Modeling Spatial-Temporal Clues in a Hybrid Deep Learning Framework for Video Classification

Zuxuan Wu, Xi Wang, Yu-Gang Jiang[*], Hao Ye, Xiangyang Xue
School of Computer Science, Shanghai Key Lab of Intelligent Information Processing,
Fudan University, Shanghai, China
{zxwu, xwang10, ygj, haoye10, xyxue}@fudan.edu.cn

ABSTRACT

Classifying videos according to content semantics is an important problem with a wide range of applications. In this paper, we propose a hybrid deep learning framework for video classification, which is able to model static spatial information, short-term motion, as well as long-term temporal clues in the videos. Specifically, the spatial and the short-term motion features are extracted separately by two Convolutional Neural Networks (CNN). These two types of CNN-based features are then combined in a regularized feature fusion network for classification, which is able to learn and utilize feature relationships for improved performance. In addition, Long Short Term Memory (LSTM) networks are applied on top of the two features to further model longer-term temporal clues. The main contribution of this work is the hybrid learning framework that can model several important aspects of the video data. We also show that (1) combining the spatial and the short-term motion features in the regularized fusion network is better than direct classification and fusion using the CNN with a softmax layer, and (2) the sequence-based LSTM is highly complementary to the traditional classification strategy without considering the temporal frame orders. Extensive experiments are conducted on two popular and challenging benchmarks, the UCF-101 Human Actions and the Columbia Consumer Videos (CCV). On both benchmarks, our framework achieves very competitive performance: 91.3% on the UCF-101 and 83.5% on the CCV.

Categories and Subject Descriptors

I.5.2 [**Pattern Recognition**]: Design Methodology—*Classifier design and evaluation*

Keywords

Video Classification; Deep Learning; CNN; LSTM; Fusion.

[*]Corresponding author.

MM'15, October 26–30, 2015, Brisbane, Australia.
© 2015 ACM. ISBN 978-1-4503-3459-4/15/10 ...$15.00.
DOI: http://dx.doi.org/10.1145/2733373.2806222.

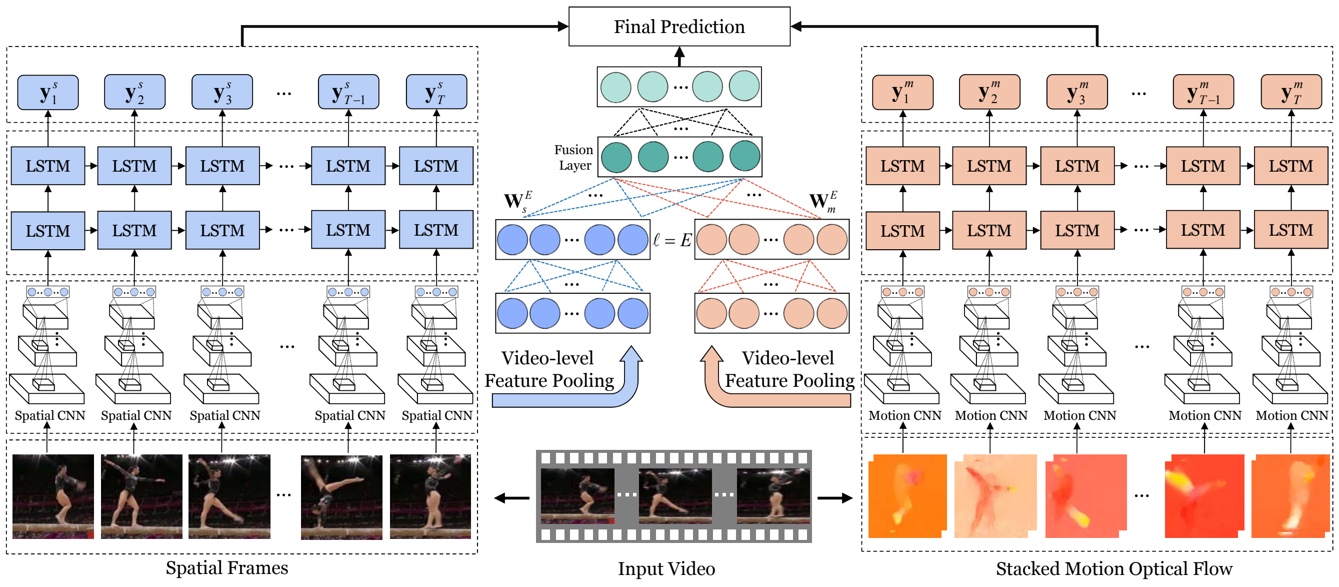
1. INTRODUCTION

Video classification based on contents like human actions or complex events is a challenging task that has been extensively studied in the research community. Significant progress has been achieved in recent years by designing various features, which are expected to be robust to intra-class variations and discriminative to separate different classes. For example, one can utilize traditional image-based features like the SIFT [27] to capture the static spatial information in videos. In addition to the static frame based visual features, motion is a very important clue for video classification, as most classes containing object movements like the human actions require the motion information to be reliably recognized. For this, a very popular feature is the dense trajectories [44], which tracks densely sampled local frame patches over time and computes several traditional features based on the trajectories.

In contrast to the hand-crafted features, there is a growing trend of learning robust feature representations from raw data with deep neural networks. Among the many existing network structures, Convolutional Neural Networks (CNN) have demonstrated great success on various tasks, including image classification [22, 38, 34], image-based object localization [7], speech recognition [5], *etc.* For video classification, Ji *et al.* [13] and Karparthy *et al.* [19] extended the CNN to work on the temporal dimension by stacking frames over time. Recently, Simonyan *et al.* [33] proposed a two-stream CNN approach, which uses two CNNs on static frames and optical flows respectively to capture the spatial and the motion information. It focuses only on short-term motion as the optical flows are computed in very short time windows. With this approach, similar or slightly better performance than the hand-crafted features like [44] has been reported.

These existing works, however, are not able to model the long-term temporal clues in the videos. As aforementioned, the two-stream CNN [33] uses stacked optical flows computed in short time windows as inputs, and the order of the optical flows is fully discarded in the learning process (cf. Section 3.1). This is not sufficient for video classification, as many complex contents can be better identified by considering the temporal order of short-term actions. Take "birthday" event as an example—it usually involves several sequential actions, such as "making a wish", "blowing out candles" and "eating cakes".

