields a gain of 7.0% over the already state-of-the-art accuracy of 77.0% achieved by Dense.

<table>
<thead>
<tr>
<th>Classifier</th>
<th>SCSampler $S$ (K clips) accuracy (%)</th>
<th>Random / Uniform / Empirical (K clips) accuracy (%)</th>
<th>Dense (all clips) accuracy (%)</th>
<th>Oracle $O$ (K clips) accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tr: MC3-18</td>
<td>72.8 / 65.5 / 65.3</td>
<td>64.5 / 63.2 / 63.9</td>
<td>66.6 / 12.9 / 12.9</td>
<td>85.1 / 5.0 / 5.0</td>
</tr>
<tr>
<td>Tr: R(2+1)D-18</td>
<td>73.9 / 63.7 / 63.9</td>
<td>63.0 / 62.9 / 63.9</td>
<td>68.7 / 13.1 / 13.1</td>
<td>87.0 / 7.0 / 7.0</td>
</tr>
<tr>
<td>Tr: R3D-18</td>
<td>70.2 / 59.8 / 60.3</td>
<td>59.8 / 59.9 / 60.3</td>
<td>65.6 / 13.3 / 13.3</td>
<td>85.0 / 5.0 / 5.0</td>
</tr>
<tr>
<td>Tr: R(2+1)D-34*</td>
<td>78.0 / 71.2 / 72.0</td>
<td>71.2 / 71.5 / 72.0</td>
<td>70.9 / 14.2 / 14.2</td>
<td>88.4 / 4.0 / 4.0</td>
</tr>
<tr>
<td>Tr: ir-CSN-152*</td>
<td>84.0 / 75.3 / 75.8</td>
<td>75.3 / 75.8 / 76.2</td>
<td>77.0 / 14.0 / 14.0</td>
<td>92.6 / 3.0 / 3.0</td>
</tr>
</tbody>
</table>

Table 2: Video-level classification on Kinetics [25] using $K$ clips selected using our SCSampler, chosen at “Random” or with “Uniform” spacing, by sampling clips according to the “Empirical” distribution computed on the training set, as well as “Dense” evaluation on all clips. Even though Kinetics videos are short (10 seconds) our sampling procedure provides consistent accuracy gains for all 6 networks, compared to Random and Uniform clip selection or even Dense evaluation. Models marked with “*” are pretrained on Sports1M, and models with “**” are pretrained as 2D CNNs on ImageNet and then 3D-inflated [6].

<table>
<thead>
<tr>
<th>Classifier</th>
<th>SCSampler $S$ (K clips) accuracy (%)</th>
<th>Random / Uniform / Empirical (K clips) accuracy (%)</th>
<th>Dense (all clips) accuracy (%)</th>
<th>Oracle $O$ (K clips) accuracy (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Tr: MC3-18</td>
<td>67.0 / 63.0 / 63.6</td>
<td>63.0 / 63.4 / 63.6</td>
<td>65.1 / 2.3 / 2.3</td>
<td>82.0 / 3.0 / 3.0</td>
</tr>
<tr>
<td>Tr: R(2+1)D-18</td>
<td>70.9 / 65.9 / 66.3</td>
<td>65.9 / 66.2 / 66.3</td>
<td>68.0 / 2.4 / 2.4</td>
<td>85.4 / 3.0 / 3.0</td>
</tr>
<tr>
<td>Tr: R3D-18</td>
<td>67.3 / 63.6 / 64.0</td>
<td>63.6 / 63.8 / 64.0</td>
<td>65.2 / 2.4 / 2.4</td>
<td>83.0 / 3.0 / 3.0</td>
</tr>
<tr>
<td>Tr: R(2+1)D-34**</td>
<td>76.7 / 73.8 / 74.1</td>
<td>73.8 / 74.0 / 74.1</td>
<td>74.1 / 3.1 / 3.1</td>
<td>82.9 / 3.0 / 3.0</td>
</tr>
<tr>
<td>Tr: I3D-RGB**</td>
<td>75.1 / 71.9 / 71.9</td>
<td>71.9 / 71.8 / 71.9</td>
<td>72.8 / 2.9 / 2.9</td>
<td>81.2 / 3.0 / 3.0</td>
</tr>
<tr>
<td>Tr: ir-CSN-152**</td>
<td>80.2 / 77.8 / 79.2</td>
<td>77.8 / 78.5 / 79.2</td>
<td>78.8 / 3.0 / 3.0</td>
<td>89.0 / 3.0 / 3.0</td>
</tr>
</tbody>
</table>

Table 3: Cross-dataset and cross-classifier performance. Numbers report MC3-18 video-level accuracy on the validation set of Kinetics (first row) and test set of Sports1M (second row). SCSampler outperforms Uniform even when optimized for a different classifier (R(2+1)D) and a different dataset (e.g., 68.5% vs 64.8% for Sports1M).

