Deep Unsupervised Key Frame Extraction for Efficient Video Classification

HAO TANG, ETH Zurich
LEI DING, University of Trento
SONGSONG WU, Guangdong University of Petrochemical Technology
BIN REN, NICU SEBE, and PAOLO ROTA, University of Trento

Video processing and analysis have become an urgent task, as a huge amount of videos (e.g., YouTube, Hulu) are uploaded online every day. The extraction of representative key frames from videos is important in video processing and analysis since it greatly reduces computing resources and time. Although great progress has been made recently, large-scale video classification remains an open problem, as the existing methods have not well balanced the performance and efficiency simultaneously. To tackle this problem, this work presents an unsupervised method to retrieve the key frames, which combines the convolutional neural network and temporal segment density peaks clustering. The proposed temporal segment density peaks clustering is a generic and powerful framework, and it has two advantages compared with previous works. One is that it can calculate the number of key frames automatically. The other is that it can preserve the temporal information of the video. Thus, it improves the efficiency of video classification. Furthermore, a long short-term memory network is added on the top of the convolutional neural network to further elevate the performance of classification. Moreover, a weight fusion strategy of different input networks is presented to boost performance. By optimizing both video classification and key frame extraction simultaneously, we achieve better classification performance and higher efficiency. We evaluate our method on two popular datasets (i.e., HMDB51 and UCF101), and the experimental results consistently demonstrate that our strategy achieves competitive performance and efficiency compared with the state-of-the-art approaches.

CCS Concepts: • Networks → Network components; Network types;
Additional Key Words and Phrases: Key frame extraction, density peaks clustering, LSTM, weight fusion, unsupervised learning, video classification

ACM Reference format:
Hao Tang, Lei Ding, Songsong Wu, Bin Ren, Nicu Sebe, and Paolo Rota. 2023. Deep Unsupervised Key Frame Extraction for Efficient Video Classification. ACM Trans. Multimedia Comput. Commun. Appl. 19, 3, Article 119 (February 2023), 17 pages. https://doi.org/10.1145/3571735

This work was supported by the PRIN project PREVUE (Prot. 2017N2RK7K) and by the EU H2020 project AI4Media under grant 951911.
Authors’ addresses: H. Tang (corresponding author), Department of Information Technology and Electrical Engineering, ETH Zurich, Zurich 8092, Switzerland; email: hao.tang@vision.ee.ethz.ch; L. Ding, B. Ren, N. Sebe, and P. Rota, Department of Information Engineering and Computer Science (DISI), University of Trento, Trento 38123, Italy; emails: {lei.ding, bin.ren}@unitn.it, sebe@disi.unitn.it, paolo.rota@unitn.it; S. Wu, School of Computer Science, Guangdong University of Petrochemical Technology, Maoming 525000, China; email: sswuai@126.com.
Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
© 2023 Association for Computing Machinery.
1551-6857/2023/02-ART119 $15.00
https://doi.org/10.1145/3571735

ACM Trans. Multimedia Comput. Commun. Appl., Vol. 19, No. 3, Article 119. Publication date: February 2023.
1 INTRODUCTION

Recent years have witnessed an exponential growth in video data availability on the web, such as YouTube. In this situation, large-scale video classification techniques [3, 5, 10, 12–14, 18, 21, 24, 31, 32, 43, 54, 57, 67, 69–71, 74–76] have also received increasing interest due to the fact that there is an increasing demand for efficient indexing and managing of these video data.

Efficient and accurate large-scale video classification performance relies on the extraction of discriminative spatial and temporal features. Conventional approaches extract information or feature using handcrafted features (e.g., Histogram of Optical Flow (HOF) [30] or Improved Dense Trajectories (iDT) [63]) from video frames, which are then encoded (e.g., Fisher Vector (FV) [63]) and pooled (e.g., average pooling) to produce a global feature representation and then passed to a classifier (e.g., SVM).

In recent years, video classification research has been influenced by the trends of deep learning. A common pipeline of these works is to use a group of frames as the input to the network, whereby the model is expected to learn spatio-temporal features. Simonyan and Zisserman [48] propose a two-stream ConvNet architecture that incorporates spatial and temporal networks to extract spatial and temporal features. Feichtenhofer et al. [15] study a number of ways of fusing ConvNet towers both spatially and temporally to best take advantage of this spatio-temporal information. Wang et al. [67] present the Temporal Segment Network (TSN), which combines a sparse temporal sampling strategy and video-level supervision to enable efficient and effective learning using the whole action video. Ji et al. [23] introduce a 3D ConvNet model, in which both spatial and temporal features are extracted by performing 3D convolution. Tran et al. [55] propose an approach for spatio-temporal feature learning using deep 3D ConvNets.

However, these approaches for video classification treat the videos in a holistic way—that is, as one data instance. Feature representation learned from the entire video unavoidably brings redundant information from the repeated and unrelated video frames, which leads to two problems. The first problem is that processing the whole video data requires excessive computational resources and time. For example, Kulhare et al. [28] report that the optical flow data on 13K videos was 1.5 TB. From the computational time perspective, these methods require extensively long periods of training time to effectively optimize millions of parameters that represent the model. The second problem is that repeated and unrelated video frames such as blurry, feature-less, and background frames always overwhelm the targeted visual patterns, which sometimes confuse the classifier.

To fix these problems, many efforts have been made to explore the key frame extraction methods that can convert video processing to image processing. For example, Donahue et al. [9] employ a Long Short-Term Memory Network (LSTM) that is connected to the output of the underlying Convolutional Neural Network (CNN). However, this framework processes 16 sample frames selected evenly with a stride of eight frames from the full-length video as the video representation. Such average selection of samples may not consider all useful motion and spatial information. Besides, Zhu et al. [75] propose a key volume mining deep framework to identify key volumes to achieve high accuracy, but with a steep increase of the computational load.

