Gesture recognition

Mohsin Vindhani
Red Hen Lab
Google Summer of Code 2016

August 17, 2016

1 Introduction

The aim of this project was to do gesture recognition for videos. The project was divided into two main stages. They are:

1. Video segmentation: This involves segmenting the pixels that have a human being in it and seperate multiple persons in the image.

2. Gesture recognition : The segregated output for each person will then be feed into the gesture recognition block for recognising the gestures.

The main work was done using the Deconvolution framework of Deep learning. The code was written in the Theano python framework.

2 Extending Theano

The Theano framework was selected after discussion because of the ease of extending and processing several modalities in python and the popularity of the Theano framework in the deep learning community. Apart from the gesture recognition task, this project made huge extensions in the Theano library which can be useful to the community. Before going to the details of the approach, this section will cover the details of the new layer developed.

The deep learning system is developed using a modular approach. Each layer has weights which when given an input does the forward pass using the weights. In the backward pass, the layer is given the gradients with respect to the output and the layer computes the gradient of the final cost with respect to the weights in that layer and passes the gradient of the cost with respects to its input to the bottom layers. The modular approach is shown below.

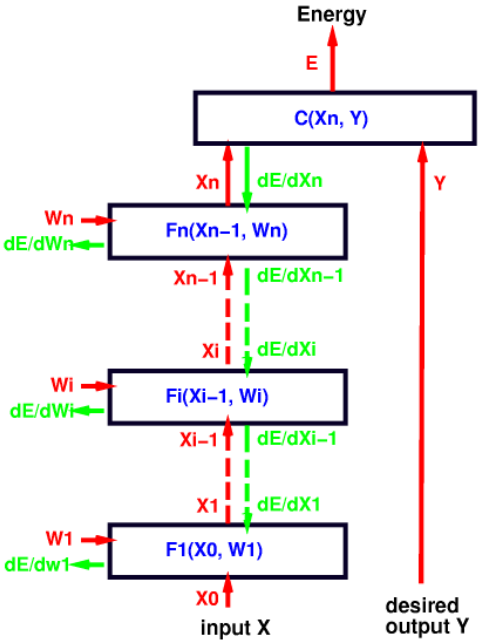

Figure 1: Modular deep learning taken from [2]

2.1 Switched pooling layer

This layer does a max-pooling but also remembers the position from where the maximum value came. The positions are returned in form of the switch variables which will be used for unpooling.

For the back propagation the normal pool gradient available in the theano package is used. The switch variables do not play a role in the back prop.

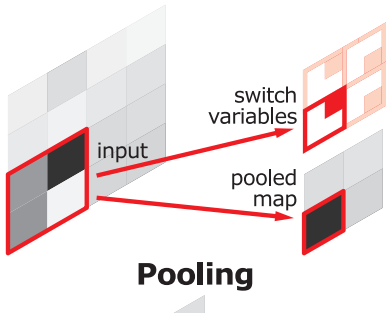

Figure 2: Switched pooling taken from [1]

The C implementation was done using the *theano.op* functionality.

2.2 Unpooling Layer

This layer takes the switch variables as the input and gives the upscaled image. Both the switched pooling and unpooling layers were tested for a pooling and stride factor of 2 each.

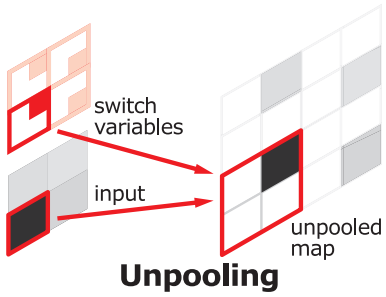

Figure 3: Unpooling layer taken from [1]

2.3 Deconvolution layer

Deconvolution layer can be viewed as the convolution layer with the forward and the backward stride reversed. This means that the forward pass of the Deconvolution layer is obtained by the back propagation through the convolution layer and vice versa. This was implemented in Theano using the *AbstractConv2d_gradInputs* op.

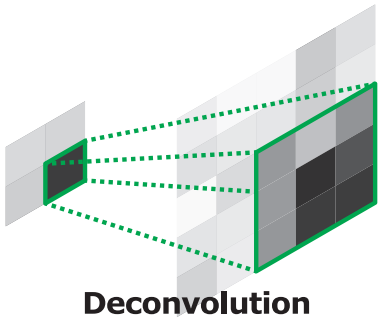

Figure 4: Deconvolution layer taken from [1]

2.4 Batch Normalization layer

In a large convolution network, the outputs from the convolution layer can be very large or very small. For efficiency in training and also for forward propagation, it is required that the output after each convolution layer be normalized. The batch normalization layer computes the mean across the entire image and

also across all the images in the batch. Then for each pixel in the image, it subtracts the computed mean.

