Exploring Various Methods of Video Categorization of Human Actions Using RNN and CNN

Zacharyah Egan Gramstad
Rice University
6100 Main St, Houston, TX 77005
zeg2@rice.edu

Akshay Ravindran
University of Houston
4800 Calhoun Rd, Houston, TX 77004
asujatharavindran@uh.edu

Zilu (Peter) Tang
Rice University
6100 Main St, Houston, TX 77005
zt8@rice.edu

Abstract

Many breakthroughs in Deep Learning result from applying biological and neurological patterns to computational learning models. Humans and other animals do not process visual information as individual discrete images, but as time-dependent sequential data. With biology as our inspiration, we chose to explore whether video classification through various experiments employing Long Short Term Convolutional Neural Networks would outperform static frame classification. We narrowed our scope to explore methods of categorizing human motion in video from a specific subset of ten actions in the UCF101 dataset [1] by performing transfer learning on our dataset with prominent image classification networks InceptionV3 and VGG19 pretrained on ImageNet. We compared frame-by-frame CNN classification to two RNN experiments. 1) Using VGG19 as a feature extractor, we saved pooling activations for internal layers. We then converted the extracted activations to sequences to be used as the input to an LSTM of the entire video length. 2) To investigate the merits of a model trained at generating data to be used as a classifier, we combined an classifier with a variational autoencoder and trained on the same dataset.

1. Background

Many efforts has been made in the effort to make better video classification model. In our paper, we will introduce the background on current usage of RCNN network structure, some methods to classify videos, saliency mapping and autoencoders before talking about our experiments

[1]The majority of the analysis was performed on 10 classes under Body-Motion Only category under UCF101. It was selected as the actions were relatively similar to one another.

1.1. Traditional RCNN network structure

Many other studies has been conducted to combine RNN and CNN together for various different purposes. In paper recently by Kriegeskorte et. al.[1], they investigated adding bottom-up (B), lateral (L), and top-down(T) connections to vanilla RCNN network, yielding 4 different architectures (B, BT, BL, and BLT). Inspired by anatomical features such as feedback connections in ventral visual pathway [2] and lateral recurrent connectivity [3], the paper successfully hypothesized and proved that top-down, lateral, and bottom-up connections are all important for object recognition, especially under situation where multiple objects occludes on another or when target object is occluded by other object fragments. In another paper by Quang and Xie in 2016 [4], a CNN was combined with Bidirectional Long Short-Term Memory Network (BLSTM) to identify functions of non-coding region of human genome. By feeding short sequences of one-hot encoded gene sequence into CNN and input the max-pooled results into BLSTM, the model outperformed current state of art model DeepSEA by being able to learn from local motifs as well as complex regulatory grammar between motifs. An CRNN architecture is also useful in detecting rare sound events [5]. The paper by Tampere University of Technology fed in spectrum graph as input of CNN, after applying filters, the layers of activations are stacked together as a 2D input for the recurrent layer activation.

1.2. Matt Harvey's Exploration

With different models trying to combine RNN and CNN, Matt Harvey, Founder of Coastline Automation, ran an experiment comparing five different techniques to classify videos on UCF101[6]. Below is a table summarizing his methods and achieved results:

Method Number	Method Description	Achieved Results
Method 1	Classify one frame at a time with a CNN (transfer learned InceptionV3 Model).	65% top 1 accuracy and 90% top 5 accuracy. This is used as benchmark accuracy for the rest of experiments
Method 2	Use a time-distributed CNN (VGG16, untrained), passing features to an RNN (GRU with 3 layers of 128 nodes), train in one network	20% top 1 and 41% top 5. May need more heavy lifting CNN. It is hard to train CNN with RNN in end from scratch. Requires large memory
Method 3	Classify using 3D convolutional network (3 layers of 32, 64, and 128 nodes).	28% top 1 and 51% top 5. Requires large memory
Method 4	Extract features with CNN (transfer learned Inception V3 Model), pass sequence to a separate RNN (4096 wide LSTM)	74% top 1 and 91% top 4. Network is learning temporal information correctly
Method 5	Extract features with CNN (transfer learned Inception V3 Model), pass sequence to a multilayer perceptron (MLP) (2 layer net with 512 nodes each)	74% top 1 and 88% top 5. MLP could organically infer temporal feature. Deeper and wider MLP result in near perfect top 5 but mediocre top 1