To solve both limitations, we present a novel framework to model video representation in an efficient and accurate manner. Instead of trying to learn features over the entire video as in previous works, which always contain redundant information, resulting in degraded performance, we consider a different manner to extract video features over several key frames of the entire video, which is a fundamental way to alleviate the computation burden. Key frames, also named representative frames, are often employed to represent the story of a video. Key frame extraction is the key technology for video abstraction, which can remove the redundant information in the video and reduce the high computing redundancy between adjacent frames. The algorithm for key frame extraction will affect the reconstruction of video content.
Fig. 1. The pipeline of the proposed large-scale video classification, which has three components: CNN, TSDPC, and LSTM. The TSDPC is an unsupervised key frame extraction method that preserves the temporal information of the video and calculates the number of key frames automatically. The CNN-TSDPC is proposed for accelerating video processing. In addition, an LSTM cell is further connected at the end of the CNN, which is able to elevate the performance of video classification. Last, the LSTM predicts the video label at each key frame, and we average these predictions for final labels.

To this end, we propose an unsupervised key frame extraction method by using CNNs and density peaks clustering [44]. More specifically, we extract a convolutional deep feature for each frame in the video and then map these deep feature maps into a high-dimensional feature space. After that, how to describe the feature space is a hard nut to crack for its uneven distribution. To solve this, we propose **Temporal Segment Density Peaks Clustering (TSDPC)** to select key frames in an unsupervised way in feature space. To preserve the temporal cues of the video, we first segment it into several segments, then select key frames on each segment using density peaks clustering. Finally, we combine all the key frames extracted from each segment to form the final key frames. TSDPC has two advantages compared with the previous clustering-based method. First, previous clustering approaches cannot detect non-spherical clusters due to the fact that they only rely on the distance between feature points to do clustering, whereas TSDPC is the approach based on the local density of feature points, which is able to detect non-spherical clusters. Second, clumsy tricks are used to ensure the number of key frames that may bring uncertainty in previous works, whereas for TSDPC, it can calculate the number of key frames automatically.

After extracting key frames, we replace the original video sequence with the key frames as a surrogate for analyzing, which could greatly enhance the time efficiency with little cost of accuracy. In addition, to improve the accuracy, an LSTM is further connected at the end of the CNN. Furthermore, a novel input network fusion strategy with different weights is presented. The pipeline of the proposed large-scale video classification framework is shown in Figure 1. Finally, experimental results show that the proposed framework is accurate and efficient for video classification on two public datasets.

The contributions of this work can be summarized as follows:

- We propose a novel method of unsupervised key frame extraction in video—CNN-TSDPC—which is comprised of CNN and TSDPC. Note that the proposed TSDPC can preserve the temporal information of a video and calculate the number of key frames automatically.
• The proposed CNN-TSDPC-LSTM framework is trained in an end-to-end fashion to improve both performance and efficiency for the video classification task.
• We present a weight fusion strategy of different input networks, in which different inputs are fused with different weights to elevate the accuracy of video classification.
• The proposed method and framework achieve competitive performance and are more efficient compared with the state-of-the-art models.

2 RELATED WORK
Video content summarization has been widely used to facilitate the indexing of large videos. In earlier works on video summarization, key frames are selected either by sampling video frames randomly or uniformly at certain time intervals. To make a key frame extraction algorithm effective, the extracted key frames should represent the whole video content without missing important information such as people and objects. At the same time, video content information of these key frames should not be similar, to avoid content redundancy. In recent years, many algorithms have been proposed for key frame extraction in videos. There are three categories of key frame extraction algorithms.

Segmentation Based. These methods detect abrupt changes in terms of similarity between successive frames. The key frames are equidistant in the video curve with respect to Iso-Content Distance, Iso-Content Error, and Iso-Content Distortion in the work of Panagiotakis et al. [39]. In the work of Ejaz et al. [11], a key frame is extracted if and only if the inter-frame difference overcomes a certain threshold. A key frame selection method based on key points is presented in the work of Guan et al. [20], in which a global pool of key points based on the SIFT feature extracted from all frames is generated and those frames that best cover the global key point pool are selected as key frames. In other words, this type of key frame extraction method has the disadvantage that is, it may extract similar key frames if the same content reappears during a video.

Dictionary Based. These types of approaches convert key frame extraction into a sparse dictionary selection problem. In the work of Cong et al. [6], a key frame selection method is proposed, which is based on the sparse dictionary selection with the loss in $L_{2,1}$ norm. Besides, the key frames are selected based on the true sparse constraint $L_0$ norm to represent the whole video in the work of Mei et al. [36]. The $L_{2,0}$ constrained sparse dictionary selection model is proposed to solve the problems that the solution based on the convex relaxation cannot guarantee the sparsity of the dictionary, and it selects key frames in a local point of view in the work of Mei et al. [35]. In the work of Meng et al. [37], a video is summarized into a few key objects by selecting representative object proposals generated from video frames based on the sparse dictionary selection method. To find key frames with both diversity and representativeness, the objective function in the work of Wang et al. [61] consists of a reconstruction error and three structured regularizers: a group sparsity regularizer, diversity regularizer, and locality-sensitivity regularizer.

Clustering Based. These approaches cluster frames into groups and then select the frames closest to the cluster centers as key frame. In the work of Zhuang et al. [77], key frames are detected using unsupervised clustering based on visual variations, whereas in the work of Kuanar et al. [26], dynamic Delaunay clustering is adopted to extract key frames. Key frames are selected based on color feature using the $k$-means clustering algorithm in the work of Avila et al. [7]. In the work of Vázquez-Martín and Bandera [59], spectral clustering on spatio-temporal features is employed to extract key frames. Moreover, the mutual information values of these consecutive frames are clustered into several groups using a split-merge method in the work of Cernekova et al. [4]. In the work of Panda et al. [40], the problem is modeled as a graph clustering problem, and it is solved
using a skeleton graph. Besides, the authors present a key frame extraction approach based on local description and graph modularity clustering in the work of Gharbi et al. [19].

However, the proposed CNN-TSDPC framework has three advantages compared with these approaches. First, our framework can capture the discriminative spatial information by using a sophisticated CNN feature extractor. Second, our framework can represent the temporal information by using a temporal segmentation strategy. Third, our framework can extract more representative key frames with less redundant information than previous works.

