Attention Based Hierarchical RNN Model

Markus Woodson
Electrical and Computer Engineering
mwoodson@andrew.cmu.edu

Xiang Xu
Electrical and Computer Engineering
xiangxu@andrew.cmu.edu

Yubo Zhang
Civil and Environmental Engineering
yuboz@andrew.cmu.edu

Abstract

Inspired by recent work in soft attention and hierarchical recurrent neural network models, we propose an attention based hierarchical model for image captioning and video recognition. Our model extracts features from different layers of the Convolutional Neural Network and enables implicitly regularizing the network based on previous attentions. We show that our model achieves state-of-the-art performance on MS COCO while learning where to focus at different time steps.

1. Introduction

We address the task of image captioning and action recognition in videos. With applications in social activity analysis, surveillance, event detection, auto summary and etc, it is obvious that having a model which can understand and represent image or frames from an video well is an important research task. A good model not only needs to learn to capture visual information across the spatial and temporal content but also needs to learn how to translate the visual information to natural language.

There has been many research on how to exploit the spatial information in an given image. One recent success is the the convolutional neural network architectures which builds upon a hierarchy model of local spatial information.[8, 14, 5] Because convolutional neural network is very good at recognizing objects in a given image, many image caption models and video recognition models use it as a visual information extractor.[20, 3]

There has also been previous work on how to exploit the temporal information found in video data. Some choose to incorporate temporal information by hand-crafting features [18, 11]. Others have focused on learning spatial-temporal filters [7, 16, 17]. More interestingly, some have used two-stream architectures where a networks are trained on RGB frames and optical frames and then fused together before performing the classification [13]. In most of these works they incorporate temporal information via some type of pooling such as average or max pooling [13, 7, 17, 21]. This strategy ignores temporal structure in longer videos. What is needed is a model which can learn to exploit multiple temporal scales jointly.

A natural class of model one would consider for modeling long term sequence such as a video would be the Recurrent Neural Network. One variant of the original RNN that works well in practice is the Long Short-Term Memory (LSTM) networks, pictured in Figure 1. Originally proposed in [6], LSTM networks have had much success in sequence modeling of both textual and video data [3, 21, 15, 12]. LSTM networks are a type of recurrent end-to-end architecture which receives sequence data as input and preserves an internal memory cell that remembers what has happened over the time of the sequence. Another popular variant of the RNN is the Gated Recurrent Unit, or GRU[2]. It has less parameters than LSTM but has shown similar performance for many tasks. Many caption models also use recurrent neural network to generate the caption while using the features extracted from

10807 Class Project (2016).

convolutional neural network as the input.[20, 1, 3, 21, 13] This kind of model can also be viewed as an encoder-decoder model. The convolutional neural network is the encoder that encodes visual information to the feature map. The role of the recurrent neural network then is to decode the feature into language caption which describes the image or a video in the case of video captioning.[2]

One question that researchers try to address with such model is how much information should flow from the encoder to the decoder. Convolutional neural network is a powerful feature extractor model, it has many layers of feature map with an increasing spatial size from top to lower layers. Early model only uses the fully connected layer at the top of the CNN model as inputs to RNN. This however, proves to be inefficient due to the lack of spatial information. Later models used the top convolution feature and this usually improves the performance. However, the convolution feature layer has a much larger dimension than the fully connected layer. Using all of the feature map from the top convolution layer is not only slow for computation but also need a model with more parameters and are prone to over-fitting. Another downside of such a model is that it in general LSTM or GRU cannot represent data with very long term dependencies. This is especially worse in tasks involving video data.

One solution to modeling large data has been the use of an attention mechanism. Much like in human brains, attention mechanisms allow the models to learn to focus on only the most important parts of the data. Works such as [9, 4] applied such an attention mechanism to the sequence modeling tasks of machine translation,image generation and image captioning. Others have approached the tasks of modeling video data using a soft-attention mechanism for action recognition [12, 10]. All of these works have shown that the use of an attention mechanism can often improve performance on various tasks but one common problem is these attention mechanisms are learned at a single spatial layer or at a single temporal scale.

In this paper, we proposed a attention based hierarchical model to address the previous concerns. We approach this by incorporating attention mechanism into the encoder in order to alleviate the amount of work that the decoder needs to do. We follow the similar approach from other papers and treat a convolutional neural network as the encoder and a recurrent neural network as the decoder. The only difference is that the decoder also tells the encoder which part of the feature map to pay attention to. We denote this as the attention mechanism in our model. So far, our method is similar to the one described in [20]. However, unlike their method which only applies attention mechanism on the top convolution feature layer, our model can generate hierarchical attention priors for many feature layers based on ones computed before. The model can then use the different feature maps to make better prediction. Our model is also simpler than the ones described in [20] both in terms of parameters per attention model and the overall pipeline.
The main contributions of this paper are the following:

(1) We introduced an attention mechanism for extracting local interest regions within different layers of feature maps. We also used the prior attention probability as an implicit method to regularize later attention region.