Method 4 clearly outperformed the othe

1.3. Saliency mapping

In order to understand the information that our CNN is extracting from the individual frames, we used the method of saliency map in aid of our analysis. Saliency map, first introduced by Zisserman. Et. al (source), is calculating, the gradient of output category with respect to input category [7].

$$\frac{\partial output}{\partial input}$$

This should be able to inform which part of the input image is contributing towards the classification result. In another word, saliency map informs which part of the image the CNN is basing its judgment on.

1.4. Autoencoders

Autoencoders is a neural network which is typically used for unsupervised learning of efficient coding of the data. It is composed of an input layer, multiple hidden layers, and an output layer of the same size as the input layer. The input is typically transformed into a lower dimensional space or is forced to learn a sparse representation of the original input without degrading the features captured in the input(encoder). Then this encoded information is expanded to obtain the initial data by another network called the decoder. The autoencoder can inform meaningful information about the structure of the original data, such as correlations between inputs. This exploration effort prompted us to explore more on using pre-trained CNN as feature extractor, and use LSTM to learn on the activations of CNN layers.

2. Methods and Experiments

2.1. Experiments 1 - Time-Independent Classification

For the time-independent CNN frame-by-frame classification, we chose the state of the art VGG-19 architecture. It consists of 16 convolutional layers to capture the low level features, followed by 3 fully connected layers. We selected 10 body-motion only classes from UCF101 dataset so that the similarity between videos makes the classification problem challenging. We then performed transfer learning on our selected dataset. We selected the hyper parameters for the model by observing the validation loss and accuracy w.r.t. different hyper parameter combinations. The hyper parameters tested were learning rate, optimization technique and the batch size. Keeping all the parameters except for one fixed, we tried a total of 56 combinations to arrive at the optimal hyper parameters. We trained the model for the same number of epochs (200). Some of the performance graphs are shown in Fig3. If we simply go for the parameter that gave larger validation accuracy, we would choose Adagrad or Adam with a learning rate of 0.003. However,

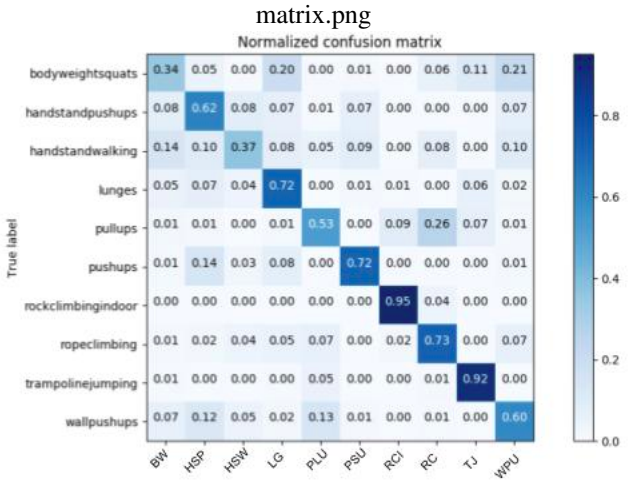

Figure 1. the confusion matrix of pure CNN classifying video content through frame by frame prediction. Rockclimbingindoor was the most accurate class with 0.95 accuracy and bodweightsquats is the least accurate with only 0.34 accuracy