3 METHOD

We assume that a video $V$ can be divided into $K$ segments $\{V_1, V_2, \ldots, V_K\}$ of equal durations. For each segment $V_K$, we assume that a frame in $V_K$ can be represented by $f^K_i$, where $i \in [1, 2, \ldots, N]$ and $N$ is the number of frames in $V_K$,

$$V_K = \{ f^K_i \}_{i=1}^N. \quad (1)$$

Hence, the key frame set $f^K_{m_k}$ is defined as follows:

$$f^K_{m_k} = \Theta(V_K), k \in [1, 2, \ldots, n_c], \quad (2)$$

where $m_k$ is the index of the key frames, $n_c \geq 1$ is the number of key frames, and $\Theta$ denotes the key frame extraction procedure, and the objective is to remove the redundant data that will significantly reduce the amount of information to be processed. It is necessary to discard the frames with repetitive or redundant information during the extraction. Thus, key frame extraction is the fundamental step in video analysis applications.

3.1 Convolutional Deep Feature Extraction

We try to find a proper descriptive index to evaluate each frame in a segment $V_K$, facilitating key frame extraction. Informative frames could better summarize the whole video, but how to quantify the information each frame contains is a hard nut to crack. Previous works such as that of Avila et al. [7] adopt the color feature to represent each frame. Tang et al. [54] use the image entropy as a feature representation for hand action recognition. However, we argue that these feature extractors are not powerful to extract discriminative information. In this article, we use CNNs to extract the deep feature. CNNs are powerful due to their ability to extract the semantic features of an image. The first few convolutional layers can identify lines and corners, and then we pass these patterns down through more convolutional layers and start recognizing more complex and abstract semantic features as we go deeper. This property makes CNNs really good at extracting features in images and videos.

For each frame $f^K_i$ in the segment $V_K$, we adopt ResNet [22] as the deep feature extractor. This pre-trained model of ResNet is trained on a subset of the ImageNet dataset [45]. The model is trained on more than 1 million images and can classify images into 1,000 object categories. The feature vector $x^K_i$ can be obtained as follows:

$$x^K_i = \Delta(f^K_i), i \in [1, 2, \ldots, N], \quad (3)$$

where $\Delta$ denotes CNN feature extraction operation and $x^K_i$ is the corresponding feature vector of frame $f^K_i$. After extracting deep features of each frame via the feature extractor, feature vectors $x^K_i, i \in [1, 2, \ldots, N]$ are then mapped to the points in high-dimension feature space $F$.

3.2 Temporal Segment Density Peaks Clustering

We define the following symbols for the sake of simplicity:

$$S = \{ x^K_i \}_{i=1}^N, I_S = \{ 1, 2, \ldots, N \}. \quad (4)$$
We consider that the distribution of $S$ in $F$ should have the following two characteristics: (i) the cluster centers of $S$ are surrounded by neighbors with a lower local density, and (ii) these centers have a relatively large distance from any points with a higher local density. Thus, we adopt density peaks clustering to further cluster these feature vector $x^K_i$. Density peaks clustering \cite{44} could better catch the delicate spherical structure of space where points reside than traditional clustering strategies, such as $K$-means, in which features are grouped to the nearest cluster center.

For each point $x^K_i$ in $S$, we compute two quantities: the local density $\rho_i$ and its distance $\delta_i$ from points of higher density. Both of these quantities depend only on the distance $d_{ij}$ between points in $F$:

$$d_{ij} = \text{dist} \left( x^K_i, x^K_j \right).$$

(5)

The local density $\rho_i$ of point $x_i$ is defined as

$$\rho_i = \sum_{j \in \{I_S - \{i\}\}} \chi(d_{ij} - d_c),$$

(6)

where

$$\chi(x) = \begin{cases} 1, & x < 0, \\ 0, & x \geq 0, \end{cases}$$

(7)

and $d_c$ is a cutoff distance. Basically, $\rho_i$ equals to the number of points that are closer than $d_c$ to point $x^K_i$. The algorithm is sensitive only to the relative magnitude of $\rho_i$ in different points, which implies that the results of the analysis are robust with respect to the choice of $d_c$ for a large dataset.

A different way to define $\rho_i$ is

$$\rho_i = \sum_{j \in \{I_S - \{i\}\}} e^{-\frac{d_{ij}^2}{2d^2}},$$

(8)

in which a Gaussian kernel is used to calculate the local density. We can see from these two definitions that the cutoff kernel in Equation (6) is a discrete value, whereas the Gaussian kernel in Equation (8) is a continuous value, which guarantees a smaller probability of conflict. In other words, the probability that different points have the same local densities, $\rho_i$, is smaller.

Another important quantity is $\delta_i$, which is measured by the minimum distance between the point $x^K_i$ and any other point with higher density:

$$\delta_i = \begin{cases} \min_{j \in I_i^S} d_{ij}, & I_i^S \neq \emptyset, \\ \max_{j \in I_S} d_{ij}, & I_i^S = \emptyset, \end{cases}$$

(9)

where

$$I_i^S = \{ k \in I_S : \rho_k > \rho_i \}.$$  

(10)

Obviously, the $I_i^S = \emptyset$ if $\rho_i = \max_{j \in I_S} \rho_j$.

Consequently, for each point $x^K_i$ in $S$, we can calculate binary pair $(\rho_i, \delta_i)$, where $i \in I_S$. The definition of quantity $\gamma_i$ that considers both $\rho_i$ and $\delta_i$ is as follows:

$$\gamma_i = \rho_i \delta_i, i \in I_S.$$  

(11)

We select the point with the larger value of $\gamma_i$ as the cluster center. Figure 2 shows the selection of $\rho$, $\delta$, and $\gamma_i$ on two videos of the HMDB51 and UCF101 datasets, respectively.

In experiments, according to previous works on temporal modeling \cite{17, 64, 67}, we set $K$ to 3. Since we observe that when $K = 1$, the key frames extracted by the proposed method cannot preserve the temporal information in the whole input video. We do the same operations on other
segments and combine all the key frames from each segment to form the final key frames $f = \{f^1, f^2, \ldots, f^K\}$ of the video $V$.

