Active Vision for Early Recognition of Human Actions
Supplementary Material for CVPR 2020 paper

Boyu Wang[1], Lihan Huang[1], Minh Hoai[1,2]
[1]Stony Brook University, [2]VinAI Research
{boywang, lihahuang, minhhoai}@cs.stonybrook.edu

1. Early Recognition Performance

Fig. 1 shows the entire performance curves of several policies on the NTU dataset, plotting the recognition accuracy as a function of the observational ratio. To reduce clutter, we only plot the top performing policies in this figure.

The policy learned by reinforcement learning outperforms all other policies. Note that the upper bound for all the policies can be obtained by assuming that all views are available all the time. In this case, the obtained recognition accuracy is 81.28% and the average early recognition $\overline{\text{acc}}$ is 61.77%. The learned policy has a recognition accuracy of 79.62% and the early recognition accuracy $\overline{\text{acc}}$ of 58.01%, which are not too far from these upper bound values, even though the learned policy only uses one third of the video frames.

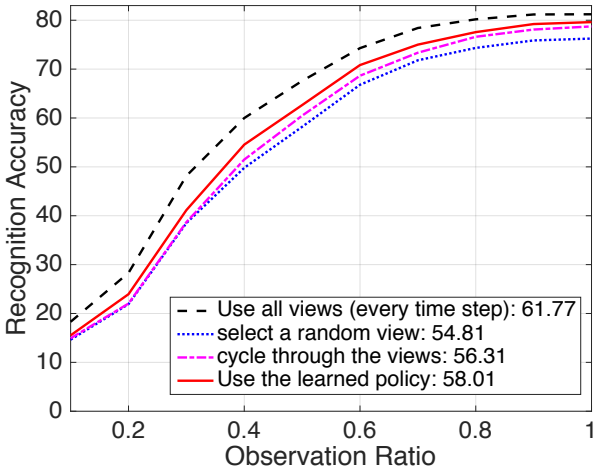

Figure 1: **Early recognition performance on the NTU dataset.** This shows the recognition accuracy against the observational ratio, which is the proportion of an action that has been observed when the recognition decision is made. The learned policy outperforms random-view and cycle-view policies.

2. Handling missing frames on IXMAS and nvGesture dataset

In this section, we demonstrate the effect of mfLSTM for handling unobserved video frames on the IXMAS dataset and the nvGesture dataset. We observe similar trends as for the NTU dataset.

Following Section 5.3, we measure the classification performance of our proposed method and other baselines for two test scenarios. In Test Scenario 1, the test sequences have variable frame rates due to the random frame dropping; the dropping rate ranges from 20% to 70%. In Test Scenario 2, the test sequences are the original test sequences; every frame is observed.

Table 1 and Table 3 show the experiment results for Test Scenario 1 on the IXMAS dataset and nvGesture dataset respectively. It reports the action classification accuracy of three methods on each camera views separately. The proposed approach mfLSTM for handling missing frames achieves the best performance. Compared to LSTM without data augmentation, mfLSTM is around 40% better for IXMAS dataset and 20% better for nvGesture dataset on average. Compared to LSTM with data augmentation, mfLSTM is around 3% better for IXMAS dataset and 2% better for nvGesture dataset on average.

The proposed approach mfLSTM also improves the classification performance on Test Scenario 2, where there are no missing video frames. Table 2 and Table 4 show the classification performance on the test sequences without dropped frames on the IXMAS dataset and nvGesture dataset respectively. mfLSTM outperforms the normal LSTM. The better generalization ability of the proposed method can be attributed to having augmented training data, proactively preparing the classifier for a wide range of cases. However, as can be seen from Table 2, the augmented training data hurts the performance of the LSTM classifier. This is due to having the wrong type of augmented data: the test data has no missing frames, while the generated training data is severely corrupted. On the other hand, the proposed mfLSTM with learnable decay pa-

Method	View 1	View 2	View 3	View 4	View 5
LSTM	44.39	50.90	42.87	48.78	35.45
LSTM + data augmentation	83.78	84.39	**87.57**	87.72	68.63
mfLSTM + data aug. (proposed)	**87.57**	**87.87**	86.45	**91.51**	**69.99**

Table 1: **Handling missing frames – Test Scenario 1**. This shows the classification accuracies of several methods on the test sequences that are corrupted by random frame dropping (the dropping rate ranges from 20% to 70%) on the **IXMAS** dataset. The proposed mfLSTM with augmented training data achieves the best performance.

