Temporal Difference Networks for Video Action Recognition

Joe Yue-Hei Ng   Larry S. Davis
University of Maryland, College Park
{yhng,lsd}@umiacs.umd.edu

Abstract

Deep convolutional neural networks have been great success for image based recognition tasks. However, it is still unclear how to model the temporal evolution of videos effectively by deep networks. While recent deep models for videos show improvement by incorporating optical flow or aggregating high level appearance across frames, they focus on modeling either the long term temporal relations or short term motion. We propose Temporal Difference Networks (TDN) that model both long term relations and short term motion from videos. We leverage a simple but effective motion representation: difference of CNN features in our network and jointly modeling the motion at multiple scales in a single CNN. It achieves state-of-the-art performance on three different video classification benchmarks, showing the effectiveness of our approach to learn temporal relations in videos.

1. Introduction

Video action recognition is one of the fundamental problems in computer vision. One of the main challenges is to model the complex temporal relations in addition to image appearance. Unlike images, which have a fixed size input, videos and their corresponding actions are of arbitrary length. While convolutional neural networks (CNNs) have been very sucessful in image based recognition tasks [17, 42, 30, 11], it is still unclear how to model the temporal evolution of videos effectively by deep networks. In recent work, there are two main approaches to model the temporal dimension of videos - model the short term motion and model the longer term temporal relations.

Motion modeling usually focuses on video clips that span less than one second. Optical flow has been used extensively [24, 7, 6, 36, 37, 20] as a motion representation and shows improvement in conjunction with RGB image inputs for action recognition. While it is an effective hand-crafted feature as input for CNNs to recognize action, it is still unclear how to learn motion features directly from raw pixels without hand-crafted inputs. Furthermore, optical flow, which is defined at the pixel level, can only model short term motion effectively, but does not capture longer range and high level temporal dependency.

Another line of work models the high level representation of frames from the output of deep CNNs by models such as Long-short-term memory (LSTM). These approaches rely on the strong high level CNN features but do not consider the low level correspondence between frames. While they can sucessfully combine semantic information over long videos, we believe that CNN features capture high level abstract concepts and are unsuitable for learning motion, as the apperance of consecutive frames in a video might be very similar and the precise spatial details may be lost after multiple pooling layers in the network.

We believe successful video action recognition models require temporal reasoning on multiple levels of appearance. Current approaches, which focus on modeling either the long term temporal relations or short term motion, are insufficient as they model on a fixed level of appearance. In this work, we propose a novel deep network architecture - *Temporal Difference Network (TDN)* - to model temporal relations in videos. Instead of leveraging independent techniques for modeling short term motion [24] and high level temporal relations [20, 37, 36], our framework unifies these two strategies and learns video motion representations from multiple levels of apperance abstraction, leveraging low level to high level image features.

While pixel level motion can be represented using optical flow, the motion of mid-level concepts is not well-defined. We consider an alternative representation of motion - *Eulerian motion* - which is defined in terms of image differences. In addition to differences of raw input images, we consider the Eulerian motion of mid-level to high-level image features, and then combine the multiple layers of motion information in a single CNN. By forcing the network to model the motion directly instead of implicitly learn from class label supervision, our network effectively models the temporal relations between frames rather than just aggregates apperance information over a video.

We test our model on three public video classification benchmarks and achieve state-of-the-art results. Remarkably, the improvement in the RGB stream is significant, demonstrating the effectiveness of modeling temporal relations in our network from raw pixels.

2. Related Work

Video action recognition has been studied extensively in computer vision. Please refer to the survey by Poppe [23] and Wu *et al*. [39] for complete background. Recent work on action recognition falls into two main categories: 1) long range temporal relations modeling and 2) short term motion representation.

Long range temporal relations modeling. Recurrent neural networks have been used to model the temporal relations in video sequences [20, 5, 19, 27, 1]. Ng *et al*. learned long range temporal information in videos by LSTMs for long video classification [20]. Donahue *et al*. similarly learns LSTM models for action recognition. Mahasseni and Todorovic regularized the LSTM model with 3D human-skeleton sequences [19]. Srivastava *et al*. learns LSTM in an unsupervised manner for action recognition. Another strategy to aggregate information across frames is pooling. Ng *et al*. applied max pooling at the last convolutional layer for long video classification [20]. Wang *et al*. proposed temporal segment networks to combine the final categorical classification scores from multiple frames by average pooling [36]. Wang *et al*. represent actions by a transformation from the initial state to final state of videos and treat the temporal location of the states as latent variables [37]. Rank pooling has been proposed to capture temporal evolution of videos by considering the temporal ordering of video frames [9, 10, 8].

All these methods learn temporal relations only based on high level CNN features, i.e. the last convolutional layer or fully connected layers, which lose detailed spatial information and do not effecitvely learn small motions. In contrast, our work operates over multiple level of appearance within a single network, and learns both small motions and high level temporal relations.

Ballas *et al*. exploit multiple layers of image features to train GRU-RNN for video representations [1]. Instead of using recurrent networks, we train a feed-forward network to directly model the motion between frames.