The pipeline of the proposed CNN-TSDPC framework is summarized in Algorithm 1. The CNN-TSDPC method comprises three steps. The first step is to segment a video into several volumes equally (step 1.1). The second step is convolutional deep feature extraction, in which we extract deep feature vector $x_i$ using Equation (3) and then map feature vector $x_i$ to feature space $\mathcal{F}$ (step 2.1 and step 2.2). After that, we conduct density peaks clustering operation. We first calculate the distance $d_{ij}$ between $x_i$ and $x_j$, then calculate the cutoff distance $d_c$ with the given $t$ (step 3.1 and step 3.2). According to the assertions in the work of Rodriguez and Laio [44], $d_c$ can be selected as the rule that the average number of neighbors is around 1% to 2% of the total number of points in the dataset. To obtain $\gamma_i$, we need to calculate $\rho$ and $\delta$ with Equation (6) or Equation (8) and Equation (9) (step 3.3 and step 3.4), then multiply $\rho$ and $\delta$ to obtain $\gamma$ using Equation (11) (step 3.5). Next, we rank $\gamma_i$ in descending order and choose the $n_c$ largest $\gamma$ values as the clustering centers. In the end of our algorithm, the index of the key frames $m_k$ and key frames $\{f_{m_k}\}_{k=1}^{n_c}$ of $V_K$ are returned for further processing.

4 LARGE-SCALE VIDEO CLASSIFICATION

To perform the sequential image classification, we employ LSTMs, which are added at the top of CNNs. LSTMs have been widely used to advance the state of the art of many difficult problems.
ALGORITHM 1: The pipeline of the proposed CNN-TSDPC method

Require:  The video $V$.
Ensure: The key frames $f = \{f^1, f^2, \ldots, f^K\}$.

Step 1 Temporal segmentation.

1.1 Divide a video $V$ into $\{V_1, V_2, \ldots, V_K\}$; for each segment in $\{V_1, V_2, \ldots, V_K\}$, we do the following steps.

Step 2 Convolutional deep feature extraction.

2.1 Extract deep feature vector $x^K_i \leftarrow \text{Equation (3)}$.
2.2 Map $x^K_i$ to feature space $F$.

Step 3 Density peaks clustering.

3.1 Calculate $d_{ij}$ and $d_{ij} = d_{ji}, i < j, i, j \in I_S$.
3.2 Given parameter $t \in (0, 1]$ to calculate $d_c$, $d_c = d_{f(Mt)}$, (12)
where $f(Mt)$ denotes the integer after rounding off $Mt$ and $M = \frac{1}{2}N(N-1)$ and $d_1 \leq d_2 \leq \cdots \leq d_M$.
3.3 Calculate $\rho_i \leftarrow \text{Equation (6)}$ or Equation (8).
3.4 Calculate $\delta_i \leftarrow \text{Equation (9)}$.
3.5 Calculate $\gamma_i \leftarrow \text{Equation (11)}$.

return Index of the key frames $m_k$ and key frames $\{f^K_{m_k}\}_{k=1}^{n_c}$.

because they are effective at capturing long-term temporal dependencies. We adopt the LSTM unit in our large-scale video classification framework, which comprises six important parts: block input, input gate, forget gate, cell state, output gate, and block output. Let $\odot$ denote the point-wise multiplication of two vectors, let $\sigma(x) = \frac{1}{1+e^{-x}}$ be the sigmoid non-linear activation function that squashes real-valued inputs to a $[0, 1]$ range, and let $\phi = \frac{e^x - e^{-x}}{e^x + e^{-x}} = 2\sigma(2x) - 1$ be the hyperbolic tangent non-linearity activation function, which can also squash its inputs to the range of $[-1, 1]$.

The LSTM in timestep $t$ given inputs $x_t$ and $c_{t-1}$ is defined as follows:

$$
g_t = \sigma(W_{xc}x_t + W_{hc}h_{t-1} + b_c),
$$

$$
i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i),
$$

$$
f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f),
$$

$$
c_t = i_t \odot g_t + f_t \odot c_{t-1},
$$

$$
o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o),
$$

$$
h_t = o_t \odot \phi(c_t).
$$

In this work, three layers of LSTM are adopted. The proposed framework of large-scale video classification is composed of two phases—training and testing—and it is summarized in Algorithm 2.

In the training stage, we first obtain the key frames of video $V$ via Algorithm 1, then the LSTM model takes over (step 1.1). Next, the LSTM model predicts the video class $P_f$ at each key frame (step 1.2),

$$
P_f = \frac{\exp(W_{hc}h_{t,c} + b_c)}{\sum \exp(W_{hc}h_{t,c'} + b_c)},
$$

and we average these predictions for final classification (step 1.3),

$$
R_V = \frac{1}{n_c} \sum_{k=1}^{n_c} P_f.
$$
**ALGORITHM 2**: The large-scale video classification framework

**Require**: $L$ videos for training, corresponds to the labels $L_{\text{label}}$; $T$ testing videos.

**Ensure**: Testing accuracy $T_{\text{acc}}$.

**Step 1** TRAINING STAGE:
1.1 Extract key frames $f$ of $V$ ← Algorithm 1.
1.2 The video class $P_f$ of each key frame ← Equation (19).
1.3 The final classification $R_V$ ← Equation (20).
1.4 Calculate the weights $w$ for different input ← Equations (21) and (22).

**Step 2** TESTING STAGE:
2.1 Extract key frames ← Algorithm 1.
2.2 The $P_f$ of each key frame ← Equation (19).
2.3 The $R_V$ ← Equation (20).
2.4 Computing classification rates of different input.
2.5 Combine the weight $w$ with each rate.

