Large-Scale Video Classification with Elastic Streaming Sequential Data Processing System

Yao Peng*
Shanghai Advanced Research Institute, CAS & Qiniu AI Lab
pengyao@qiniu.com

Hao Ye*
Shanghai Advanced Research Institute, CAS, China
yeh@sari.ac.cn

Yining Lin
Qiniu AI Lab
Shanghai, China
linyining@qiniu.com

Yixin Bao
Qiniu AI Lab
Shanghai, China
baoyixin@qiniu.com

Zhijian Zhao
Qiniu AI Lab
Shanghai, China
zhaozhijian@qiniu.com

Haonan Qiu
East China Normal University
Shanghai, China
51164500051@stu.ecnu.edu.cn

Yao Lu
University of Washington
Seattle, USA
luyao@cs.washington.edu

Li Wang
Shanghai Advanced Research Institute, CAS, China
wangli@sari.ac.cn

Yingbin Zheng†
Shanghai Advanced Research Institute, CAS, China
zhengyb@sari.ac.cn

ABSTRACT

Videos are dominant on the Internet. Current systems to process large-scale videos are suboptimal due to the following reasons: (1) machine learning modules such as feature extractors and classifiers generate huge intermediate data and place heavy burden to the storage and network, and (2) task scheduling is explicit; manually configuring the machine learning modules on the cluster is tedious and inefficient. In this work, we propose Elastic Streaming Sequential data Processing system (ESSP) that supports automatic task scheduling; multiple machine learning components are automatically parallelized. Further, our system prevents extensive disc I/O by applying the in-memory dataflow scheme. Evaluation on real-world video classification datasets shows many-fold improvements.

KEYWORDS

Large-scale video classification, Elastic Streaming Sequential Data Processing System, two-stream network.

*These authors contributed equally to this work.
†Corresponding author.

LSVC'17, October 27, 2017, Mountain View, CA, USA
© 2017 Copyright held by the owner/author(s). Publication rights licensed to Association for Computing Machinery.
ACM ISBN 978-1-4503-5537-7/17/10. . . $15.00
https://doi.org/10.1145/3134263.3134264

1 INTRODUCTION

With the prevalent use of mobile devices and the rapid development of transmission and storage systems, a massive number of multimedia contents are uploaded onto the Internet. A recent report shows that 67% of all Internet traffic are video in 2016, and the percentage will increase to 80% by 2021[1]. As more and more videos are generated, distributed, and made accessible all over the world, efficient ways to analyze and understand the video content are of fundamental importance. In particular, video classification concentrates on automatically assigning video clips with semantic concepts such as actions, events, activities, and emotions based on the visual contents, which is a prerequisite for a broad array of high-level vision tasks, including video retrieval and surveillance [27, 44].

In the past years there has been much attention in the video classification domain. Descriptors like frame-based appearance features [3, 24, 26, 30], spatio-temporal visual features [5, 22, 41, 42], and audio-visual joint representations [1, 15, 53] are designed for video contents. Recently, deep-neural-network-based visual models make breakthrough on several computer vision tasks [11, 20, 37], and video classification with deep networks also achieves significant improvement on the performance. Deep networks such as convolutional neural network (CNN) [18, 36, 55] and Long Short Term Memory (LSTM) [6, 50, 51] are applied to the video domain and achieve performance gain over traditional approaches.

Current deep network paradigms for classification involve learning the models on a large dataset of labeled data. Researches on video classification also stimulate largely by the video datasets and a few benchmarks are released [2, 12, 13, 16–18, 21, 23, 25, 33, 38, 54]. We list some popular manually labeled datasets in Table 1. The quantitative parameters of

[1]CISCO VNI Global Fixed and Mobile Internet Traffic Forecasts 2017. https://www.cisco.com/c/en/us/solutions/service-provider/vni-network-traffic-forecast/infographic.html

Table 1: Statistics of popular video classification benchmarks (sorted by the released year). The total durations are measured in hours.

Dataset	Duration	#Video	#Class	Year
KTH [33]	3	600	6	2004
Hollywood [23]	2	430	8	2008
HMDB51 [21]	6	6,766	51	2011
CCV [17]	210	9,317	20	2011
UCF-101 [38]	27	13,320	101	2012
ActivityNet [12]	849	27,901	203	2015
FCVID [16]	4232	91,223	239	2015

these datasets, *i.e.*, video clip numbers, category number, and total durations, increase over years. Large scale video database, incorporating with the well-designed deep learning based models, effectively promote current video classification systems. Meanwhile, the learning frameworks also bring forward new challenges. In general, there are mainly two steps included in a video classification framework, *i.e.*, generating multimodal features and multimodal clue fusion. The intermediate video data and features not only consume large storage space but also reduce the efficiency of whole system, thus this step is usually considered as the bottleneck, especially when the amount of video are large.