(2) We used the attention mechanism to extract features from different layers of the VGG16 net and show that having more layers improve performance on image captioning task.

(3) We apply our hierarchy attention model to video recognition.

2. Related Work

There are mainly three dominating approaches to leverage temporal information in tasks of action recognition. Some researches integrate spatial features and temporal features together when training detectors for action recognition [11, 18]. Christian et.al. [11] represented videos with local space-time features, and classified motions with SVM. Ju Sun et.al. modeled spatio-temporal context in hierarchical way by including local features, transition descriptor, and trajectory proximity descriptor. Considering spatio-temporal features increases performance of action recognition, since information along time scale reveals large quantities of motion features in videos; However, such methods only considers local spatial information and short term motion information without taking into account

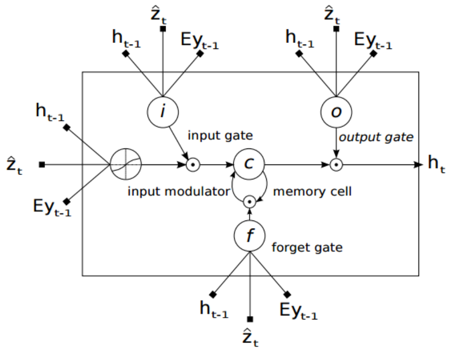


Figure 1: An image of a LSTM with memory cell

long-term structure. Recently, instead of using hand crafted features and generative models, some research focused on learning spatial-temporal filters [7, 13, 16, 17]. Shuiwang Ji et.al performed 3D convolution in Convolutional Neural Network which captures temporal information in adjacent frames. Karpathy et.al. [7] multiple models to expand connectivity in Convolutional Neural Network by using fusion of multiple frames or fusion of two single-frame networks. Simonyan et.al. [13] proposed a two-stream Convolutional Neural Network where one network utilized multi-frames' dense optical flow to explore temporal information. Such models apply simple temporal pooling or work on a sequence of frames directly, which lacks deep modeling on time dimension. Recurrent Neural Network(RNN) in perceptual applications enables modeling long term sequences in videos [6, 3]. LSTM [6] is a recurrent end-to-end architecture that takes a sequence of data as input and preserves internal memory cells which stores information along time series. Donahue et.al. [3] proposed Long-term Recurrent Convolutional Network(LRCN) that combines CNN as feature extractors with LSTM which enables modeling synthesize temporal dynamics.

The attention model enables Recurrent Neural Network to sequencely focus on a subset of inputs that are most important for model to use [4, 9, 10, 12, 7]. The image structure is easier to capture when analyzing partial images than when scanning all over images. Gregor et.al. [4]developed a structure with dynamicly updated attention mechanism which applies multiple scalable Gaussian filters. However, the algorithm alligns the attention center to the image center which fails when the object is on the side. [20] extracted features from Convolutional Neural Network and applied Recurrent Neural Network with attention based on the feature maps from CNN. The model incorporates attention on the level of frame and generate discription from object directly. All the works have shown that utilization of attention mechanism improves performance on various tasks. Our method does not only take advantage of spatial information by applying attention model but also combines multiple temporal scales into a hierarchical structure on both spatial and temporal dimensions.

3. Methods

In this section we will describe our image caption model and video recognition model. We made our code publicly available at: https://github.com/samxuxiang/Attention-Based-Hierarchical-Model-

3.1. Image Caption with Attention Based Hierarchical Model

we will first describe the detailed implementations of our image caption model. This includes the hierarchical attention mechanism, the encoder and the decoder.

3.1.1 Encoder

We use VGG16 as the encoder for extracting the features from an image. For simplicity, we shall assume that we will be using two layers of the convolution features as input to our decoding recurrent neural network. The CNN extractor produces L_1 vectors of dimension D_1 for the top layer (first