2.5 Crop layer

This layer outputs a subset of the image. Given a image and the offset, this operation removes the offset number of pixels along the border. So given a image with the shape (h, w) and offset d, the output shape of the image will be $(h - 2d, w - 2d)$.

To compute the fast back propagation, the gradient of this layer is also implemented as the operation in C.

2.6 Fuse-sum layer

This layer computes the sum of the two inputs of the same shape and same batch size. It computes the sum at the pixel level. The gradient for this layer is simple. It is just the same for both the inputs and the value is equal to the gradient coming from the top layers.

2.7 Temporal Convolution layer

This layer implements convolution in the spatio-temporal domain. This was done using the conv3d layer of theano. To implement the stride, the subset of the output is selected according to stride. This approach was not the optimal one but it gave the best way to implement convolution using the existing theano functionality.

2.8 Temporal Deconvolution layer

This was the step which took the largest time in the project. Many approaches were tried to have a deconvolution layer with spatio-temporal stride. However a simple solution helped solve the problem. As deconvolution is convolution with forward and backward stride exchange, this layer was implemented using the above Temporal convolution layer. The gradient of the output of the convolution layer w.r.t to the input was calculated and used as a forward pass of this layer.

3 Human segmentation

The input to this module is a raw image coming from a video or an image file. The output should be the segmented regions where there were human beings in the image. Recent advances in deep learning have resulted in great efficiency in this task. Two approaches were developed to solve this problem and both of them are based on the deconvolution paradigm. Most of these systems are trained using the PASCAL VOC data set[3]

3.1 Deconvolution network

This network used the convolution and switched pooling layers with the fully connected layers followed by the deconvolution and the unpooling layers. The convolution and switched pooling layers form a mirror image about the fully

connected layers. The inference part of this model is complex as it requires a lot of proposals to generate the final segmentation. Hence it was not used in the final pipeline of the project.

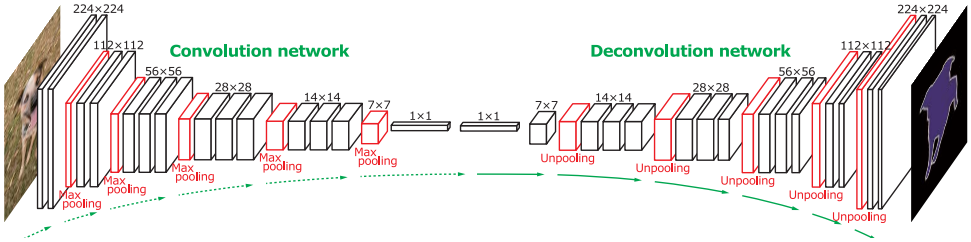

Figure 5: Deconvolution network taken from [1]

This network was coded in Theano using the new layers that were implemented. The weights were available online [4].

3.2 Fully Convolution Network

This network[5] is also based on the deconvolution frame work. The first half of the network does convolution followed pooling up to a fully connected layer. After the fully connected layer, the network in deconvolution fuses the information from various layers at different depth with the aim of merging the high level and low level information of the pixel. The full architecture of the network is as below.

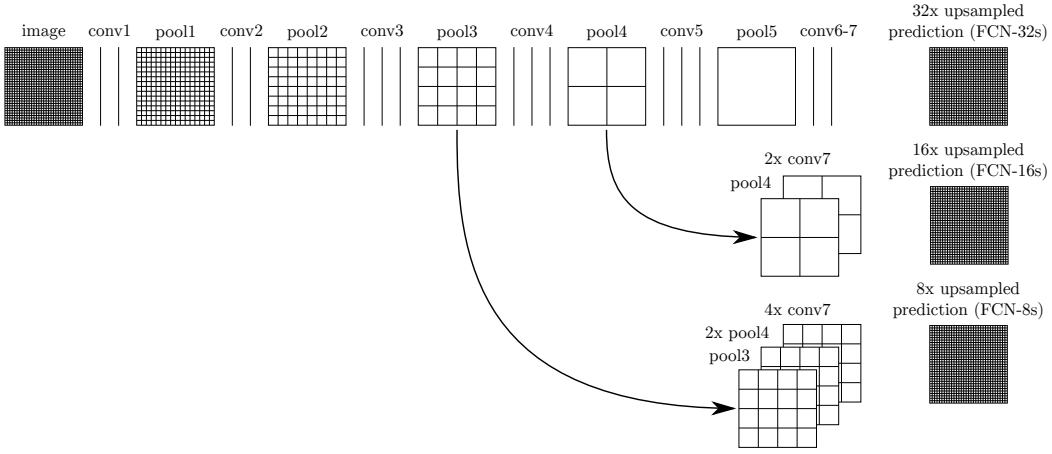

Figure 6: Deconvolution network taken from [1]

The advantage of this network is that it does not require complex proposal generation method to generate the final segmentation.