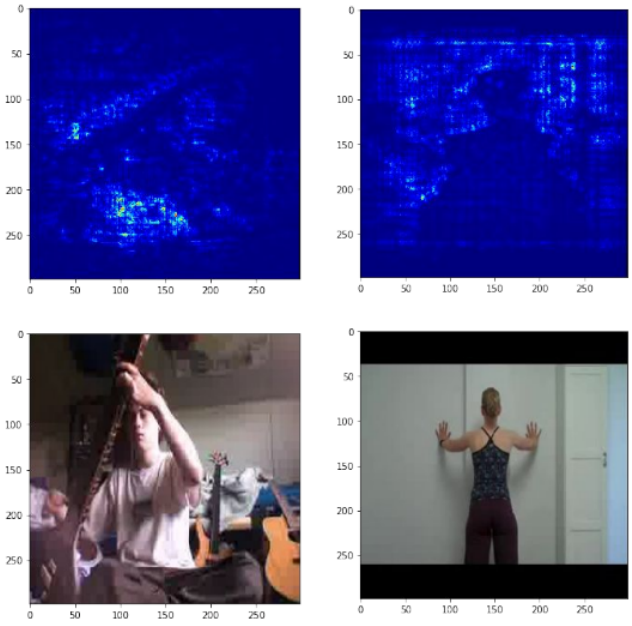

Figure 2. Guided Saliency map for a frame from class playingsitar (Left). Guided Saliency map for a frame from class wallpushup (Right) As seen, the sitar shape may not perfectly align with the angle in real frame, but the general shape did indicate the network was learning from the sitar. In classifying wallpushup, the network, however, learned the surrounding of the person performing the action

checking the validation loss we observed that the model gets over fitted easily if we use those hyper parameters. Considering this as a trade off, we selected Adam optimization with a learning rate of 0.00005 and batch size of 32.

The network averaged 61% accuracy on testing data for the 10 classifications (see [Figure 1] for the class breakdown) after training for a total of 50 epochs with an early stopping condition to stop training if there exists no improvement in the validation loss for 5 consecutive epochs. Checking the confusion matrix is Fig1, it makes sense intuitively as to why some of the classes were mislabelled. For e.g. pullups were mostly misclassified as rock climbing, since both involves the person stretching up. Similarly pushups were misclassified as handstand pushups. In order to perform similar analysis in a more meaningful way, we attempted to visualize what and how the model was learning via generating saliency maps for some of the images. In general, vanilla saliency maps did not provide easily understandable results, but guided saliency maps (mapping activations starting only from the classification neuron to the pixel layer) were slightly more understandable. Based on our observations, in a lot of cases the network learned the background of the image rather than the subject performing the action (Figure 2 – right). In other cases (Figure 2 – left), the network seems have learned the actual object in question.

2.2. Experiment 2 - LSTM Trained on Features Extracted from CNN

Inspired by Matt Harveys post on video classification, we decided to train VGG-19 on 40 frames per video, then extract the activations from the pooling layers for each convolutional block and perform Global Average Pooling on each convolution block activations. Then, we combined and sequenced the activations from each video, and trained an LSTM on these stacked activations(40 frames x activation dimension). We used an LSTM layer with N1 nodes,a dense layer with nodes N1, followed by a softmax layer as shown in Fig 3. The number of nodes N1 and N2 was selected in a similar fashion as the hyper parameter selection for CNN. Finally an architecture with N1=2048 and N2=64 was selected as the optimal one.

While Harvey only used the final pooling layer of InceptionV3, we experimented with multiple pooling layers of VGG-19 (2, 3, 4, and 5) in order to explore the effectiveness of various depths. For the earlier layers of activation, the global average pooling converted the 3D vector into 1D so we can feed it into LSTM. Even though we spent a lot of time tuning hyperparameters (Figure 3), layers 3 and 4 perform poorly while layer 5 produced similar results as static CNN (Figure 4).

A small deviation from our normal experiment is the dataset we were using initially. Before using the current dataset involving only body-motion only classes, we chose 2 classes out of 5 general categories of the data set which includes: human-object interaction, body-motion only, human-human interaction, playing musical instru-

