Action Recognition and Video Description using Visual Attention

by

Shikhar Sharma

A thesis submitted in conformity with the requirements
for the degree of Master of Science
Graduate Department of Computer Science
University of Toronto

© Copyright 2016 by Shikhar Sharma

Abstract

Action Recognition and Video Description using Visual Attention

Shikhar Sharma
Master of Science
Graduate Department of Computer Science
University of Toronto
2016

We propose soft attention based models for the tasks of action recognition in videos and generating natural language descriptions of videos. We use multi-layered Recurrent Neural Networks (RNNs) with Long Short-Term Memory (LSTM) units which are deep both spatially and temporally. Our model learns to focus selectively on parts of the video frames and classifies videos after taking a few glimpses. It is also able to generate sentences describing the videos using spatio-temporal glimpses across them. The model essentially learns which parts in the frames are relevant for the task at hand and attaches higher importance to them. We evaluate the action recognition model on UCF-11 (YouTube Action), HMDB-51 and Hollywood2 datasets and analyze how the model focuses its attention depending on the scene and the action being performed. We evaluate the description generation model on YouTube2Text dataset and visualize the model's attention as it generates words.

Acknowledgements

This work was supported by IARPA and Raytheon BBN Contract No. D11PC20071. We would like to thank Nitish Srivastava for valuable discussions and Yukun Zhu for his assistance with the CNN packages. A part of this work encompassing most of the action recognition task has been accepted as a contribution in NIPS Time Series Workshop 2015 and ICLR Workshop 2016 and was co-authored by Ryan Kiros.

Contents

Chapter 1

Introduction and Related Work

In this chapter, we will motivate soft attention models for action recognition and video description and discuss relevant attention models in the literature.

1.1 Introduction

It has been noted in visual cognition literature that humans do not focus their attention on an entire scene at once (Rensink, 2000). Instead, they focus sequentially on different parts of the scene to extract relevant information. Most traditional computer vision algorithms do not employ attention mechanisms and are indifferent to various parts of the image/video. With the recent surge of interest in deep neural networks, attention based models have been shown to achieve promising results on several challenging tasks, including caption generation (Xu et al., 2015), machine translation (Bahdanau et al., 2015), game-playing and tracking (Mnih et al., 2014), as well as image recognition (e.g. Street View House Numbers dataset (Ba et al., 2015b)). Many of these models have employed LSTM (Hochreiter & Schmidhuber, 1997) based RNNs and have shown good results in learning sequences.

Attention models can be classified into soft attention and hard attention models. Soft attention models are deterministic and can be trained using backpropagation, whereas hard attention models are stochastic and can be trained by the REINFORCE algorithm (Williams, 1992; Mnih et al., 2014), or by maximizing a variational lower bound or using importance sampling (Ba et al., 2015b;a). Learning hard attention models can become computationally expensive as it requires sampling. In soft attention approaches, on the other hand, a differentiable mapping can be used from all the locations output to the next input.

Attention based models can also potentially infer the action happening in videos by focusing only on the relevant places in each frame. For example Fig. 1.1a shows four frames from the UCF-11 video sequence belonging to the "golf swinging" category. The model tends to focus on the ball, the club, and the human, which allows the model to correctly recognize the activity as "golf swinging". In Fig. 1.1b, our model attends to the trampoline, while correctly identifying the activity as "trampoline jumping".

The ability to describe our different sensory experiences, in language, is key to human communication and propagation of knowledge. For machine learning systems, the task of understanding and describing videos is an important challenge and a complex problem as it involves reasoning about long-term and short-term video dependencies. It has numerous applications including automatic generation of subtitles,

(a) Correctly classified as "golf swinging" (b) Correctly classified as "trampoline jumping"

Figure 1.1: Attention over time: The white regions show what the model is attending to and the brightness indicates the strength of focus. Best viewed in color. The intensity of the attention overlay has been scaled uniformly for clarity of visualization.

assistive technology for the visually challenged, video summarization and better indexing. Our attention over different objects in a scene varies as we describe the entire scene and this would seem to be a desirable property for machine learning systems to have as well. Such attention based models would have the flexibility to focus on different regions of the video at different stages of the description generation process and we analyze the behavior of soft attention models for this task.

1.2 Related Work

Convolutional Neural Networks (CNNs) have been highly successful in image classification and object recognition tasks (Ren et al., 2015; Wu et al., 2015). Classifying videos instead of images adds a temporal dimension to the problem of image classification. Learning temporal dynamics is a difficult problem and earlier approaches have used optical flow, HOG and hand-crafted features to generate descriptors with both appearance and dynamics information encoded. LSTMs have been recently shown to perform well in the domain of speech recognition (Graves et al., 2013), machine translation (Sutskever et al., 2014), image description (Xu et al., 2015; Vinyals et al., 2015) and video description (Yao et al., 2015; Venugopalan et al., 2015). They have also started picking up momentum in action recognition (Srivastava et al., 2015; Ng et al., 2015).

Most of the existing approaches also tend to have CNNs underlying the LSTMs and classify sequences directly or do temporal pooling of features prior to classification (Donahue et al., 2015; Ng et al., 2015). LSTMs have also been used to learn an effective representation of videos in unsupervised settings (Srivastava et al., 2015) by using them in an encoder-decoder framework. More recently, Yao et al. (2015) have proposed to use 3-D CNN features and an LSTM decoder in an encoder-decoder framework to generate video descriptions. Their model incorporates attention on a video level by defining a probability distribution over frames used to generate individual words. They, however, do not employ an attention mechanism on a frame level (i.e. within a single frame).

Karpathy et al. (2014) used a multi-resolution CNN architecture to perform action recognition in videos. They mention the concept of fovea but they fix attention to the center of the frame. A recent work of Xu et al. (2015) used both soft attention and hard attention mechanisms to generate image descriptions. Their model actually looks at the respective objects when generating their description. More recently, Jaderberg et al. (2015) have proposed a soft attention mechanism called the *Spatial Transformer* module which they add between the layers of CNNs. Instead of weighting locations using

a softmax layer which we do, they apply affine transformations to multiple layers of their CNN to attend to the relevant part and get state-of-the-art results on the Street View House Numbers dataset (Netzer et al., 2011). Yeung et al. (2015) perform dense action labeling using a temporal attention based model on the input-output context and report higher accuracy and better understanding of temporal relationships in action videos.

Xu et al. (2015) explored caption generation for image datasets using both soft and hard attention based models and reported state-of-the-art results and most of the video description approaches are based on this work. We next describe some of these recent approaches which have been state-of-the-art in video description at some point in time. Venugopalan et al. (2015) use CNNs to extract features for all video frames, perform mean pooling over the features to create a fixed dimensional vector representation and then have an LSTM-RNN decoder generate descriptions for the videos. Pan et al. (2015) create a sentence representation by concatenating all the word embedding vectors and a video embedding comprising of mean pooled CNN and 3-D CNN features which is used as context for the decoder. The loss function which they use to train the decoder is a linear combination of relevance loss (distance between these embeddings) and coherence loss (log probability loss used in statistical machine translation systems) along with regularization. Yao et al. (2015) use both 2-D and 3-D CNNs for feature extraction and have a temporal attention mechanism in an LSTM-RNN decoder for generating descriptions of videos. Yu et al. (2015) use hierarchical RNNs with Gated Recurrent Units (GRUs) and a spatio-temporal attention model (similar to the spatial attention mechanism used by Xu et al. (2015)) to get state-of-the-art results on video description tasks. The hidden state of their GRU-RNN decoder is not conditioned on the weighted video features which gave them higher performance. Ballas et al. (2015) use RNNs with GRUs with percepts extracted from several convolutional layers of CNNs and report performance comparable to state-of-the-art approaches without using 3-D CNN features.

In general, it is rather difficult to interpret internal representations learned by deep neural networks. Attention models add a dimension of interpretability by capturing where the model is focusing its attention when performing a particular task. Our work directly builds upon the work of Xu et al. (2015). However, while they primarily worked on caption generation in static images, in this thesis, we focus on using a soft attention mechanism for activity recognition in videos and describing videos.

Chapter 2

Action Recognition using Visual Attention

In this chapter, we describe the soft attention based model which we use for action recognition. We describe the datasets that we use - UCF-11, Hollywood2 and HMDB-51. We present quantitative analysis of our model's performance against baselines and state-of-the-art models and qualitative analysis of why it looks where it looks.

2.1 The Model and the Attention Mechanism

2.1.1 Convolutional Features

We extract the last convolutional layer obtained by pushing the video frames through GoogLeNet model (Szegedy et al., 2015) trained on the ImageNet dataset (Deng et al., 2009). This last convolutional layer has D convolutional maps and is a feature cube of shape $K \times K \times D$ ($7 \times 7 \times 1024$ in our experiments). Thus, at each time-step t, we extract K^2 D-dimensional vectors. We refer to these vectors as feature slices in a feature cube:

$$\mathbf{X}_t = [\mathbf{X}_{t,1}, \ldots, \mathbf{X}_{t,K^2}], \qquad \mathbf{X}_{t,i} \in \mathbb{R}^D.$$

Each of these K^2 vertical feature slices maps to different overlapping regions in the input space and our model chooses to focus its attention on these K^2 regions.