To address the above limitation, this paper proposes a hybrid deep learning framework for video classification, which is able to harness not only the spatial and short-term motion features, but also the long-term temporal clues. In or-



Final Prediction
Fusion Layer
$\mathbf{W}_s^E$
$\mathbf{W}_m^E$
$\ell = E$
Video-level Feature Pooling
Video-level Feature Pooling
$\mathbf{y}_1^s$ $\mathbf{y}_2^s$ $\mathbf{y}_3^s$ $\mathbf{y}_{T-1}^s$ $\mathbf{y}_T^s$
$\mathbf{y}_1^m$ $\mathbf{y}_2^m$ $\mathbf{y}_3^m$ $\mathbf{y}_{T-1}^m$ $\mathbf{y}_T^m$
LSTM
Spatial CNN
Motion CNN
Spatial Frames
Input Video
Stacked Motion Optical Flow

Figure 1: An overview of the proposed hybrid deep learning framework for video classification. Given an input video, two types of features are extracted using the CNN from spatial frames and short-term stacked motion optical flows respectively. The features are separately fed into two sets of LSTM networks for long-term temporal modeling (Left and Right). In addition, we also employ a regularized feature fusion network to perform video-level feature fusion and classification (Middle). The outputs of the sequence-based LSTM and the video-level feature fusion network are combined to generate the final prediction. See texts for more discussions.

der to leverage the temporal information, we adopt a Recurrent Neural Networks (RNN) model called Long Short Term Memory (LSTM), which maps the input sequences to outputs using a sequence of hidden states and incorporates memory units that enable the network to learn when to forget previous hidden states and when to update hidden states with new information. In addition, many approaches fuse multiple features in a very "shallow" manner by either concatenating the features before classification or averaging the predictions of classifiers trained using different features separately. In this work we integrate the spatial and the short-term motion features in a deep neural network with carefully designed regularizations to explore feature correlations. This method can perform video classification within the same network and further combining its outputs with the predictions of the LSTMs can lead to very competitive classification performance.

Figure 1 gives an overview of the proposed framework. Spatial and short-term motion features are first extracted by the two-stream CNN approach [33], and then input into the LSTM for long-term temporal modeling. Average pooling is adopted to generate video-level spatial and motion features, which are fused by the regularized feature fusion network. After that, outputs of the sequence-based LSTM and the video-level feature fusion network are combined as the final predictions. Notice that, in contrast to the current framework, alternatively one may train a fusion network to combine the frame-level spatial and motion features first and then use a single set of LSTM for temporal modeling. However, in our experiments we have observed worse results using this strategy. The main reason is that learning dimension-wise feature correlations in the fusion network requires strong and reliable supervision, but we only have video-level class labels which are not necessarily always related to the frame semantics. The main contributions of this work are summarized as follows:

- We propose a hybrid deep learning framework for video classification, which can model not only the short-term spatial-motion patterns but also the long-term temporal clues with variable-length video sequences as inputs.
- We adopt the LSTM to model long-term temporal clues on top of both the spatial and the short-term motion features. We show that both features work well with the LSTM, and the LSTM based classifiers are very complementary to the traditional classifiers without considering the temporal frame orders.
- We fuse the spatial and the motion features in a regularized feature fusion network that can explore feature correlations and perform classification. The network is computationally efficient in both training and testing.
- Through an extensive set of experiments, we demonstrate that our proposed framework outperforms several alternative methods with clear margins. On the well-known UCF-101 and CCV benchmarks, we attain very competitive performance.

The rest of this paper is organized as follows. Section 2 reviews related works. Section 3 describes the proposed hybrid deep learning framework in detail. Experimental results and comparisons are discussed in Section 4, followed by conclusions in Section 5.

2. RELATED WORKS

Video classification has been a longstanding research topic in multimedia and computer vision [16]. Successful classification systems rely heavily on the extracted video features, and hence most existing works focused on designing robust features. Many video representations were motivated by the advances in image domain, which can be extended to utilize the temporal dimension of the video data. For instance, Laptev [24] extended the 2D Harris corner detector [10] into 3D space to find space-time interest points. Klaser *et al.* proposed HOG3D by extending the idea of integral images for fast descriptor computation [20]. Wang *et al.* reported that dense sampling at regular positions in space and time outperforms the detected sparse interest points on video classification tasks [45]. Partly inspired by this finding, they further proposed the dense trajectory features, which densely sample local patches from each frame at different scales and then track them in a dense optical flow field over time [44]. This method has demonstrated very competitive results on major benchmark datasets. In addition, further improvements may be achieved by using advantageous feature encoding methods like the Fisher Vectors [30] or adopting feature normalization strategies, such as RootSift [1]. Note that these spatial-temporal video descriptors only capture local motion patterns within a very short period, and popular descriptor quantization methods like the bag-of-words entirely destroy the temporal order information of the descriptors.

To explore the long-term temporal clues, graphical models have been popularly used, such as hidden Markov models (HMM), Bayesian Networks (BN), Conditional Random Fields (CRF), *etc.* For instance, Li *et al.* proposed to replace the hidden states in HMMs with visualizable salient poses estimated by Gaussian Mixture Models [25], and Tang *et al.* introduced latent variables over video frames to discover the most discriminative states of an event based on a variable duration HMM [39]. Zeng *et al.* exploited multiple types of domain knowledge to guide the learning of a Dynamic BN for action recognition [52]. Instead of using directed graphical models like the HMM and BN, undirected graphical models have also been adopted. Vail *et al.* employed the CRF for activity recognition in [41]. Wang *et al.* proposed a max-margin hidden CRF for action recognition in videos, where a human action is modeled as a global root template and a constellation of several "parts" [46].

Many related works have investigated the fusion of multiple features, which is often effective for improving classification performance. The most straightforward and popular ways are early fusion and late fusion. Generally, the early fusion refers to fusion at the feature level, such as feature concatenation or linear combination of kernels of individual features. For example, in [54], Zhang *et al.* computed nonlinear kernels for each feature separately, and then fused the kernels for model training. The fusion weights can be manually set or automatically estimated by multiple kernel learning (MKL) [2]. For the late fusion methods, independent classifiers are first trained using each feature separately, and outputs of the classifiers are then combined. In [50], Ye *et al.* proposed a robust late fusion approach to fuse multiple classification outputs by seeking a shared low-rank latent matrix, assuming that noises may exist in the predictions of some classifiers, which can possibly be removed by using the low-rank matrix.

Both early and late fusion fail to explore the correlations shared by the features and hence are not ideal for video classification. In this paper we employ a regularized neural network tailored feature fusion and classification, which can automatically learn dimension-wise feature correlations. Several studies are related. In [15, 12], the authors proposed to construct an audio-visual joint codebook for video classification, in order to discover and model the audio-visual feature correlations. There are also studies on using neural networks for feature fusion. In [37], the authors employed deep Boltzmann machines to learn a fused representation of images and texts. In [29], a deep denoised auto-encoder was used for cross-modality and shared representation learning. Very recently, Wu *et al.* [47] presented an approach using regularizations in neural networks to exploit feature and class relationships. The fusion approach in this work differs in the following. First, instead of using the traditional handcrafted features as inputs, we adopt CNN features trained from both static frames and motion optical flows. Second and very importantly, the formulation in the regularized feature fusion network has a much lower complexity compared with that of [47].

Researchers have attempted to apply the RNN to model the long-term temporal information in videos. Venugopalan *et al.* [43] proposed to translate videos to textual sentences with the LSTM through transferring knowledge from image description tasks. Ranzato *et al.* [31] introduced a generative model with the RNN to predict motions in videos. In the context of video classification, Donahua *et al.* adopted the LSTM to model temporal information [6] and Srivastava *et al.* designed an encoder-decoder RNN architecture to learn feature representations in an unsupervised manner [36]. To model motion information, both works adopted optical flow "images" between nearby frames as the inputs of the LSTM. In contrast, our approach adopts stacked optical flows. Stacked flows over a short time period can better reflect local motion patterns, which are found to be able to produce better results. In addition, our framework incorporates video-level predictions with the feature fusion network for significantly improved performance, which was not considered in these existing works.

