Where and When Counts: Action Recognition in Videos

Abstract

Videos usually contain redundant information that not only results in computation complexity but also introduces disturbance for action recognition task. The redundant information comes from two ways. The first is temporal-level redundancy: for a video, some frames have little relevance to the action. The second is spatial-level redundancy: in a frame, some regions have nothing to do with the action. We believe only attending to the relevant information has the potential to boost accuracy for video action recognition. In this work, we propose an attention agent that can decide **where** and **when** to focus on, and we build our spatiotemporal attention agent upon Long-term recurrent convolutional network[2]. We evaluate our model on UCF-11, and find our model improves the accuracy compared with the baseline methods.

1 Introduction

Video recognition is a fundamental research topic in high-level computer vision research, required for true perceptual understanding in practical scenario where image streams are processed. Video recognition has seen exciting progress over recent three years. First, deep learning have been demonstrated as an effective models for understanding video content, for example, deep 3-dimensional convolutional Neural Networks (3D ConvNets) [1] and Long-term Recurrent Convolutional Networks (LRCN) [2] have been proposed as a powerful models for learning the spatiotemporal features. Second, thanks to the introduction of large amounts of annotated data and powerful hardware, training of deep networks with large number of parameters and great learning capacity becomes possible.

However, previous models [1][2] are indifferent to various parts of video, and they don't model the video information redundancy which is a very import property of video. First, for the purpose of computation efficiency, the recognition model should have the ability to selectively absorb input information. As we can see, the objects in consecutive frames of video don't change significantly in appearance, so the model don't have to take in all the frames. Second, redundancies can be disturbances for video recognition. For example, it's common that in a video, there are many people being active in the scene but only a small subset contributing to an actual event, thus taking the non-relevant people's action into consideration can lead to wrong recognition result. For the above two reasons, we believe it's necessary to model the information redundancy into recognition method. The redundant information comes from two ways. The first is temporal-level redundancy: for a video, some frames have little relevance to the action. The second is spatial-level redundancy: in a frame, some regions have nothing to do with the action.

Efforts have been paid in seeking the design of attentional models that can dynamically focus on voxels that are most relevant by eliminating or down-weight voxels that are not important or non-relevant for the task at hand. The key intuition of attention model is originate from cognition filed, cognition researchers think that humans do not focus their attention on an entire scene at once. Instead, they focus sequentially on different parts of the scene to extract relevant information. Thus, the process of recognize an action is one of continuous, iterative observation and refinement. The attention models kind of mimic human's vision pattern that only attend to parts of the inputs and dynamically change the attended voxels in order to precisely understand the action. However, existing works on attention model for video analysis either attend to spatial level [3] or temporal level [4][5] information. In our project, we design a spatiotemporal level attention agent that is able to focus on spatiotemporal volumes, i.e., our proposed attention agent can simultaneously attend to the relevant frames within a video and relevant regions within a frame. Our attention agent is built on the LRCN [2]. LRCN is doubly deep since it can learn compositional representations in space and time. It learns the frame-level features through 2D CNN and then encodes temporal dependencies by forwarding those frame features to RNN. Long-term dependencies and dynamics can be learned by adopting Long short-term memory (LSTM) units that can overcome the vanishing and exploding gradients problem of vanilla RNN. As we have discussed, LRCN treat all the voxels indifferently. By adding our spatiotemporal attention agent, the model can dynamically pool the convolutional features and outputs of each time-step LSTM unit.

The rest of the report is organized as follows: in approach section, we will introduce the basic LRCN model, describe the architecture of the spatiotemporal attention based LRCN and two kinds of formulation; in experiment section, we evaluate the performance of our model both quantitatively and qualitatively; in conclusion section, we point out the future work.