**return** $T_{\text{acc}}$.

Moreover, we propose an approach for the weighted fusion of several input networks (i.e., RGB, RGB difference, optical flow, and warped flow) that automatically estimates the weight of each input. The weights reflect the relevance of each input for the specific video shot [53]. Note that the proposed weight fusion method is different from the method proposed in the work of Tang et al. [53]. The method in an earlier work by Tang et al. [53] tries to fusion different patch of the same video, whereas the proposed method in this article tries to combine different modalities of input features. First, we obtain $t$ classification rates $r = \{r_1, r_2, \ldots, r_t\}$ for different input networks. We assume that the higher the rate is, the better the feature representation is [53], then the weights can be calculated as follows:

$$t_0 = \frac{r - \min(r)}{(100 - \min(r))/10}. \quad (21)$$

The weight of the lowest rate is set to 1, and the other weights are set proportional to 1 according to the ratios to the lowest ratio:

$$t_1 = \text{round}(t_0),$$
$$t_2 = \frac{t_0 \times (\max(t_1) - 1)}{\max(t_0)} + 1, \quad (22)$$
$$w = \text{round}(t_2),$$

where $w$ is the weight vector of different inputs (step 1.4). During the testing stage, key frames of the testing video are selected in the same way as the training stage (step 2.1). Then we calculate the $P_f$ and $R_V$ in the same way as the training stage as well (step 2.2 and 2.3). After that, we compute the classification rates of different inputs and combine the rates with the weights $w$ (step 2.4 and 2.5). At the end of the framework, the testing accuracy $T_{\text{acc}}$ of the whole dataset is returned.

**5 EXPERIMENTS**

To evaluate the effectiveness of the proposed method, we conduct experiments with two popular public datasets. More comparisons are shown in Table 1.
Table 1. Key Characteristics of the Datasets Used in the Experiments

| Dataset   | Resolution | Classes | Training | Testing | Total  |
|-----------|------------|---------|----------|---------|--------|
| HMDB51    | 320 × 240  | 51      | 5,263    | 1,530   | 6,766  |
| UCF101    | 320 × 240  | 101     | 9,586    | 3,734   | 13,320 |

5.1 Datasets
The HMDB51 dataset\(^1\) contains a total of 6,766 video clips distributed in a large set of 51 action categories collected from various sources, mostly from movies, public datasets, and YouTube. Each category contains a minimum of 101 video clips. The videos are taken with the resolution of 320 × 240 with 30 fps.

The UCF101 dataset\(^2\) is a widely used dataset for action recognition. It comprises realistic videos collected from YouTube lasting 7 seconds on average. Videos have a spatial resolution of 320 × 240 pixels with 25 fps. UCF101 gives the largest diversity in terms of actions and with the presence of large variations in object appearance, scale and pose, camera motion, viewpoint, cluttered background, and illumination conditions.

5.2 Setups
For a fair comparison, we set \( t = 0.2 \) as in the work of Rodriguez and Laio \(^{44}\). All the experiments are run on Ubuntu with two TITAN Xp GPUs. For one video, the selection process is quite fast with the help of GPUs. We follow existing methods (e.g., \(^3\)) and train our model by using cross-entropy loss.

5.3 Experimental Results
We present extensive experimental results to demonstrate the necessity and efficiency of the proposed method and framework on the large-scale video classification task.

Comparison of Other Extraction Methods. We first compare different key frame extraction methods on the large-scale video classification task. The column “Input Type” of Table 2 shows the performance comparison between our method and the uniform sampling method with different rates (8, 16, 32 frames), \( k \)-means \(^{34}\), S-RNN \(^{47}\), Joint Unsupervised Learning \(^{72}\), single frame \(^9\), and LRCN \(^9\) on the UCF101 dataset (split 1). Note that we follow Wang et al. \(^{67}\) and adopt four different types of information (i.e., RGB image, RGB difference, optic flow, and warped flow) as inputs. The results show that our method CNN-TSDPC-LSTM consistently outperforms all the baselines with significant improvements, which validates that there is significant informative motion and spatial information available around key frames. Finally, to better understand and evaluate the proposed key frame extraction method, we show one video example from UCF101 in Figure 3.

Combination with Weight Strategy. We then add the proposed weight strategy to our framework to test whether they benefit for this task. As shown in the column “Weighted” of Table 2, the proposed weight method outperforms the baseline method LRCN \(^9\) when two inputs (RGB Image

\(^1\)http://serre-lab.clps.brown.edu/resource/hmdb-a-large-human-motion-database/.
\(^2\)http://crcv.ucf.edu/data/UCF101.php.
### Table 2. Exploration of Different Input for the Proposed Framework on the UCF101 Dataset

| Model                                         | Input Type          | Weighted | Time (s) |
|-----------------------------------------------|---------------------|----------|----------|
| Uniform Sampling (8 frames)                   | RGB Image: 54.36, 52.96 | 1/2, 1/2 | 1.63     |
| Uniform Sampling (16 frames)                  | RGB Difference: 58.67 | 1/3, 2/3 |          |
| Uniform Sampling (32 frames)                  | Optic Flow: 56.39    | 2.32     |
| K-means [34]                                 | Warped Flow: 65.35   | 3.53     |
| S-RNNN [47]                                  |                      |          |
| Joint Unsupervised Learning [72]             |                      |          |
| Single Frame [9]                             | Uniform Sampling (8 frames): 54.36 | 1/2, 1/2 | 1.63     |
| Single Frame (ave.) [9]                      |                      |          |
| LRCN-fc [9]                                  |                      |          |
| LRCN-fc [9]                                  |                      |          |
| CNN-TSDPC-LSTM (Ours)                        |                     |          |
| CNN-TSDPC-LSTM (Ours)                        |                     |          |
| CNN-TSDPC-LSTM (Ours)                        |                     |          |
| CNN-TSDPC-LSTM (Ours)                        |                     |          |

Fig. 3. A video from the UCF101 dataset, which contains 80 frames. The key frames extracted by the proposed method are in red boxes.

and Optic Flow) are employed. Note that "1/2, 1/2" and "1/3, 2/3" are proposed in the work of Donahue et al. [9] and are two different weight settings that represent the fixed weights of two different inputs. Besides, when we adopt three inputs (i.e., RGB image, optic flow, and warped flow), we achieve the best performance compared with other baselines. However, we observe that introducing RGB difference degrades the performance, which is also observed in TSN [67].