Many previous research leverages the temporal structure in videos for action recognition [21, 31, 22, 28, 12]. Niebles *et al*. uses latent SVM to discover the temporal structure of videos [21]. Tang *et al*. model the temporal structure using a varient of HMM [31]. Pirsiavash and Ramanan represent actions by segmental grammars [22]. However, their approaches are not end-to-end learnable for temporal structure modeling and thus cannot fully utilize the advantage of deep networks.

Short term motion representation. Kaparthy *et al*. trained a “Slow Fusion” network to learn motion from large number of labeled videos [15]. Ji *et al*. and Tran *et al*. use multiple 3D convolutional layers for learning motion features from raw pixels [13, 32]. Varol *et al*. train 3D convolutional networks for longer clips and show improvements in recognition. Carreira and Zisserman recently trained 3D convolutional networks [3] on newly released Kinetics dataset [16]. However, training these models requires large labeled video datasets and is extremely computationally expensive, which limits the size of the network that could be trained. Our work reuses the pretrained static image appearance model and learns motion based on that, which could easily adapt to very deep image based convolutional networks.

Bilen *et al*. computed dynamic image by rank pooling as motion representation as input to the network [2]. Our Temporal Difference Networks learn motion from multiple levels and is an end-to-end model.

Simonyan and Zisserman feed stacked optical flow frames as input to the network and showed that combining optical flow network and ImageNet pretrained network significantly improves action recognition performance over the single frame appearance model [24]. We also train networks for both RGB and optical flow frames as input, but in addition to modeling short term motion, our network learns higher level temporal relations as well. Feichtenhofer *et al*. further improves the original two stream networks by combining two input modalities into a single network [7, 6]. Our network architecture is similar to [6], but we exploit the difference of image features to learn temporal relations instead of using optical flow as input for motion.

Image difference, also known as the Eulerian motion, has been used to represent motion of images. Sun *et al*. and Wang *et al*. exploited image differences as inputs to the network [29, 36] to complement RGB inputs. Xue *et al*. represented motion by Eulerican motion to synthesize future video frames from a single image [40]. Villegas *et al*. decompose videos into motion and content by representing motion as image differences for future video sequence prediction [34]. Wu *et al*. magnify the Eulerian motion of videos to visualize subtle change in videos [38]. Extending these previous work, we compute the Eulerian motion not only of the input frames, but also the intermediate CNN features to capture high level motion.

3. Approach

3.1. Eulerian Motion of Features

The image difference, also known as the Eulerian motion, of two images is defined as:

$$v = I_2 - I_1$$

where I_1 and I_2 are consecutive frames in a video. While image differences capture some short term motion information, they do not effectively model longer range temporal relation in videos.

Instead, we encode motion additionally by differences of image features, which can be regarded as the Eulerian motion of image features:

$$v^{(\ell)} = f^{(\ell)}(I_2) - f^{(\ell)}(I_1)$$

where $f^{(\ell)}(I)$ is an appearance feature extractor for image I at layer ℓ in a CNN. Compared to raw images, the image features are more robust to translation and appearance changes, and thus their differences are more suitable for capturing higher level temporal relations across longer periods of time. The difference of features can also be seen as a special case of rank pooling [9], where only two frames are considered instead of multiple frames in the video. Similarly, image difference can be seen as a special case of dynamic image [2] from two frames.

In a convolutional neural network (CNN), different layers capture different levels of appearance abstraction [42]. For shorter time periods, the difference of lower level features should be more informative as small motion can be captured better in lower layers; and for longer time periods, higher level features should be more useful to model the temporal relations between frames. While the best level of abstraction is unclear and situation dependent, we use the differences over multiple layers in the CNN features to capture temporal relations over all scales.

3.2. Temporal Difference Network

In this section, we describe the architecture of our proposed Temporal Difference Network (TDN).

Jointly learning motion and appearance on video action recognition datasets is challenging, since image appearance provides abundant information to the model to overfit the dataset, while ignoring motion which corresponds to the actual action. While separating the motion into an independent optical flow CNN shows great improvement to the RGB inputs [24], motions, however, clearly depend on appearanace which suggests appearance model should help learning motion. Therefore, in our proposed Temporal Difference Network, we leverage the image appearance models to learn video motions, and force the network to learn the motion by explicitly model the Eulerian motion of image features as inputs. In addition to modeling motion in a fixed layer, we aggregate the multi-level feature differences in one single network.

We build the Temporal Difference Network on a well trained image appearance model. We use a 50 layers Residual Network (ResNet-50) as our base model [11], which provides a good trade-off between accuracy and training time. Our approach is flexible to adopt other architectures and further recognition improvement is possible with deeper and more accurate pretrained networks. The TDN consists of two subnetworks: the image subnetwork and the difference subnetwork. The architecture is illustrated in Figure 1.

Image subnetwork. The image subnetwork is a standard residual network which takes a single frame (or stacked consecutive frames) as input. At the end of the network, the prediction scores from different frames are averaged before softmax similar to [36]. The parameters of the image subnetwork of different input frames are shared. This is essential to force the network to learn from the feature differences instead of using the appearance from one of the image subnetworks.