In this paper, we demonstrate an end-to-end classification system for large-scale video classification, and the Elastic Streaming Sequential data Processing system, or ESSP in short, is proposed. Particularly, a video frame container is used upon the video storage as the initial stage of the data loader, and the frame can be loaded to the key-value map. The model trainer, fusion and feature extractor are executed according to the available system resource. All the data are transmitted from one block to another to assemble a dataflow in which none of the intermediate representation is stored on the disk.

In summary, the main contributions of this paper include:

- Unlike previous video classification systems, ESSP is able to decode videos and extract basic features simultaneously, so that less storage is required during video analysis.
- ESSP incorporates multi-node caching and load balance, which ensures the computational efficiency for video classification, comparing with traditional deep-network-based systems.
- We also propose novel strategies for prefetching the data for training deep neural models.
- We evaluate our system on two real-world video classification datasets, *i.e.*, UCF-101 [38] and FCVID [16]. Results indicate improvement over the traditional pipeline.

The remainder of this paper is organized as follows. Section 2 reviews the related works on video classification. Section 3 describes our proposed ESSP system. In Section 4, we demonstrate the quantitative study and preliminary results on large-scale video classification challenge. And finally, we conclude our work in Section 5.

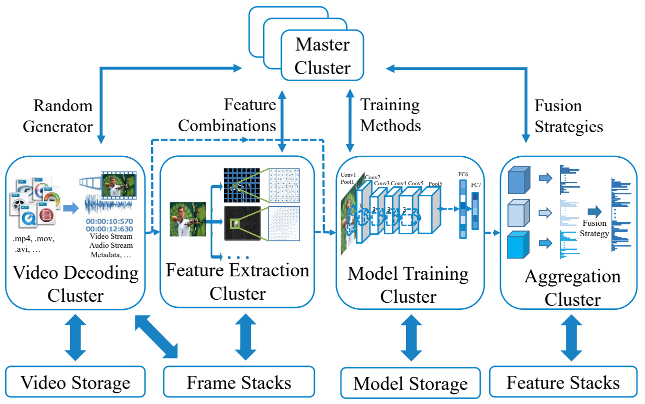

Figure 1: The architecture of the ESSP system.

2 RELATED WORKS

Video classification is among the top interests of computer vision, which requires discriminative feature descriptors to represent the videos. In recent decades, video feature engineering has witnessed the improvement of the video analysis.

Up until recently, the handcrafted appearance features and motion features are still the dominate video representations. For instance, Laptev introduced the spatio-temporal interest points [22], Dollár *et al.* extended the Gabor filters into 3D including time dimension [5]. Wang *et al.* proposed dense trajectory (DT) and improved dense trajectory (iDT) which extracted the trajectory tracks from the densely sampled interesting points [42, 43]. To describe all these interesting points and trajectories, bag-of-features (BOF) methods are adopted due to their simplicity [24]. Several alternatives are proposed including histograms of flow orientations (HOF) [23], histograms of 3D gradients (HO3DG) [19, 34], and motion boundary histogram (MBH) [4]. DT and improved iDT features together with MBH and HOF descriptor achieve the state-of-the-art performance.

With the recent advance of image classification [11, 20, 28, 37, 52], video spatial features encoded from neural networks are used for video classification [18]. To learn the spatio-temporal features, Du *et al.* [40] and Ji *et al.* [14] developed 3D convolution neural network to capture the motion information from the adjacent frames. Further spatio-temporal feature learning mimics the two-pathway model revealed from neuroscience's study on brain. Simonyan proposed the two-stream neural network that learns the temporal features from the stacked optical flows and fuses spatial and temporal features into one feature descriptor for video classification [36, 55]. Inspired by these work, Wang *et al.* stacked the trajectory-pooled features to learn motion information and also achieved the state-of-the-art [45]. In a more subtle extension of two-stream framework, Wang [48] and Peng [31] proposed to incorporate human and object proposal features which leverage the semantic cues to learn spatio-temporal features.