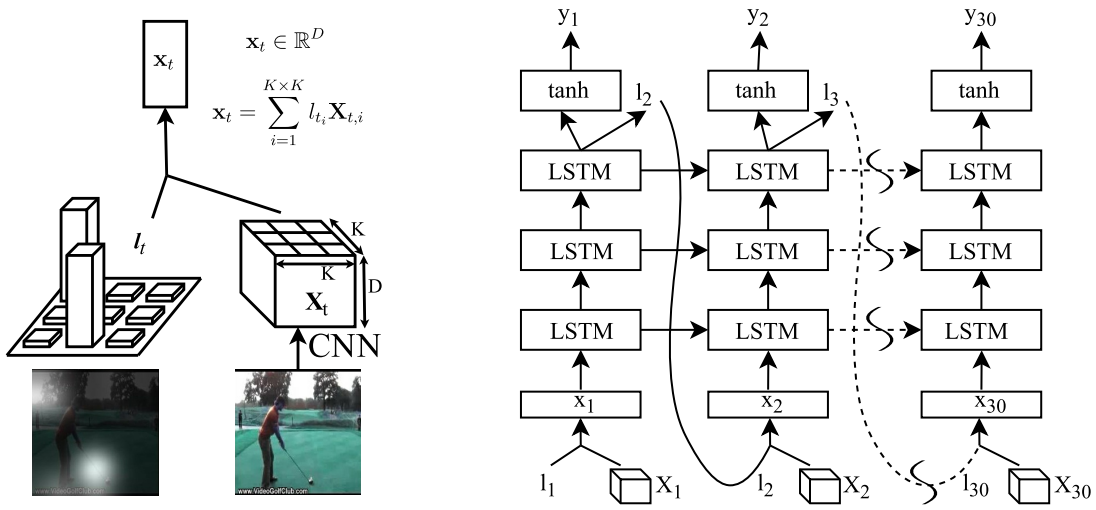
2.1.2 The LSTM and the Attention Mechanism

We use the LSTM implementation discussed in Zaremba et al. (2014) and Xu et al. (2015):

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} M \begin{pmatrix} \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix}, \tag{2.1}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \mathbf{g}_t, \tag{2.2}$$

$$\mathbf{h}_t = \mathbf{o}_t \odot \tanh(\mathbf{c}_t) \tag{2.3}$$



(a) The soft attention mechanism

(b) Our recurrent model

Figure 2.1: (2.1a) The CNN takes the video frame as its input and produces a feature cube. The model computes the current input $\mathbf{x}_t$ as an average of the feature slices weighted according to the location softmax $\mathbf{l}_t$ (2.1b). At each time-step t, our recurrent network takes a feature slice $\mathbf{x}_t$, generated as in (2.1a), as the input. It then propagates $\mathbf{x}_t$ through three layers of LSTMs and predicts the next location probabilities $\mathbf{l}_{t+1}$ and the class label $\mathbf{y}_t$.

where $\mathbf{i}_t$ is the input gate, $\mathbf{f}_t$ is the forget gate, $\mathbf{o}_t$ is the output gate, and $\mathbf{g}_t$ is calculated as shown in Eq. 2.1. $\mathbf{c}_t$ is the cell state, $\mathbf{h}_t$ is the hidden state, and $\mathbf{x}_t$ (see Eqs. 2.4, 2.5) represents the input to the LSTM at time-step t. $M : \mathbb{R}^a \to \mathbb{R}^b$ is an affine transformation consisting of trainable parameters with $a = d + D$ and $b = 4d$, where d is the dimensionality of all of $\mathbf{i}_t$, $\mathbf{f}_t$, $\mathbf{o}_t$, $\mathbf{g}_t$, $\mathbf{c}_t$, and $\mathbf{h}_t$.

At each time-step t, our model predicts $\mathbf{l}_{t+1}$, a softmax over $K \times K$ locations, and $\mathbf{y}_t$, a softmax over the label classes with an additional hidden layer with *tanh* activations (see Fig. 2.1b). The location softmax is defined as follows:

$$l_{t,i} = p(\mathbf{L}_t = i|\mathbf{h}_{t-1}) = \frac{\exp(W_i^\top \mathbf{h}_{t-1})}{\sum_{j=1}^{K\times K} \exp(W_j^\top \mathbf{h}_{t-1})} \qquad i \in 1 \dots K^2, \tag{2.4}$$

where W_i are the weights mapping to the i^{th} element of the location softmax and $\mathbf{L}_t$ is a random variable which can take 1-of-K^2 values. This softmax can be thought of as the probability with which our model believes the corresponding region in the input frame is important. After calculating these probabilities, the soft attention mechanism (Bahdanau et al., 2015) computes the expected value of the input at the next time-step $\mathbf{x}_t$ by taking expectation over the feature slices at different regions (see Fig. 2.1a):

$$\mathbf{x}_t = \mathbb{E}_{p(\mathbf{L}_t|\mathbf{h}_{t-1})}[\mathbf{X}_t] = \sum_{i=1}^{K^2} l_{t,i} \mathbf{X}_{t,i}, \tag{2.5}$$

where $\mathbf{X}_t$ is the feature cube and $\mathbf{X}_{t,i}$ is the i^{th} slice of the feature cube at time-step t. Note that in the hard attention based models, we would sample $\mathbf{L}_t$ from a softmax distribution of Eq. 2.4. The input $\mathbf{x}_t$ would then be the feature slice at the sampled location instead of taking expectation over all the slices. Thus, hard attention based models are not differentiable and have to resort to some form of sampling.

We use the following initialization strategy (see Xu et al. (2015)) for the cell state and the hidden state of the LSTM for faster convergence:

$$\mathbf{c}_0 = f_{\text{init,c}}\left(\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{K^2}\sum_{i=1}^{K^2}\mathbf{X}_{t,i}\right)\right) \quad \text{and} \quad \mathbf{h}_0 = f_{\text{init,h}}\left(\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{K^2}\sum_{i=1}^{K^2}\mathbf{X}_{t,i}\right)\right), \tag{2.6}$$

where $f_{\text{init,c}}$ and $f_{\text{init,h}}$ are two multilayer perceptrons and T is the number of time-steps in the model. These values are used to calculate the first location softmax $\mathbf{l}_1$ which determines the initial input $\mathbf{x}_1$. In our experiments, we use multi-layered deep LSTMs, as shown in Fig. 2.1b.

2.1.3 Loss Function and the Attention Penalty

We use cross-entropy loss coupled with the doubly stochastic penalty introduced in Xu et al. (2015). We impose an additional constraint over the location softmax, so that $\sum_{t=1}^{T} l_{t_i} \approx 1$. 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:

$$L = -\sum_{t=1}^{T}\sum_{i=1}^{C} y_{t,i}\log\hat{y}_{t,i} + \lambda\sum_{i=1}^{K\times K}(1-\sum_{t=1}^{T} l_{t_i})^2 + \gamma\sum_{i}\sum_{j}\theta_{i,j}^2, \tag{2.7}$$

where y_t is the one hot label vector, $\hat{y}_t$ is the vector of class probabilities at time-step t, T is the total number of time-steps, C is the number of output classes, λ is the attention penalty coefficient, γ is the weight decay coefficient, and θ represents all the model parameters.

2.2 Experiments

2.2.1 Datasets

We have used UCF-11, HMDB-51 and Hollywood2 datasets in our experiments. UCF-11 is the YouTube Action dataset consisting of 1600 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 975 videos for training and 625 videos for testing.