Besides the above discussions of related studies on feature fusion and temporal modeling with the RNN, several representative CNN-based approaches for video classification should also be covered here. The image-based CNN features have recently been directly adopted for video classification, extracted using off-the-shelf models trained on large-scale image datasets like the ImageNet [11, 32, 53]. For instance, Jain *et al.* [11] performed action recognition using the SVM classifier with such CNN features and achieved top results in the 2014 THUMOS action recognition challenge [17]. A few works have also tried to extend the CNN to exploit the motion information in videos. Ji *et al.* [13] and Karpathy *et al.* [19] extended the CNN by stacking visual frames in fixed-size time windows and using spatial-temporal convolutions for video classification. Differently, the two-stream CNN approach by Simonyan *et al.* [33] applies the CNN separately on visual frames (the spatial stream) and stacked optical flows (the motion stream). This approach has been found to be more effective [51], which is adopted as the basis of our proposed framework. However, as discussed in Section 1, all these approaches [13, 19, 33] can only model short-term motion, not the long-term temporal clues.

3. METHODOLOGY

In this section, we describe the key components of the proposed hybrid deep learning framework shown in Figure 1, including the CNN-based spatial and short-term motion features, the long-term LSTM-based temporal modeling, and the video-level regularized feature fusion network.

3.1 Spatial and Motion CNN Features

Conventional CNN architectures take images as the inputs and consist of alternating convolutional and pooling layers, which are further topped by a few fully-connected (FC) layers. To extract the spatial and the short-term motion features, we adopt the recent two-stream CNN approach [33]. Instead of using stacked frames in short time windows like [13, 19], this approach decouples the videos into spatial and motion[1] streams modeled by two CNNs separately. Figure 2 gives an overview. The spatial stream is built on sampled individual frames, which is exactly the same as the CNN-based image classification pipeline and is suitable for capturing the static information in videos like scene backgrounds and basic objects. The motion counterpart operates on top of stacked optical flows. Specifically, optical flows (displacement vector fields) are computed between each pair of adjacent frames, and the horizontal and vertical components of the displacement vectors can form two optical flow images. Instead of using each individual flow image as the input of the CNN, it was reported that stacked optical flows over a time window are better due to the ability of modeling the short-term motion. In other words, the input of the motion stream CNN is a $2L$-channel stacked optical flow image, where L is the number of frames in the window. The two CNNs produce classification scores separately using a softmax layer and the scores are linearly combined as the final prediction.

Like many existing works on visual classification using the CNN features [32], we adopt the output of the first FC layer of the two CNNs as the spatial and the short-term motion features.

3.2 Temporal Modeling with LSTM

During the training process of the spatial and the motion stream CNNs, each sweep through the network takes one visual frame or one stacked optical flow image, and the temporal order of the frames is fully discarded. To model the long-term dynamic information in video sequences, we leverage the LSTM model, which has been successfully applied to speech recognition [8], image captioning [6], *etc.* LSTM is a type of RNN with controllable memory units and is effective in many long range sequential modeling tasks without suffering from the "vanishing gradients" effect like traditional RNNs. Generally, LSTM recursively maps the input representations at the current time step to output labels via a sequence of hidden states, and thus the learning process of LSTM should be in a sequential manner (from left to right in the two sets of LSTM of Figure 1). Finally, we can obtain a prediction score at each time step with a softmax transformation using the hidden states from the last layer of the LSTM.

More formally, given a sequence of feature representations $(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$, an LSTM maps the inputs to an output

[1]Note that the authors of the original paper [33] used the name "temporal stream". We call it "motion stream" as it only captures short-term motion, which is different from the long-term temporal modeling in our proposed framework.

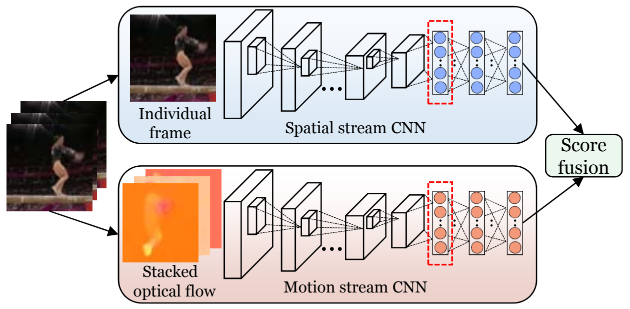


Figure 2: The framework of the two-stream CNN. Outputs of the first fully-connected layer in the two CNNs (outlined) are used as the spatial and the short-term motion features for further processing.

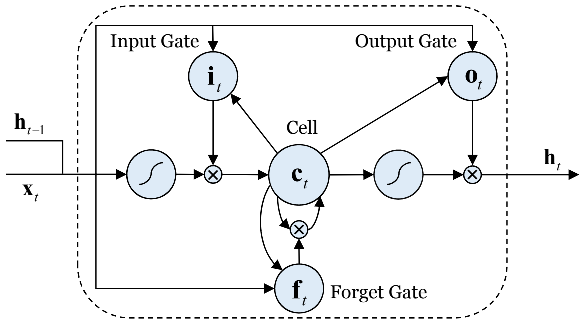


Figure 3: The structure of an LSTM unit.

sequence $(\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T)$ by computing activations of the units in the network with the following equations recursively from $t = 1$ to $t = T$:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i),\\
\mathbf{f}_t &= \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{W}_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f),\\
\mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c),\\
\mathbf{o}_t &= \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_t + \mathbf{b}_o),\\
\mathbf{h}_t &= \mathbf{o}_t \tanh(\mathbf{c}_t),
\end{aligned}$$

where $\mathbf{x}_t, \mathbf{h}_t$ are the input and hidden vectors with the subscription t denoting the t-th time step, $\mathbf{i}_t, \mathbf{f}_t, \mathbf{c}_t, \mathbf{o}_t$ are respectively the activation vectors of the input gate, forget gate, memory cell and output gate, $\mathbf{W}_{\alpha\beta}$ is the weight matrix between α and β (e.g., $\mathbf{W}_{xi}$ is weight matrix from the input $\mathbf{x}_t$ to the input gate $\mathbf{i}_t$), $\mathbf{b}_\alpha$ is the bias term of α and σ is the sigmoid function defined as $\sigma(x) = \frac{1}{1+e^{-x}}$. Figure 3 visualizes the structure of an LSTM unit.

The core idea behind the LSTM model is a built-in memory cell that stores information over time to explore long-range dynamics, with non-linear gate units governing the information flow into and out of the cell. As we can see from the above equations, the current frame $\mathbf{x}_t$ and the previous hidden states $\mathbf{h}_{t-1}$ are used as inputs of four parts at the t-th time step. The memory cell aggregates information from two sources: the previous cell memory unit $\mathbf{c}_{t-1}$ multiplied by the activation of the forget gate $\mathbf{f}_t$ and the squashed inputs regulated with the input gate's activation $\mathbf{i}_t$, the combination of which enables the LSTM to learn to forget information from previous states or consider new information.

In addition, the output gate $\mathbf{o}_t$ controls how much information from the memory cell is passed to the hidden states $\mathbf{h}_t$ for the following time step. With the explicitly controllable memory units and different functional gates, LSTM can explore long-range temporal clues with variable-length inputs. As a neural network, the LSTM model can be easily deepened by stacking the hidden states from a layer $l-1$ as inputs of the next layer l.