Fusing the spatial and temporal features has also proved to improve the classification performance significantly. Wu *et al.* proposed to fuse multi-stream features by using the hybrid neural networks and achieved the state-of-the-art [50, 51]. Feichtenhofer *et al.* investigated various ways of fusing ConvNet towers both spatially and temporally for the purpose of best taking advantage of spatio-temporal information [9]. Wang *et al.* introduced the spatio-temporal compact bilinear operator in fusing the two types of features [47]. Feichtenhofer *et al.* invented the motion gating method to combine the appearance and motion pathways of a two-stream architecture which can be trained end-to-end [8]. Duta *et al.* introduced a spatio-temporal vector of locally max features, which is a fast vector-based encoding for local deep features [7]. Zhang *et al.* used the enhanced motion vector for fast video classification [56]. Girdhar *et al.* jointly aggregated the spatio-temporal features into a set of action primitives over the appearance and motion stream of a video, named Action-VLAD layer, which can be trained end-to-end [10]. Besides, Recurrent Neural Networks are popular for aggregating the entire video features due to their attention mechanism. Ng *et al.* explored various convolutional temporal feature pooling architectures to combine image information across the entire video [29]. Donahue *et al.* described several recurrent convolutional architectures suitable for large-scale video understanding [6].

3 THE ESSP SYSTEM

Several challenges exist to apply the two-stream framework [9] and the Temporal Segment Network [46] for large-scale video classification. The first is data storage on hard disk. For instance, the dataset of Large-Scale Video Classification Challenge [49] is 3TB for the videos. During the training process for the spatial models, the video frames are extracted as a prerequisite step and the data volumes are more than 10 times for original videos. Besides, the intermediate video representations, such as the MBH and HOF features, are also stored on the hard drive. The videos, frames, as well as features, consume more than 50TB spaces in total. On the other hand, if such data is stored on the distributed cluster, the networking encounters another challenge. Suppose the video frames are resized to 256×256 pixels, the volume for the image is 192KB, while that for the corresponding optical flow with optical flow stack width L is $128L$KB. The data to transmit have to be compressed in a distributed training system with $L = 10$ and batch size 256.

To handle these challenges, we design the Elastic Streaming Sequential data Processing system (ESSP) in this work. Figure 1 is the whole architecture of the ESSP system and the details will be introduced in the following sections.

3.1 Master Cluster and Infrastructures

The goals of the master cluster are to dispatch tasks, to supervise and schedule the independent modules, and to synchronize the tasks. The master cluster consists of several components. The first component is based on the in-memory data structure store *Redis*[2] for caching the extracted video frames and features. To dispatch jobs and to coordinate the production-consuming rates among the workers, a message queue is implemented based on the distributed streaming platform *Kafka*[3] due to its distributed capacity and high throughput. Besides, as the volume of the videos and the trained deep models are usually large, we employ the *Qiniu Cloud*[4] distributed file system to host these data. The general dataflow controlled by the master cluster includes the following steps:

- Video decoding and feature extraction workers obtain job messages from the Kafka queue, fetch videos from the video storage, compute the specific task, and then push the results (video frames and features) back into Redis.
- Model training workers fetch the frames and features as inputs – after obtaining the job message from Kafka, the dataloader component fetches data from Redis and then feed them into the training process. The intermediate and final models are sent to the storage. The master cluster directly controls the training workers by sending model training strategies.
- The job messages from Kafka and information from the master cluster (like the fusion strategies) are therefore sent to the feature aggregation workers.
- The control signals such as start, stop, and pause are also made by master cluster and transmitted through Kafka.

The ESSP system supports automatic task scheduling; multiple machine learning components are automatically parallelized. The system is built upon microservices. For both master cluster and worker clusters, *Kubernetes*[5] is used as docker orchestration system for container scheduling, providing expansive methods for service deploy, maintenance and extension.

3.2 Video Decoding Cluster

For large-scale video classification, video decoding is essential. The decoding workers respond to decoding requests, get the required video and corresponding information (*e.g.*, offset and time window), and then deliver the generated frames to the following task.