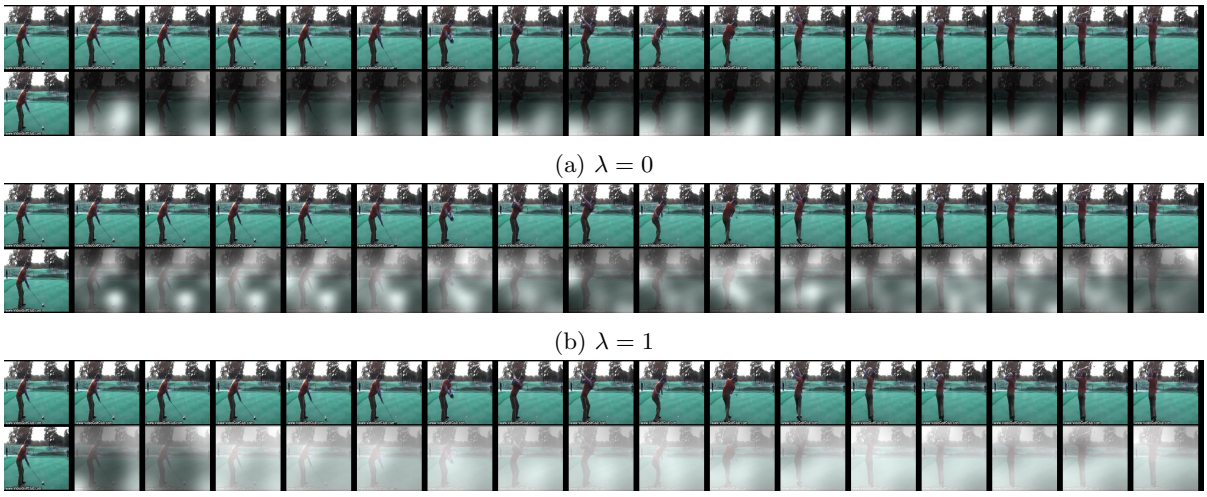
HMDB-51 Human Motion Database dataset provides three train-test splits each consisting of 5100 videos. These clips are labeled with 51 classes of human actions like Clap, Drink, Hug, Jump, Somersault, Throw and many others. Each video has only one action associated with it. The training set for each split has 3570 videos (70 per category) and the test set has 1530 videos (30 per category). The clips have a frame rate of 30 fps.

Hollywood2 Human Actions dataset consists of 1707 video clips collected from movies. These clips are labeled with 12 classes of human actions - AnswerPhone, DriveCar, Eat, FightPerson, GetOutCar, HandShake, HugPerson, Kiss, Run, SitUp, SitDown and StandUp. Some videos have multiple actions associated with them. The training set has 823 videos and the testing set has 884 videos.

All the videos in the datasets were resized to 224×224 resolution and fed to a GoogLeNet model trained on the ImageNet dataset. The last convolutional layer of size $7 \times 7 \times 1024$ was used as an input to our model.

Table 2.1: Performance on UCF-11 (acc %), HMDB-51 (acc %) and Hollywood2 (mAP %)

Model	UCF-11	HMDB-51	Hollywood2
Softmax Regression (full CNN feature cube)	82.37	33.46	34.62
Avg pooled LSTM (@ 30 fps)	82.56	40.52	43.19
Max pooled LSTM (@ 30 fps)	81.60	37.58	43.22
Soft attention model (@ 30 fps, $\lambda = 0$)	84.96	41.31	43.91
Soft attention model (@ 30 fps, $\lambda = 1$)	83.52	40.98	43.18
Soft attention model (@ 30 fps, $\lambda = 10$)	81.44	39.87	42.92



(a) $\lambda = 0$

(b) $\lambda = 1$

(c) $\lambda = 10$

Figure 2.2: Variation in the model's attention depending on the value of attention penalty λ. The white regions are where the model is looking and the brightness indicates the strength of focus. Setting $\lambda = 0$ corresponds to the model that tends to select a few locations and stay fixed on them. Setting $\lambda = 10$ forces the model to gaze everywhere, which resembles average pooling over slices.

2.2.2 Training Details and Evaluation

In all of our experiments, model architecture and various other hyper-parameters were set using cross-validation. In particular, for all datasets 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 both UCF-11 and Hollywood2 and 1024 for HMDB-51. We also experimented with models having one LSTM layer to five LSTM layers, but did not observe any significant improvements in model performance. For the attention penalty coefficient we experimented with values 0, 1, 10. While reporting results, we have set the weight decay penalty to 10^{-5} and use dropout (Srivastava et al., 2014) of 0.5 at all non-recurrent connections. All models were trained using Adam optimization algorithm (Kingma & Ba, 2015) for 15 epochs over the entire datasets. However, we found that Adam usually converged after 3 epochs. Our implementation is based in Theano (Bastien et al., 2012) which also handles the gradient computation and our code is available at `https://github.com/kracwarlock/action-recognition-visual-attention`.

For both training and testing our model takes 30 frames at a time sampled at fixed fps rates. We split each video into groups of 30 frames starting with the first frame, selecting 30 frames according to the fps rate, and then moving ahead with a stride of 1. Each video thus gets split into multiple

Table 2.2: Comparison of performance on HMDB-51 and Hollywood2 with state-of-the-art models

Model		HMDB-51 (acc %)	Hollywood2 (mAP %)
Spatial stream ConvNet	(Simonyan & Zisserman, 2014)	40.5	-
Soft attention model	(Our model)	41.3	43.9
Composite LSTM Model	(Srivastava et al., 2015)	44.0	-
DL-SFA	(Sun et al., 2014)	-	48.1
Two-stream ConvNet	(Simonyan & Zisserman, 2014)	59.4	-
VideoDarwin	(Fernando et al., 2015)	63.7	73.7
Multi-skIp Feature Stacking	(Lan et al., 2014)	65.1	68.0
Traditional+Stacked Fisher Vectors	(Peng et al., 2014)	66.8	-
Objects+Traditional+Stacked Fisher Vectors	(Jain et al., 2015)	71.3	66.4

(a) Correctly classified as "cycling"

(b) Correctly classified as "walking with a dog"

Figure 2.3: Attention over time. The model learns to look at the relevant parts - the cycle frame in (a) and the human and the dogs in (b)

(a) Incorrectly classified as "diving"

(b) Incorrectly classified as "volleyball"

Figure 2.4: Video frames for a few time-steps for an example of soccer played on a basketball court. Different glimpses can result in different predictions. Best viewed in color.

30-length samples. 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 clip, we average the predictions from all 30 frame blocks in the video.

Baselines

The softmax regression model uses the complete $7 \times 7 \times 1024$ feature cube as its input to predict the label at each time-step t, while all other models use only a 1024-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 location softmax. The inputs at

(a) Correctly classified as "swinging" (b) Correctly classified as "horse back riding"

Figure 2.5: Video frames where the model pays more attention to the background compared to the foreground and still classifies them correctly

each time-step for these models are obtained by doing average or max pooling over the $7 \times 7 \times 1024$ cube to get 1024 dimensional slices, whereas our soft attention model dynamically weights the slices by the location softmax (see Eq. 2.5).

2.2.3 Quantitative analysis

Table 2.1 reports accuracies on both UCF-11 and HMDB-51 datasets and mean average precision (mAP) on Hollywood2. Even though the softmax regression baseline is given the complete $7 \times 7 \times 1024$ cube as its input, it performs worse than our model for all three datasets and worse than all models in the case of HMDB-51 and Hollywood2. The results from Table 2.1 demonstrate that our attention model performs better than both average and max pooled LSTMs.

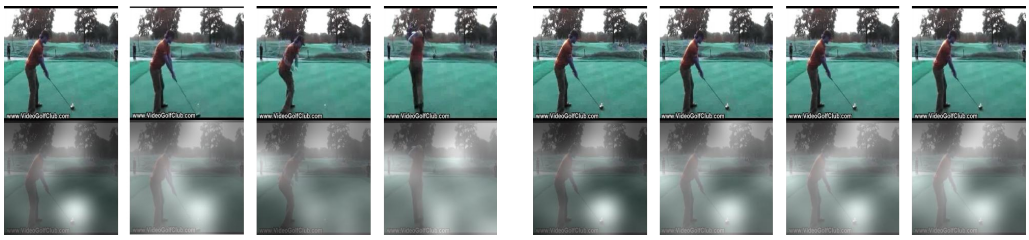
We next experimented with doubly stochastic penalty term λ (see Eq. 3.6). Fig. 2.2a shows that with no attention regularization term, $\lambda = 0$, the model tends to vary its attention less. Setting $\lambda = 1$ encourages the model to further explore different gaze locations. The model with $\lambda = 10$ looks everywhere (see Fig. 2.2c), in which case its behavior tends to become similar to the average pooling case. Values in between these correspond to dynamic weighted averaging of the slices. The models with $\lambda = 0$ and $\lambda = 1$ perform better than the models with $\lambda = 10$.

In Table 2.2, we compare the performance of our model with other state-of-the-art action recognition models. We do not include UCF-11 here due to the lack of standard train-test splits. We have divided the table into three sections. Models in the first section use only RGB data while models in the second section use both RGB and optical flow data. The model in the third section uses both RGB, optical flow, as well as object responses of the videos on some ImageNet categories. Our model performs competitively against deep learning models in its category (models using RGB features only), while providing some insight into where the neural network is looking.