Let us now consider a model of K layers, the feature vector $\mathbf{x}_t$ at the t-th time step is fed into the first layer of the LSTM together with the hidden state $\mathbf{h}_{t-1}^1$ in the same layer obtained from the last time step to produce an updated $\mathbf{h}_t^1$, which will then be used as the inputs of the following layer. Denote f_W as the mapping function from the inputs to the hidden states, the transition from layer $l-1$ to layer l can be written as:

$$\mathbf{h}_t^l = \begin{cases} f_W(\mathbf{h}_t^{l-1}, \mathbf{h}_{t-1}^l), & l > 1; \\ f_W(\mathbf{x}_t, \mathbf{h}_{t-1}^l), & l = 1. \end{cases}$$

In order to obtain the prediction scores for a total of C classes at a time step t, the outputs from last layer of the LSTM are sent to a softmax layer estimating probabilities as:

$$prob_c = \frac{\exp(\mathbf{w}_c{}^T \mathbf{h}_t^K + b_c)}{\sum_{c' \in C} \exp(\mathbf{w}_{c'}{}^T \mathbf{h}_t^K + b_{c'})},$$

where $prob_c$, $\mathbf{w}_c$ and b_c are respectively the probability prediction, the corresponding weight vector and the bias term of the c-th class. Such an LSTM network can be trained with the Back-Propagation Through Time (BPTT) algorithm [9], which "unrolls" the model into a feed forward neural net and back-propagates to determine the optimal weights.

As shown in Figure 1, we adopt two LSTM models for temporal modeling. With the two-stream CNN pipeline for feature extraction, we have a spatial feature set $(\mathbf{x}_1^s, \mathbf{x}_2^s, \ldots, \mathbf{x}_T^s)$ and a motion feature set $(\mathbf{x}_1^m, \mathbf{x}_2^m, \ldots, \mathbf{x}_T^m)$. The learning process leads to a set of predictions $(\mathbf{y}_1^s, \mathbf{y}_2^s, \ldots, \mathbf{y}_T^s)$ for the spatial part and another set $(\mathbf{y}_1^m, \mathbf{y}_2^m, \ldots, \mathbf{y}_T^m)$ for the motion part. For both LSTM models, we adopt the last step output $\mathbf{y}_T$ as the video-level prediction scores, since the outputs at the last time step are based on the consideration of the entire sequence. We empirically observe that the last step outputs are better than pooling predictions from all the time steps.

3.3 Regularized Feature Fusion Network

Given both the spatial and the motion features, it is easy to understand that correlations may exist between them since both are computed on the same video (e.g., person-related static visual features and body motions). A good feature fusion method is supposed to be able to take advantages of the correlations, while also can maintain the unique characteristics to produce a better fused representation. In order to explore this important problem rather than using the simple late fusion as [33], we employ a regularized feature fusion neural network, as shown in the middle part of Figure 1. First, average pooling is adopted to aggregate the frame-level CNN features into video-level representations, which are used as the inputs of the fusion network. The input features are non-linearly mapped to another layer and then fused in a feature fusion layer, where we apply regularizations in the learning of the network weights.

Denote N as the total number of training videos with both the spatial and the motion representations. For the n-th sample, it can be written as a 3-tuple as $(\mathbf{x}_n^s, \mathbf{x}_n^m, \mathbf{y}_n)$, where $\mathbf{x}_n^s = \sum_{t=1}^T \mathbf{x}_{n,t}^s \in \mathbb{R}^{d_s}$ and $\mathbf{x}_n^m = \sum_{t=1}^T \mathbf{x}_{n,t}^m \in \mathbb{R}^{d_m}$ represent the averaged spatial and motion features respectively, and $\mathbf{y}_n$ is the corresponding label of the n-th sample.

For the ease of discussion, let us consider a degenerated case first, where only one feature is available. Denote $g(\cdot)$ as the mapping of the neural network from inputs to outputs. The objective of network training is to minimize the following empirical loss:

$$\min_{\mathbf{W}} \sum_{i=1}^N \|g(\mathbf{x}_i) - \mathbf{y}_i\|^2 + \lambda_1 \Phi(\mathbf{W}), \tag{1}$$

where the first term measures the discrepancy between the output $g(\mathbf{x}_i)$ and the ground-truth label $\mathbf{y}_i$, and the second term is usually a Frobenius norm based regularizer to prevent over-fitting.

We now move on to discuss the case of fusion and prediction with two features. Note that the approach can be easily extended to support more than two input features. Specifically, we use a fusion layer (see Figure 1) to absorb the spatial and temporal features into a fused representation. To exploit the correlations of the features, we regularize the fusion process with a structural ℓ_{21} norm, which is defined as $\|\mathbf{W}\|_{2,1} = \sum_i \sqrt{\sum_j w_{ij}^2}$. Incorporating the ℓ_{21} norm in the standard deep neural network formulation, we arrive at the following optimization problem:

$$\min_{\mathbf{W}} \mathcal{L} + \lambda_1 \Phi(\mathbf{W}) + \frac{\lambda_2}{2} \left\|\mathbf{W}^E\right\|_{2,1}, \tag{2}$$

where $\mathcal{L} = \sum_{i=1}^N \|g(\mathbf{x}_i^s, \mathbf{x}_i^m) - \mathbf{y}_i\|^2$, $\mathbf{W}^E = [\mathbf{W}_s^E, \mathbf{W}_m^E] \in \mathbb{R}^{P \times D}$ denotes the concatenated weight matrix from the Eth layer (i.e., the last layer of feature abstraction in Figure 1), $D = d_s + d_m$ and P is the dimension of the fusion layer.

Different from the objective in Equation (1), here we have an additional ℓ_{21} norm that is used for exploring feature correlations in the E-th layer. The term $\|\mathbf{W}^E\|_{2,1}$ computes the 2-norm of the weight values across different features in each dimension. Therefore, the regularization attains minimum when $\mathbf{W}^\mathbf{E}$ contains the smallest number of non-zero rows, which is the discriminative information shared by distinct features. That is to say, the ℓ_{21} norm encourages the matrix $\mathbf{W}^E$ to be row sparse, which leads to similar zero/nonzero patterns of the columns of the matrix $\mathbf{W}^E$. Hence it enforces different features to share a subset of hidden neurons, reflecting the feature correlations.

However, in addition to seeking for the correlations shared among features, the unique discriminative information should also be preserved so that the complementary information can be used for improved classification performance. Thus, we add an additional regularizer to Equation (2) as following:

$$\min_{\mathbf{W}} \mathcal{L} + \lambda_1 \Phi(\mathbf{W}) + \frac{\lambda_2}{2} \left\|\mathbf{W}^E\right\|_{2,1} + \lambda_3 \left\|\mathbf{W}^E\right\|_{1,1}. \tag{3}$$

The term $\|\mathbf{W}^E\|_{1,1}$ can be regarded as a complement of the $\|\mathbf{W}^E\|_{2,1}$ norm. It provides the robustness of the ℓ_{21} norm from sharing incorrect features among different representations, and thus allows different representations to emphasize different hidden neurons.