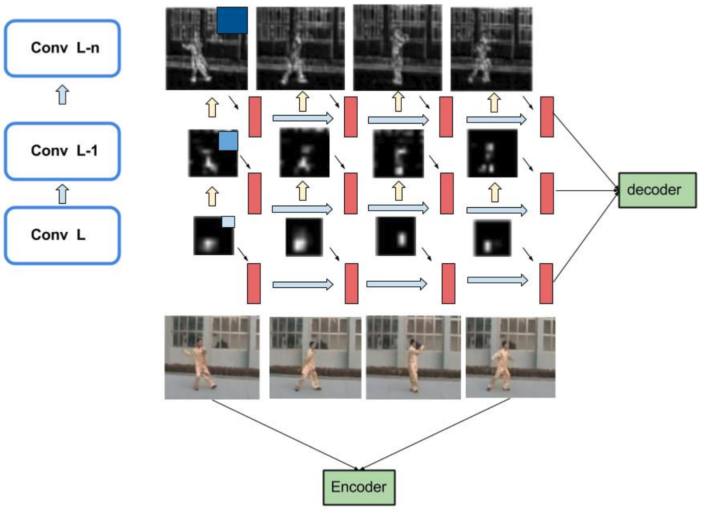


Figure 2: Attention based hierarchy RNN for both image captioning and video recognition. The red rectangles are the recurrent layers. Each image or frame will take the attention region probability from previous layer. The top layer probability is passed on to lower layer. Finally hidden states from different layers are used as the input to the decoder to predict next word. The blue region is a demo of how the probability score will propagate with an increase in receptive field. Note: for image captioning, use only one column and omit the vertical video frame RNN

layer) and L_2 vectors of dimension D_2 for the lower layer (second layer). For this paper, we use a VGG16 net and the top layer is pool5 feature map and the lower layer is pool4 feature map. Now we have two different feature maps:

$$\mathbf{a_1} = (a_1, ..., a_{L_1}), \quad a_i \in \mathbb{R}^{D_1}$$
$$\mathbf{a_2} = (a_2, ..., a_{L_2}), \quad a_i \in \mathbb{R}^{D_2}$$

For this paper, we do not fine-tuned the VGG16 net

3.1.2. Decoder

We use LSTM as the decoder. A picture of it is shown in figure 1. It generates a new caption y_t each step conditioned on the previous hidden state h_{t-1}, the current context z_t,and the previously generated word y_{t-1}. We also map each word through a word embedding matrix E. We compute the LSTM as follow where T is a transform matrix:

$$\left\{\begin{array}{c}\mathbf{i}_t\\ \mathbf{f}_t\\ \mathbf{o}_t\\ \mathbf{g}_t\end{array}\right\} = \left\{\begin{array}{c}\sigma\\ \sigma\\ \sigma\\ tanh\end{array}\right\} * T * \left\{\begin{array}{c}E * y_{t-1}\\ h_{t-1}\\ z_t\end{array}\right\}$$

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

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

Note that the context is computed from the hierarchical attention mechanism and thus originally have two different contexts of dimension D_1 and dimension D_2 from each feature map. In here we simply combine the two through a simple multi-layer neural network and the output is denoted z_t.

To generate a new word, we pass the previous word y_{t-1}, the current combined context z_t and the current hidden state h_t through a multi-layer neural network. The loss is the cross entropy logits loss with correct word as one hot vector.

For initialization of the hidden state and memory cell for LSTM. We pass the normalized first layer feature map through a multi-layer neural network.

3.1.3. Hierarchical Attention Mechanism

We follow the rule that we always process the top convolution feature first and then the lower convolution features. This has the advantage that top layer feature can add some prior belief about which region we should focus on and we can propagate that prior probability into lower layers. An example of this is shown in figure 2. From our recurrent neural network, at step t we have h_{t-1}, the hidden state from previous step. Using $\mathbf{h}_{t-1}$, we shall compute the attention probability for the current state $\mathbf{p}_t$. $\mathbf{p}_t$ has a dimension L_1 for the first layer feature and L_2 for the second layer feature. It represents the probability that each location in the feature map is important for generating captioning.

To compute $\mathbf{p}_{t,1}$, we first pass h_{t-1} through a one layer neural network so that the output dimension matches the feature map dimension D_1. The activation function is sigmoid and we view this as a feature extracting gate. If it's close to 0, then that feature unit will not be taken into consideration. Imagine that we have concluded based on h_{t-1} that the next word that we are looking for has a high probability of being an animal. Thus after parsing h_{t-1} into the extracting gate, we expect units that are representation of animals being closed to 1 and those that are not being closed to 0. To achieve this, we also add a L1 loss to encourage sparsity. We then perform a convolution between the feature map and the extractor by treating the extractor as the filter. The remaining part is the extracted features that we are interested in. The exact equation is as follow:

$$\mathbf{a_1} \odot sigmoid(h_{t-1} * w + b)$$

We then pass it through a multi-layer neural network and a softmax to transform it into probability score $\mathbf{p}_{t,1}$. We use the soft attention mechanism proposed in [20] to compute the final context value $\mathbf{z_1}$ for the first feature layer. To compute $\mathbf{p}_{t,2}$, we first reshape $\mathbf{p}_{t,1}$ into the same size as $\mathbf{p}_{t,2}$. We treat $\mathbf{p}_{t,1}$ as the prior probability of where the second layer should focus on. Because the units in each layer represent different features, we do not constraint the two probability to match exactly. Instead, we add first layer attention probability that are above average to the newly computed second layer attention probability. This will try to force the second layer to pay attention to where we have previously paid attention to without constraining it. The new second layer attention probability before adding the prior and the final context value $\mathbf{z_2}$ is computed in the same way as before. The procedure for adding more layers into consideration is also the same as this one. The exact equation is as follow:

$$softmax(\mathbf{p}_{t,2} + relu(\mathbf{p'}_{t,1} - (1.0/L_1)))$$

where $\mathbf{p'}_{t,1}$ is the resized image of $\mathbf{p}_{t,1}$ from L_1 to L_2

3.2. Video Recognition with Attention Based Hierarchical Model

we will now describe the detailed implementations of our video recognition model. In our project we build upon the work proposed in [19] by expanding the recurrent neural network to a hierarchy of recurrent networks. We argue that with knowledge at multiple temporal scales, the network should be able to out-perform other networks which only use one temporal scale or networks that learn knowledge at different temporal scales separately instead of jointly. The attention model also has a hierarchical architecture and it is built together with the recurrent networks.

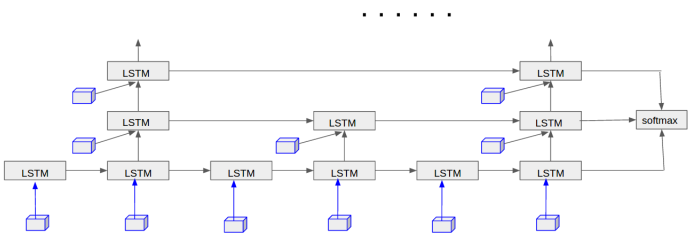


Figure 3: Here we show a 3 layer of our model. The outputs of each layer are then feed via a residual-like connection to a softmax output for prediction. At each successive layer we also feed convolutional features from layers lower in the hierarchy of a convolutional neural network, such as VGG to enforce that the higher layers should learn broader features. We only show the half of the network with RGB inputs but this network is replicated for the optical flow as well.

3.2.1 Hierarchical RNN

Our network has L layers of GRU or LSTM units followed by a dense linear layer with softmax activation which performs the prediction. Layer $L = 0$ receives $\mathbf{X}_{rgb}^{(0)}$ or $\mathbf{X}_{opt}^{(0)}$ as input. Layers $L = i$ will receive $\mathbf{X}_{rgb}^{(i)}$ or $\mathbf{X}_{opt}^{(i)}$ as input. We also proposed a more complicated hierarchical RNN model with skip connections. The new RNN model is shown in Figure 3. Different from our previous model, Layers $L = 1, 2, ...$ will receive a subset of the outputs from the lower layer. We compute this subset by skipping every kth encoded output from the previous layer. Here we just set $k = 2$. The reason why we want to use this model is because frames next to each other tend to contain similar information. We exploit this local temporal property in order to reduce the number of computation steps. Another thing that we want to try is to let the RNN network itself determine when to skip an entire frame. We can use another attention model to predict the probability that a frame is important or not. We can choose to delete the unimportant frames. However by doing so we will need to sample from the data, which can lead to increase in training time.

4. Results

In this section we outline the datasets used and our evaluation methods. We then give some details of our training procedure and present our results compared to the state-of-the-art.

4.1. Datasets

We evaluated our network on 2 different tasks, action recognition and image captioning. The standard benchmark for action recognition is the UCF-101 dataset []. This dataset contains $13,000$ videos spanning 101 different classes. Each video contains only a single action so no segmentation is necessary. We follow the first train/test split provided by the UCF-101 maintainers which is standard practice. For image captioning we test on the Microsoft COCO dataset which contains roughly 80,000 images for training and 40,000 for validation and testing. We use all training images for training, however there's no clear instruction on how to split the validation and test data. We follow the conventions in [20] and use roughly 5000 images for validation and 5000 images for testing. We eliminate words that occur less than 3 times and sentences longer than 15 words. The final vocabulary is approximately 10,000 words. We extract convolution features using a pre-trained VGG16 net. We resized images to 224x224 first and forward it through VGG16. We extracted pool5 layer of size 7x7x512 and pool4 layer of size 14x14x512. We also normalized the feature map by subtracting the mean and dividing it by the stand deviation.