As mentioned at the beginning of this section, traditional approaches put the video frames on disks and pick them up when needed. The disadvantage is obvious; the learning system needs order-of-magnitudes storage space for the frames than the video storage. Another approach is to only have videos on the disk and decode frames on-the-fly; this solution introduces great pressure on the computing infrastructures. We aim to trade-off between the computation and the storage upon the master cluster. The general video decoding work flow includes the following steps: initially, the system

[2]https://redis.io/
[3]http://kafka.apache.org/
[4]https://www.qiniu.com/
[5]https://kubernetes.io/

checks whether the decoding workers are ready and sends notifications; after receiving the notification in random sequence, videos are decoded and frames are serialized; finally, the frames are pushed into the key-value store or sent to the subsequent cluster directly.

Video decoder cluster uses a random sequence generator to produce random video segment seeds. The video learning system shuffles the data in each epoch. We set the sequence generator once for an epoch, producing a video segment seed before each epoch begins. Following class balance in training [35], videos are fed with three random selection, *i.e.*, a randomly chosen class ID, a random video ID from the chosen class, and a random frame offset from the chosen video. Therefore, the random sequence contains three random elements: class ID, video ID in current class, frame begin offset in the video. As frame stack is always needed in the following phase, it comes with another constant parameter and frame count to decode. The reason we made an independent module for decoding is that, the decoding and training rates are not always matching; one decoding module may deal with multiple training and is followed by several feature extractors.

3.3 Feature Extraction Cluster

Feature extraction worker are responsible for extracting the features during training. As we believe there will be novel features in the future, we design the feature extraction cluster as an open-registration module, so that a new extractor can be attached and registered from the master cluster. The default feature is the single video frame, which is already extracted by the video decoder. Here we take single optical flow as an example; we pack the flow extractor into a docker image, and the master cluster only maintains a feature-image table. The core work flow of a feature extractor image is as follows: a) it obtains the decode sequence from the master node; b) it then fetches the corresponding frames from Redis; c) features are extracted and serialized next; d) finally the features are pushed into Redis cluster.

The docker-based micro-service framework simplifies the resource scheduling. To add a new extractor, we build a new feature extractor image and add a record in the table. To expand specified extractors, we change the corresponding container number, and the Load-Balance will take charge of rest. This scheme reduces our work for attaching features. Currently, the system includes the following feature extraction images:

- Optical flow: dense optical flow is the most popular temporal feature in the literature. The standard dense optical flow is sampled as [9]. We leverage advantages of online sampling and online decoding and perform experiments on different optical flow stack width. Also, we use test optical flow variants as [56], such as warped flow, bi-directional flow, and dense trajectory.
- Frame stack and frame difference stack are also brought into feature extraction because of their simplicity. Our system allows different extractors and parameters; the only thing needed is attaching new services.

3.4 Model Training Cluster

The model training cluster is to manage multiple training procedures and to prevent a single node from failure. The training module obtains frames or features from a kv-store where pre-computing modules are stored. The training module usually contains several CNN training, and each one corresponds to different features. As different training has different rates of convergence, computing resource between them should be allocated dynamically. For simplicity, we make this allocation on GPU-level, assigning different training with different GPU amount. The parameters are chosen by experience aforehand and are adjusted according to the validation loss. We evaluate the necessity for parameter turning after each training iteration, which most avoids disturbs to training.

The key factor of the distributed trainer is the parameter synchronization speed, so a distributed system is necessary. We also use micro-service framework mentioned in the previous section to carry the training systems. So the Trainer-Registration Table is also needed. There are several training instances accommodated in a heterogeneous architecture. Training instances can be single-CPU or GPU container, multi-GPU in single node, or multi-GPU in distributed configuration. When instantiated, containers should be assigned with specified computing resource. However, distributed training cannot be assigned directly on docker level; a unified interface is built to accommodate the distributed system management. The mainly used training framework in video learning is the two-stream [9] framework. Spatial and temporal cues are trained separately and the final aggregation is carried out in next module, which we will discuss later. The spatial cues are used on single frames. We implemented it in PyTorch Framework, following the ImageNet classification scheme and fine-tuning from 22k pretrained model. Temporal cues are represented by optical flow and frame stacks. The advantage of our distributed trainer is that we can try as many trainers as we want. Each trainer is equipped with a prefetch queue to preload data from the memcache cluster, avoiding the transforming bottleneck. If the bottleneck still exists, we can expend the memcache cluster.