2 Approach

2.1 Formulation of basic LRCN model[2]

[2] proposes a Long-term Recurrent Convolutional Network model combining a deep hierarchical visual feature extractor (such as CNN) with a model that can learn to recognize and synthesize temporal dynamics for tasks involving sequential data (inputs or outputs), visual, linguistic, or otherwise. Fig depict the core of the approach. LRCN works by passing each visual input x_t (an image in isolation, or a frame from a video) through a feature transformation $\o_V(\cdot)$ with parameters V, usually a CNN, to produce a fixed-legth vector representation $\o_V(x_t)$. The outputs of $\o_V$ are then passed into a recurrent sequence learning module.

In its most general form, a recurrent model has parameters W, and maps an input x_t and a previous time step hidden state h_{t-1} to an output z_t and updated hidden state h_t. Therefore, inference must be run sequentially, by computing in order: $h_1 = f_W(x_1, h_0) = f_W(x_1, 0)$, then $h_2 = f_W(x_2, h_0)$, etc., up to h_T. Some of their model stack multiple LSTMs a top one another. To predict a distribution $P(y_t)$ over outcome $y_t \in C$(where C is a discrete, finite set of outcomes) at time step t, the outputs $z_t \in R^{d_x}$ of the sequential model are passed through a linear prediction layer $\hat{y} = W_z z_t + b_z$, where $W_z \in R^{|C| \times d_x}$ and $b_z \in R^{|C|}$ are learned parameters. Finally, the predicted distribution $P(y_t)$ is computed by taking the softmax of $\hat{y_t} : P(y_t = C) = softmax(y_t) = \frac{exp(\hat{y_t}, c)}{\sum_{c' \in C} exp(\hat{y_t}, c)}$.

The visual feature transformation øcorresponds to the activations in some layer of a deep CNN. Using a visual transformation $\o_V(\cdot)$ which is time-invariant and independent at each time step has the important advantage of making the expensive convolutional inference and training parallelizable over all time steps of the input, facilitating the use of fast contemporary CNN implementations whose efficiency relies on independent batch processing, and end-to-end optimization of the visual and sequential model parameters V and W.

To produce a single label prediction for an entire video clip, they average the label probabilities-the outputs of the network's softmax layer-across all frames and choose the most probable label, which implicitly means that they treat every frames indifferently.

2.2 Attention based LRCN

The architecture of the proposed spatiotemporal attention based LRCN is shown in Figure 1. We will describe the spatial attention and temporal attention separately.

2.2.1 Spatial Attention

As mentioned before, videos contain spatial-level redundancy, that means in a frame, some regions have nothing to do with the true action. The spatial attention agent is shown in Figure 1 gray frame.

For each frame extracted from the action video, we divided them into 7×7 patches, and fed each

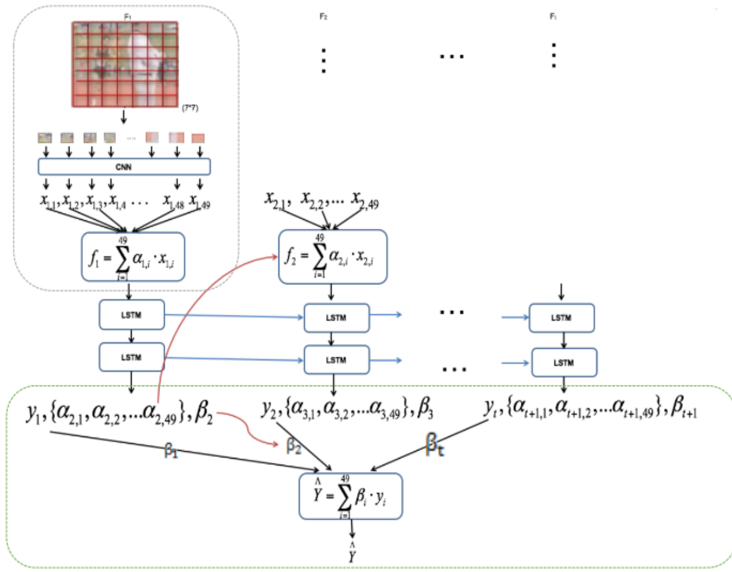

Figure 1: Action recognition model with spatial and temporal attention.