Method	View 1	View 2	View 3	View 4	View 5
LSTM	93.03	89.09	**91.51**	89.09	76.66
LSTM + data augmentation	85.75	86.06	90.30	87.57	72.42
mfLSTM + data aug. (proposed)	**93.63**	**90.90**	90.60	**90.90**	**76.66**

Table 2: **Handling missing frames – Test Scenario 2**. This shows the classification accuracies of several methods for the second test scenario where every frame of the test sequences is observed on **IXMAS** dataset. The proposed mfLSTM with augmented training data achieves the best performance. For the original LSTM network, training with the augmented data hurts its performance due to the discrepancy between the augmented training data and the test data.

Method	RGB	Flow	Depth	Duo
LSTM	47.09	50.87	61.53	43.73
LSTM + data augmentation	64.52	71.49	74.89	62.86
mfLSTM + data aug. (proposed)	**66.35**	**73.20**	**75.93**	**63.86**

Table 3: **Handling missing frames – Test Scenario 1**. This shows the classification accuracies of several methods on the test sequences that are corrupted by random frame dropping (the dropping rate ranges from 20% to 70%) on the **nvGesture** dataset. The proposed mfLSTM with augmented training data achieves the best performance.

Method	RGB	Flow	Depth	Duo
LSTM	69.70	75.10	79.49	66.59
LSTM + data augmentation	69.29	74.68	79.25	65.98
mfLSTM + data aug. (proposed)	**70.95**	**76.97**	**79.87**	**67.01**

Table 4: **Handling missing frames – Test Scenario 2**. This shows the classification accuracies of several methods for the second test scenario where every frame of the test sequences is observed on **nvGesture** dataset. The proposed mfLSTM with augmented training data achieves the best performance. For the original LSTM network, training with the augmented data hurts its performance due to the discrepancy between the augmented training data and the test data.

rameters has the right architecture to take advantages of the augmented training data.

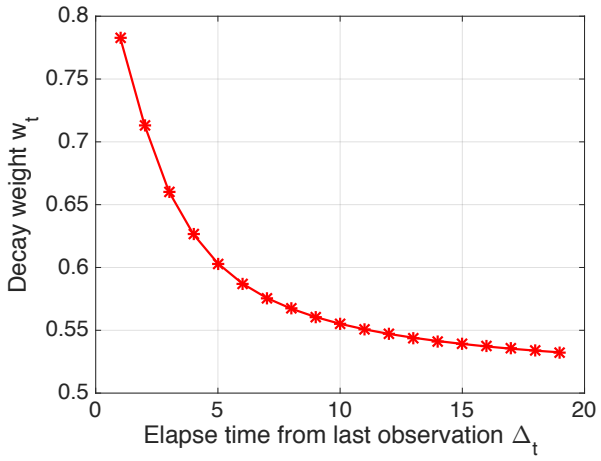

Figure 2: **The decay weight parameter $\mathbf{w}_t$ as a function of Δ_t.** $\mathbf{w}_t$ controls the contribution of the last observed values toward the estimation of the missing values. The contribution decreases as the difference between two time steps increases.

3. Visualize weight parameter in mfLSTM

Recall that the mfLSTM network uses a weight parameter $\mathbf{w}_t$ to control how much the missing values will be estimated based on the last observed values. $\mathbf{w}_t$ is a multi-dimensional function of the elapsed time Δ_t. Figure 2 plots the average value of $\mathbf{w}_t$ (averaged over its dimensions) for different elapsed times. The values of $\mathbf{w}_t$ decrease as the elapsed time increases. This is desirable because the significance of observation should decay over time.

4. Visualize the learned policy

We show the behavior of the learned policy at test time in Fig. 3. There is no particular pattern for the behavior of the policy; it does not stick to any particular view and it does not cycle through the views in any order. But the learned policy outperforms the random policy in our experiments, so it must take into account what is occurring and what has been observed to make the decisions.