Our approach does also better than “Dense” prediction, which averages the action classification predictions over all non-overlapping clips. To the best of our knowledge the accuracy of 77.0% achieved by ir-CSN-152 using Dense evaluation is currently the best published result on this benchmark. SCSampler provides an additional gain of 7.0% over this state-of-the-art model, pushing the accuracy to 84.0%. We note that when using ir-CSN-152, Dense requires 14 days whereas SCSampler achieves better accuracy and requires only 0.65 days to run inference on the Sports1M test set. Finally, we report also the performance of the “Oracle” $O$, which selects the $K$ clips that yield the highest classification score for the true class of the test video. This is an impractical model but it gives us an informative upper bound on the accuracy achievable with an ideal sampler.

Fig. 2 (left) shows the histogram of the clip temporal locations using $K = 10$ samples per video for the test set of Sports1M (after remapping the temporal extent of each video to $[0, 1]$). Oracle and SCSampler produce similar distributions of clip locations, with the first section and especially the last section of videos receiving many more samples. It can be noted that Empirical shows a different sample distribution compared to Oracle. This is due to the fact that it computes the histogram from the training set which in this case appears to have different statistics from the test set.

Thumbnails of top-ranked and bottom-ranked clips for two test videos are shown in Fig. 3.

4.1.3 Evaluation on Kinetics

We further evaluate SCSampler on the Kinetics [25] dataset. Kinetics is a large-scale benchmark for action recognition containing 400 classes and 280K videos (240K for training and 40K for testing), each about 10 seconds long. The results are reported in Table 2. Kinetics videos are short and thus in principle the recognition model should only marginally better than Random and Uniform.
not benefit from a clip-sampling scheme such as ours. Nevertheless, we see that for all architectures SCSampler provides accuracy gains over Random/Uniform/Empirical selection and Dense evaluation, although the improvements are understandably less substantial than in the case of Sports1M. To the best of our knowledge, the accuracy of 80.2% achieved by ir-CSN-152 with our SCSampler is the best reported result so far on this benchmark.

Note that [43] reports an accuracy of 79.0% using Uniform (instead of the 78.5% we list in Table 2, row 6) but this accuracy is achieved by applying the clip classifier spatially in a fully-convolutional fashion on frames of size 256x256, whereas here we use a single center spatial crop of size 224x224 for all our experiments. Sliding the clip classifier spatially in a fully-convolutional fashion (as in [43]) raises the accuracy of SCSampler to 81.1%.

Fig. 2 (right) shows the histogram of clip temporal locations on the validation set of Kinetics. Compared to Sports1M, the Oracle and SCSampler distributions here is much more uniform.

4.1.4 Unseen Action Classifiers and Novel Classes

While our SCSampler has low computational cost, it adds the procedural overhead of having to train a specialized clip selector for each classifier and each dataset. Here we evaluate the possibility of reusing a sampler $s(\cdot)$ that was optimized for a classifier $f$ on a dataset $D$, for a new classifier $f'$ on a dataset $D'$ that contains action classes different from those seen in $D$. In Table 3, we present cross-dataset performance of an SCSampler trained on Kinetics but then used to select clips on Sports1M (and vice-versa). We also report cross-classifier performance obtained by optimizing SCSampler with pseudo-ground truth labels (see section 3.2.2) generated by R(2+1)D-18 but then used for video-level prediction with action classifier MC3-18. On the Kinetics validation set, using an SCSampler that was trained using the same action classifier (MC3) but a different dataset (Sports1M) causes a drop of about 2% (65.0% vs 67.0%) while training using a different action classifier (R(2+1)D) to generate pseudo-ground truth labels on the the same dataset (Kinetics) causes a degradation of 1.1% (65.9% vs 67.0%). The evaluation on Sports1M shows a similar trend, where cross-dataset accuracy (69.2%) is lower than cross-classifier accuracy (72.1%). Even in the extreme setting of cross-dataset and cross-classifier, the accuracies achieved with SCSampler are still better than those obtained with Random or Uniform selection. Finally, we note that samplers trained using the AC loss (section 3.2.1) do not require pseudo-labels and thus are independent of the action classifier by design.

4.2. Evaluating Design Choices for SCSampler

In this subsection we evaluate the different choices in the design of SCSampler. Given the many configurations to assess, we make this study more computationally feasible by restricting the evaluation to a subset of Sports1M, which we name miniSports. The dataset is formed by randomly choosing for each class 280 videos from the training set and 69 videos from the test set. This gives us a class-balanced set of 136,360 training videos and 33,603 test videos. All videos are shortened to the same length of 2.75 minutes. For our assessment, we restrict our choice of action classifier to MC3-18, which we retrain on our training set of miniSports. We assess the SCSampler design choices in terms of how they affect the video-level accuracy of MC3-18 on the test set of miniSports, since our aim is to find the best configuration for video classification.