3.5 Feature Aggregation Cluster

Aggregating multiple modality cues improves video classification performance. However, aggregating multi-modality features usually requires the sequence of features, which is different from training the spatial ConvNets. For instance, training LSTM for aggregation requires the video features in sequence. The target of this module is to adjust the IO pattern from fully random to partially sequential. During the aggregation stage, each stream features can be stored in the sequential cache. The IO pattern is dynamically adjusted based on the aggregation algorithm. When the aggregation is focusing on spatial aspect, the IO pattern is more random. If the aggregation is on temporal aspect, the IO pattern is more sequential. Moreover, in order to support distributed training,

Figure 2: Sample video frames for the datasets.

the IO pattern is designed to be virtualized to facilitate the dispatching of the sequential features.

3.6 Prediction

During the prediction, each stream of the video features are extracted according to the time sequence; then the streams are fed to the aggregation model to obtain the video descriptors. At this stage, many models are ensembled by using various kinds of weights. To improved the classification accuracy, action localization is optional for untrimmed videos to detect the key motion chunks.

4 EVALUATIONS

4.1 Dataset and Evaluation Metrics

We adopt the popular two-stream pipeline for large scale video classification. For the training phase, we randomly select batches of frames and stack optical flows from the the feature pool which is automatically maintained by the ESSP feature extracting process. We train the spatial and temporal CNN models and evaluate the performance on two datasets in different scales.

UCF-101. [38] The UCF-101 dataset is a widely used benchmark for action recognition in videos, which consists of 13,320 video clips (27 hours in total). There are 101 annotated classes divided into five types: Human-Object Interaction, Body-Motion Only, Human-Human Interaction, Playing Musical Instruments and Sports. We perform evaluation according to the popular 3 train/test splits following [13]. Results are measured by classification accuracy on each split and mean accuracy over the 3 splits. For some evaluations, we only report results on the first split due to the computation limitation.

FCVID [16] contains 91,223 web videos annotated manually in 239 categories. The categories in FCVID cover a wide range of topics like social events (e.g., "tailgate party"), procedural events (e.g., "making cake"), objects (e.g., "panda"), scenes (e.g., "beach"), etc. These categories are defined carefully and organized in a hierarchy of 11 high-level groups. Specifically, FCVID uses the organization structures on YouTube and Vimeo as references, and browses numerous videos to identify categories that satisfy the following three criteria: (1) utility - high relevance in supporting practical application needs; (2) coverage - a good coverage of the contents that people record; and (3) feasibility - likely to be automatically recognized in the next several years, and a high frequency of occurrence that is sufficient for training a recognition algorithm.

4.2 Neural Network Architectures

The neural network architecture is essential to the performance of a deep learning model and many cutting edge network architectures show their outstanding performance in various image understanding tasks. Recent studies demonstrate that better recognition accuracies are achieved by deepening the CNN architectures [37, 39];, deeper architectures can lead to progressively more discriminative features at higher layers. To evaluate the performance of different network architectures, we try and evaluate a *medium* network structure (VGG19) and a very recent *deeper* network architecture (ResNet). For the temporal network, we also try VGG19 and ResNet.

VGG19 [37] VGG19 not only reduces the size of convolutional filters and the stride, but more importantly, it extends the depth of the network. Specifically, VGG19 consists of 19 layers, including 16 convolutional layers and 3 fully connected layers. In addition, the size of all the convolutional filters decreases to 3×3 and the stride reduces to only 1 pixel, which enables the network to explore finer-grained details from the feature maps. With this deep architecture, VGG19 possesses strong capabilities of learning more discriminative features and the high-level final predictions. It produces a 7.1% top-5 error rate on ILSVRC-2012 [32] validation set.

ResNet [11] is proposed in which the neural network learns the residual information between each layers. In ResNet, many skip shortcuts are built between the network blocks. ResNet abandons the fully-connected layers which contribute to most of the parameters; instead the adaptive pooling layers are used to generate the features. Benefit from these skip connections, ResNet can be designed with a very deep structure while still keeping small number of parameters for fast speed. ResNet achieved the best performance in ILSRVC 2016 challenges [32], where its top-5 error rate in validation set is only 5.71% produced by a 152 layers ResNet. We refer the readers to the detail structure in [11].

Experiment Configuration. We fine-tune the VGG19, ResNet50 and ResNet101 pre-trained on ILSVRC2012. We set the dropout rate to 0.9 for VGG19, and there is no fully-connected layers in ResNet. In the fine-tuning case, the learning rate starts from 10^{-3} and decreases to 10^{-4} after 14K iterations, then to 10^{-5} after 20K iterations on UCF-101. On FCVID, we only fine-tune for 5 epochs using the learning rate 10^{-3}.