For example the raw image is as below:

Figure 7: Example image taken from [3]

The output by the fcn algorithm for the above image is :

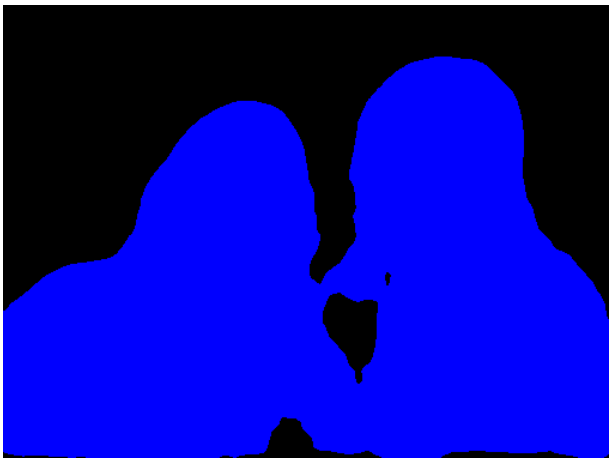

Figure 8: Output from FCN

4 Pipeline

After the human labels are segregated in the image, the next step in the video is to separate different persons in the image and also keep track of the person across the frames in the video. This approach makes an assumption that the persons in the video do not undergo a large pixel movement across videos. So the position of the existing person can not be replaced by a new person in small number of frames.

Given a video frame, first the faces in the image are identified using the Viola and Jones descriptor. The face detection algorithm gives the number and

the location of the faces in the image. These parameters are used to initiate a K-means algorithm on the human segmented data. Only one iteration of the K-means algorithm is enough to cluster the different persons existing in the image. This ensures that the process is smooth.

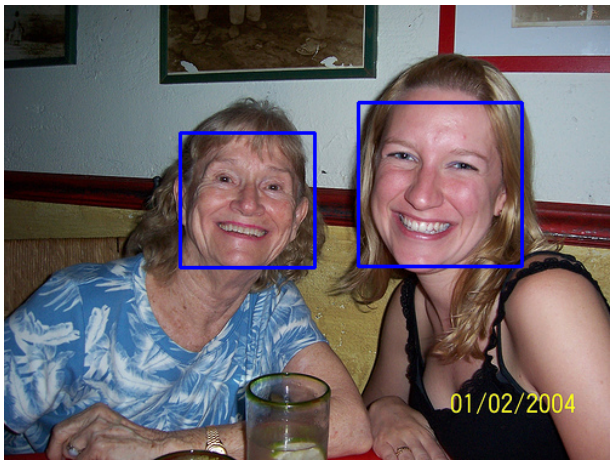

Figure 9: Output from clustering

After clustering the persons, it is required to map the clusters consistently across frames. This is done by keeping track of the face centers of the person in the previous frames. Given a new frame, first the weighted center for each cluster using the face centers of the previous frame is calculated. The weights are assigned in linear fashion such that the frame which came first gets weight equal to one and weights of the consecutive frames are increased by one. The weighted center for a cluster is then:

$$(x_w, y_w) = \frac{\sum_{i=1}^{n} w_i(x_i, y_i)}{\sum_{i=1}^{n} w_i}$$

A new cluster is then tracked by assigning it to the weighted cluster it closely associates with. The assumption made is required for this tracking to work.

The output by the pipeline for the image used in the above section is :

Figure 10: Output from clustering

Once the persons are clustered, they can be easily separated into different images and fed to the gesture recognition block.

5 Gesture recognition

Object recognition for images have been well studied and very high accuracy have been achieved. Gesture recognition for videos is an active topic for research in the deep learning community. The main challenge in the gesture recognition for videos is to combine the information across the frames in time.

The challenge for training the network is the unavailability of large segmented video data set. So an approach "multi-velocity network"[6] based on semi-supervised learning was selected. The paper in order to achieve the state of the art result uses an extensive data set with over 2700 labeled sequences and 6.5 million unlabeled clips which is still not made public. Hence it was decided to use the largest publicly available data set which is Cohn-Kanade data set. This data set has only 593 sequences out of which 327 are labeled. The selected data set had sequences with varying length whereas the network requires a pre-defined number of sequence. In order to solve this, the length of the input sequence was fixed and the frames were deleted from both the start and the end till the required number of frames are remaining. If the number of frames are less then required, that sequence is ignored.