4.2. Training Procedures and Quantitative Analysis

In this section, we will describe our train procedures for the two models as well as their evaluated performance

4.2.1. Video Action Recognition

We follow the feature extraction process outlined above. We use VGG19 as our convolutional network of choice and extract features from the frames at the last pooling layer. We sample the videos at an fps of $fs = 2$ and skipped every 5 frames between sequences extracted. The sequence length M was evaluated at $M = 30$ and $M = 60$ which can be seen in Table 1. Due to time constraints we were not able to integrate optical flow into our model so no features from the optical flow network were extracted.

fps	***UCF-101***
30	73%
60	75%

Table 1: Performance on the UCF-101 dataset action recognition task using different fps of the video.

Our first layer of our hierarchical LSTM model we use $h = 512$ hidden units. As with most networks, our network benefits more from increased number of layers so we evaluate performance with $l = 1, 2, 3$ layers using and fps of $M = 30$ in Table 2. Though it is possible to increase the number of layers since we halve the number of time steps per layer the contribution of higher layers starts to diminish in importance. We note that the diminishing returns from more levels in our hierarchy can be averted by increasing the sequence length of our data but this increases computation time significantly.

Layers	***UCF-101 Test Performance***
1	73%
2	77.4%
3	79.8%

Table 2: The impact of the number of layers on the model is consistent with other deep learning models. In particular we observe that more layers in our hierarchy leads to improved performance. We also observe diminishing returns on increased layers due to our fixed time scale of only 30 frames.

As we have noted before it has been shown that adding a soft attention mechanism in models involving images tends to improve performance. In Table 3 we show our results compared to state of the art using our hierarchical attention mechanism and without said attention mechanism. You can see that attention improves performance overall but our hierarchical attention does even better than a normal soft attention mechanism using outputs from only one layer in the network.

Model	***UCF-101 Test Performance***
Histogram of Oriented Gradient	72.4%
Improved Dense Trajectories	85.9%
spatial-temporal CNN	65.4%
two stream CNN	88.0%
two stream CNN + LSTM	88.6%
soft attention + LSTM	84.96%
Hierarchical RNN	79.8%
Hierarchical RNN + spatial soft attention	82.0%
Hierarchical RNN + Hierarchical attention	82.6%

Table 3: Comparison with different state of the art methods on the UCF-101 dataset

4.2.2. Image Captioning

We apply dropout of 0.5 during training for the decoding layer in LSTM. For regularization, we add the L1 sparsity loss for the feature extractor. We do not explicitly add any other weight decay to the loss. We also add two regularization based on the generated attention probability. The first one is the doubly stochastic attention loss from [20] where we encourage the model to look at different part of an image when generating different words. We also add a new entropy loss where we compute the entropy for each attention probability value. This allows the model to try and focus on a smaller region during each time step. For the LSTM, we set the hidden dimension to 1024, the word embedded dimension to 512, and the combined context dimension to 512. We use adam and a learning rate of 0.001 for optimization. We also use early stopping. The batch size is 128. We show our performance using both BLEU score and METEOR score. We compared our two layer hierarchy attention based model that uses pool5 and pool4 convolution feature to the single layer hierarchy attention model that only uses pool5 feature. We see that our two layer model gains significant improvements over the single layer model. The performances from all two models are shown in Table 5. Visualization of the generated attention image is presented at the very end.

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4	METEOR
One Layer (pool5)	69.7	48.9	34.2	23.9	22.7
Two Layer (pool5 + pool4)	70.3	49.5	34.7	24.5	23.2

Table 4: Attention Based Hierarchy Model

Discussion

For video recognition, we note that a hierarchy of LSTM units performs better than naively stacking LSTM layers. We argue this type of architecture forces the network to learn to exploit the data at multiple temporal scales in a joint fashion instead of an interactive fashion that a simple stacked network would. We also note that our novel hierarchical attention model out performs the same model using only a naive spatial attention model. Unfortunately our results for the action recognition task were not competitive but we can attribute this to not having time to integrate a optical flow stream into our network. This optical flow stream would provide more temporal information which has been shown to improve performance, as demonstrated by the prevalence of other two-stream networks we compare our results to. For image captioning, we also see that our hierarchical attention based model performs significantly better than using just one layer. Even though we are only using pool5 and pool4 features from VGG16, our performance is close or even better than the that from [20] which uses con5-3 from VGG19. We believe that this is due to the design of our model which uses prior probability as an implicit prior for lower feature layer as well as the regularization which uses both doubly stochastic attention loss and entropy loss.

References

[1] Nicolas Ballas and et al. Delving deeper into convolutional networks for learning video representations. *arXiv preprint arXiv:1511.06432*, 2015.

[2] Kyunghyun Cho and et al. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*, 2014.