Table 2: Spatial stream results of two network architectures on UCF-101.

Models	UCF-101 (split-1)
VGG19	80.41%
ResNet101	80.44%
ResNet152	81.10%

Table 3: Temporal stream results of two network architectures on UCF-101.

Models	UCF-101 (split-1)
VGG19	78.22%
ResNet50	77.63%

Table 4: Speed of each operation. Video decoding and optical flow computing speed is measured on single CPU core. The convolution time is measured on single Titan X (Pascal) by using ResNet152.

Operation	Speed
Decode & Optical Flow	0.04 sec.
Convolution	0.001 sec.

We only train VGG19 and ResNet50 on temporal streams from the scratch of UCF-101. The learning rate is set to 10^{-2} initially, and then decreases to 10^{-3} after 100K iterations, then to 10^{-4} after 200K iterations. The augmentation is similar to [36].

Feature Aggregation. Aggregating multiple clues is a standard technique in video analysis, which can often lead to better performance. The most common aggregate method is fusing the different stream features by averaging the predictions. In this work, we focus on modality feature aggregation.

4.3 Performance Analysis

Classification Performance. Table 2 shows the mean average precision (mAP) on spatial ConvNets for UCF-101. We learn from the table that the accuracy raises up with the increase depth of the pre-trained network. The performance of the temporal stream models is in Table 3. When training VGG19 and ResNet50, we use pre-trained networks to initialize all the layers except for the first input convolution layer and the last softmax layer. After we fuse the two stream modalities, we get the ultimate mAP of 87.8%, where the spatial ConvNet (ResNet152) reaches 81.1% and the temporal ConvNet (ResNet50) reaches 77.63%. In general, our results are comparable with current two-stream approaches such as [55]. We also conduct preliminary experiment (spatial stream) on FCVID by fine-tuning pre-trained ResNet50 for 5 epochs, and got mAP of 62.7% on FCVID validation set.

System Performance. Training on the ESSP system, we obtain a large elastic potential on training scalable sequential data. The main challenge of training spatial or temporal ConvNet is the intermediate storage of the images decoded from the video. The storage size may explode even on small video datasets like UCF-101, where $1.5TB$ is needed to save the full extracted information. Although we can use multi-stage pipeline and data compressing techniques to overcome this difficulty, the increasing scale of the video dataset, which requires hundreds of gigabyte storage, makes the training infeasible.

Table 5: Training time on UCF-101 by using different GPUs. The spatial ConvNet is fine-tuned from pre-trained ResNet101. The temporal ConvNet is ResNet50 trained from scratch.

Network	4 GPUs	8 GPUs
Spatial ConvNet	12 hours	8 hours
Temporal ConvNet	18 hours	10 hours

We evaluate our system's decoding and prefetching efficiency. In video analysis, decoding the video and computing the optical flow are the fundamental procedures. Decoding and computing the optical flow on-the-fly using the off-the-shelf implementation such as ffmpeg[6] and OpenCV[7] may introduce bottleneck on modern GPUs. Table 4 shows the speed of each operation. As we can see from the table, the data prefetching speed restricts the entire convolutional network. In our ESSP system, we load the optical flow and images from the cache pool with no overhead between data loading and training. We train the temporal model on various multi-GPU settings to evaluate the efficiency of our system.

5 CONCLUSIONS

We introduce an elastic system for large-scale video classification. The pre-processing module (video decoding) and machine learning modules (feature extraction, model training, and feature aggregation) in the worker clusters collaborate and are controlled by the master cluster. The system supports automatic task scheduling and multiple machine learning components are automatically parallelized. The dataflow between consecutive worker clusters is executed in an in-memory manner, which avoids the heavy burden to the storage and networking. Preliminary results on two real-world video classification datasets show the efficiency of the proposed system. For our future work, we plan to extend our system to support multi-node caching to facilitate a distributed training framework.

ACKNOWLEDGMENTS

The authors would like to thank the organizers of Large-Scale Video Classification Challenge for providing the dataset and organizing the challenge. This work was supported in part by grants from National Natural Science Foundation of China (No. 61602459) and Science and Technology Commission of Shanghai Municipality (No. 16511104802 and No. 17511101902).

[6]http://ffmpeg.org/
[7]http://opencv.org/

REFERENCES

[1] M. Beal, N. Jojic, and H. Attias. 2003. A graphical model for audiovisual object tracking. *IEEE Trans. PAMI* (2003).