**Comparison with State-of-the-Art Methods.** We assemble these three inputs and all the techniques described as our final video classification method, and we test it on the HMDB51 and UCF101 datasets. For the HMDB51 and UCF101 datasets, we compare the proposed framework (i.e., CNN-TSDPC-LSTM-Fusion) with the state-of-the-art traditional approaches (e.g., iDT + HSV [41] and MoFAP [66]). We also compare the proposed method with deep learning representation methods such as LRCN [9], KVMF [75], and TSN [67]. The results are summarized in Table 3, and our
### Table 3. Mean Classification Performance of the State-of-the-Art Approaches on the HMDB51 and UCF101 Datasets

| Method                                      | HMDB51  | UCF101 |
|---------------------------------------------|---------|--------|
| iDT + StackFV [42]                          | 66.8%   | -      |
| DT + MVSV [2]                               | 55.9%   | 83.5%  |
| iDT + FV [63]                               | 57.2%   | 85.9%  |
| iDT + SFV + STP [62]                        | 60.1%   | 86.0%  |
| iDT + HSV [41]                              | 61.1%   | 87.9%  |
| MoFAP [66]                                  | 61.7%   | 88.3%  |
| iDT + MIPS [29]                             | 65.1%   | 89.1%  |
| VideoDarwin [16]                            | 63.7%   | -      |
| MPR [38]                                    | 65.5%   | -      |
| VGAN [60]                                   | -       | 52.1%  |
| Deep Networks, Sports 1M pre-training [25]  | -       | 65.2%  |
| C3D (1 net), Sports 1M pre-training [55]    | -       | 82.3%  |
| LRCN (fc6) [9]                              | -       | 82.92% |
| C3D (3 nets), Sports 1M pre-training [55]   | -       | 85.2%  |
| Res3D [56]                                  | 54.9%   | 85.8%  |
| Two Stream [48]                             | 59.4%   | 88.0%  |
| FSTCN (SCI fusion) [51]                     | 59.1%   | 88.1%  |
| Two Stream + LSTM[73]                       | -       | 88.6%  |
| Dynamic Image Networks + IDT [1]            | 65.2%   | 89.1%  |
| AdaScan+Two Stream [24]                     | 54.9%   | 89.4%  |
| C3D (3 nets) + IDT, Sports 1M pre-training [55] | -       | 90.1%  |
| TDD + FV [65]                               | 63.2%   | 90.3%  |
| AdaScan+iDT+last fusion [24]                | 61.0%   | 91.3%  |
| TDD + iDT [65]                              | 65.9%   | 91.5%  |
| LTC [58]                                    | 64.8%   | 91.7%  |
| RGB-3D, miniKinetics pre-training [3]       | 66.4%   | 91.8%  |
| Actions Trans [68]                          | 62.0%   | 92.0%  |
| Convolutional Two Stream [15]               | 65.4%   | 92.5%  |
| Hybrid-iDT [8]                              | 70.4%   | 92.5%  |
| KVMF [75]                                   | 63.3%   | 93.1%  |
| AdaScan+iDT+C3D+last fusion [24]            | 66.9%   | 93.2%  |
| TSN (2 modalities, BN-Inception) [67]       | 68.5%   | 94.0%  |
| Spatiotemporal Multiplier Network [13]      | 68.9%   | 94.2%  |
| TSN (3 modalities, BN-Inception) [67]       | 69.4%   | 94.2%  |
| Cool-TSN [43]                               | 69.5%   | 94.2%  |
| ST-VLMPF(DF) [10]                           | 73.1%   | 94.3%  |
| Spatiotemporal Pyramid Network [69]         | 68.9%   | 94.6%  |
| Spatiotemporal ResNets + IDT [12]           | 70.3%   | 94.6%  |
| Flow-3D, miniKinetics pre-training [3]      | 72.4%   | 94.7%  |
| Attention Fusion [33]                       | -       | 94.8%  |
| Spatiotemporal Multiplier Network + iDT [13]| 72.2%   | 94.9%  |
| RGB-3D, Kinetics pre-training [3]           | 74.8%   | 95.6%  |
| Optical Flow guided Feature [52]            | 74.2%   | 96.0%  |
| Flow-3D, Kinetics pre-training [3]          | 77.1%   | 96.7%  |
| Two-Stream 3D, miniKinetics pre-training [3]| 76.3%   | 96.9%  |
| 3D RGB + DMC-Net (3D) [46]                  | 77.8%   | 96.5%  |
| Two-Stream 3D, Kinetics pre-training [3]    | 80.7%   | 98.0%  |
| CNN-TSDPC-LSTM (Ours)                       | 75.52%  | 95.86% |
| CNN-TSDPC-LSTM* (Ours)                      | 81.44%  | 98.45% |

Methods on the horizontal line are traditional video classification methods, and the approaches under the horizontal line are deep learning methods. The asterisk (*) means the proposed method uses the two-stream 3D as the backbone and uses Kinetics for pre-training.
Table 4. Compression Ratio (%) on the HMDB51 and UCF101 Datasets

| Dataset   | #Total Frames | #Avg. Frames | #Total Key Frames | CR (%)  |
|-----------|---------------|--------------|-------------------|---------|
| HMDB51    | 634,552       | 93.8         | 108,256           | 82.94   |
| UCF101    | 2,485,519     | 186.6        | 213,120           | 91.43   |

Table 5. Time Comparison of Different Models on the UCF101 Dataset

| Model                                           | Time  |
|-------------------------------------------------|-------|
| CNN-TSDPC-LSTM (Ours)                           | 1.00  |
| Optical Flow guided Feature [52]                 | 1.13x |
| Two-Stream I3D [3]                               | 1.35x |

Table 6. Computational Complexity of Different Networks

| Network Architecture    | GFLOPs |
|-------------------------|--------|
| C3D [55]                | 38.5   |
| Res3D-18 [56]           | 19.3   |
| ResNet-152 [22]         | 11.3   |
| ResNet-18 [22]          | 1.78   |
| PWC-Net [50]            | 36.15  |
| CNN-TSDPC-LSTM (Ours)   | 1.45   |

The method achieves competitive classification accuracy compared with these methods. Note that the classification performance of the proposed framework is worse than several of the baselines, such as Two-Stream I3D [3]. Carreira and Zisserman [3] achieve the best performance after pre-training on extra data (i.e., Kinetics). However, the proposed method is more efficient and has a higher compression ratio, which is defined as the relative amount of savings provided by the summary representation. The definition of the compression ratio is $\text{CR}(V) = 1 - \frac{n_c}{N}$, where $n_c$ and $N$ are the number of key frames and the number of frames in the original video $V$, respectively. The results are shown in Table 4. Generally, a high compression ratio means a compact video summary and also means less video processing time. Finally, to further prove the effectiveness of our proposed method, we use the two-stream I3D as our backbone, and use Kinetics for pre-training. As can be seen from Table 3, this model achieves state-of-the-art results on both datasets.