Difference subnetwork. The difference subnetwork has the exact same size as the image subnetwork. At the layers right before reducing the spatial resolution, we compute the Eulerian motion of the features from the image subnetwork, which is the difference between two feature maps. The differences are then combined with the bottom up activations from the difference subnetwork by element-wise addition, inspired by [6], as the difference of features and the CNN features have the exact same dimensions. Specifically, in our implementation the difference features have spatial resolution of 224×224 (input images), 56×56, 28×28 and 14×14. More layers in the network can be used for computing the image feature differences. The difference in the last convolutional layer is not used because it would be immediately fed into the final classifier without additional transformation, which makes it unlikely to help classification.

While the difference in the features is basically a linear operation and could be learned by 3D convolutions, which has been explored previously in [32, 20], it is not an effective method to model the motion between frames since the frame based image appearance already provides extremely strong cues to the classification task, and the model can simply learn to aggregate the appearance context for classification instead of learning the action itself. By explicitly taking the difference of the features, the network is forced to focus on the motion and temporal evolution of the videos, instead of relying on strong appearance cues for classification.

We employ a simple summation to combine the bottom up activations of the difference subnetwork and the differences of image features. We have additionally experimented with adding a 1×1 convolutional layer to the difference features before the addition, or dynamicly computing the weights of two inputs by a gating mechanism, but observed no significant difference in performance.

By taking the difference of higher level features, we are able to not only model the motion between consecutive frames, but also over longer time periods. Since the mid-level features are more robust to translation and view point changes, the model can then focus on the difference in mid-level concepts like pose. Therefore, we sparsely sample frames throughout the whole video as input, instead of

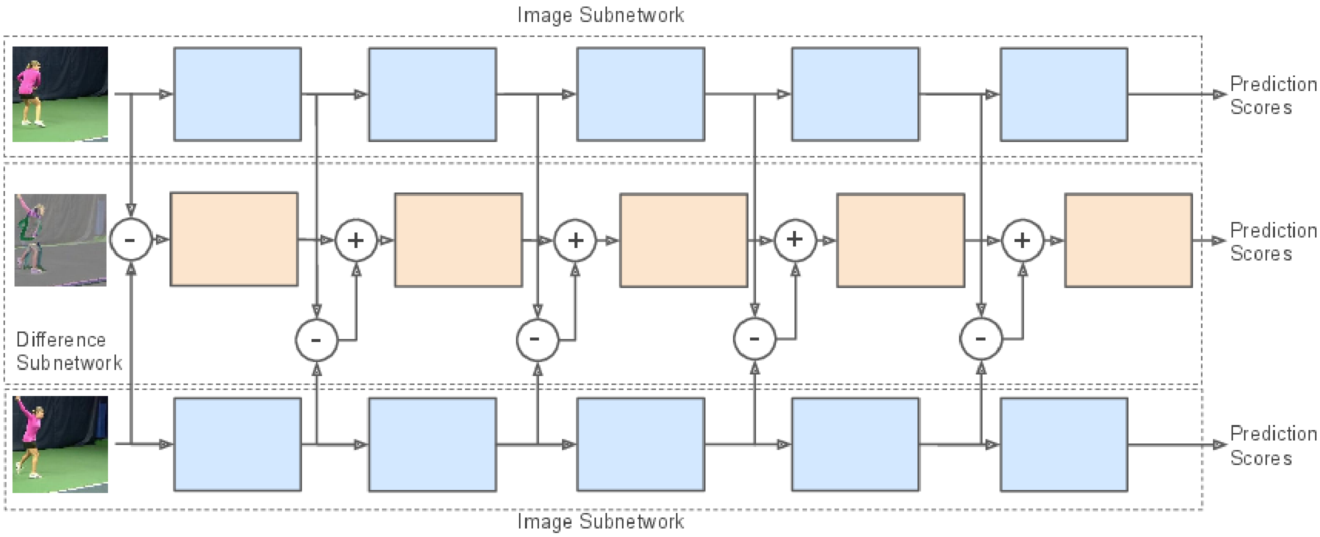


Figure 1: The figure shows our Temporal Difference Network architecture. Each rectangle in the figure represents the convolutional layers with max pooling or stacked residual modules. The blue blocks represent layers in the image subnetwork and the orange blocks represent layers in the difference subnetwork. The circles represent element-wise add or subtraction operation. At the layers before reducing the spatial resolution, the difference of the convolutional feature maps from the image subnetwork is computed and then added into the difference subnetwork. The class prediction scores are obtained at the end of each subnetwork.

only considering consecutive frames.

Final prediction. There are many ways to combine the final classification outputs of the image subnetwork and difference subnetwork during testing. We found that simple averaging works very well and this is used in all experiments.

TDN with TSN. Our TDN can be trained with more than two frames as a temporal segment network (TSN) [36]. As a TSN with s input snippets, our TDN produces s outputs from image subnetworks for each frame, and $s-1$ outputs from difference subnetworks for each pair of adjacent frames. Following [36], we use average pooling as the segmental consensus function to combine the prediction scores of each network during training. Figure 2 illustrates an example with three input frames ($s = 3$). By taking more frames, the network captures more context in the video during training before classification. This is important especially for training the difference subnetwork, since the difference of only two video frames may not have enough information for recognizing actions where TSN reduce the noise in training by considering more frames at once.