2.2.4 Qualitative analysis

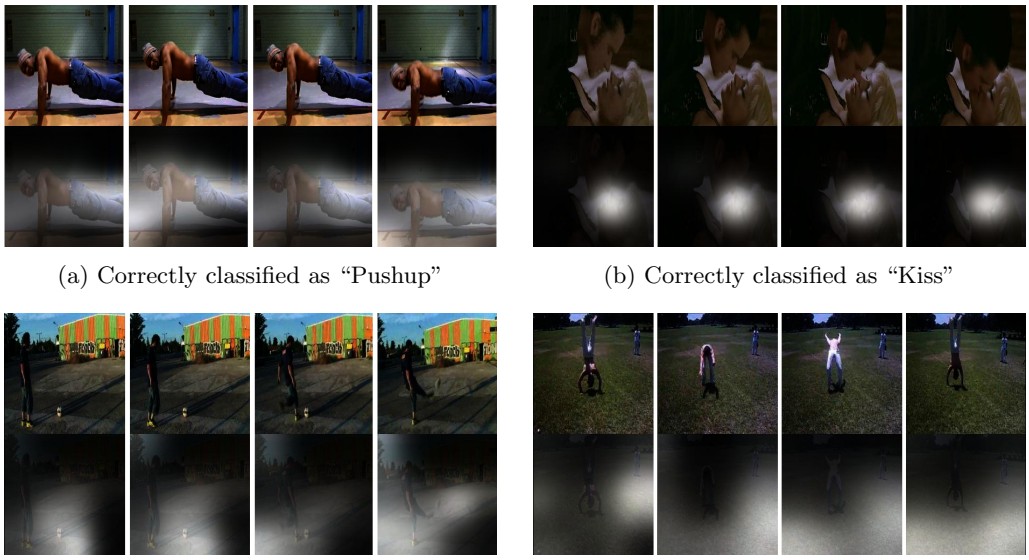
Fig. 2.3 shows some test examples of where our model attends to on UCF-11 dataset. In Fig. 2.3a, we see that the model was able to focus on parts of the cycle, while correctly recognizing the activity as "cycling". Similarly, in Fig. 2.3b, the model attends to the dogs and classifies the activity as "walking with a dog".

We can also better understand failures of the model using the attention mechanism. For example, Fig. 2.4a shows that the model mostly attends to the background like the light blue floor of the court. The model incorrectly classifies the example as "diving". However, using a different manually specified



(a) "golf swinging" (@ 6fps, $\lambda = 1$) (b) "golf swinging" (@ 30 fps, $\lambda = 1$)

Figure 2.6: The model's focus of attention visualized over four equally spaced timesteps at different fps rates. (a) plays faster and when the ball is hit and the club disappears, the model searches around to find them. (b) plays slower and the model stays focused on the ball and the club.



(a) Correctly classified as "Pushup" (b) Correctly classified as "Kiss"

(c) Inorrectly classified as "Somersault" (d) Incorrectly classified as "Hit"

Figure 2.7: Visualization of the focus of attention for four videos from HMDB-51 and Hollywood2 datasets over time. The white regions are where the model is looking and the brightness indicates the strength of focus.

glimpse, as shown in Fig. 2.4b, the model classifies the same example as "volleyball spiking". It is quite interesting to see that we can better understand the success and failure cases of this deep attention model by visualizing where it attends to.[1] The model does not always need to attend to the foreground. In many cases the camera is far away and it may be difficult to make out what the humans are doing or what the objects in the frames are. In these cases the model tends to look at the background and tries to infer the activity from the information in the background. For example, the model can look at the basketball court in the background and predict the action being performed. Thus, depending on the video both foreground and background might be important for activity recognition. Some examples are shown in Fig. 2.5, where the model appears to look everywhere.

It is also interesting to observe that in some cases, the model is able to attend to important objects in the video frames and attempts to track them to some extent in order to correctly identify the performed activity. In Fig. 2.6b, the video is sampled at 30fps and subsequent frames are almost identical. In this case the model stays focused on the golf ball, club, and the human. However, when we change the

[1] All the figures are from our best performing models with $\lambda = 0$ unless otherwise mentioned.

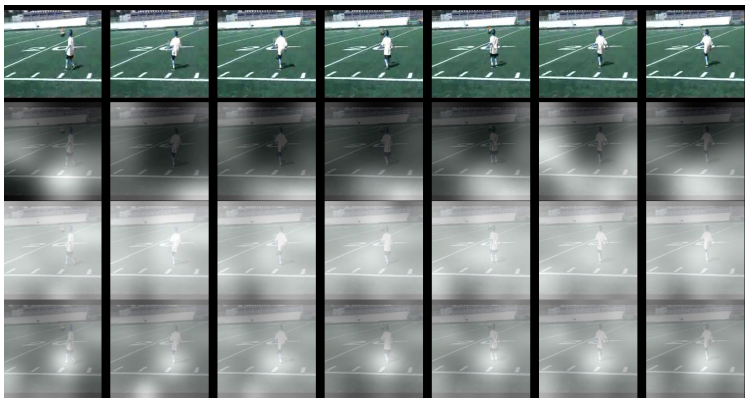


Figure 2.8: (**First**) The original video frames for a "soccer juggling" example from UCF-11 (**Second**). Glimpse of model with $\lambda = 1$ overlayed on the frames; predicted incorrectly as "tennis swinging" (**Third**) Randomly initialized glimpse overlayed on the frames; predicted incorrectly as "tennis swinging" (**Fourth**) The first glimpse at which the action is correctly predicted as "soccer juggling", overlayed on the frames

sampling rate to 6fps, as shown in Fig. 2.6a, we find that the video frames change quickly. The model now remains focused on the ball before it disappears. After the person hits the ball, we see that the model tries to look at other places, possibly to track the ball and the golf club. We next examined the model's performance on the HMDB-51 dataset.[2] In Fig. 2.7a the model attempts to focus on the person performing push-ups to recognize "Pushup" activity. In Fig. 2.7c the model classifies the example of "KickBall" incorrectly as "Somersault" despite attending to the location where the action is happening. In some cases, however, the model fails to even attend to the relevant location (see Fig. 2.7d). For Hollywood2, Fig. 2.7b shows an example of a short clip belonging to the "Kiss" action. It appears that the model correctly anticipates that a kiss is going to take place and attempts to focus on the region between the man and the woman.

In our final set of experiments, we have tried to examine some failure cases of our attention mechanism. As an example, Fig. 2.8 shows a test video clip of "soccer juggling" (top row). Our model focuses on the white boundaries of the field (second row), while incorrectly recognizing the activity as "tennis swinging". To see whether we can potentially correct the model's mistake by forcing it to look at the relevant locations, we took a trained model and initialized the location softmax weights to uniform random numbers between the minimum and maximum in the original model. The model's glimpse in this case is shown in the third row of Fig. 2.8. We next optimized only the softmax weights, or the location variables, for this specific example of "soccer juggling" to find the glimpse for which the model would predict it correctly. All the other model parameters were kept fixed. Note that this only changes the sequences of glimpses, or where the model attends to, and not the model itself. It is interesting to see that in order to classify this video clip correctly, the glimpse the model learns (the fourth row of Fig. 2.8) tends to focus on the soccer player's legs.

[2]More examples of our model's attention are available in Appendix A and at `http://www.cs.toronto.edu/~shikhar/projects/action-recognition-attention`.

Chapter 3

Video Description using Visual Attention

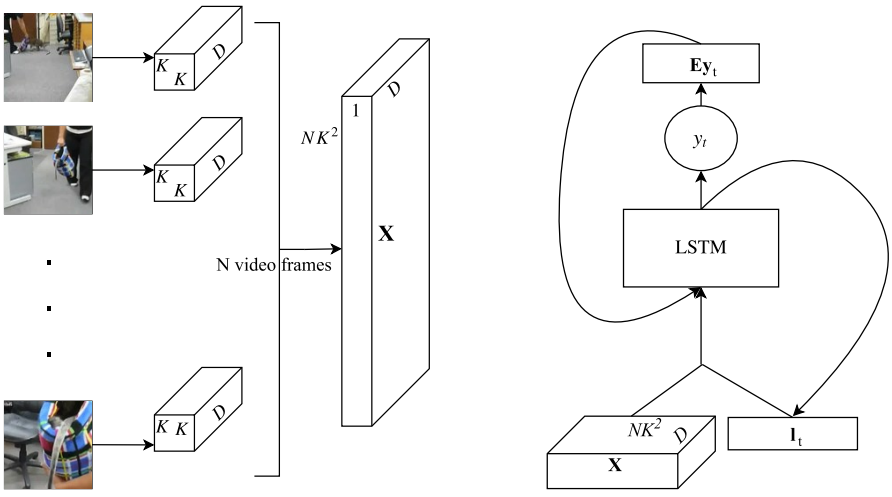
In this chapter, we describe the encoder-decoder soft attention based model that we use to generate descriptions of videos. We describe the YouTube2Text dataset. We present quantitative analysis of our model's performance against baselines and state-of-the-art models and qualitative analysis of the attention mechanism.