Efficiency Comparison. We also measure the average classification time on the UCF101 dataset. The column “Time” of Table 2 lists the average time of different sampling approaches. The results show that we can obtain high classification accuracy while keeping the complexity low compared with the state-of-the-art key frame extraction methods such as S-RNN [47] and Joint Unsupervised Learning [72]. Consequently, the proposed method is time-efficient both online and offline, and can be readily adopted in real-world applications. Moreover, as we expected, we observe that more inputs lead to longer training and testing times. We also provide time comparison with the state-of-the-art methods on the UCF101 dataset. Results are shown in Table 5, and we can see that the proposed method is faster than Optical Flow guided Feature [52] and Two-Stream I3D [3]. In Table 6, we also provide the GFLOPs results compared with other network architectures for video analysis, including ResNet-18 [22], ResNet-152 [22], C3D [55], and Res3D [56]. We observe that the complexity of the proposed method is smaller compared to that of other architectures, which makes it run much faster.
6 CONCLUSION

We presented an unsupervised key frame extraction method (i.e., TSDPC) for video key frame extraction, which can greatly reduce the redundant information of the original video while preserving the temporal information through selecting the number of key frames automatically. In addition, a framework CNN-TSDPC-LSTM aimed at large-scale video classification that consists of CNN, TSDPC, and LSTM was proposed. Moreover, a fusion strategy of different input networks was presented to boost the accuracy of video classification. Experimental results on a variety of public datasets demonstrated that (i) our framework is capable of summarizing video efficiently regardless of the visual content of videos, and (ii) our framework has high efficiency while the mean classification accuracy is comparable with other state-of-the-art methods. Note that the proposed method is generic, which thus can be beneficial to other video processing tasks.

REFERENCES

[1] Hakan Bilen, Basura Fernando, Efstratios Gavves, Andrea Vedaldi, and Stephen Gould. 2016. Dynamic image networks for action recognition. In Proceedings of CVPR.
[2] Zhuowei Cai, Limin Wang, Xiaojiang Peng, and Yu Qiao. 2014. Multi-view super vector for action recognition. In Proceedings of CVPR.
[3] Joao Carreira and Andrew Zisserman. 2017. Quo Vadis, action recognition? A new model and the kinetics dataset. In Proceedings of CVPR.
[4] Zuzana Cernekova, Ioannis Pitas, and Christophoros Nikou. 2006. Information theory-based shot cut/fade detection and video summarization. IEEE Transactions on Circuits and Systems for Video Technology 16, 1 (2006), 82–91.
[5] Vasilieos Choutas, Philippe Weinzaepfel, Jérôme Revaud, and Cordelia Schmid. 2018. Potion: Pose motion representation for action recognition. In Proceedings of CVPR.
[6] Yang Cong, Junsong Yuan, and Jiebo Luo. 2012. Towards scalable summarization of consumer videos via sparse dictionary selection. IEEE Transactions on Multimedia 14, 1 (2012), 66–75.
[7] Sandra Eliza Fontes De Avila, Ana Paula Brandão Lopes, Antonia da Luz, and Arnaldo de Albuquerque Araújo. 2011. VSUMM: A mechanism designed to produce static video summaries and a novel evaluation method. Pattern Recognition Letters 32, 1 (2011), 56–68.
[8] César Roberto de Souza, Adrien Gaidon, Eleonora Vig, and Antonio Manuel López. 2016. Sympathy for the details: Dense trajectories and hybrid classification architectures for action recognition. In Proceedings of ECCV.
[9] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. 2015. Long-term recurrent convolutional networks for visual recognition and description. In Proceedings of CVPR.
[10] Ionut Cosmin Duta, Bogdan Ionescu, Kiyoharu Aizawa, and Nicu Sebe. 2017. Spatio-temporal vector of locally max pooled features for action recognition in videos. In Proceedings of CVPR.
[11] Naveed Ejaz, Tayyab Bin Tariq, and Sung Wook Baik. 2012. Adaptive key frame extraction for video summarization using an aggregation mechanism. Journal of Visual Communication and Image Representation 23, 7 (2012), 1031–1040.
[12] Christoph Feichtenhofer, Axel Pinz, and Richard Wildes. 2016. Spatiotemporal residual networks for video action recognition. In Proceedings of NIPS.
[13] Christoph Feichtenhofer, Axel Pinz, and Richard P. Wildes. 2017. Spatiotemporal multiplier networks for video action recognition. In Proceedings of CVPR.
[14] Christoph Feichtenhofer, Axel Pinz, Richard P. Wildes, and Andrew Zisserman. 2018. What have we learned from deep representations for action recognition? In Proceedings of CVPR.
[15] Christoph Feichtenhofer, Axel Pinz, and Andrew Zisserman. 2016. Convolutional two-stream network fusion for video action recognition. In Proceedings of CVPR.
[16] Basura Fernando, Efstratios Gavves, Jose M. Oramas, Amir Ghodrati, and Tinne Tuytelaars. 2015. Modeling video evolution for action recognition. In Proceedings of CVPR.
[17] Adrien Gaidon, Zaid Harchaoui, and Cordelia Schmid. 2013. Temporal localization of actions with actoms. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 11 (2013), 2782–2795.
[18] Ruohan Gao, Bo Xiong, and Kristen Grauman. 2018. Im2Flow: Motion hallucination from static images for action recognition. In Proceedings of CVPR.
[19] Hana Gharbi, Sahbi Bahroun, Mohamed Massaoudi, and Ezzeddine Zagrouba. 2017. Key frames extraction using graph modularity clustering for efficient video summarization. In Proceedings of ICASSP.
Deep Unsupervised Key Frame Extraction for Efficient Video Classification