[2] M. Blank, L. Gorelick, E. Shechtman, M. Irani, and R. Basri. 2005. Actions as space-time shapes. In *ICCV*, Vol. 2. 1395–1402.

[3] N. Dalal and B. Triggs. 2005. Histograms of oriented gradients for human detection. In *CVPR*, Vol. 1. 886–893.

[4] N. Dalal, B. Triggs, and C. Schmid. 2006. Human detection using oriented histograms of flow and appearance. In *ECCV*. 428–441.

[5] P. Dollár, V. Rabaud, G. Cottrell, and S. Belongie. 2005. Behavior recognition via sparse spatio-temporal features. In *IEEE VS-PETS*. 65–72.

[6] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko. 2015. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*. 2625–2634.

[7] I. C. Duta, B. Ionescu, K. Aizawa, and N. Sebe. 2017. Spatio-Temporal Vector of Locally Max Pooled Features for Action Recognition in Videos. In *CVPR*.

[8] C. Feichtenhofer, A. Pinz, and R. P Wildes. 2017. Spatiotemporal Multiplier Networks for Video Action Recognition. In *CVPR*.

[9] C. Feichtenhofer, A. Pinz, and A. Zisserman. 2016. Convolutional two-stream network fusion for video action recognition. In *CVPR*.

[10] R. Girdhar, D. Ramanan, A. Gupta, J. Sivic, and B. Russell. 2017. ActionVLAD: Learning spatio-temporal aggregation for action classification. In *CVPR*.

[11] K. He, X. Zhang, S. Ren, and J. Sun. 2016. Deep residual learning for image recognition. In *CVPR*. 770–778.

[12] F. C. Heilbron, V. Escorcia, B. Ghanem, and J. C. Niebles. 2015. Activitynet: A large-scale video benchmark for human activity understanding. In *CVPR*. 961–970.

[13] H. Idrees, A. R. Zamir, Y.-G. Jiang, A. Gorban, I. Laptev, R. Sukthankar, and M. Shah. 2017. The THUMOS Challenge on Action Recognition for Videos "in the Wild". *CVIU* (2017).

[14] S. Ji, W. Xu, M. Yang, and K. Yu. 2013. 3D convolutional neural networks for human action recognition. *IEEE Trans. PAMI* (2013).

[15] W. Jiang, C. Cotton, S.-F. Chang, D. Ellis, and A. Loui. 2009. Short-term audio-visual atoms for generic video concept classification. In *ACM Multimedia*. 5–14.

[16] Y.-G. Jiang, Z. Wu, J. Wang, X. Xue, and S.-F. Chang. 2017. Exploiting feature and class relationships in video categorization with regularized deep neural networks. *IEEE Trans. PAMI* (2017).

[17] Y.-G. Jiang, G. Ye, S.-F. Chang, D. Ellis, and A. C. Loui. 2011. Consumer Video Understanding: A Benchmark Database and An Evaluation of Human and Machine Performance. In *ICMR*.

[18] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *CVPR*. 1725–1732.

[19] A. Klaser, M. Marszałek, and C. Schmid. 2008. A spatio-temporal descriptor based on 3d-gradients. In *BMVC*.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton. 2012. ImageNet classification with deep convolutional neural networks. In *NIPS*.

[21] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. 2011. HMDB: a large video database for human motion recognition. In *ICCV*. 2556–2563.

[22] I. Laptev. 2005. On space-time interest points. *IJCV* (2005).

[23] I. Laptev, M. Marszalek, C. Schmid, and B. Rozenfeld. 2008. Learning realistic human actions from movies. In *CVPR*.

[24] S. Lazebnik, C. Schmid, and J. Ponce. 2006. Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. In *CVPR*. 2169–2178.

[25] A. Loui, J. Luo, S.-F. Chang, D. Ellis, W. Jiang, L. Kennedy, K. Lee, and A. Yanagawa. 2007. Kodak's consumer video benchmark data set: concept definition and annotation. In *MIR*. 245–254.

[26] D. G. Lowe. 2004. Distinctive Image Features from Scale-Invariant Keypoints. *IJCV* 60, 2 (2004), 91–110.

[27] Y. Lu, A. Chowdhery, and S. Kandula. 2016. Optasia: A relational platform for efficient large-scale video analytics. In *ACM SoCC*.