[3] Jeffrey Donahue, Lisa Anne Hendricks, Sergio Guadarrama, Marcus Rohrbach, Subhashini Venugopalan, Kate Saenko, and Trevor Darrell. Long-term recurrent convolutional networks for visual recognition and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2625–2634, 2015.

[4] Karol Gregor, Ivo Danihelka, Alex Graves, Danilo Jimenez Rezende, and Daan Wierstra. Draw: A recurrent neural network for image generation. *arXiv preprint arXiv:1502.04623*, 2015.

[5] Xiangyu Zhang Shaoqing Ren He, Kaiming and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv:1512.03385*, 2015.

[6] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

[7] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. Large-scale video classification with convolutional neural networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 1725–1732, 2014.

[8] Ilya Sutskever Krizhevsky, Alex and Geoffrey Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 2012.

[9] Minh-Thang Luong, Hieu Pham, and Christopher D Manning. Effective approaches to attention-based neural machine translation. *arXiv preprint arXiv:1508.04025*, 2015.

[10] Álvaro Peris, Marc Bolaños, Petia Radeva, and Francisco Casacuberta. Video description using bidirectional recurrent neural networks. *CoRR*, abs/1604.03390, 2016.

[11] Christian Schuldt, Ivan Laptev, and Barbara Caputo. Recognizing human actions: a local svm approach. In *Pattern Recognition, 2004. ICPR 2004. Proceedings of the 17th International Conference on*, volume 3, pages 32–36. IEEE, 2004.

[12] Shikhar Sharma, Ryan Kiros, and Ruslan Salakhutdinov. Action recognition using visual attention. *CoRR*, abs/1511.04119, 2015.

[13] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. In *Advances in Neural Information Processing Systems*, pages 568–576, 2014.

[14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[15] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhutdinov. Unsupervised learning of video representations using lstms. *CoRR, abs/1502.04681*, 2, 2015.

[16] Graham W Taylor, Rob Fergus, Yann LeCun, and Christoph Bregler. Convolutional learning of spatio-temporal features. In *European conference on computer vision*, pages 140–153. Springer, 2010.

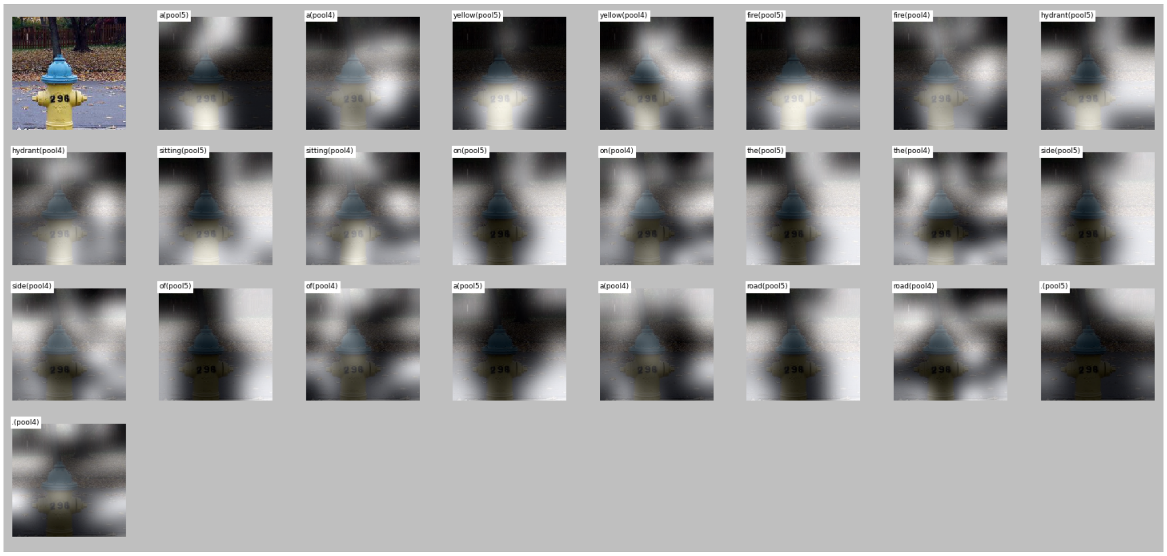
[17] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 4489–4497. IEEE, 2015.