patch into CNN model, get last convolutional layer for the patch feature. For the whole frame, we get a feature cube of shape $K \times K \times D$ which is $7 \times 7 \times 256$ here. Thus, at time-step *t*, we extract K^2 *D*-dimensional vectors. We refer to these vectors as feature slices in a feature cube:

$$X_t = [x_{t,1}, ..., x_{t,K^2}], x_{t,i} \in R^D$$

Each of these K^2 vertical feature slices maps to different regions in the input frame space and our spatial attention model learns to focus its attention on these K^2 regions. At each time-step *t*, spatial attention model has α_t, a weight vector over patch features. It indicates the importance of each patch contributing to the action in the video. The final feature for the whole frame is a weighted vector of patch features:

$$f_t = \sum_{i=1}^{49} \alpha_{t,i} \cdot x_{t,i}$$

Human vision systems can dynamically change the regions that should be paid attention, our spacial attention agent can learn the dynamic weight vector, too. We think that attended regions in consequent frames should be related in terms of their spacial positions. In order to model the spatial attention dependencies between consequent frames, we set $\alpha_{t,i} = s(\alpha_{t-1,i}, X_{t-1})$,i.e., our model predicts α_t as the output of LSTM at $t-1$ time-step.

2.2.2 Temporal Attention

For redundancy in temporal-level, we think not every frame in the video contributes to the video label, such as action type. For each frame, LSTM model will predict a frame label y_i as shown in Figure 1 green frame. Traditional video action recognition system will simply average the label results of each frame and get the final predicted video label. In our temporal attention, an attention weight is assigned to each predicted frame label,β_i, and the final video label is calculated as:

$$Y = \sum_{i=1}^{t} \beta_i \cdot y_i$$

Similarly, we think that the attention weight for frame at time-step t has some dependencies on previous frames, thus we set $\beta_t = t(X_{t-1})$, i.e., our model predict β_t as the output of LSTM at $t-1$ time-step.

2.2.3 Loss function and the attention penalty

We use cross-entropy loss coupled with the doubly stochastic penalty, We impose an additional constraint over the location softmax, so that $\sum_{t=1}^{T} \alpha_{t,i} \approx 1$. This is the attention regularization which forces the model to look at each region of the frame at some point in time. The loss function is defined as follows:

$$L = -\sum_{n=1}^{N}\sum_{i=1}^{C} y_{n,i} log\hat{y_{n,i}} + \lambda \sum_{i=1}^{K^2}(1 - \sum_{t=1}^{T} l_{t,i})^2 + \gamma \sum_{i}\sum_{j} \theta_{i,j}^2$$

where y_n is the one hot label vector, $\hat{y_n}$ is the vector of class probabilities for data n, N is the total number of training set, C is the number of output class, $\lambda$is the attention penalty coefficient,γ is the weight decay coefficient, and θ represents all the model parameters.

2.2.4 Two kinds of Formulation

Broadly speaking, attentional models can be split into two categories. The first class is represented by methods that use soft attention mechanism. This soft attention agent can learn to decrease the weight of non-relevant frames and regions, and increase the weight of relevant frames and regions. The second category embodies hard attentional methods that completely discard (as opposed to re-weight) less relevant information.

In the mathematical formulation, for soft attention mechanism:

$$\alpha_{t,j} \in [0,\ 1], \beta_t \in [0,\ 1]$$

Soft attention models are deterministic and can be trained using standard backpropagation.

for hard attention mechanism:

$$\alpha_{t,j} = 0 \; or \; 1, \beta_t = 0 \; or \; 1$$

Hard attention modeles are stochastic and non-differentiable, they can be trained by the REINFORCE algorithm.

In our project, we adopt soft attention mechanism for spatial-level attention and hard attention mechanism for temporal-level attention. Instead of using REINFORCE rule for leaning the hard temporal-level attention, we build a plug for human computation module in our system, i.e., we involve the idea of human-in-the-loop and ask crowd workers to select the relevant frames.