119:15

[20] Genliang Guan, Zhiyong Wang, Shiyang Lu, Jeremiah Da Deng, and David Dagan Feng. 2013. Keypoint-based keyframe selection. IEEE Transactions on Circuits and Systems for Video Technology 23, 4 (2013), 729–734.  
[21] Kensho Hara, Hirokatsu Kataoka, and Yutaka Satoh. 2018. Can spatiotemporal 3D CNNs retrace the history of 2D CNNs and ImageNet? In Proceedings of CVPR.  
[22] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In Proceedings of CVPR.  
[23] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2013. 3D convolutional neural networks for human action recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 35, 1 (2013), 221–231.  
[24] Amlan Kar, Nishant Rai, Karan Sikka, and Gaurav Sharma. 2017. AdaScan: Adaptive scan pooling in deep convolutional neural networks for human action recognition in videos. In Proceedings of CVPR.  
[25] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In Proceedings of CVPR.  
[26] Sanjay K. Kuanar, Rameswar Panda, and Ananda S. Chowdhury. 2013. Video key frame extraction through dynamic Delaunay clustering with a structural constraint. Journal of Visual Communication and Image Representation 24, 7 (2013), 1212–1227.  
[27] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. 2011. HMDB: A large video database for human motion recognition. In Proceedings of ICCV.  
[28] Sourabh Kulhare, Shagan Sah, Suhas Pillai, and Raymond Ptucha. 2016. Key frame extraction for salient activity recognition. In Proceedings of IJCV.  
[29] Zhengzhong Lan, Ming Lin, Xuanchong Li, Alex G. Hauptmann, and Bhiksha Raj. 2015. Beyond Gaussian pyramid: Multi-skip feature stacking for action recognition. In Proceedings of CVPR.  
[30] Ivan Laptev, Marcin Marszalek, Cordelia Schmid, and Benjamin Rozenfeld. 2008. Learning realistic human actions from movies. In Proceedings of CVPR.  
[31] Hong Liu, Hao Tang, Wei Xiao, ZiYi Guo, Lu Tian, and Yuan Gao. 2016. Sequential bag-of-words model for human action classification. CAAI Transactions on Intelligence Technology 1, 2 (2016), 125–136.  
[32] Hong Liu, Lu Tian, Mengyuan Liu, and Hao Tang. 2015. SDM-BSM: A fusing depth scheme for human action recognition. In Proceedings of ICIP.  
[33] Xiang Long, Chuang Gan, Gerard de Melo, Xiao Liu, Yandong Li, Fu Li, and Shilei Wen. 2018. Multimodal keyless attention fusion for video classification. In Proceedings of AAAI.  
[34] James MacQueen. 1967. Some methods for classification and analysis of multivariate observations. In Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Lucien M. Le Cam and Jerzy Neyman (Eds.). University of California Press, 281–297.  
[35] Shaohui Mei, Genliang Guan, Zhiyong Wang, Mingyi He, Xian-Sheng Hua, and David Dagan Feng. 2014. L2, 0 constrained sparse dictionary selection for video summarization. In Proceedings of ICME.  
[36] Shaohui Mei, Genliang Guan, Zhiyong Wang, Shuai Wan, Mingyi He, and David Dagan Feng. 2015. Video summarization via minimum sparse reconstruction. Pattern Recognition 48, 2 (2015), 522–533.  
[37] Jingjing Meng, Hongxing Wang, Junsong Yuan, and Yap-Peng Tan. 2016. From keyframes to key objects: Video summarization by representative object proposal selection. In Proceedings of CVPR.  
[38] Bingbing Ni, Pierre Moulin, Xiaokang Yang, and Shuicheng Yan. 2015. Motion part regularization: Improving action recognition via trajectory selection. In Proceedings of CVPR.  
[39] Costas Panagiotakis, Anastasios Doulaamis, and Georgios Tziritas. 2009. Equivalent key frames selection based on iso-content principles. IEEE Transactions on Circuits and Systems for Video Technology 19, 3 (2009), 447–451.  
[40] Rameswar Panda, Sanjay K. Kuanar, and Ananda S. Chowdhury. 2014. Scalable video summarization using skeleton graph and random walk. In Proceedings of ICPR.  
[41] Xiaojiang Peng, Limin Wang, Xingxing Wang, and Yu Qiao. 2016. Bag of visual words and fusion methods for action recognition: Comprehensive study and good practice. Computer Vision and Image Understanding 150 (2016), 109–125.  
[42] Xiaojiang Peng, Changqing Zou, Yu Qiao, and Qiang Peng. 2014. Action recognition with stacked Fisher vectors. In Proceedings of ECCV. 581–595.  
[43] Cesar Roberto de Souza, Adrien Gaidon, Yohann Cabon, and Antonio Manuel Lopez. 2017. Procedural generation of videos to train deep action recognition networks. In Proceedings of CVPR.  
[44] Alex Rodriguez and Alessandro Laio. 2014. Clustering by fast search and find of density peaks. Science 344, 6191 (2014), 1492–1496.  
[45] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Kruse, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, et al. 2015. ImageNet Large Scale Visual Recognition Challenge. International Journal of Computer Vision 115, 3 (2015), 211–252.  
[46] Zheng Shou, Xudong Lin, Yannis Kalantidis, Laura Sevilla-Lara, Marcus Rohrbach, Shih-Fu Chang, and Zhicheng Yan. 2019. DMC-Net: Generating discriminative motion cues for fast compressed video action recognition. In Proceedings of CVPR.
[76] Yi Zhu, Yang Long, Yu Guan, Shawn Newsam, and Ling Shao. 2018. Towards universal representation for unseen action recognition. In Proceedings of CVPR.

[77] Yueting Zhuang, Yong Rui, Thomas S. Huang, and Sharad Mehrotra. 1998. Adaptive key frame extraction using unsupervised clustering. In Proceedings of ICIP.

Received 18 April 2022; revised 17 September 2022; accepted 6 November 2022