Although the regularizer terms in Equation (3) are all convex functions, the optimization problem in Equation (3) is nonconvex due to the nonconvexity of the sigmoid function. Below, we discuss the optimization strategy using the gradient descent method in two cases:

1. For the E-th layer, our objective function has four valid terms: the empirical loss, the ℓ_2 regularizer $\Phi(\mathbf{W})$, and two nonsomooth structural regularizers, *i.e.*, the ℓ_{21} and ℓ_{11} terms. Note that simply using the gradient descent method is not optimal due to the two nonsmooth terms. We propose to optimize the E-th layer using a proximal gradient descent method, which splits the objective function into two parts:

$$p = \mathcal{L} + \lambda_1 \Phi(\mathbf{W}),$$
$$q = \frac{\lambda_2}{2} \left\| \mathbf{W}^E \right\|_{2,1} + \lambda_3 \left\| \mathbf{W}^E \right\|_{1,1},$$

where p is a smooth function and q is a nonsmooth function. Thus, the update of the i-th iteration is formulated as:

$$(\mathbf{W}^E)^{(i)} = \text{Prox}_q((\mathbf{W}^E)^{(i)} - \nabla p((\mathbf{W}^E)^{(i)})),$$

where Prox is a proximal operator defined as:

$$\text{Prox}_q(\mathbf{W}) = \arg\min_{\mathbf{V}} \|\mathbf{W} - \mathbf{V}\| + q(V).$$

The proximal operator on the combination of ℓ_{21}/ℓ_{11} norm ball can be computed analytically as:

$$\mathbf{W}^E_{r\cdot} = \left(1 - \frac{\lambda_2}{\|\mathbf{U}_{r\cdot}\|_2}\right) \mathbf{U}_{r\cdot}, \forall r = 1, \cdots, P, \quad (4)$$

where $\mathbf{U}_{r\cdot} = [|\mathbf{V}_{r\cdot}| - \lambda_3]_+ \cdot sign[\mathbf{V}_{r\cdot}]$, and $\mathbf{W}_{r\cdot}, \mathbf{U}_{r\cdot}, \mathbf{V}_{r\cdot}$ denote the r-th row of matrix $\mathbf{W}, \mathbf{U}$ and $\mathbf{V}$, respectively. Readers may refer to [49] for a detailed proof of a similar analytical solution.

2. For all the other layers, the objective function in Equation 3 only contains the first two valid terms, which are both smooth. Thus, we can directly apply the gradient descent method as in [3]. Denote $\mathbf{G}^l$ as the gradient of $\mathbf{W}^l$, the weight matrix of the lth layer is updated as:

$$\mathbf{W}^l = \mathbf{W}^l - \eta \mathbf{G}^l. \quad (5)$$

The only additional computation cost for training our regularized feature fusion network is to compute the proximal operator in the E-th layer. The complexity of the analytical solution in Equation (4) is $O(P \times D)$. Therefore, the proposed proximal gradient descent method can quickly train the network with affordable computational cost. When incorporating more features, our formulation can be computed efficiently with linearly increased cost, while cubic operations are required by the approach of [47] to reach a similar goal. In sum, the above optimization is incorporated into the conventional back propagation procedure, as described in Algorithm 1.

```
Input : $\mathbf{x}^s_n$ and $\mathbf{x}^m_n$: the spatial and motion CNN
        features of the $n$-th video sample;
        $\mathbf{y}_n$: the label of the $n$-th video sample;
        randomly initialized weight matrices $\mathbf{W}$;
1  begin
2      for epoch $\leftarrow 1$ to $M$ do
3          Get the prediction error with feedforward
           propagation;
4          for $l \leftarrow L$ to 1 do
5              Evaluate the gradients and update the
               weight matrices using Equation (5);
6              if $l == E$ then
7                  Evaluate the proximal operator
                   according to Equation (4);
8              end
9          end
10     end
11 end
```

Algorithm 1: The training procedure of the regularized feature fusion network.

3.4 Discussion

The approach described above has the capability of modeling static spatial, short-term motion and long-term temporal clues, which are all very important in video content analysis. One may have noticed that the proposed hybrid deep learning framework contains several components that are separately trained. Joint training is feasible but not adopted in this current framework for the following reason. The joint training process is more complex and existing works exploring this aspect indicate that the performance gain is not very significant. In [6], the authors jointly trained the LSTM with a CNN for feature extraction, which only improved the performance on a benchmark dataset from 70.5% to 71.1%. Besides, an advantage of separate training is that the framework is more flexible, where a component can be replaced easily without the need of re-training the entire framework. For instance, more discriminative CNN models like the GoogLeNet [38] and deeper RNN models [4] can be used to replace the CNN and LSTM parts respectively.

In addition, as mentioned in Section 1, there could be alternative frameworks or models with similar capabilities. The main contribution of this work is to show that such a hybrid framework is very suitable for video classification. In addition to showing the effectiveness of the LSTM and the regularized feature fusion network, we also show that the combination of both in the hybrid framework can lead to significant improvements, particularly for long videos that contain rich temporal clues.

4. EXPERIMENTS

4.1 Experimental Setup

4.1.1 Datasets

We adopt two popular datasets to evaluate the proposed hybrid deep learning framework.

UCF-101 [35]. The UCF-101 dataset is one of the most popular action recognition benchmarks. It consists of 13,320 video clips of 101 human actions (27 hours in total). Following [17], we conduct evaluations using 3 train/test splits, which is currently the most popular setting in using this dataset. Results are measured by classification accuracy on each split and we report the mean accuracy over the three splits.

Columbia Consumer Videos (CCV) [18]. The CCV dataset contains 9,317 YouTube videos annotated according to 20 classes, which are mainly events like “basketball”, “graduation ceremony”, “birthday party” and “parade”. We follow the convention defined in [18] to use a training set of 4,659 videos and a test set of 4,658 videos. The performance is evaluated by average precision (AP) for each class, and we report the mean AP (mAP) as the overall measure.

4.1.2 Implementation Details

For feature extraction network structure, we adopt the VGG_19 [34] and the CNN_M [33] to extract the spatial and the motion CNN features, respectively. The two networks can achieve 7.5% [34] and 13.5% [33] top-5 error rates on the ImageNet ILSVRC-2012 validation set respectively. The spatial CNN is first pre-trained with the ILSVRC-2012 training set with 1.2 million images and then fine-tuned using the video data, which is observed to be better than training from scratch. The motion CNN is trained from scratch as there is no off-the-shelf training set in the required form. In addition, simple data augmentation methods like cropping and flipping are utilized following [33].

The CNN models are trained using mini-batch stochastic gradient descent with a momentum fixed to 0.9. In the fine-tuning case of the spatial CNN, the rate starts from 10^{-3} and decreases to 10^{-4} after 14K iterations, then to 10^{-5} after 20K iterations. This setting is similar to [33], but we start from a smaller rate of 10^{-3} instead of 10^{-2}. For the motion CNN, we set the learning rate to 10^{-2} initially, and reduce it to 10^{-3} after 100K iterations, then to 10^{-4} after 200K iterations. Our implementation is based on the publicly available Caffe toolbox [14] with modifications to support parallel training with multiple GPUs in a server.