3 Experiment

3.1 Dataset

We use UCF YouTube Action dataset in our experiments. The video dataset consists of 1599 videos and 11 actions- basketball shooting, biking/cycling, diving, golf swinging, horse back riding, soccer juggling, swinging, tennis swinging, trampoline jumping, volleyball spiking, and walking with a dog. The clips have a frame rate of 29.97 fps and each video has only one action associated with it. We use 948 videos for training and 651 videos for testing.

All videos in the dataset were transformed to frames at 30 fps and fed to AlexNet model trained on the ImageNet dataset. The last convolutional layer was used as input to model.

3.2 Training Details

In our experiments, for our dataset we trained 3-layer LSTM models, where the dimensionality of the LSTM hidden state, cell state, and the hidden layer were set to 512 for dataset. For the attention penalty coefficient we experimented with values 0, 1, 10. We set the weight decay penalty to 10^-5 and use dropout of 0.5. Models are trained for 15 epochs over the entire datasets. Our implementation is based in Theano.

For both training and testing our model takes 30 frames for each video which are selected manually. At test time, we compute class predictions for each time step and then average those predictions over 30 frames. To obtain a prediction for the entire video, we average the predictions from all 30 frames in the video.

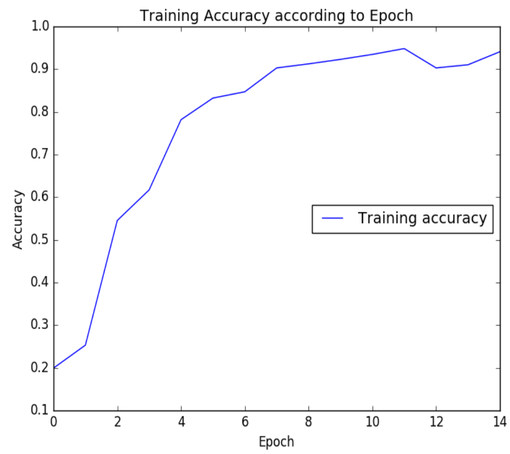

Figure 2: training accuracy-epoch curve.

Table 1: Performance on UCF-11 (acc %)

Model	**UCF-11**
Softmax Regression(full CNN featrue cube)	82.37
Avg pooled LSTM	82.56
Max pooled LSTM	81.6
Soft attention model ($\lambda = 0$)	84.96
Soft attention model ($\lambda = 1$)	83.52
Soft attention model ($\lambda = 10$)	81.44

3.3 Experiment Result

We trained our model on UCF YouTube Action dataset. The accuracies are reported in Table 1. The softmax regression model uses the complete $7 \times 7 \times 256$ feature cube as its input to predict the label at each time-step *t*, while all other models use only a 256-dimensional feature slice as their input. The average pooled and max pooled LSTM models use the same architecture as our model except that they do not have any attention mechanism and thus do not produce a weight vector.The input at each time-step for these models are obtained by doing average or max pooling over the feature cube to get 256 dimensional slices, whereas our soft attention model dynamically weights the slices by location weights. The visualization results are shown in Figure 3. The white regions are where the model is looking and the brightness indicates the strength of focus. Figures are from the best performing models with $\lambda = 0$. Setting $\lambda = 0$ corresponds to the model that tends to select a few locations and stay fixed on them.

4 Conclusion

4.1 Our work

In our project, we propose a spatiotemporal attention based LRCN for action recognition. Our proposed model can attend to the relevant voxels in a video. We show that our model performs better than baselines which do not use any attention mechanism.