3.1 The Model and the Attention Mechanism

3.1.1 The Encoder

We extract the last convolutional layer obtained by pushing the video frames through GoogLeNet trained on the ImageNet dataset. We extract these K^2 D-dimensional vectors for N frames from each video.

$$\mathbf{X} = [\mathbf{X}_1, \ldots, \mathbf{X}_{NK^2}], \qquad \mathbf{X}_i \in \mathbb{R}^D.$$



(a) Spatio-Temporal Features (b) LSTM Decoder

Figure 3.1: (3.1a) Last convolutional layer features are extracted from video frames as the encoder features. (3.1b) The recurrent decoder performs joint word-video alignment.

3.1.2 The Decoder

At each time-step t, our model generates a word $\mathbf{y}_t$ of the video description $\mathbf{y}$ using a 1-of-V encoding.

$$\mathbf{y} = \{\mathbf{y}_1, \ldots, \mathbf{y}_C\}, \qquad \mathbf{y}_t \in \mathbb{R}^V$$

where C is the length of the description and V is the vocabulary size.

We use the LSTM implementation described in Sec. 2.1.2 with a slight modification:

$$\begin{pmatrix} \mathbf{i}_t \\ \mathbf{f}_t \\ \mathbf{o}_t \\ \mathbf{g}_t \end{pmatrix} = \begin{pmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{pmatrix} M \begin{pmatrix} \mathbf{E}\mathbf{y}_{t-1} \\ \mathbf{h}_{t-1} \\ \mathbf{x}_t \end{pmatrix} \tag{3.1}$$

where $\mathbf{x}_t$ (see Eqs. 3.2, 3.3) represents the input to the LSTM at time-step t. $\mathbf{E} \in \mathbb{R}^{n \times V}$ represents the word embedding matrix. $M : \mathbb{R}^a \to \mathbb{R}^b$ is an affine transformation consisting of trainable parameters with $a = d + D + n$ and $b = 4d$, where n is the word embedding dimensionality.

At each time-step t, our model predicts $\mathbf{l}_{t+1}$, a softmax over all the space-time locations, and a softmax over all the words. The one with the highest value is the predicted word $\mathbf{y}_t$ for each iteration. In practice, we actually use beam search to generate candidate sentences and score them and select the highest scoring sentence. The location softmax is defined as follows:

$$l_{t,i} = p(\mathbf{L}_t = i | \mathbf{h}_{t-1}) = \frac{\exp(W_i^\top \mathbf{h}_{t-1})}{\sum_{j=1}^{K \times K} \exp(W_j^\top \mathbf{h}_{t-1})} \qquad i \in 1, \ldots, NK^2, \tag{3.2}$$

where $\mathbf{L}_t$ is a random variable which can take 1-of-NK^2 values. This is the same as Eq. 2.4 with the minor change that $i \in 1, \ldots, NK^2$. After calculating these probabilities, the soft attention mechanism (Bahdanau et al., 2015) computes the expected value of the input at the next time-step $\mathbf{x}_t$ by taking expectation over the feature slices at different regions (see Fig. 2.1a):

$$\mathbf{x}_t = \beta_t \mathbb{E}_{p(\mathbf{L}_t | \mathbf{h}_{t-1})}[\mathbf{X}_t] = \beta_t \sum_{i=1}^{NK^2} l_{t,i} \mathbf{X}_i, \tag{3.3}$$

where $\mathbf{X}$ is the feature cube and $\mathbf{X}_i$ is the i^{th} slice of the feature cube at time-step t. β_t is a penalty term described in Sec. 3.1.3. This equation is the same as Eq. 2.5 except for the gating scalar β_t. β_t is necessary for regulating the relative importance of the input at each time-step for generating the output word. We found that introducing the gating scalar improved performance compared to when it was absent. We use the initialization strategy from Eq. 2.6 for the cell state and the hidden state of the LSTM for faster convergence, again, with the minor change that $i \in 1, \ldots, NK^2$.

The word probability at each time-step is calculated using a deep output layer (Pascanu et al., 2013) as follows:

$$p(\mathbf{y}_t | \mathbf{X}, \mathbf{y}_{t-1}) \propto \exp(\mathbf{W}_p(\mathbf{E}\mathbf{y}_{t-1} + \mathbf{W}_h \mathbf{h}_t + \mathbf{W}_x \mathbf{x}_t)) \tag{3.4}$$

where $\mathbf{W}_p$, $\mathbf{W}_h$ and $\mathbf{W}_x$ are learned weights. $\mathbf{W}_p \in \mathbb{R}^{V \times n}$, $\mathbf{W}_h \in \mathbb{R}^{n \times d}$ and $\mathbf{W}_x \in \mathbb{R}^{n \times D}$.

3.1.3 Loss Function and the Attention Penalty

We use negative log probability of the generated description as the loss coupled with the doubly stochastic penalty introduced in Xu et al. (2015). We impose an additional constraint over the location softmax, so that $\sum_{t=1}^{T} l_{t_i} \approx 1$. This is the attention regularization which forces the model to look at each region of the frame at some point in time. At each time-step we also predict β_t, a scalar which gates $\mathbf{x}_t$.

$$\beta_t = \sigma(f_\beta(\mathbf{h}_{t-1})) \tag{3.5}$$

where σ is the sigmoid function and f_β is a multilayer perceptron.

Compared to Chapter 2 we use the negative log-likelihood instead of cross-entropy as the loss function:

$$L = -\log P(\mathbf{y}|\mathbf{x}) + \lambda \sum_{i=1}^{NK^2} (1 - \sum_{t=1}^{C} l_{t_i})^2 + \gamma \sum_i \sum_j \theta_{i,j}^2, \tag{3.6}$$

where λ is the attention penalty coefficient, γ is the weight decay coefficient, and θ represents all the model parameters.

3.2 Experiments

3.2.1 Dataset

We evaluate our model on the YouTube2Text dataset (Chen & Dolan, 2011). The YouTube2Text dataset comprises of 1,970 videos from YouTube spanning several different domains. We use the standard split used by prior work (Venugopalan et al., 2014; Yao et al., 2015) which has 1,200 training videos, 100 validation videos and 670 testing videos. There are approximately 80,000 English descriptions available for these 1,970 videos with a vocabulary of roughly 15,000 words.

3.2.2 Training Details and Evaluation

In our experiments, model architecture and various other hyper-parameters were set using the validation set. We found 1-layer LSTM models to work best, where the dimensionality of the word embedding layer n was set to 512 and the dimensionality of the LSTM hidden state, cell state, and the hidden layer d was set to 4096 for the best performance. The highest scores on BLEU-4 and METEOR were obtained when beam search width was 4. For the attention penalty coefficient *lambda* the value 1 was found to work best. We use a weight decay penalty of 10^{-5} and dropout of 0.5 at all non-recurrent connections. All models were trained using the Adam optimization algorithm for 15 epochs over the entire dataset. However, we found that Adam usually converged after 8 epochs.

To ensure comparability of results with prior work (Yao et al., 2015), we keep only the first 240 frames from each video. If the video is shorter than 240 frames we pad it with blank black frames. From these 240 frames, we extract $N = 26$ evenly spaced frames. Our spatio-temporal attention model generates descriptions for the videos while looking at these frames. In each training epoch, the model is trained using all the descriptions for each training video.

Table 3.1: Comparison of performance on YouTube2Text dataset against baselines

	BLEU				
Model	B-1	B-2	B-3	B-4	**METEOR**
STA-avg-pool	0.712	0.561	0.463	0.362	0.256
STA-first-time-step	0.712	0.554	0.437	0.264	0.233
STA-soft	0.760	0.629	0.523	0.371	0.246

Table 3.2: Comparison of performance on YouTube2Text dataset with state-of-the-art models

	BLEU				
Model	B-1	B-2	B-3	B-4	**METEOR**
LSTM-YT (Venugopalan et al., 2014)	-	-	-	0.333	0.291
STA-soft (Our model)	0.760	0.629	0.523	0.371	0.246
LSTM-E-Vgg (Pan et al., 2015)	0.749	0.609	0.506	0.402	0.295
Temporal Attention (Yao et al., 2015)	0.791	0.632	0.512	0.403	0.290
h-RNN-Vgg (Yu et al., 2015)	0.773	0.645	0.546	0.443	0.311
Temporal Attention (Yao et al., 2015)	0.800	0.647	0.526	0.419	0.296
LSTM-E (Pan et al., 2015)	0.788	0.660	0.554	0.453	0.310
h-RNN (Yu et al., 2015)	0.815	0.704	0.604	0.499	0.326

Baselines

We compare our model STA-soft against two baselines: STA-avg-pool which pays equal attention to all the feature locations and is equivalent to average pooling over the feature cube, and, STA-first-time-step which differs from our attention model in that it receives the feature input at the first time-step only and not at all time-steps.

3.2.3 Quantitative Analysis

Table 3.1 shows the comparison of performance of our model against the baselines for all the BLEU and METEOR metrics. We found that our model performed better than the STA-avg-pool model and much better than the STA-first-time-step baseline which was fed the encoder features only at the first time-step, in terms of BLEU scores.