3.3. Fusion from Multiple Modalities

Optical flow has shown improvements in conjunction with RGB inputs for video action recognition performance. Our model can be applied to different modalities including RGB and optical flow, although we expect more improvements from the RGB network as optical flow already encodes motion. Long term temporal relations, which are not encoded in optical flow, can be learned through our framework. Following previous work [36], we separately train two networks for RGB inputs as spatial stream and stacked optical flow inputs as temporal stream. At test time, we compute the weighted average of prediction scores with RGB:Flow as 1:1.5. We use the confidence scores before softmax for fusion after ℓ_1-normalization.

3.4. Training

During training, we randomly sample frames throughout the videos as in temporal segment network [36]. The input videos are split into s approximately equally sized segments, and k consecutive frames ($k = 1$ for RGB input and $k = 5$ for optical flow) are randomly picked from each segments as an input snippet.

Cross Modality Pretraining. Following [36, 20], which suggests initializing the optical flow network with image pretrained model, we initialize the weight of convolutional layers in the difference subnetwork with ImageNet pretrained model. Since the inputs to the difference subnetwork, which are the differences of image features, contain similar spatial structure to the image features, ImageNet pretrained models should possess useful representation for modeling the motion and thus help the training by initialization.

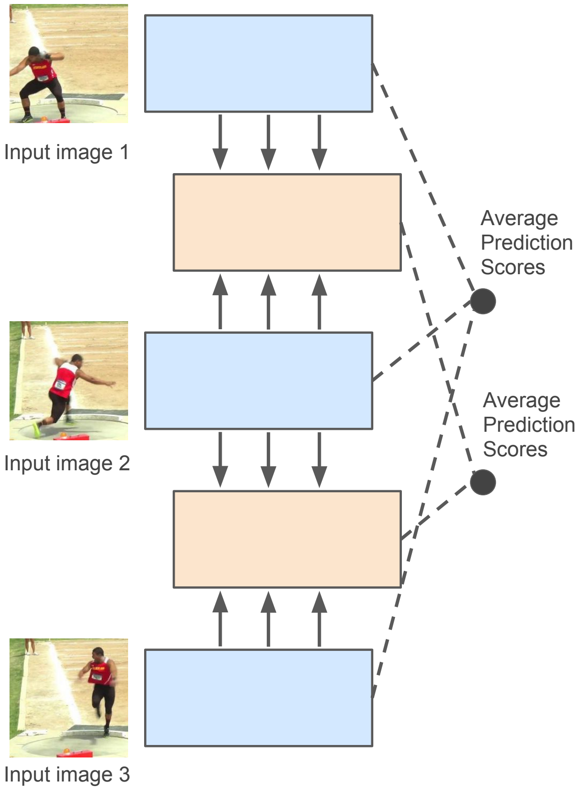


Figure 2: Temporal Difference Network as temporal segment network with $s = 3$ input snippets. The blue and orange blocks represent the image subnetworks and difference subnetworks respectively. The class prediction scores of image subnetworks and difference subnetworks are averaged across frames independently.

Two phase Training. To improve training stability, we employ a two phase training strategy. We first fix the image subnetwork and only train the classifier in the image subnetwork and the entire difference subnetwork. After convergence, we jointly finetune both image subnetwork and difference subnetwork.

When using optical flow inputs, we initialize the "image" subnetwork with a pretrained temporal segment network on optical flow inputs, and the difference subnetwork is still initialized with ImageNet pretrained weights. We similarly fix the already trained "image" subnetwork and train the difference subnetwork first, and then finetune the entire network.

4. Experiments

4.1. Datasets

We test our Temporal Difference Networks on three video datasets: HMDB51, ACT and FCVID.

The HMDB51 dataset [18] contains 6,766 video clips taken from mostly movies with 51 action classes. The standard three splits averaged accuracy is reported. This is a challenging action recognition dataset since many action classes, for example "turn" and "kiss", cannot be recognized from a static frame or backgroud context.

The ACT dataset [37] contains 11,234 video clips with 43 action classes. The videos are donwloaded from the web and then labeled by a commercial crowdsourcing organization. We evalute our model on the first task for the dataset, which is the standard action classification proposed in [37]. We follow the train and test split by the dataset authors with 7,260 training videos and 3,974 for testing.

The Fudan-Columbia Video Dataset (FCVID) [14] contains 91,223 web videos with 239 categories including social events, procedural events, objects and scenes. The average video length is 167 seconds. It is computationally very expensive to process such a large dataset, so the frames are sampled every three seconds and resized to 256 × 256. We evaluate our models using the same train and test split as [14], which contains approximately half of the videos for training and half for testing, and compute the mean average precision (mAP) across categories.

We do not test on the UCF101 dataset [26] as it heavily relies on appearance and context information, and the performance has already been saturated with more than 94% accuracy by [36, 6] and 98% with pretraining on the Kinetics dataset [3, 16].