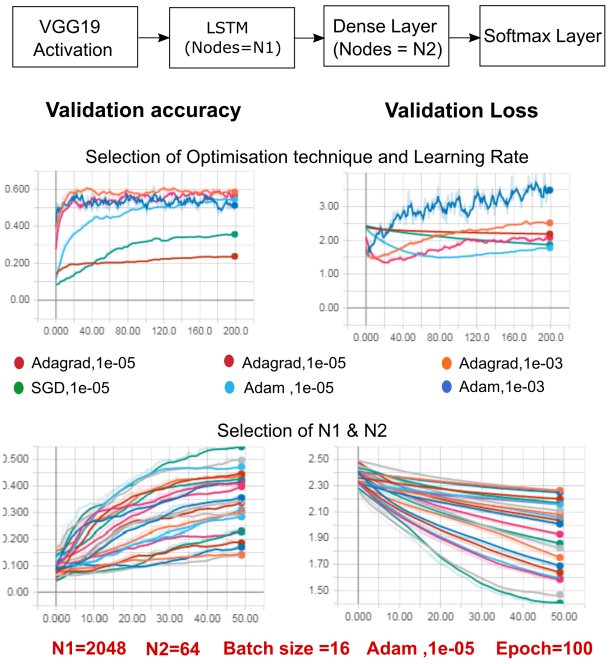

Figure 3. hyperparameter tuning for LSTM training on CNN activations. Out of all optimization algorithms, we chose Adam because of its steady increase in accuracy without over-fitting. We chose to have 2048 nodes in LSTM and 64 nodes in dense layer

ments, and sports. The resulting dataset was more heterogeneous and naturally easier to predict. We used the same method with VGG19 extracting average pool layer activation and use LSTM to classify the result on our current dataset and discovered very little difference between pure CNN method and CNN, LSTM combination (probably because it is very easy to distinguish difference through frame by frame already). Therefore we decided to subsample a different dataset involving harder to distinguish classes.

2.3. Experiment 3 - LSTM Classification and Variational Autoencoder

We experimented with adding autoencoder at the end of CNN to capture more variations within the CNN structure. Instead of feeding the final average pool layer activations into LSTM, we fed the inputs frames into a variational autoencoder forcing sparse activation. The variational autoencoder consists of 3 convolutional layer with 64 filters of size 3x3 and stride 1, followed by 2 dense layers of size 128 and 2 to act as the encoder and a decoder which is the transposed version of the encoder. We trained the model to simultaneously predict the class as well as reconstruct the image. We were able to achieve slightly better accuracy with this approach (Figure 5). Some of the reconstructed images based on the latent variables are shown in (Figure 6).

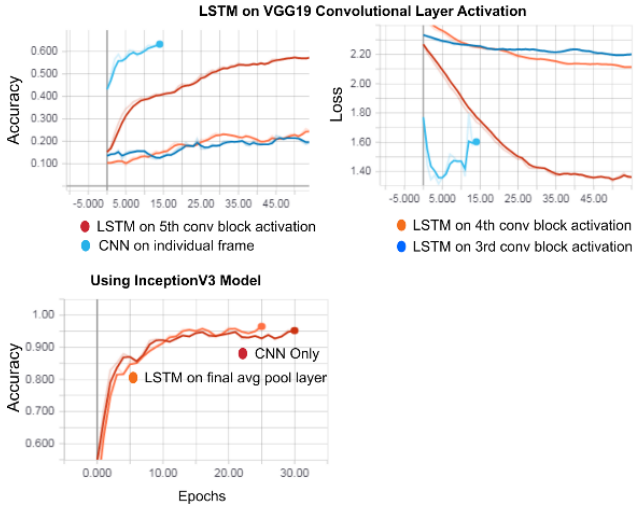

Figure 4. Training accuracy and loss for LSTM model based on layer 3, 4, 5 activation from VGG-19 (Top). Training accuracy of CNN only method compared to LSTM on final average pool layer activation of CNN model (Bottom)

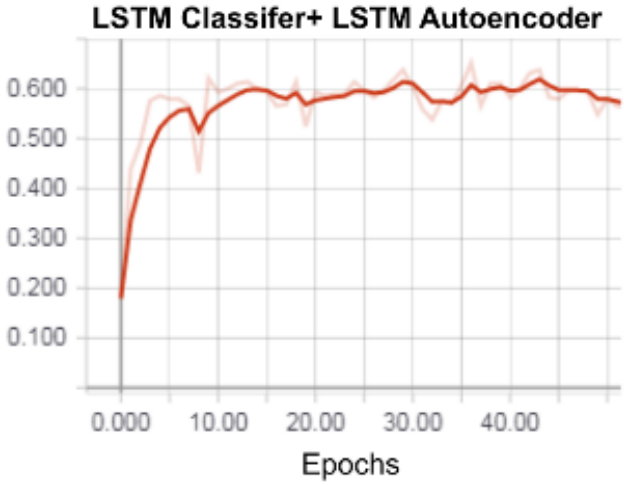

Figure 5. Training accuracy of LSTM with Autoencoder in the end. The accuracy plateaued around 0.6 and never exceeded the simple CNN model