[28] Y. Lu, W. Zhang, K. Zhang, and X. Xue. 2012. Semantic context learning with large-scale weakly-labeled image set. In *CIKM*.

[29] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. 2015. Beyond Short Snippets: Deep Networks for Video Classification. In *CVPR*.

[30] A. Oliva and A. Torralba. 2001. Modeling the Shape of the Scene: A Holistic Representation of the Spatial Envelope. *IJCV* (2001).

[31] X. Peng and C. Schmid. 2016. Multi-region two-stream R-CNN for action detection. In *ECCV*. 744–759.

[32] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and L. Fei-Fei. 2015. ImageNet Large Scale Visual Recognition Challenge. *IJCV* 115, 3 (2015), 211–252.

[33] C. Schuldt, I. Laptev, and B. Caputo. 2004. Recognizing human actions: a local SVM approach. In *ICPR*, Vol. 3. 32–36.

[34] P. Scovanner, S. Ali, and M. Shah. 2007. A 3-dimensional sift descriptor and its application to action recognition. In *ACM Multimedia*. 357–360.

[35] L. Shen, Z. Lin, and Q. Huang. 2016. Relay backpropagation for effective learning of deep convolutional neural networks. In *ECCV*. 467–482.

[36] K. Simonyan and A. Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. In *NIPS*. 568–576.

[37] K. Simonyan and A. Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*.

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

[39] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. Going deeper with convolutions. In *CVPR*.

[40] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. 2015. Learning spatiotemporal features with 3d convolutional networks. In *ICCV*. 4489–4497.

[41] F. Wang, Y.-G. Jiang, and C.-W. Ngo. 2008. Video event detection using motion relativity and visual relatedness. In *ACM Multimedia*. 239–248.

[42] H. Wang, A. Kläser, C. Schmid, and C.-L. Liu. 2011. Action recognition by dense trajectories. In *CVPR*. 3169–3176.

[43] H. Wang and C. Schmid. 2013. Action recognition with improved trajectories. In *ICCV*. 3551–3558.

[44] L. Wang, Y. Lu, H. Wang, Y. Zheng, H. Ye, and X. Xue. 2017. Evolving Boxes for fast Vehicle Detection. In *ICME*.

[45] L. Wang, Y. Qiao, and X. Tang. 2015. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*.

[46] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. 2016. Temporal segment networks: Towards good practices for deep action recognition. In *ECCV*. 20–36.

[47] Y. Wang, M. Long, J. Wang, and P. S Yu. 2017. Spatiotemporal Pyramid Network for Video Action Recognition. In *CVPR*.

[48] Y. Wang, J. Song, L. Wang, L. Van Gool, and O. Hilliges. 2016. Two-Stream SR-CNNs for Action Recognition in Videos.. In *BMVC*.

[49] Z. Wu, Y.-G. Jiang, L S Davis, and S.-F. Chang. 2017. LSVC2017: Large-Scale Video Classification Challenge. http://bigvid.fudan.edu.cn/LSVC_MM17/. (2017).

[50] Z. Wu, Y.-G. Jiang, X. Wang, H. Ye, and X. Xue. 2016. Multi-stream multi-class fusion of deep networks for video classification. In *ACM Multimedia*. 791–800.

[51] Z. Wu, X. Wang, Y.-G. Jiang, H. Ye, and X. Xue. 2015. Modeling spatial-temporal clues in a hybrid deep learning framework for video classification. In *ACM Multimedia*. 461–470.

[52] X. Xue, W. Zhang, J. Zhang, B. Wu, J. Fan, and Y. Lu. 2011. Correlative multi-label multi-instance image annotation. In *ICCV*.

[53] G. Ye, I-H. Jhuo, D. Liu, Y.-G. Jiang, DT Lee, and S.-F. Chang. 2012. Joint audio-visual bi-modal codewords for video event detection. In *ICMR*. 39.

[54] G. Ye, Y. Li, H. Xu, D. Liu, and S.-F. Chang. 2015. Eventnet: A large scale structured concept library for complex event detection in video. In *ACM Multimedia*. 471–480.

[55] H. Ye, Z. Wu, R.-W. Zhao, X. Wang, Y.-G. Jiang, and X. Xue. 2015. Evaluating two-stream CNN for video classification. In *ICMR*. 435–442.

[56] B. Zhang, L. Wang, Z. Wang, Y. Qiao, and H. Wang. 2016. Real-time action recognition with enhanced motion vector CNNs. In *CVPR*. 2718–2726.