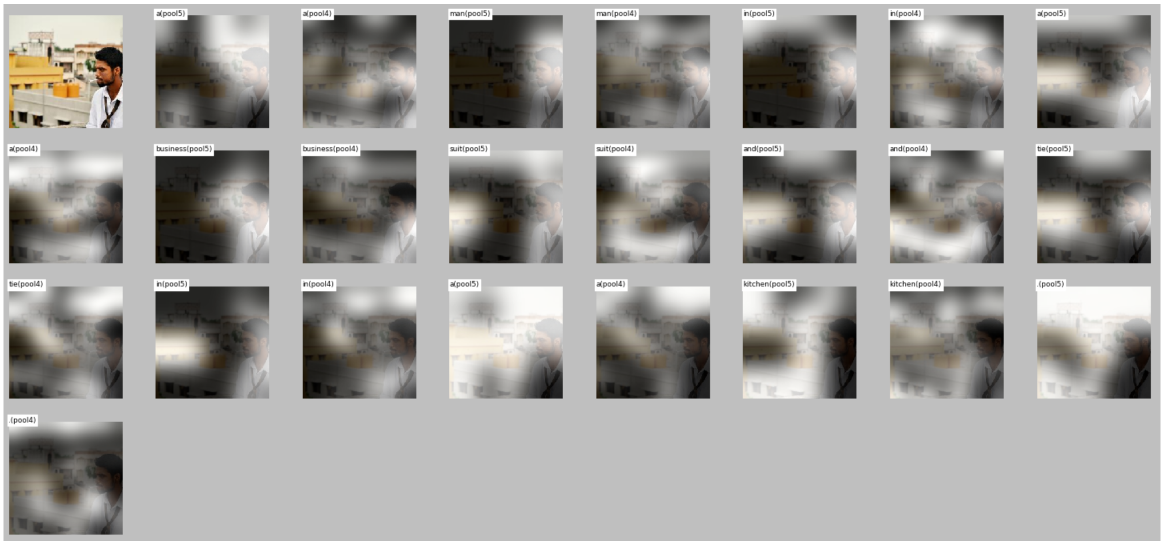
[18] Heng Wang and Cordelia Schmid. Action recognition with improved trajectories. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 3551–3558, 2013.

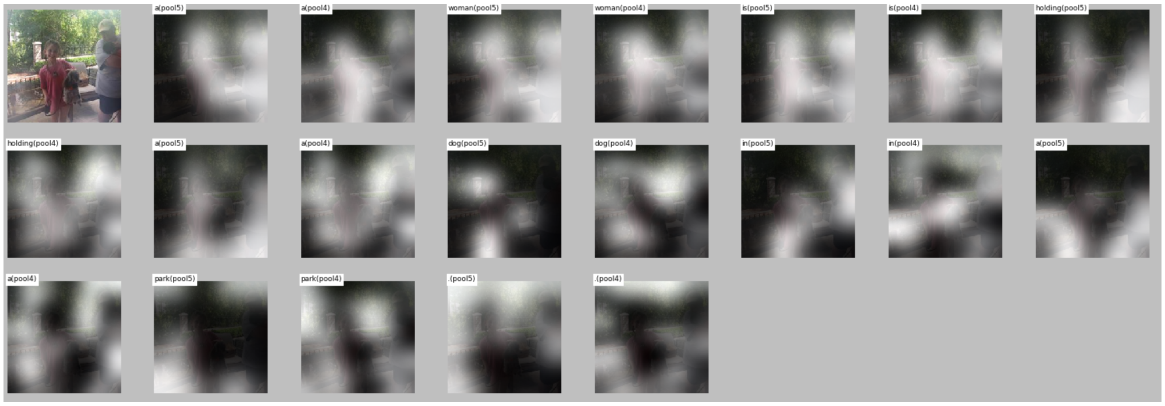
[19] Yilin Wang, Suhang Wang, Jiliang Tang, Neil O'Hare, Yi Chang, and Baoxin Li. Hierarchical attention network for action recognition in videos. *CoRR*, abs/1607.06416, 2016.

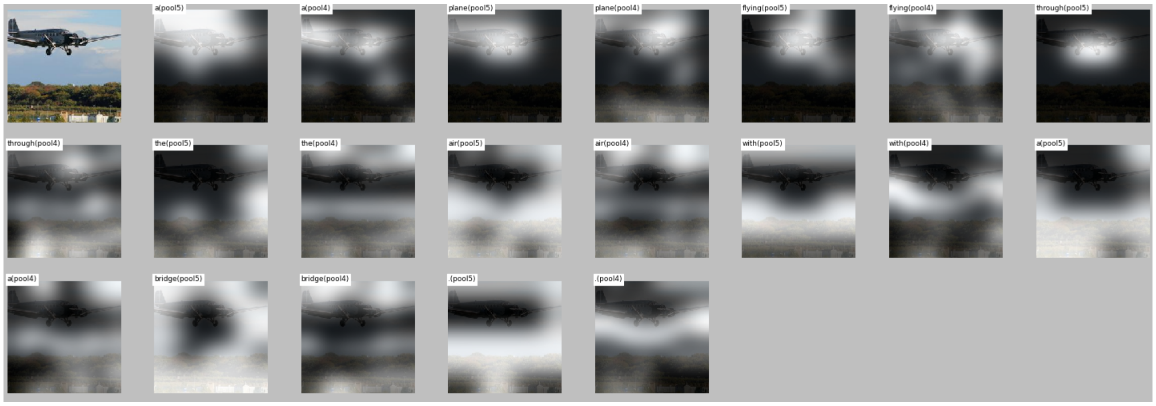
[20] Jimmy Ba Ryan Kiros Kyunghyun Cho Aaron Courville Ruslan Salakhutdinov Richard S. Zemel Xu, Kelvin and Yoshua Bengio. Show, attend and tell: Neural image caption generation with visual attention. *arXiv preprint arXiv:1502.03044*, 2015.