3. Observations and Discussions

Though our project was mostly explorational, our hypothesis that the LSTMs would classify the videos more accurately than the time independent CNNs was not supported as well as we expected. However, the results that we did have were interesting nonetheless and reveal the need for more exploration into the area of video classification.

3.1. Misclassification on Frame-by-Frame VGG-19

One interesting observation was that some very similar frames within the same video were misclassified (Figure 7).

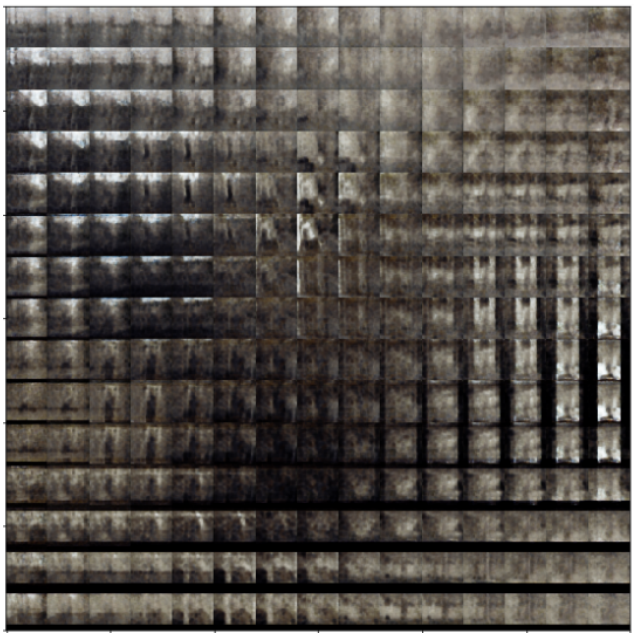

Figure 6. Visualization of autoencoder representation. Each image represent a predicted image based on the latent variables.

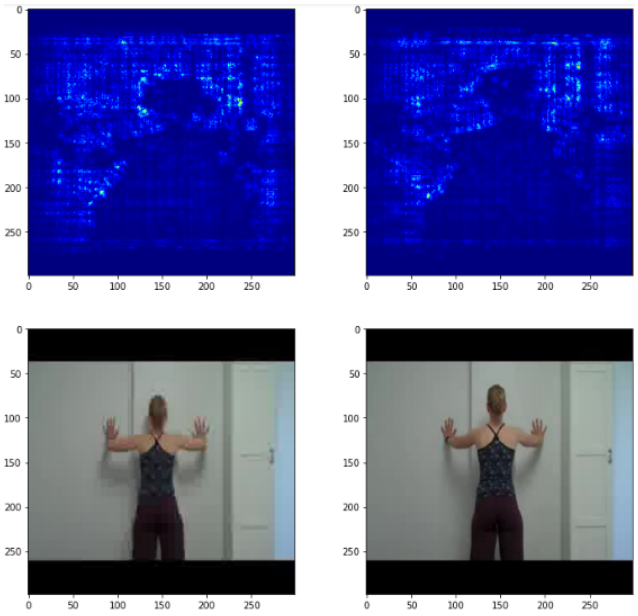

Figure 7. Model classified left frame as 0.63 chance being pullup, and right frame as 0.75 chance being wallpushup. The two frames are pulled from the same clips of video

In the case of the Wall Push-up/Pull-up misclassification, not only were the images similar, but the guided saliency maps were very similar, meaning that the network was learning very similar structures for totally different classes. However, this is not entirely unexpected as the data chosen was intentionally chosen to be similar. In the specific example of Figure 7, the misclassification could be due to the 90-degree angle that the womans arms are in, which resembles a pull up.