This network uses the Deconvolution framework in the spatio-temporal domain to combine the features. Making the theano support for this was the most challenging part in the project. To the knowledge of the author, this is the first attempt to write spatio-temporal deconvolution layer for theano.

The network has three deconvolution network blocks to which the video is passed at different speeds. As the gesture may be performed at different speeds, the video is then speed up or down using the spline interpolation. The network combines the fully connected layer of all the blocks and apply a Multi-layer-Perceptron(MLP) to identify the gesture. For training, the loss is a combination

of the reconstruction error and the error in gesture recognition. The error for reconstruction is the L2 norm difference and for gesture recognition it is softmax. The network combines the temporal information using the slow fusion approach.

The full architecture of the system is shown below.

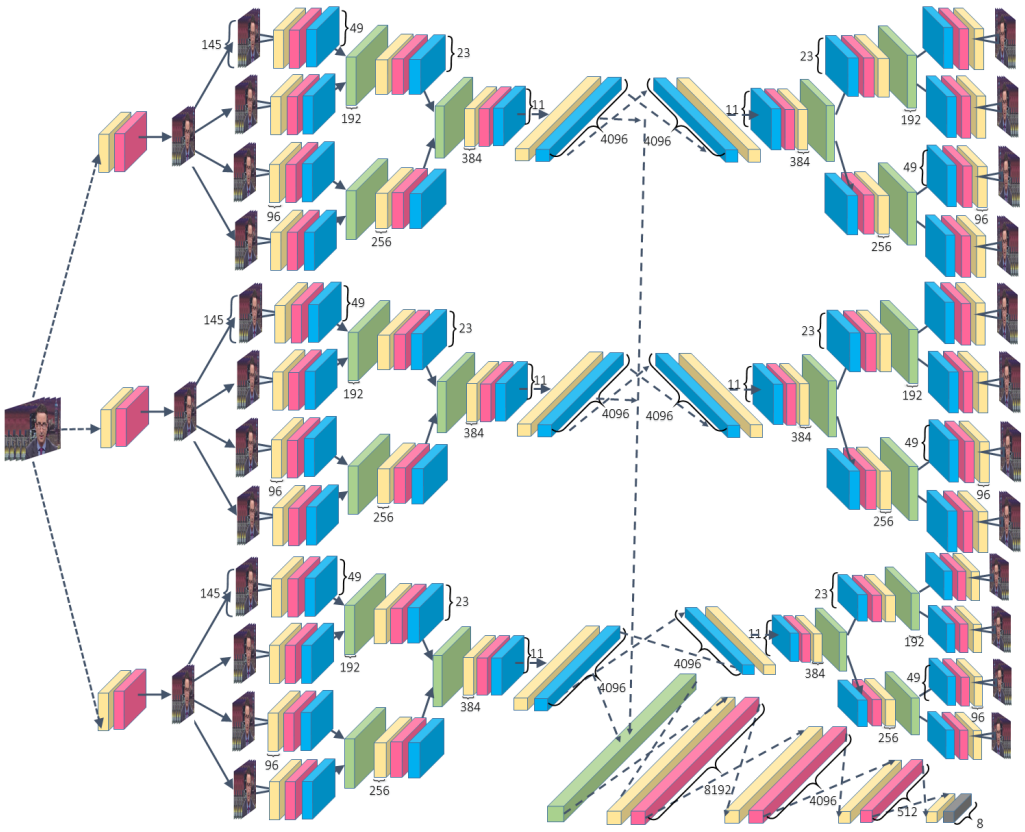

Figure 11: Multi velocity network [6]

For training the hyper parameters of the system are:

- Number of input frames
- Sampling factor
- Stride in the temporal domain for the layers
- The relative weighting of the error from reconstruction and gesture recognition.
- usual back prop parameters : learning rate, batch size.

6 References

1. http://arxiv.org/abs/1505.04366 "DeconvNet for semantic segmentation"

2. http://cilvr.cs.nyu.edu/lib/exe/fetch.php?media=deeplearning:dl-backprop1.

3. http://host.robots.ox.ac.uk/pascal/VOC/

4. https://github.com/HyeonwooNoh/DeconvNet

5. https://github.com/shelhamer/fcn.berkeleyvision.org

6. arxiv.org/pdf/1603.06829 "multi-velocity neural network"

7. http://arxiv.org/abs/1604.02647 ""

8. http://vis-www.cs.umass.edu/lfw/index.html "LFW data set"

7 Acknowledgments

I am extremely grateful to Karan Singla, Prof. Francis Steen and Prof. Mark Turner for their patient guidance and support throughout the period of entire summer. I am extremely indebted to them and Red Hen Lab for giving me this wonderful opportunity of working on gesture recognition for videos using deep learning. Without the support and valuable insight of my mentors, this work would not have been possible.