For temporal modeling, we adopt two layers in the LSTM for both the spatial and the motion features. Each LSTM has 1,024 hidden units in the bottom layer and 512 hidden units in the other layer. The network weights are learnt using a parallel implementation of the BPTT algorithm with a mini-batch size of 10. In addition, the learning rate and momentum are set to 10^{-4} and 0.9 respectively. The training is stopped after 150K iterations for both datasets.

For the regularized feature fusion network, we use four layers of neurons as illustrated in the middle of Figure 1. Specifically, we first use one layer with 200 neurons for the spatial and motion feature to perform feature abstraction separately, and then one layer with 200 neurons for feature fusion with the proposed regularized structural norms. Finally, the fused features are used to build a logistic regression model in the last layer for video classification. We set the learning rate to 0.7 and fix λ_1 to 3×10^{-5} in order to prevent over-fitting. In addition, we tune λ_2 and λ_3 in the same range as λ_1 using cross-validation.

4.1.3 Compared Approaches

To validate the effectiveness of our approach, we compare with the following baseline or alternative methods: (1) **Two-stream CNN**. Our implementation produces similar overall results with the original work [33]. We also report the results of the individual spatial-stream and motion-stream CNN models, namely **Spatial CNN** and **Motion CNN**, respectively; (2) **Spatial LSTM**, which refers to the LSTM trained with the spatial CNN features; (3) **Motion LSTM**, the LSTM trained with the motion CNN features; (4) **SVM-based Early Fusion (SVM-EF)**. χ^2-kernel is computed for each video-level CNN feature and then the two kernels are averged for classification; (5) **SVM-based Late Fusion (SVM-LF)**. Separate SVM classifiers are trained for each video-level CNN feature and the prediction outputs are averaged; (6) **Multiple Kernel Learning (SVM-MKL)**, which combines the two features with the ℓ_p-norm MKL [21] by fixing $p = 2$; (7) **Early Fusion with Neural Networks (NN-EF)**, which concatenates the two features into a long vector and then use a neural network for classification; (8) **Late Fusion with Neural Networks (NN-LF)**, which deploys a separate neural network for each feature and then uses the average output scores as the final prediction; (9) **Multimodal Deep Boltzmann Machines (M-DBM)** [29, 37], where feature fusion is performed using a neural network in a *free* manner without regularizations; (10) **RDNN** [47], which also imposes regularizations in a neural network for feature fusion, using a formulation that has a much higher complexity than our approach.

The first three methods are a part of the proposed framework, which are evaluated as baselines to better understand the contribution of each individual component. The last seven methods focus on fusing the spatial and the motion features (outputs of the first fully-connected layer of the CNN models) for improved classification performance. We compare our regularized fusion network with all the seven methods.

4.2 Results and Discussions

4.2.1 Temporal Modeling

We first evaluate the LSTM to investigate the significance of leveraging the long-term temporal clues for video classification. The results and comparisons are summarized in Table 1. The upper two groups in the table compare the LSTM models with the two-stream CNN, which performs classification by pooling video-level representations without considering the temporal order of the frames. On UCF-101, the Spatial LSTM is better than the spatial stream CNN, while the result of the Motion LSTM is slightly lower than that of the motion stream CNN. It is interesting to see that, on the spatial stream, the LSTM is even better than the state-of-the-art CNN, indicating that the temporal information is very important for human action modeling, which is fully discarded in the spatial stream CNN. Since the mechanism of the LSTM is totally different, these results are fairly appealing because it is potentially very complementary to the video-level classification based on feature pooling.

On the CCV dataset, the LSTM models produce lower performance than the CNN models on both streams. The reasons are two-fold. First, since the average duration of the CCV videos (80 seconds) is around 10 times longer than that of the UCF-101 and the contents in CCV are more complex and noisy, the LSTM might be affected by the noisy video segments that are irrelevant to the major class of a video. Second, some classes like "wedding reception" and "beach" do not contain clear temporal order information (see Figure 4), for which the LSTM can hardly capture helpful clues.

We now assess the performance of combining the LSTM and the CNN models to study whether they are complementary, by fusing the outputs of the two types of models trained on the two streams. Note that the fusion method adopted here is the simple late average fusion, which uses the average prediction scores of different models. More advanced fusion methods will be evaluated in the next subsection.

Results are reported in the bottom three rows of Table 1. We observe significant improvements from model fusion on

	UCF-101	CCV
Spatial CNN	80.1%	75.0%
Spatial LSTM	83.3%	43.3%
Motion CNN	77.5%	58.9%
Motion LSTM	76.6%	54.7%
CNN + LSTM (Spatial)	84.0%	77.9%
CNN + LSTM (Motion)	81.4%	70.9%
CNN + LSTM (Spatial & Motion)	**90.1%**	**81.7%**

Table 1: Performance of the LSTM and the CNN models trained with the spatial and the short-term motion features respectively on UCF-101 and CCV. "+" indicates model fusion, which simply uses the average prediction scores of different models.

	UCF-101	CCV
Spatial SVM	78.6%	74.4%
Motion SVM	78.2%	57.9%
SVM-EF	86.6%	75.3%
SVM-LF	85.3%	74.9%
SVM-MKL	86.8%	75.4%
NN-EF	86.5%	75.6%
NN-LF	85.1%	75.2%
M-DBM	86.9%	75.3%
Two-stream CNN	86.2%	75.8%
RDNN	88.1%	75.9%
Non-regularized Fusion Network	87.0%	75.4%
Regularized Fusion Network	**88.4%**	**76.2%**

Table 2: Performance comparison on UCF-101 and CCV, using various fusion approaches to combine the spatial and the short-term motion clues.

both datasets. On UCF-101, the fusion leads to an absolute performance gain of around 1% compared with the best single model for the spatial stream, and a gain of 4% for the motion stream. On CCV, the improvements are more significant, especially for the motion stream where an absolute gain of 12% is observed. These results confirm the fact that the long-term temporal clues are highly complementary to the spatial and the short-term motion features. In addition, the fusion of all the CNN and the LSTM models trained on the two streams attains the highest performance on both datasets: 90.1% and 81.7% on UCF-101 and CCV respectively, showing that the spatial and the short-term motion features are also very complementary. Therefore, it is important to incorporate all of them into a successful video classification system.

4.2.2 *Feature Fusion*

Next, we compare our regularized feature fusion network with the alternative fusion methods, using both the spatial and the motion CNN features. The results are presented in Table 2, which are divided into four groups. The first group reports the performance of individual features extracted from the first fully connected layer of the CNN models, classified by SVM classifiers. This is reported to study the gain from the SVM based fusion methods, as shown in the second group of results. The individual feature results using the CNN, i.e., the Spatial CNN and the Motion CNN, are already reported in Table 1. The third group of results in Table 2 are based on the alternative neural network fusion methods. Note that NN-EF and NN-LF take the features from the CNN models and perform fusion and classification using separate neural networks, while the two-stream CNN approach performs classification using the CNN directly with a late score fusion (Figure 2). Finally, the last group contains the results of the proposed fusion network.

As can be seen, the SVM based fusion methods can greatly improve the results on UCF-101. On CCV, the gain is consistent but not very significant, indicating that the short-term motion is more important for modeling the human actions with clearer motion patterns and less noises. SVM-MKL is only slightly better than the simple early and late fusion methods, which is consistent with observations in recent works on visual recognition [42].