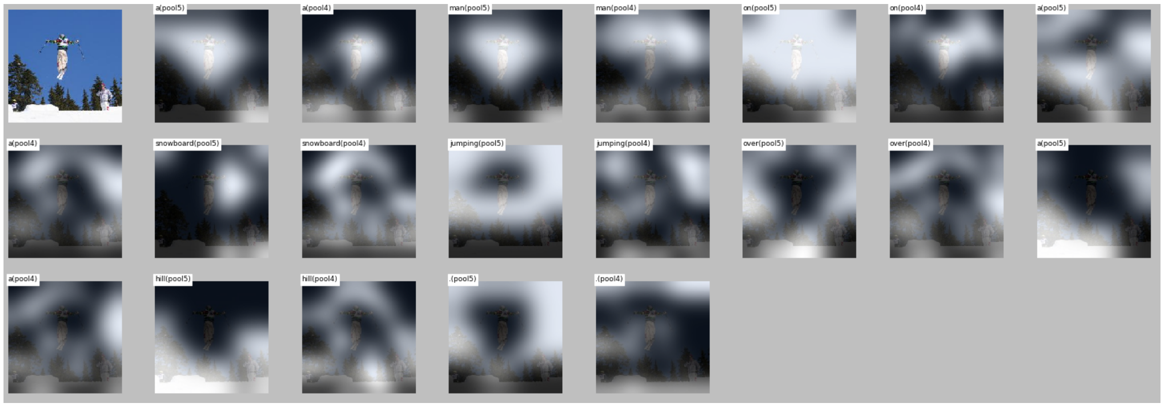
[21] Joe Yue-Hei Ng, Matthew Hausknecht, Sudheendra Vijayanarasimhan, Oriol Vinyals, Rajat Monga, and George Toderici. Beyond short snippets: Deep networks for video classification. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4694–4702, 2015.


a(pool5)
a(pool4)
yellow(pool5)
yellow(pool4)
fire(pool5)
fire(pool4)
hydrant(pool5)
hydrant(pool4)
sitting(pool5)
sitting(pool4)
on(pool5)
on(pool4)
the(pool5)
the(pool4)
side(pool5)
side(pool4)
of(pool5)
of(pool4)
a(pool5)
a(pool4)
road(pool5)
road(pool4)
.(pool5)
.(pool4)


a(pool5)
a(pool4)
man(pool5)
man(pool4)
in(pool5)
in(pool4)
a(pool5)
a(pool4)
business(pool5)
business(pool4)
suit(pool5)
suit(pool4)
and(pool5)
and(pool4)
tie(pool5)
tie(pool4)
in(pool5)
in(pool4)
a(pool5)
a(pool4)
kitchen(pool5)
kitchen(pool4)
.(pool5)
.(pool4)


a(pool5)
a(pool4)
woman(pool5)
woman(pool4)
is(pool5)
is(pool4)
holding(pool5)
holding(pool4)
a(pool5)
a(pool4)
dog(pool5)
dog(pool4)
in(pool5)
in(pool4)
a(pool5)
a(pool4)
park(pool5)
park(pool4)
.(pool5)
.(pool4)


a(pool5)
a(pool4)
plane(pool5)
plane(pool4)
flying(pool5)
flying(pool4)
through(pool5)
through(pool4)
the(pool5)
the(pool4)
air(pool5)
air(pool4)
with(pool5)
with(pool4)
a(pool5)
a(pool4)
bridge(pool5)
bridge(pool4)
.(pool5)
.(pool4)


a(pool5)
a(pool4)
man(pool5)
man(pool4)
on(pool5)
on(pool4)
a(pool5)
a(pool4)
snowboard(pool5)
snowboard(pool4)
jumping(pool5)
jumping(pool4)
over(pool5)
over(pool4)
a(pool5)
a(pool4)
hill(pool5)
hill(pool4)
.(pool5)
.(pool4)