4.2. Implementation

We implement our network with Torch7 [4] using multiple GPUs for data-parallelism. SGD is used for training with mini-batch size 128 and momentum set to 0.9. To regularize the network on small datasets, we follow the good practice from [36] and use high dropout rate (drop probability 0.8), corner cropping, scale jittering and partial-BN, which fixes the mean and variance in batch normalization layers except the first one, to train the model on the HMDB51 and ACT datasets. We also apply weight decay with rate 0.0005 and color jittering [30] for data augmentation.

We roughly follow [36] for numbers of training iterations and learning rate decay schedules to train the network for the HMDB51 dataset. As the size of ACT is similar to UCF101, we adopt their settings for training on ACT. For FCVID, we first train the TSN for 30,000 iterations with initial learning rate 0.01, and divide the learning rate by a factor of 10 every 10,000 iterations. The TDN is trained with the same settings as TSN, and we jointly fine-tune the whole network for another 10,000 iterations. We use $s = 3$ input snippets for ACT and FCVID, and $s = 2$ for HMDB51.

For the temporal stream networks, we compute the optical flow with the TV-L1 algorithm [41] using the OpenCV implementation with CUDA and save the flow as images after discretized into $[0, 255]$ range following [24]. We sample 5 consecutive optical flow frames as input to the flow network following [36].

Druing testing, we randomly sample 25 clips and perform 10 crop data augmentation, which crop the 4 corners and 1 center with their horizontal reflections, and compute the average of the prediction scores as final prediction.

4.3. Results

We present the results of our Temporal Difference Networks on various datasets. For fair comparisons, we train ResNet-50 temporal segmental networks [36] in all three evaluation datasets as our baselines.

4.3.1 HMDB51 Dataset

We compare our TDN with previous two-stream based networks including [24, 29, 37, 36, 7, 6, 35, 44, 33] that uses ImageNet as the only external dataset. The 3 splits averaged accuracy of the models are shown in Table 1. We exclude the fusion results with extra input modality like warped optical flow fields or hand-crafted features like improved dense trajectories for fair comparison. We compare to methods that only uses ImageNet as external datasets. Since different based networks are used in previous work including VGG, BN-Inception and ResNet-50, we additionally trained temporal segmental networks with ResNet-50 as baseline for comparison. We would also want to note that the recent work ST-ResNet [6] is also based on ResNet-50 and therefore can be directly compared.

Our implementation of TSN with ResNet-50 is better than the previous state-of-the-art in [36], which verifies the strength of our baseline. Our Temporal Difference Networks further improve over TSNs on both RGB and optical flow streams. Remarkably, our model significantly improves the RGB network by 4.4%. The improvement on the temporal stream with optical flow inputs is marginal, which is reasonable since the stacked optical flow already encodes motion information. Overall, our TDN achieves better accuracy than previous work with improvement of 1.9% (70.4% vs 68.5%).

Recently, Carreira and Zisserman trained I3D models [3] on newly released Kinetics dataset [16]. While they obtained strong recognition performance, their models have access to external video data thus could not be directly compared. Since training on Kinetics dataset is extermely computationally expensive, we leave the experiments with Kinetics dataset as future work.

Method	RGB	Flow	Fusion
Two Stream [24]	40.5	54.6	59.4
Two Stream (VGG) [25, 37]	42.2	55.0	58.5
F$_{\text{ST}}$CN (SCI fusion) [29]	-	-	59.1
Actions $\sim$ Transformation [37]	44.1	57.1	62.0
TDD + FV [35]	50.0	54.9	63.2
Key Volumne Mining [44]	-	-	63.3
LTC [33]	-	59.0	64.8
Two-stream Fusion [7]	-	-	65.4
ST-ResNet [6]	-	-	66.4
BN-Inception TSN [36]	51.0	64.2	68.5
ResNet-50 TSN	51.1	64.6	69.6
ResNet-50 TDN (ours)	**55.5**	**64.8**	**70.4**

Table 1: Classification accuracies on HMDB51 (3 splits average). Our model achieves state-of-the-art accuracy on the HMDB51 dataset. In particular, the RGB model significantly improves the TSN baseline.

4.3.2 ACT Dataset

We test our models on ACT and observe similar improvement. The evaluation results are shown in Table 2. Our ResNet-50 TSN baseline is the better than previous two-stream based networks with precondition and effect modeling by [37], and our TDNs again outperform the TSN baseline. The improvement in RGB stream is especially significant (75.9% vs 72.0%), showing that our model is capable of learning motion from the difference subnetwork. Overall, our TDNs improve on both RGB and optical flow inputs, and give slight improvement to TSN after fusion. Our model performs significantly better than previous work by a large margin (85.1% vs 80.6%).

Method	RGB	Flow	Fusion
Two Stream	66.8	71.4	78.7
LSTM + Two Stream	68.7	72.1	78.6
Actions $\sim$ Transformation [37]	69.5	73.7	80.6
ResNet-50 TSN	72.0	76.1	84.3
ResNet-50 TDN (ours)	**75.9**	**77.0**	**85.1**

Table 2: Performance comparison for the first task on ACT dataset. All baselines are trained by [37] with VGG-16 [25]. Our models significantly outperform previous methods.