Our proposed regularized feature fusion network (the last row in Table 2) is consistently better than the alternative neural network based fusion methods shown in the third group of the table. In particular, the gap between our results and that of the M-DBM and the two-stream CNN confirms that using regularizations in the fusion process is helpful. Compared with the RDNN, our formulation produces slightly better results but with a much lower complexity as discussed earlier.

In addition, as the proposed formulation contains two structural norms, to directly evaluate the contribution of the norms, we also report a baseline using the same network structure without regularization ("non-regularized fusion network" in the table), which is similar to the M-DBM approach in its design but differs slightly in network structures. We see that adding regularizations in the same network can improve 1.4% on UCF-101 and 0.8% on CCV.

4.2.3 *The Hybrid Framework*

Finally, we discuss the results of the entire hybrid deep learning framework by further combining results from the temporal LSTM and the regularized fusion network. The prediction scores from these networks are fused linearly with weights estimated by cross-validation. As shown in the last row of Table 3, we achieve very strong performance on both datasets: 91.3% on UCF-101 and 83.5% on CCV. The performance of the hybrid framework is clearly better than that of the Spatial LSTM and the Motion LSTM (in Table 1). Compared with the Regularized Fusion Network (in Table 2), adding the long-term temporal modeling in the hybrid framework improves 2.9% on UCF-101 and 7.3% on CCV, which are fairly significant considering the difficulties of the two datasets. In contrast to the fusion result in the last row of Table 1, the gain of the proposed hybrid framework comes from the use of the regularized fusion, which again verifies the effectiveness of our fusion method.

To better understand the contributions of the key components in the hybrid framework, we further report the per-class performance on CCV in Figure 4. We see that, although the performance of the LSTM is clearly lower, fusing it with the video-level predictions by the regularized fusion network can significantly improve the results for almost all the classes. This is a bit surprising because some classes do not seem to require temporal information to be recognized. After checking into some of the videos we find that there could be helpful clues which can be modeled, even for

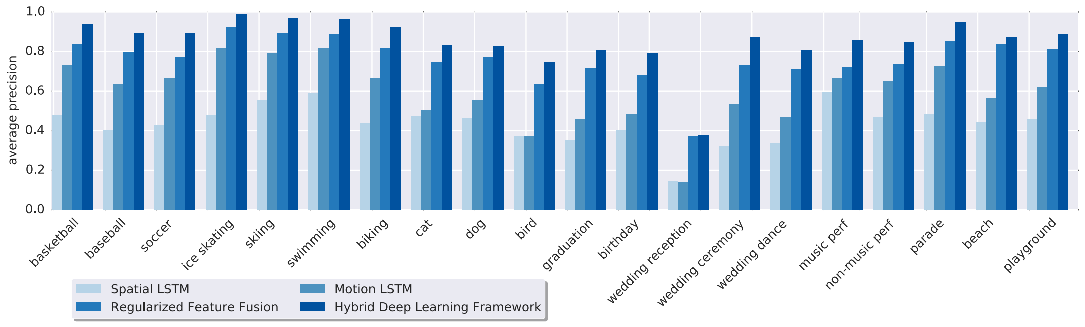


Figure 4: Per-class performance on CCV, using the Spatial and Motion LSTM, the Regularized Fusion Network, and their combination, i.e., the Hybrid Deep Learning Framework.

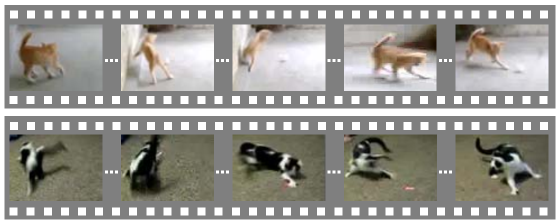


Figure 5: Two example videos of class "cat" in the CCV dataset with similar temporal clues over time.

UCF-101		CCV	
Donahue *et al.* [6]	82.9%	Xu *et al.* [48]	60.3%
Srivastava *et al.* [36]	84.3%	Ye *et al.* [50]	64.0%
Wang *et al.* [44]	85.9%	Jhuo *et al.* [12]	64.0%
Tran *et al.* [40]	86.7%	Ma *et al.* [28]	63.4%
Simonyan *et al.* [33]	88.0%	Liu *et al.* [26]	68.2%
Lan *et al.* [23]	89.1%	Wu *et al.* [47]	70.6%
Zha *et al.* [53]	89.6%	/	
Ours	**91.3%**	Ours	**83.5%**

Table 3: Comparison with state-of-the-art results.

object-related classes like "cat" and "dog". For instance, as shown in Figure 5, we observe that quite a number of "cat" videos contain only a single cat running around on the floor. The LSTM network may be able to capture this clue, which is helpful for classification.

Efficiency. In addition to achieving the superior classification performance, our framework also enjoys high computational efficiency. We summarize the average testing time of a UCF-101 video (duration: 8 seconds) as follows. The extraction of the frames and the optical flows takes 3.9 seconds, and computing the CNN-based spatial and short-term motion features requires 9 seconds. Prediction with the LSTM and the regularized fusion network needs 2.8 seconds. All these are evaluated on a single NVIDIA Telsa K40 GPU.

4.2.4 Comparison with State of the Arts

In this subsection, we compare with several state-of-the-art results. As shown in Table 3, our hybrid deep learning framework produces the highest performance on both datasets. On the UCF-101, many works with competitive results are based on the dense trajectories [44, 53], while our approach fully relies on the deep learning techniques. Compared with the original result of the two-stream CNN in [33], our framework is better with the additional fusion and temporal modeling functions, although it is built on our implementation of the CNN models which are slightly worse than that of [33] (our two-stream CNN result is 86.2%). Note that a gain of even just 1% on the widely adopted UCF-101 dataset is generally considered as a significant progress. In addition, the recent works in [6, 36] also adopted the LSTM to explore the temporal clues for video classification and reported promising performance. However, our LSTM results are not directly comparable as the input features are extracted by different neural networks.

On the CCV dataset, all the recent approaches relied on the joint use of multiple features by developing new fusion methods [48, 50, 12, 28, 26, 47]. Our hybrid deep learning framework is significantly better than all of them.

5. CONCLUSIONS

We have proposed a novel hybrid deep learning framework for video classification, which is able to model static visual features, short-term motion patterns and long-term temporal clues. In the framework, we first extract spatial and motion features with two CNNs trained on static frames and stacked optical flows respectively. The two types of features are used separately as inputs of the LSTM network for long-term temporal modeling. A regularized fusion network is also deployed to combine the two features on video-level for improved classification. Our hybrid deep learning framework integrating both the LSTM and the regularized fusion network produces very impressive performance on two widely adopted benchmark datasets. Results not only verify the effectiveness of the individual components of the framework, but also demonstrate that the frame-level temporal modeling and the video-level fusion and classification are highly complementary, and a big leap of performance can be attained by combining them.

Although deep learning based approaches have been successful in addressing many problems, effective network architectures are urgently needed for modeling sequential data like the videos. Several researchers have recently explored this direction. However, compared with the progress on image classification, the achieved performance gain on video classification over the traditional hand-crafted features is much less significant. Our work in this paper represents one of the few studies showing very strong results. For future work, further improving the capability of modeling the temporal dimension of videos is of high priority. In addition, audio features which are known to be useful for video classification can be easily incorporated into our framework.