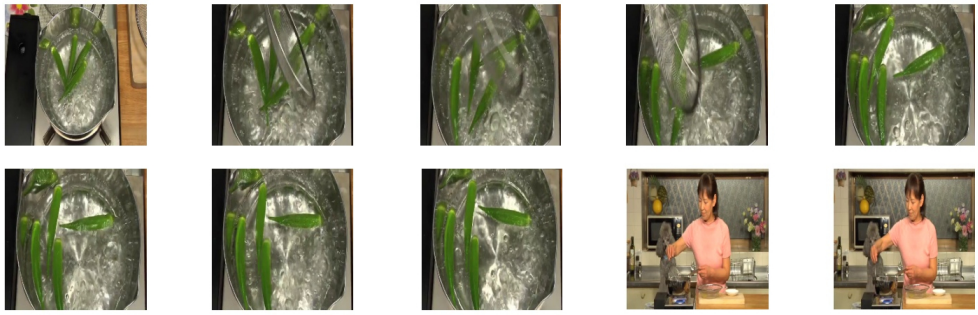
In Table 3.2, we compare the performance of our model with other state-of-the-art video description models. We have divided the table into two sections. Models in the first section use only RGB data while models in the second section use other features apart from RGB including C3D, optical flow, HOG, HOF, MBH. Our model does not beat the state-of-the-art results but it provides some insight into where the neural network is looking.

3.2.4 Qualitative Analysis

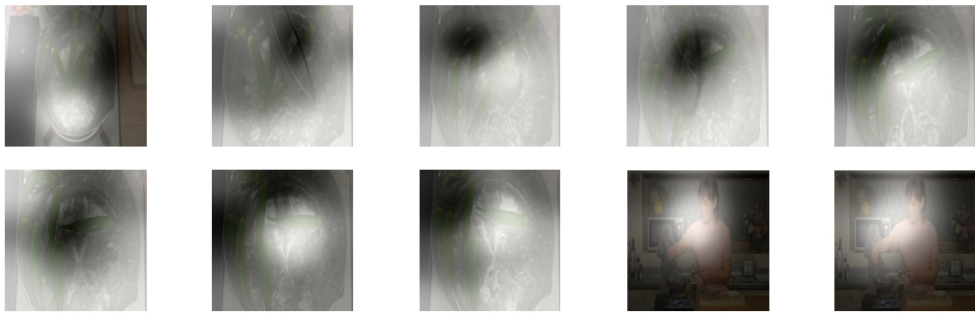
Fig. 3.2 and Fig. 3.3 illustrate the attention maps which tell us where our model attends to, across all the frames while generating different words in the description. They also show β_t for each time-step. We see that the model attaches highest importance to the first time-step and looks at all the relevant areas for generating a description and incorporates them in its hidden state. The β_t value continues to

decrease significantly after that, as it appears that the model has enough information to generate the description.

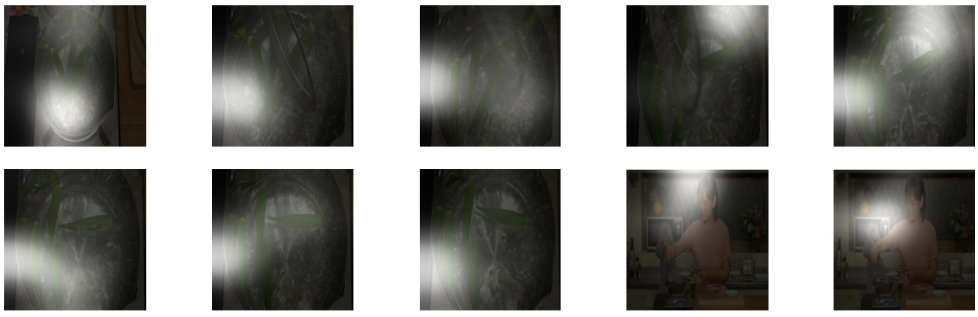
Fig. 3.2b represents the attention for the first time-step where the model generates the word 'a'. In the second time-step shown in Fig. 3.2c, the model generates the word 'woman'. However, $\beta_2 = 0.04$ so the glimpse at $t = 2$ doesn't affect the model's decision much and it has already looked at the woman at $t = 1$ as shown in Fig. 3.2b. We can thus see that the model pays attention to the woman and the food being cooked in the initial time-step in Fig. 3.2 and generates the sentence "a woman is cooking". In the second example it looks at the woman and the food being sliced in Fig. 3.3 while generating "a woman is slicing an onion". In general, for this dataset, we found that the model actually tries to focus on objects appearing in the predicted description within the first time-step.



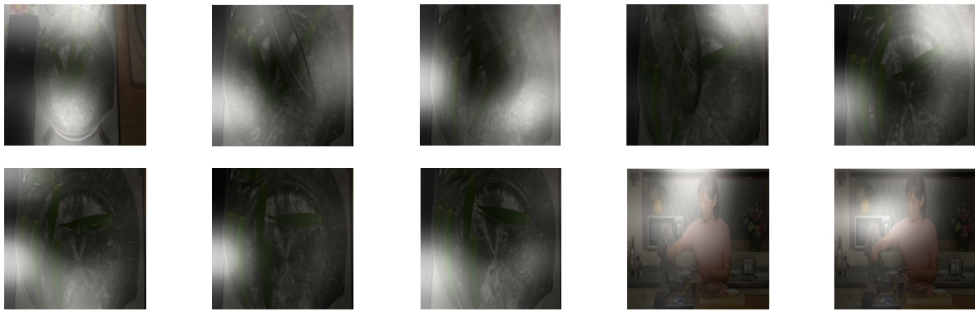
(a) Original video frames



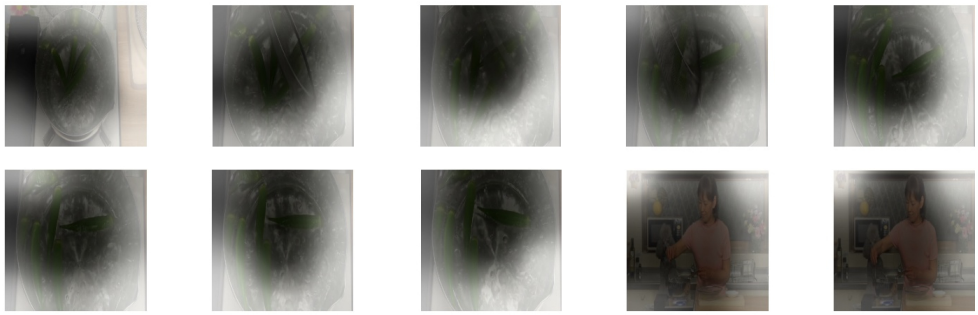
(b) Attention map for $t = 1$. $\beta_1 = 1.00$



(c) Attention map for $t = 2$. $\beta_2 = 0.04$



(d) Attention map for $t = 3$. $\beta_3 = 0.01$

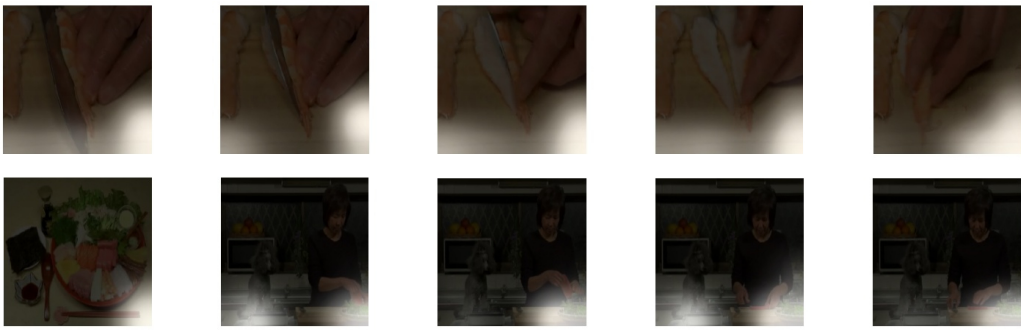


(e) Attention map for $t = 4$. $\beta_4 = 0.00$

Figure 3.2:
Attention map for our model STA-soft
Generated: a woman is cooking
GT: a woman is boiling okra