4.3.3 FCVID

We compare our models to previous results reported by [14]. They provide strong baselines by combining static CNN features, improved dense trajectories and audio features with various fusion techniques. In particular, they proposed rDNN to exploit features and class relationships with deep networks to combine the predictions from multiple modalities.

As the dataset is very large and computing optical flow for the whole dataset is very time consuming, we only train our network for raw image inputs. We lower the weight decay rate to 0.0001 and do not use regularization techniques like dropout, corner cropping and partial-BN suggested in [36] since the dataset is already very large.

The recognition results are summarized in Table 3. The TSN baseline alone is already much better than the previous state-of-the-art rDNN from [14], which fused multiple features, by 5.8% in mAP. We believe the difference in performance should be due to: 1) the advancement in CNN architecture (ResNet vs AlexNet); 2) end-to-end finetuning rather than feature extraction in [14]; and 3) training as TSN rather than single frame network.

Our TDN further improves the TSN model significantly by 1.9%, which demonstrates that our network is able to learn temporal relationships effectively even on large scale settings with unconstrained and noisy videos.

Method	mAP (%)
Static CNN [14]	63.8
rDNN (Static CNN + Motion + Audio) [14]	76.0
ResNet-50 TSN	81.8
ResNet-50 TDN (ours)	**83.7**

Table 3: Performance comparison on the FCVID. Our TDN substantially improves on the strong TSN baseline, and significantly outperforms previous work.

We compare the class-wise average precision of TSN and TDN. The top 5 improving classes from our TDN are paperCutting (+14%), dumbbellWorkout (+13%), makingIceCream (+10%), pushUps (+10%) and makingHotdog (+10%). We can clearly observe that all of them are actions instead of scenes or objects, showing that our model improves action recognition by modeling the temporal relations in videos.

4.4. Influence of Multiple Layers

One natural question to the TDN architecture is whether the multiple layers of feature differences are really helpful in action recognition in addition to image difference. To answer this question, we train TDNs with different settings on the FCVID dataset. We train multiple models by incrementally adding layers of feature differences into the network. As shown in Table 4, incorporating multiple layers of feature differences gradually improves the performance. This shows that our network benefits from the differences in higher level features in addition to image difference, and incorporating motions in multiple layers are indeed important to achieve good classification performance.

4.5. Visualization

We visualize the network outputs to understand what is learned in the Temporal Difference Network. As our base network is a residual network which includes a global average pooling layer before the final linear classifier, we can compute the Class Activation Mapping (CAM) [43] of the image subnetwork and difference subnetwork respectively, by removing the average pooling layer and applying the linear classifier in all spatial locations. We then use bilinear upsampling to enlarge the heatmaps back to the input size 224×224 and overlaid with the input images.

TDN Layers	mAP (%)
`input`	82.7
`input` – `conv1`	82.9
`input` – `res2`	83.3
`input` – `res3`	**83.8**
`input` – `res4`	83.7

Table 4: Effects of incorporating multiple layers of motion. The "`input`" row represents the network only taking difference of the RGB frames, and "`input` – `conv1`" represents the network taking difference of RGB frames and `conv1` outputs into the network and so on. Adding more layers improves recognition performance.