4.2 Future work

First, as we have discussed in 2.2.2 and 2.2.3 section, we set $\alpha_{t,i} = s(\alpha_{t-1,i}, X_{t-1})$ and $\beta_t = t(X_{t-1})$, while the modeling of both α and β could be more complex. $\alpha_{t,i}$ may not only have

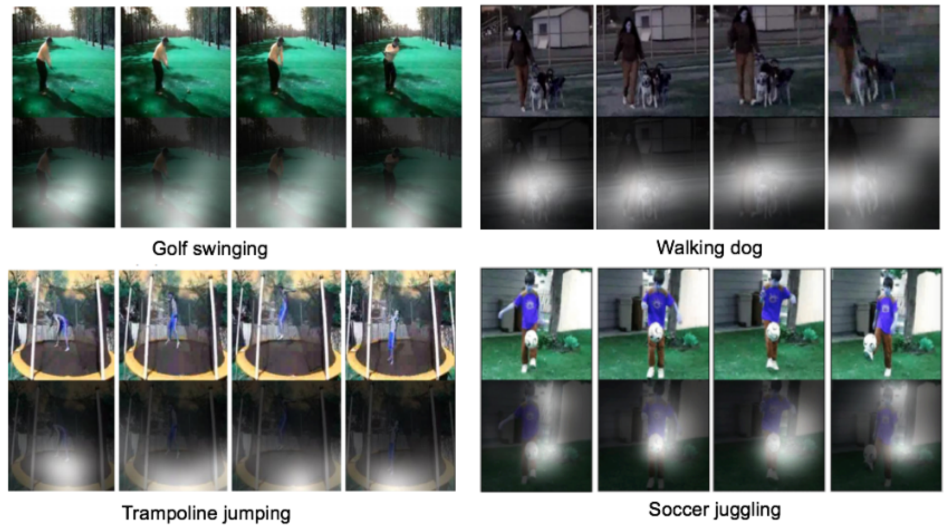

Figure 3: Visualization of the focus of attention for four videos from UCF-11 datasets, the white regions are where the model is looking and the brightness indicates the strength of focus.

dependencies on $\alpha_{t-1,i}, X_{t-1}$, but also depend on X_t.β have the same case. Thus, we can also set $\alpha_{t,i} = s(\alpha_{t-1,i}, X_{t-1}, X_t)$ and $\beta_t = t(\beta_{t-1}, X_{t-1}, X_t)$ to encode more complex dependencies.

Second, our project use human computation to manually select attended frames for the implementation of temporal attention. However, it's difficult to hand craft the criteria for relevant frames, so it's necessary to algorithmically select the attended frames. We plan to explore REINFORCE rule to learn the hard attention agent, and compare the learned frames with the manually selected frames.

Third, we consider attention agent as an implicit method to model human gaze, and we don't explicitly make the attention model attend to the true attentional voxels. Instead, we tune the attention weights by the specific task at hand(in our project, the task refers to action r ecognition). Indeed, there's another direction to reduce the computation and decrease the disturbances introduced by the video redundancies. That is explicitly localize the relevant regions, and usually ground truth of localizations are available. We may further compare the implicit method and explicit method, hoping to find their underlying similarities and provide some intuitions for vision cognition researchers.

References

[1] Tran D, Bourdev L, Fergus R, et al. Learning spatiotemporal features with 3d convolutional networks[C]//2015 IEEE International Conference on Computer Vision (ICCV). IEEE, 2015: 4489-4497.

[2] Donahue J, Anne Hendricks L, Guadarrama S, et al. Long-term recurrent convolutional networks for visual recognition and description[C]//Proceedings of the IEEE Conference on Computer Vision and Pattern Recog-nition. 2015: 2625-2634..

[3] Sharma S, Kiros R, Salakhutdinov R. Action recognition using visual attention[J]. arXiv preprint arXiv:1511.04119, 2015.

[4] Yao L, Torabi A, Cho K, et al. Describing videos by exploiting temporal structure[C]//Proceedings of the IEEE International Conference on Computer Vision. 2015: 4507-4515.

[5] Yeung S, Russakovsky O, Mori G, et al. End-to-end Learning of Action Detection from Frame Glimpses in Videos[J]. arXiv preprint arXiv:1511.06984, 2015.