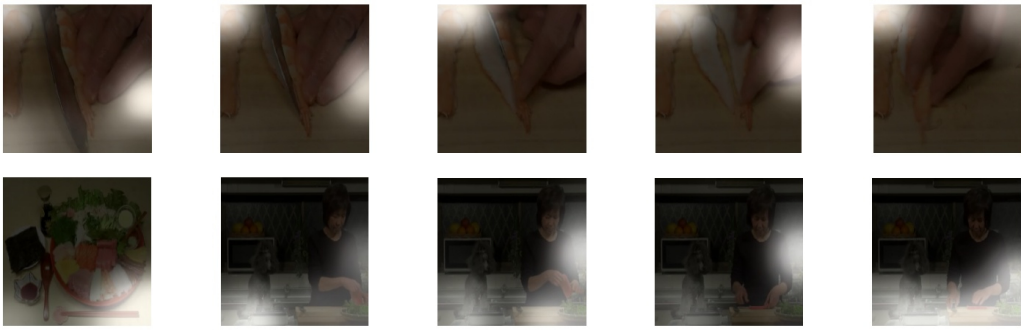
(a) Original video frames

(b) Attention map for $t = 1$. $\beta_1 = 1.00$

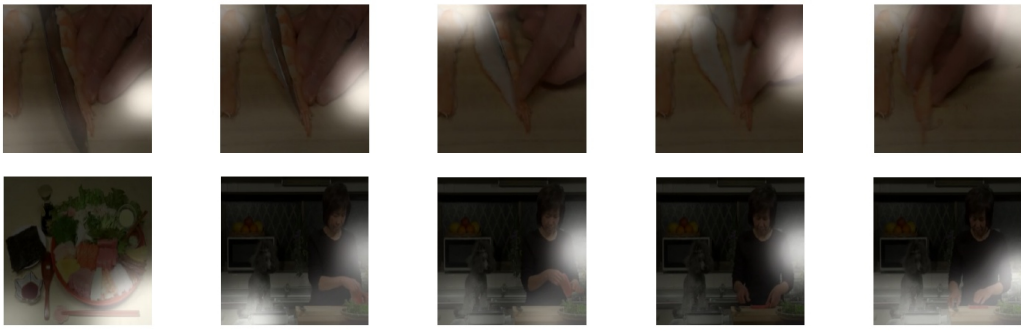


(c) Attention map for $t = 2$. $\beta_2 = 0.70$

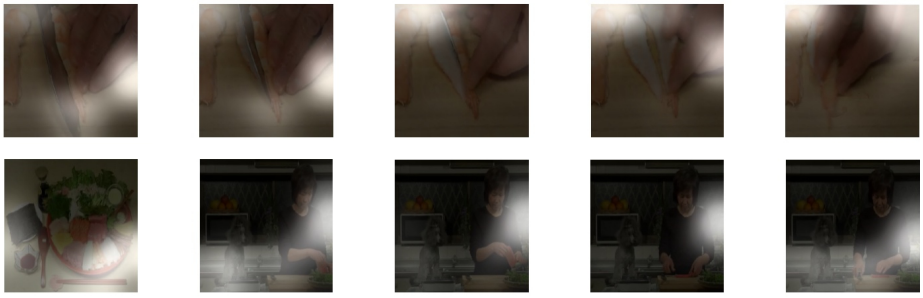
(d) Attention map for $t = 3$. $\beta_3 = 0.83$



(e) Attention map for $t = 4$. $\beta_4 = 0.65$



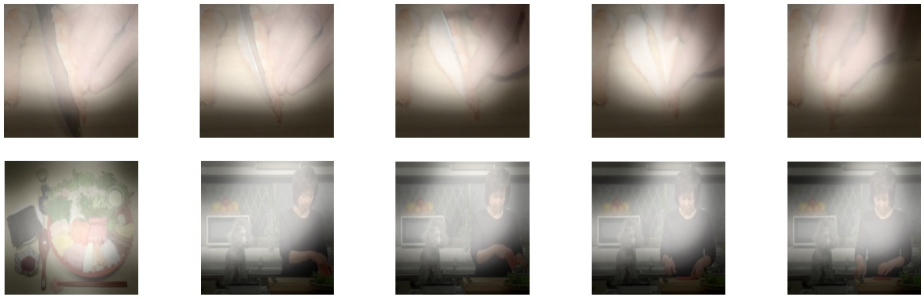
(f) Attention map for $t = 5$. $\beta_5 = 0.35$



(g) Attention map for $t = 6$. $\beta_6 = 0.19$

Figure 3.3:
Attention map for our model STA-soft
Generated: a woman is slicing an onion
GT: a woman is slicing shrimps

(a) Original video frames



(b) Attention map for $t = 1$

Figure 3.4:
Attention map for the baseline STA-first-time-step
Generated: a woman is slicing a guitar
GT: a woman is slicing shrimps

Figure Fig. 3.4 shows an example of the same video shown in Fig. 3.3 for the baseline STA-first-time-step. We see that the model tries to look everywhere when it is restricted to just an initial glimpse of the input. In this example, however, the model incorrectly predicts the word 'guitar'. This baseline performs worse than our model STA-soft for BLEU-4 by a wide margin but it is close for all the other metrics.

In general, for video description, we found that we couldn't gain much value from our attention mechanism. With better attention mechanisms and more general and larger datasets in the future, we expect the performance of soft attention based models to improve.

Chapter 4

Conclusion and Future Work

In this chapter we summarize the work done in this thesis and we present conclusions drawn from the conducted experiments which are described in Chapter 2 and Chapter 3. We also outline future directions for continuing work on attention models for video action recognition and multimodal video description generation tasks.

4.1 Conclusion

In this thesis we developed recurrent soft attention based models for action recognition and video description and analyzed where they focus their attention. Our proposed model tends to recognize important elements in video frames based on the activity that is being performed. We also showed that our models perform comparably with baselines which do not use any attention mechanism and use similar features. For action recognition on the HMDB-51 dataset, our model beat the average and max pooled baselines by a small margin ($\sim$1%) and provided insight about the model's attention and was $\sim$8% more accurate than the regression baseline. We observed a similar improvement of $\sim$1% in BLEU-4 scores over the average pooling baseline in generating video descriptions on the YouTube2Text dataset. While this improvement is not very significant, these models seem to recognize and attend to important objects. There is no perfect way yet to attend to video dynamics like humans do but with better attention models we expect performance to rise significantly as has been seen on Image Captioning and other tasks.

Soft attention models, though impressive, are still computationally expensive since they require computing convolutional features at all locations to perform dynamic averaging. The size of video datasets has been increasing to leverage deep learning to its full potential and overcoming their computational cost is one of the most significant challenges faced by attention models.

4.2 Future Work

In the future, we plan to explore other soft attention architectures, hard attention models as well as hybrid soft and hard attention models which can improve performance on action recognition and description generation tasks. Some of these would potentially have lower computational cost compared to our current model and can scale to larger datasets like UCF-101, Sports-1M and the DVS dataset.

Experiments done on these datasets would be more representative of the performance of attention models and can help to avoid overfitting which we observed occurred very quickly on datasets we have used in this thesis.

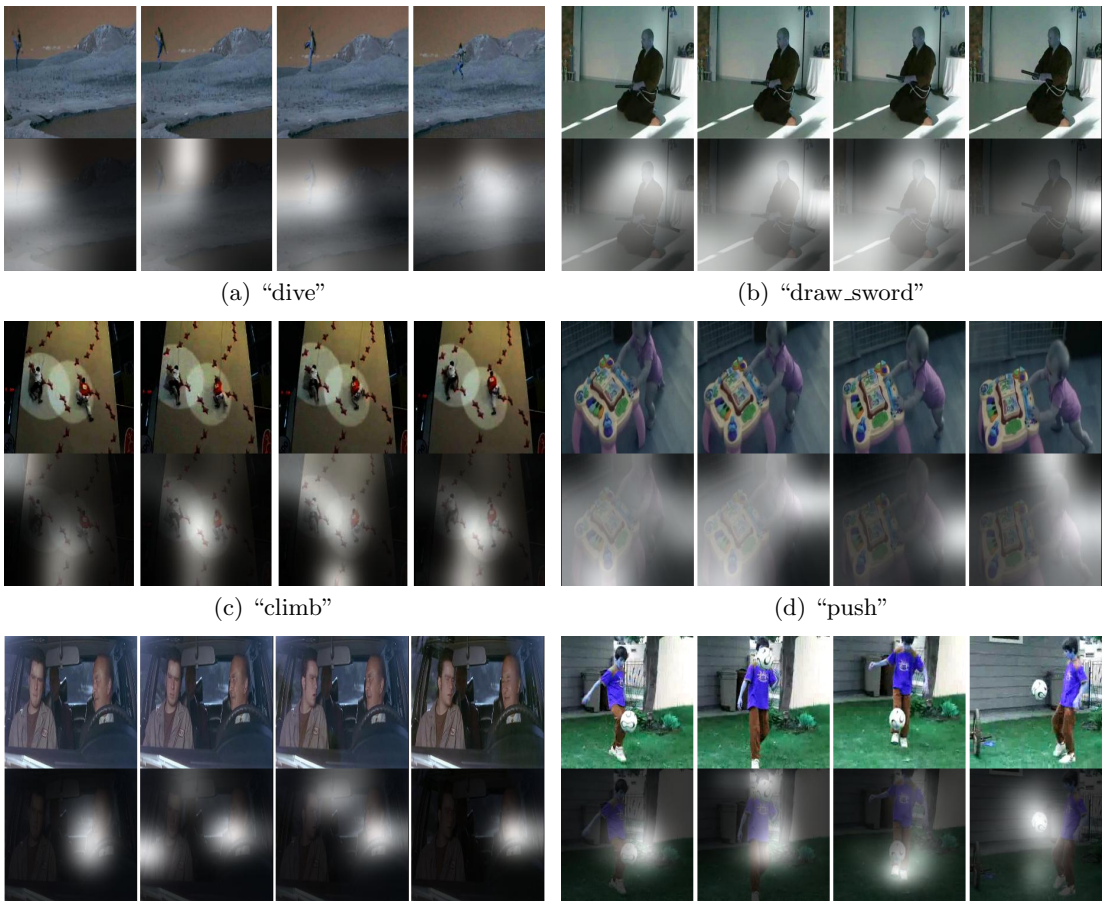
Allowing our model access to lower convolutional layers as well in a multi-resolution setting would be an extension to the current attention mechanism which can be explored further. We believe that being able to focus on both high-level and low-level features in video frames explicitly would be beneficial for performance and would shed further light on what these models choose to attend to. Soft attention models which attend only to the top-k locations also show some promise. Soft and hard attention models both have separate advantages and disadvantages. Thus, we also plan to explore and compare the performance of hard attention models with the current soft attention model. Another promising direction would be to explore hybrid approaches like sampling multiple locations using a hard attention mechanism and then taking a soft weighted average over their features.