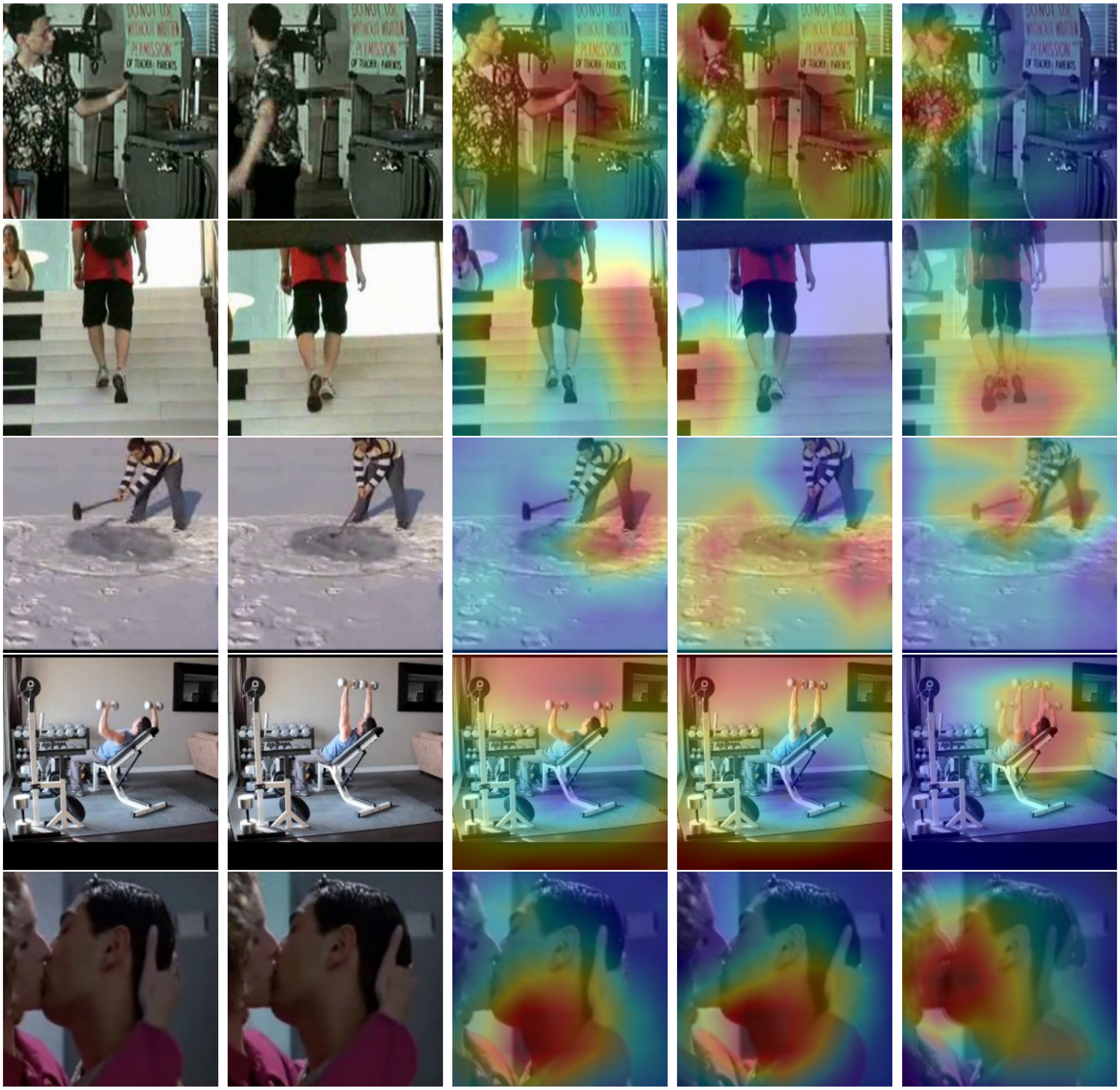
The visualizations are shown in Figure 3. Although the output heatmaps only have 7×7 resolution restricted by the size of the last convolutional layer outputs, we can clearly see what is salient to the network with respect to the action classes. The image subnetwork focuses more on the appearance and the background context, and the difference subnetwork focuses on the motions and actions. This shows that the image subnetwork and the difference subnetwork learn complementary features that help action classification when combined.

5. Conclusion

We present a novel network architecture - Temporal Difference Network - for learning temporal relations from videos for action recognition. Instead of learning temporal relation at a fixed level, we capture the Eulerian motions of image features at multiple levels and combine with a single CNN to jointly model motions in multiple scales. We obtain state-of-the-art performance on three public video action recognition benchmarks, demonstrating the effectiveness of our approach.

Acknowledgments

This research was supported in part by funds provided from the Office of Naval Research under grant N000141612713 entitled "Visual Common Sense Reasoning for Multi-agent Activity Prediction and Recognition".



(a) Input image 1

(b) Input image 2

(c) CAM of image subnetwork on image 1

(d) CAM of image subnetwork on image 2

(e) CAM of difference subnetwork (overlaid on the average of two input images)

Figure 3: Class Activation Mapping (CAM) for the image subnetwork and difference subnetwork. The action classes from top to bottom are: turn, climbing-chairs, hit, lifting-benchpress and kiss. Our difference subnetwork can capture the motion regions effectively while the image subnetwork focuses on the apppearance and background context. Note the camera motion and large movement between frames may make optical flow between two frames ineffective, but our TDN sucessfully learns from the large difference on multiple CNN layers.

References

[1] N. Ballas, L. Yao, C. Pal, and A. Courville. Delving deeper into convolutional networks for learning video representations. *ICLR*, 2016. 2

[2] H. Bilen, B. Fernando, E. Gavves, A. Vedaldi, and S. Gould. Dynamic image networks for action recognition. In *CVPR*, 2016. 2, 3

[3] J. Carreira and A. Zisserman. Quo vadis, action recognition? a new model and the kinetics dataset. In *CVPR*, 2017. 2, 5, 6

[4] R. Collobert, K. Kavukcuoglu, and C. Farabet. Torch7: A matlab-like environment for machine learning. In *BigLearn, NIPS Workshop*, 2011. 5

[5] 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. In *CVPR*, 2015. 2

[6] C. Feichtenhofer, A. Pinz, and R. Wildes. Spatiotemporal residual networks for video action recognition. In *NIPS*, 2016. 1, 2, 3, 5, 6

[7] C. Feichtenhofer, A. Pinz, and A. Zisserman. Convolutional two-stream network fusion for video action recognition. In *CVPR*, 2016. 1, 2, 6

[8] B. Fernando, P. Anderson, M. Hutter, and S. Gould. Discriminative hierarchical rank pooling for activity recognition. In *CVPR*, 2016. 2

[9] B. Fernando, E. Gavves, J. Oramas, A. Ghodrati, and T. Tuytelaars. Rank pooling for action recognition. *IEEE TPAMI*, 2016. 2, 3

[10] B. Fernando and S. Gould. Learning end-to-end video classification with rank-pooling. In *ICML*, 2016. 2

[11] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *CVPR*, 2016. 1, 3

[12] H. Izadinia and M. Shah. Recognizing complex events using large margin joint low-level event model. In *ECCV*, 2012. 2