Acknowledgment

This work was supported in part by a National 863 Program (#2014AA015101), a grant from the NSF China (#61201387), a grant from the STCSM, China (#13PJ1400400), and the EU FP7 QUICK project under Grant Agreement No. PIRSES-GA-2013-612652.

6. REFERENCES

[1] R. Arandjelovic and A. Zisserman. Three things everyone should know to improve object retrieval. In *CVPR*, 2012.

[2] F. R. Bach, G. R. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *ICML*, 2004.

[3] Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural Networks: Tricks of the Trade*. Springer, 2012.

[4] J. Chung, Ç. Gülçehre, K. Cho, and Y. Bengio. Gated feedback recurrent neural networks. *CoRR*, 2015.

[5] G. E. Dahl, D. Yu, L. Deng, and A. Acero. Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition. *IEEE TASLP*, 2012.

[6] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, K. Saenko, and T. Darrell. Long-term recurrent convolutional networks for visual recognition and description. *CoRR*, 2014.

[7] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[8] A. Graves, A. Mohamed, and G. E. Hinton. Speech recognition with deep recurrent neural networks. In *ICASSP*, 2013.

[9] A. Graves and J. Schmidhuber. Framewise phoneme classification with bidirectional lstm and other neural network architectures. *Neural Networks*, 2005.

[10] C. Harris and M. J. Stephens. A combined corner and edge detector. In *Alvey Vision Conference*, 1988.

[11] M. Jain, J. van Gemert, and C. G. M. Snoek. University of amsterdam at thumos challenge 2014. In *ECCV THUMOS Challenge Workshop*, 2014.

[12] I.-H. Jhuo, G. Ye, S. Gao, D. Liu, Y.-G. Jiang, D. T. Lee, and S.-F. Chang. Discovering joint audio-visual codewords for video event detection. *Machine Vision and Applications*, 2014.

[13] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. In *ICML*, 2010.

[14] Y. Jia, E. Shelhamer, J. Donahue, S. Karayev, J. Long, R. Girshick, S. Guadarrama, and T. Darrell. Caffe: Convolutional architecture for fast feature embedding. In *ACM Multimedia*, 2014.

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

[16] Y.-G. Jiang, S. Bhattacharya, S.-F. Chang, and M. Shah. High-level event recognition in unconstrained videos. *IJMIR*, 2013.

[17] Y.-G. Jiang, J. Liu, A. Roshan Zamir, G. Toderici, I. Laptev, M. Shah, and R. Sukthankar. THUMOS challenge: Action recognition with a large number of classes. `http://crcv.ucf.edu/THUMOS14/`, 2014.

[18] Y.-G. Jiang, G. Ye, S.-F. Chang, D. Ellis, and A. C. Loui. Consumer video understanding: A benchmark database and an evaluation of human and machine performance. In *ICMR*, 2011.

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

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

[21] M. Kloft, U. Brefeld, S. Sonnenburg, and A. Zien. Lp-norm multiple kernel learning. *The Journal of Machine Learning Research*, 2011.

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

[23] Z. Lan, M. Lin, X. Li, A. G. Hauptmann, and B. Raj. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. *CoRR*, 2014.

[24] I. Laptev. On space-time interest points. *IJCV*, 2007.

[25] W. Li, Z. Zhang, and Z. Liu. Expandable data-driven graphical modeling of human actions based on salient postures. *IEEE TCSVT*, 2008.

[26] D. Liu, K.-T. Lai, G. Ye, M.-S. Chen, and S.-F. Chang. Sample-specific late fusion for visual category recognition. In *CVPR*, 2013.

[27] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *IJCV*, 2004.

[28] A. J. Ma and P. C. Yuen. Reduced analytic dependency modeling: Robust fusion for visual recognition. *IJCV*, 2014.

[29] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Ng. Multimodal deep learning. In *ICML*, 2011.

[30] D. Oneata, J. Verbeek, C. Schmid, et al. Action and event recognition with fisher vectors on a compact feature set. In *ICCV*, 2013.

[31] M. Ranzato, A. Szlam, J. Bruna, M. Mathieu, R. Collobert, and S. Chopra. Video (language) modeling: a baseline for generative models of natural videos. *CoRR*, 2014.

[32] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. CNN features off-the-shelf: an astounding baseline for recognition. *CoRR*, 2014.

[33] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *NIPS*, 2014.

[34] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, 2014.

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

[36] N. Srivastava, E. Mansimov, and R. Salakhutdinov. Unsupervised learning of video representations using LSTMs. *CoRR*, 2015.

[37] N. Srivastava and R. Salakhutdinov. Multimodal learning with deep boltzmann machines. In *NIPS*, 2012.

[38] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. *CoRR*, 2014.

[39] K. Tang, L. Fei-Fei, and D. Koller. Learning latent temporal structure for complex event detection. In *CVPR*, 2012.

[40] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. C3d: Generic features for video analysis. *CoRR*, 2014.

[41] D. L. Vail, M. M. Veloso, and J. D. Lafferty. Conditional random fields for activity recognition. In *AAAMS*, 2007.

[42] A. Vedaldi, V. Gulshan, M. Varma, and A. Zisserman. Multiple kernels for object detection. In *ICCV*, 2009.

[43] S. Venugopalan, H. Xu, J. Donahue, M. Rohrbach, R. J. Mooney, and K. Saenko. Translating videos to natural language using deep recurrent neural networks. *CoRR*, 2014.

[44] H. Wang and C. Schmid. Action recognition with improved trajectories. In *ICCV*, 2013.

[45] H. Wang, M. M. Ullah, A. Klaser, I. Laptev, and C. Schmid. Evaluation of local spatio-temporal features for action recognition. In *BMVC*, 2009.

[46] Y. Wang and G. Mori. Max-margin hidden conditional random fields for human action recognition. In *CVPR*, 2009.

[47] Z. Wu, Y.-G. Jiang, J. Wang, J. Pu, and X. Xue. Exploring inter-feature and inter-class relationships with deep neural networks for video classification. In *ACM Multimedia*, 2014.

[48] Z. Xu, Y. Yang, I. Tsang, N. Sebe, and A. Hauptmann. Feature weighting via optimal thresholding for video analysis. In *ICCV*, 2013.

[49] H. Yang, M. R. Lyu, and I. King. Efficient online learning for multitask feature selection. *ACM SIGKDD*, 2013.

[50] G. Ye, D. Liu, I.-H. Jhuo, and S.-F. Chang. Robust late fusion with rank minimization. In *CVPR*, 2012.

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

[52] Z. Zeng and Q. Ji. Knowledge based activity recognition with dynamic bayesian network. In *ECCV*, 2010.

[53] S. Zha, F. Luisier, W. Andrews, N. Srivastava, and R. Salakhutdinov. Exploiting image-trained cnn architectures for unconstrained video classification. *CoRR*, 2015.

[54] J. Zhang, M. Marszałek, S. Lazebnik, and C. Schmid. Local features and kernels for classification of texture and object categories: A comprehensive study. *IJCV*, 2007.