This is an exciting time for research on multimodal language and visual representations. The Computer Vision and Natural Language Processing communities have been very active in this space and we are seeing state-of-the-art methods being beaten quickly and repeatedly. We believe that attention models would contribute significantly in the future of learning better multimodal representations.

Appendix A

Appendix

We present some more correctly classified examples from UCF-11, HMDB-51 and Hollywood2 in Fig. A.1 and incorrectly classified examples in Fig. A.2.



(a) "dive" (b) "draw_sword"

(c) "climb" (d) "push"

(e) "DriveCar" (f) "soccer juggling"

Figure A.1: Correctly classified video frames showing attention over time: The white regions are where the model is looking and the brightness indicates the strength of focus. The model learns to look at relevant parts.

(a) “pour” misclassified as “push” (b) “laugh” misclassified as “smile”

Figure A.2: Incorrectly classified video frames showing attention over time: The white regions are where the model is looking and the brightness indicates the strength of focus.

Bibliography

Ba, J., Grosse, R., Salakhutdinov, R., and Frey, B. Learning wake-sleep recurrent attention models. In *NIPS*, 2015a.

Ba, J., Mnih, V., and Kavukcuoglu, K. Multiple object recognition with visual attention. *ICLR*, 2015b.

Bahdanau, D., Cho, K., and Bengio, Y. Neural machine translation by jointly learning to align and translate. *ICLR*, 2015.

Ballas, N., Yao, L., Pal, C., and Courville, A. C. Delving deeper into convolutional networks for learning video representations. *CoRR*, abs/1511.06432, 2015.

Bastien, F., Lamblin, P., Pascanu, R., Bergstra, J., Goodfellow, I. J., Bergeron, A., Bouchard, N., Warde-Farley, D., and Bengio, Y. Theano: new features and speed improvements. *CoRR*, abs/1211.5590, 2012.

Chen, D. and Dolan, W. B. Collecting highly parallel data for paraphrase evaluation. In *ACL HLT*, 2011.

Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., and Li, F.-F. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.

Donahue, J., Hendricks, L. A., Guadarrama, S., Rohrbach, M., Venugopalan, S., Saenko, K., and Darrell, T. Long-term recurrent convolutional networks for visual recognition and description. In *CVPR*, 2015.

Fernando, B., Gavves, E., Oramas, J., Ghodrati, A., and Tuytelaars, T. Modeling video evolution for action recognition. In *CVPR*, 2015.

Graves, A., Jaitly, N., and Mohamed, A.-r. Hybrid speech recognition with deep bidirectional LSTM. In *2013 IEEE Workshop on Automatic Speech Recognition and Understanding*, pp. 273–278. IEEE, 2013.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural Computation*, 9(8):1735–1780, 1997.

Jaderberg, M., Simonyan, K., Zisserman, A., and Kavukcuoglu, K. Spatial transformer networks. *CoRR*, abs/1506.02025, 2015.

Jain, M., v. Gemert, J. C., and Snoek, C. G. M. What do 15,000 object categories tell us about classifying and localizing actions? In *CVPR*, June 2015.

Karpathy, A., Toderici, G., Shetty, S., Leung, T., Sukthankar, R., and Li, F.-F. Large-scale video classification with convolutional neural networks. In *CVPR*, 2014.

Kingma, D. P. and Ba, J. Adam: A method for stochastic optimization. *ICLR*, 2015.

Lan, Z.-Z., Lin, M., Li, X., Hauptmann, A. G., and Raj, B. Beyond gaussian pyramid: Multi-skip feature stacking for action recognition. *CoRR*, abs/1411.6660, 2014.

Mnih, V., Heess, N., Graves, A., and Kavukcuoglu, K. Recurrent models of visual attention. In *NIPS*, 2014.

Netzer, Y., Wang, T., Coates, A., Bissacco, A., Wu, B., and Ng, A. Y. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning 2011*, 2011.

Ng, J. Y.-H., Hausknecht, M. J., Vijayanarasimhan, S., Vinyals, O., Monga, R., and Toderici, G. Beyond short snippets: Deep networks for video classification. In *CVPR*, 2015.

Pan, Y., Mei, T., Yao, T., Li, H., and Rui, Y. Jointly modeling embedding and translation to bridge video and language. *CoRR*, abs/1505.01861, 2015.

Pascanu, R., Gülçehre, Ç., Cho, K., and Bengio, Y. How to construct deep recurrent neural networks. *CoRR*, abs/1312.6026, 2013.

Peng, X., Zou, C., Qiao, Y., and Peng, Q. Action recognition with stacked fisher vectors. In *ECCV*, volume 8693, pp. 581–595. Springer, 2014.

Ren, S., He, K., Girshick, R. B., Zhang, X., and Sun, J. Object detection networks on convolutional feature maps. *CoRR*, abs/1504.06066, 2015.

Rensink, R. A. The dynamic representation of scenes. *Visual Cognition*, 7(1-3):17–42, 2000.

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

Srivastava, N., Hinton, G. E., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. Dropout: a simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

Srivastava, N., Mansimov, E., and Salakhutdinov, R. Unsupervised learning of video representations using LSTMs. *ICML*, 2015.

Sun, L., Jia, K., Chan, T.-H., Fang, Y., Wang, G., and Yan, S. DL-SFA: deeply-learned slow feature analysis for action recognition. In *CVPR*, 2014.

Sutskever, I., Vinyals, O., and Le, Q. V. V. Sequence to sequence learning with neural networks. In *NIPS*. 2014.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., and Rabinovich, A. Going deeper with convolutions. In *CVPR*, 2015.

Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R. J., and Saenko, K. Translating videos to natural language using deep recurrent neural networks. *CoRR*, abs/1412.4729, 2014.

Venugopalan, S., Xu, H., Donahue, J., Rohrbach, M., Mooney, R. J., and Saenko, K. Translating videos to natural language using deep recurrent neural networks. In *NAACL HLT*, 2015.

Vinyals, O., Toshev, A., Bengio, S., and Erhan, D. Show and tell: A neural image caption generator. In *CVPR*, 2015.

Williams, R. J. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8:229–256, 1992.

Wu, R., Yan, S., Shan, Y., Dang, Q., and Sun, G. Deep image: Scaling up image recognition. *CoRR*, abs/1501.02876, 2015.

Xu, K., Ba, J., Kiros, R., Cho, K., Courville, A. C., Salakhutdinov, R., Zemel, R. S., and Bengio, Y. Show, attend and tell: Neural image caption generation with visual attention. *ICML*, 2015.

Yao, L., Torabi, A., Cho, K., Ballas, N., Pal, C., Larochelle, H., and Courville, A. Describing videos by exploiting temporal structure. *CoRR*, abs/1502.08029, 2015.

Yeung, S., Russakovsky, O., Jin, N., Andriluka, M., Mori, G., and Li, F.-F. Every moment counts: Dense detailed labeling of actions in complex videos. *CoRR*, abs/1507.05738, 2015.

Yu, H., Wang, J., Huang, Z., Yang, Y., and Xu, W. Video paragraph captioning using hierarchical recurrent neural networks. *CoRR*, abs/1510.07712, 2015.

Zaremba, W., Sutskever, I., and Vinyals, O. Recurrent neural network regularization. *CoRR*, abs/1409.2329, 2014.