4.2.1 Learning objective

We begin by studying the effect of the loss function used for training SCSampler, by considering the two loss variants described in section 3.2. For this evaluation, we assess separately the visual sampler and the audio sampler. The video sampler is based on two ResNet-18 networks with MD and RGB-R features, respectively. These 2 networks are pretrained on ImageNet and then finetuned on the training set of miniSport for each of the three different SCSampler loss functions. The audio sampler is our VGG network pretrained for classification on AudioSet [16] and then finetuned on the training set of miniSports. The MC3-18 video classification accuracy is 73.1% when the visual sampler is trained with the Action Classification (AC) loss whereas it is 64.8% when it is trained with the Saliency

---

Figure 2: Histogram of clip-sample locations on the test set of Sports1M (left) and validation set of Kinetics (right). The distribution of SCSampler matches fairly closely that of the Oracle.

Figure 3: Top-ranked and bottom-ranked clips by SCSampler for two test videos from Sports1M. Top-ranked clips often show the sports in action, while bottom-ranked clips tend to be TV-interviews or static segments with scoreboard. Clips are shown as thumbnails. To see the videos please visit http://scsampler.ai.
Ranking (SAL-RANK) loss. Conversely, we found that the audio sampler is slightly more effective when trained with the SAL-RANK loss as opposed to the AC loss (video-level accuracy is 67.8% with SAL-RANK and 66.4% with AC). A possible explanation for this difference in results is that the AC loss defines a more challenging problem to address (action classification vs binary ranking) but provides more supervision (multiclass vs binary labels). The model using compressed video features is a stronger model that can benefit from the AC supervision and do well on this task (as already shown in [52]) but the weaker audio model does better when trained on the simpler SAL-RANK problem.

4.2.2 Sampler architecture and features

In this subsection we assess different architectures and features for the sampler. For the visual sampler, we use the AC loss and consider two different lightweight architectures: ResNet-18 and ShuffleNet26. Each architecture is trained on each of the 3 types of video-compression features described in section 3.3.1: IF, MD and RGB-R. We also assess performance of combination of these three features by averaging the scores of classifiers based on individual features. The results are reported in Table 4. We can observe that given the same type of input features, ResNet-18 provides much higher accuracy than ShuffleNet-26 at a runtime that is only marginally higher. It can be noticed that MD and RGB-R features seem to be quite complementary: for ResNet-18, MD+RGB-R yields an accuracy of 73.1% whereas these individual features alone achieve an accuracy of only 68.0% and 63.5%. However, adding IF features to MD+RGB-R provides a modest gain in accuracy (74.9 vs 73.1) but impacts noticeably the runtime. Considering these tradeoffs, we adopt ResNet-18 trained on MD+RGB-R as our visual sampler on all subsequent experiments.

We perform a similar ablation study for the audio sampler. Given our VGG audio network pretrained for classification on AudioSet, we train it on miniSport using the following two options: finetuning the entire VGG model vs training a single FC layer on several VGG activations. Finetuning the audio sampler yields the best classification accuracy (see detailed results in supplementary material).

4.2.3 Combining audio and visual saliency

In this subsection we assess the impact of our different schemes for combining audio-based and video-based saliency scores (see 3.3.3). For this we use the best configurations of our visual and audio sampler (described in 4.1.1). Table 5 shows the video-level action recognition accuracy achieved for the different combination strategies.

Perhaps surprisingly, the best results are achieved with AV-union-list, which is the rather naïve solution of taking \( K' \) clips based on the video sampler and \( K - K' \) different clips based on the audio sampler \( (K' = 8 \) is the best value when \( K = 10 \)). The more sophisticated approach of joint training AV-joint-training performs nearly on-par with it. Overall, it is clear that the visual sampler is a better clip selector than the audio sampler. But considering the small cost of audio-based sampling, the accuracy gain provided by AV-union-list over visual only (76.0 vs 73.1) warrants the use of this combination.

5. Discussion

We presented a very simple scheme to boost both the accuracy and the speed of clip-based action classifiers. It leverages a lightweight clip-sampling model to select a small subset of clips for analysis. Experiments show that, despite its simplicity, our clip-sampler yields large accuracy gains and big speedups for 6 different strong action recognizers, and it retains strong performance even when used on novel classes. Future work will investigate strategies for optimal sample-set selection, by taking into account clip redundancies. It would be interesting to extend our sampling scheme to models that employ more sophisticated aggregations than simple averaging, e.g., those that use a set of contiguous clips to capture long-range temporal structure. SC-Sampler scores for the test videos of Kinetics and Sports1M are available for download at \http://scsampler.ai\.
References