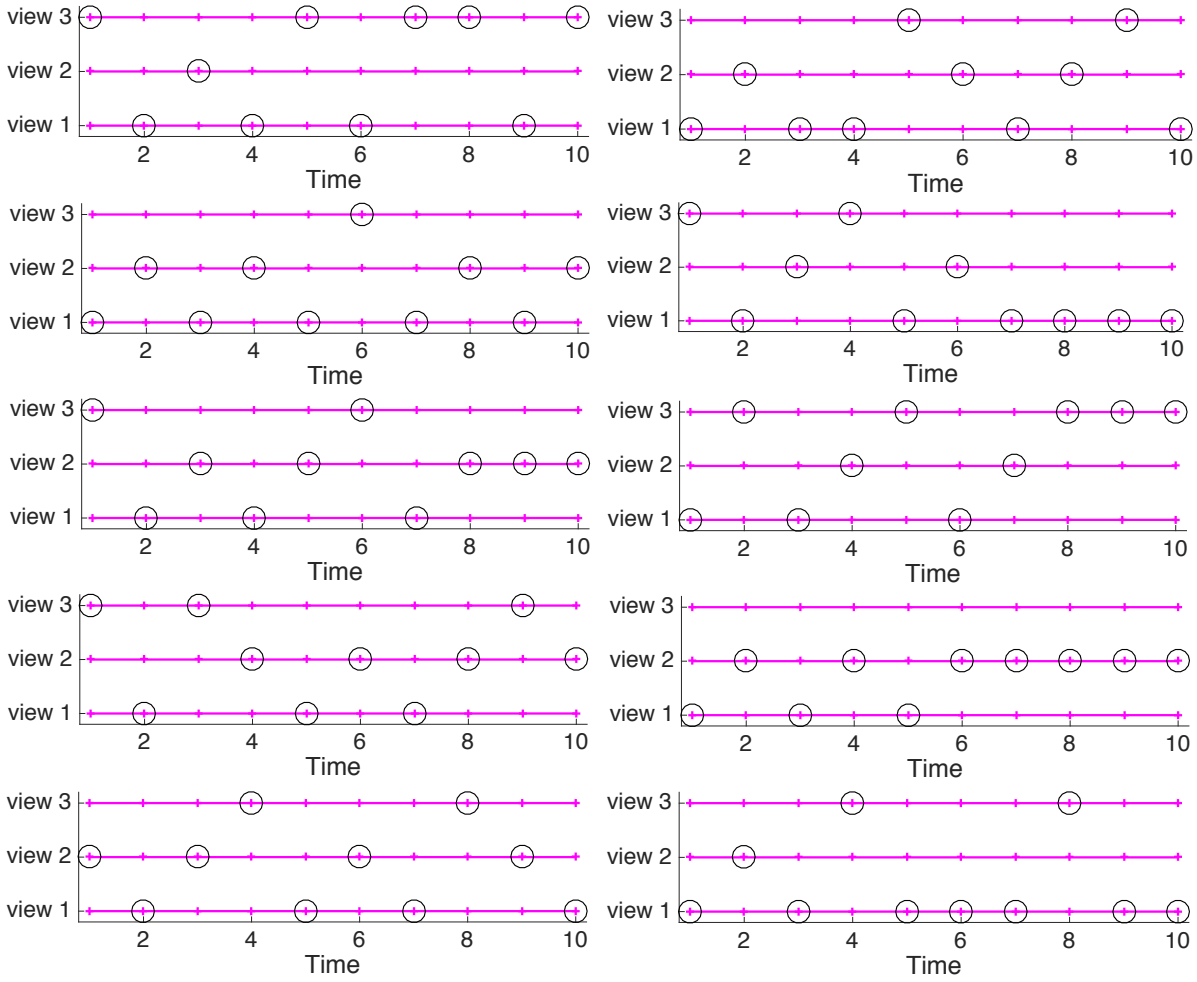

Figure 3: **A sequence of decision by the learned policy.** This shows 10 random examples for which views the learned policy selects at test time. The circles indicate the selected views. There is no particular pattern for the behavior of the policy. It appears to be random, but it is not.