[13] S. Ji, W. Xu, M. Yang, and K. Yu. 3d convolutional neural networks for human action recognition. *IEEE TPAMI*, 35(1):221–231, 2013. 2

[14] Y.-G. Jiang, Z. Wu, J. Wang, X. Xue, and S.-F. Chang. Exploiting feature and class relationships in video categorization with regularized deep neural networks. *arXiv preprint arXiv:1502.07209*, 2015. 5, 6, 7

[15] 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. 2

[16] W. Kay, J. Carreira, K. Simonyan, B. Zhang, C. Hillier, S. Vijayanarasimhan, F. Viola, T. Green, T. Back, P. Natsev, et al. The kinetics human action video dataset. *arXiv preprint arXiv:1705.06950*, 2017. 2, 5, 6

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *NIPS*, 2012. 1

[18] H. Kuehne, H. Jhuang, E. Garrote, T. Poggio, and T. Serre. HMDB: A large video database for human motion recognition. In *ICCV*, 2011. 5

[19] B. Mahasseni and S. Todorovic. Regularizing long short term memory with 3d human-skeleton sequences for action recognition. In *CVPR*, 2016. 2

[20] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici. Beyond Short Snippets: Deep Networks for Video Classification. In *CVPR*, 2015. 1, 2, 3, 4

[21] J. C. Niebles, C.-W. Chen, and L. Fei-Fei. Modeling temporal structure of decomposable motion segments for activity classification. In *ECCV*, 2010. 2

[22] H. Pirsiavash and D. Ramanan. Parsing videos of actions with segmental grammars. In *CVPR*, 2014. 2

[23] R. Poppe. A survey on vision-based human action recognition. *Image and vision computing*, 28(6):976–990, 2010. 2

[24] K. Simonyan and A. Zisserman. Two-Stream Convolutional Networks for Action Recognition in Videos. In *NIPS*, 2014. 1, 2, 3, 6

[25] K. Simonyan and A. Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition. In *ICLR*, 2014. 6

[26] K. Soomro, A. R. Zamir, and M. Shah. UCF101: A Dataset of 101 Human Action Classes From Videos in The Wild. In *CRCV-TR-12-01*, 2012. 5

[27] N. Srivastava, E. Mansimov, and R. Salakhudinov. Unsupervised learning of video representations using lstms. In *ICML*, 2015. 2

[28] C. Sun and R. Nevatia. Active: Activity concept transitions in video event classification. In *CVPR*, 2013. 2

[29] L. Sun, K. Jia, D.-Y. Yeung, and B. E. Shi. Human action recognition using factorized spatio-temporal convolutional networks. In *ICCV*, 2015. 2, 6

[30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going Deeper with Convolutions. In *CVPR*, 2015. 1, 5

[31] K. Tang, L. Fei-Fei, and D. Koller. Learning latent temporal structure for complex event detection. In *CVPR*, 2012. 2

[32] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri. Learning Spatiotemporal Features with 3D Convolutional Networks. In *ICCV*, 2015. 2, 3

[33] G. Varol, I. Laptev, and C. Schmid. Long-term temporal convolutions for action recognition. *arXiv preprint arXiv:1604.04494*, 2016. 6

[34] R. Villegas, J. Yang, S. Hong, X. Lin, and H. Lee. Decomposing motion and content for natural video sequence prediction. In *ICLR*, 2017. 2

[35] L. Wang, Y. Qiao, and X. Tang. Action recognition with trajectory-pooled deep-convolutional descriptors. In *CVPR*, 2015. 6

[36] L. Wang, Y. Xiong, Z. Wang, Y. Qiao, D. Lin, X. Tang, and L. Van Gool. Temporal segment networks: towards good practices for deep action recognition. In *ECCV*, 2016. 1, 2, 3, 4, 5, 6, 7

[37] X. Wang, A. Farhadi, and A. Gupta. Actions˜ transformations. In *CVPR*, 2016. 1, 2, 5, 6

[38] H.-Y. Wu, M. Rubinstein, E. Shih, J. Guttag, F. Durand, and W. Freeman. Eulerian video magnification for revealing subtle changes in the world. *ACM Transactions on Graphics (TOG)*, 31(4):65, 2012. 2

[39] Z. Wu, T. Yao, Y. Fu, and Y.-G. Jiang. Deep learning for video classification and captioning. *arXiv preprint arXiv:1609.06782*, 2016. 2

[40] T. Xue, J. Wu, K. Bouman, and B. Freeman. Visual dynamics: Probabilistic future frame synthesis via cross convolutional networks. In *NIPS*, 2016. 2

[41] C. Zach, T. Pock, and H. Bischof. A Duality Based Approach for Realtime TV-L1 Optical Flow. In *Pattern Recognition*. 2007. 5

[42] M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In *ECCV*, 2014. 1, 3

[43] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba. Learning deep features for discriminative localization. In *CVPR*, 2016. 7

[44] W. Zhu, J. Hu, G. Sun, X. Cao, and Y. Qiao. A key volume mining deep framework for action recognition. In *CVPR*, 2016. 6