Representation Flow for Action Recognition
- Supplemental Materials -

AJ Piergiovanni and Michael S. Ryoo
Department of Computer Science, Indiana University, Bloomington, IN 47408
{ajpiergi,mryoo}@indiana.edu

A. Training and Implementation Details

Implementation Details When applying the representation flow layer within a CNN, we first applied a 1x1 convolutional layer to reduce the number of channels from C to 32. CNN feature maps often have hundreds of channels, but computing the representation flow for hundreds of channels is computationally expensive. We found 32 channels to be a good trade-off between performance and speed. The flow layer produces output with 64 channels, x and y flows for the 32 input channels, which are concatenated together. We apply a 3x3 convolutional layer to this representation to produce C output channels. This allows us to apply the rest of the standard CNN to the representation flow feature.

Two-stream networks stack 10 optical flow frames to capture temporal information [1]. However, we found that stacking representation flows did not perform well. Instead, we computed the flow for sequential images and averaged the predictions from a sequence of 16 frames. We found this outperformed stacking flow representations.

Training Details We trained the network using stochastic gradient descent with momentum set to 0.9. For Kinetics and Tiny-Kinetics, the initial learning rate was 0.1 and decayed by a factor of 10 every 50 epochs. The model was trained for 200 epochs. The 2D CNNs were trained using a batch size of 32 on 4 Titan X GPUs. The 3D and (2+1)D CNNs were trained with a batch size of 24 using 8 V100 GPUs. When fine-tuning on HMDB, the learning rate started at 0.005 and decayed by a factor of 10 every 20 epochs. The network was fine-tuned for 50 epochs. When learning the optical flow parameters, the learning rate for the parameters (i.e., λ, τ, θ, divergence kernels and Sobel filters) was set of $0.01 \cdot$ lr, otherwise the model produced poor predictions. This is likely due to the accumulation of gradients from the many iterations of the algorithm. For Kinetics and Tiny-Kinetics, we used dropout at 0.5 and for HMDB it was set to 0.8.

Testing Details For the results reported in Table **??**, we classified actions by applying our model to 25 different random croppings of each video. As found in many previous works, this helps increase the performance slightly. In all the other experiments (i.e., Tables **??**-**??**), random cropping was not used. Also notice that only the results in Table **??** uses our full model with $32 \times 224 \times 224$ input resolution. The other experiments uses spatially and/or temporally smaller models.

References

[1] K. Simonyan and A. Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems (NIPS)*, pages 568–576, 2014. 1