3.2. Importance of Depth of Extracted Features

As is evident in Figure 3, the 5th pooling layer performed substantially better than pooling layers 3 and 4 in Experiment 2. Also, [trying an experiment] more similar to Matt Harveys, we chose 10 quite varied classes in the UCF101 dataset to train and test on InceptionV3 and compare the accuracy against a final layer feature extraction of InceptionV3 fed into an LSTM. Both performed very well on the set, but the LSTM outperformed the frame-by-frame CNN by a small margin. Thus, again, a deeper hidden layer performs very well. There then seems to be a correlation between the layer depth of the extracted features and the LSTM accuracy. The exact reason for this is still unknown to us, but we developed a few hypotheses.

The deeper the layers, the more abstract and complicated the features. Thus, it takes a certain level of abstract features in order to actually understand the picture well enough to make accurate predictions.

There is less data loss in extracting deeper layers because to feed them into the LSTM involved converting them to a single 1-dimensional vector of global average pooled filters. Since earlier layers have larger spatial dimension while performing the global max-pooling operation, some of the useful data might be lost in the extraction, resulting in the LSTM having a worse representation of what the layer actually learned.

3.3. Efficacy of Time in Video Classification

Video involves a temporal relation between different frames. One would think that time should matter when classifying a persons actions to some extent. However, it seems that in our experiments the CNN without any sort of temporal dimension performed equally as well as, if not better than, the LSTMs. Intuitively, it makes sense that for certain videos a single frame would be enough to classify the video. (The saying a picture is worth a thousand words is apt.) However, the fact that single frame image classification performs equally well or better overall is unintuitive. It might be that our data is flawed that there is enough variability in the invariant parts of within videos that the classifier does not need the actors movements to accurately classify. Exploring data with actions that might involve more temporal aspects ones that might be difficult to classify as humans without more context would be a clear way to determine whether or not this was the case. Another reason could be traced back to the guided saliency map of VGG19. Since in many cases the network is actually capturing the non-moving part of the image (Figure 7), it is unlikely that

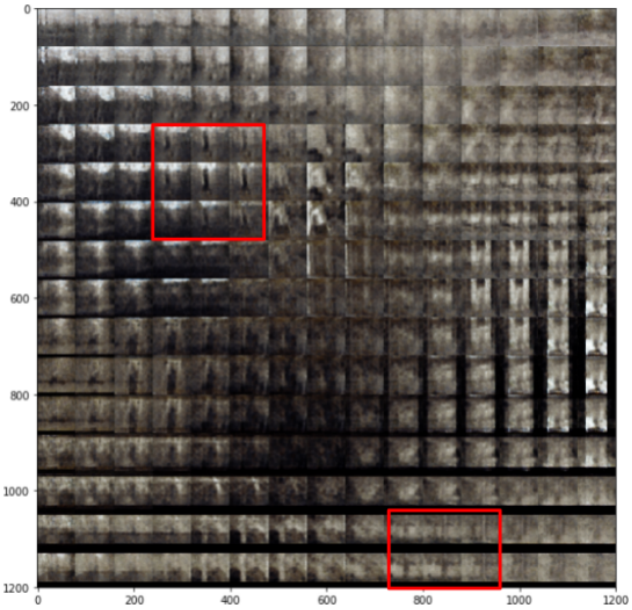

Figure 8. Visualization of Variational Autoencoder. The top left box circles the frames that seems to be specialized in distinguishing handstand while bottom right frames seems to be specialized in pullup or wallpushup

stacking average pool layer feature of non-moving feature across time is able to give LSTM anymore information than a single frame can. Since the LSTM is using CNN activations as input, and if the CNN is capturing background information more compared to the rest, we expect LSTM to not be of advantage in those cases. A way to combat this problem is to use optic-flow and force CNN to learn on the part of the frame that has more movement in temporal dimension.

3.4. Internal Representation of Videos within Autoencoders

The autoencoder in our experiment performed fairly well compared to CNN only method and LSTM on activation method. Looking across frames horizontally, we see almost sequences of action playing frame by frame, indicating that the VAE are trained to represent one frame of the action over time using different latent variables. Another derivative of this hypothesis is that by imposing a constraint upon the number of nodes, the autoencoder can then be forced to learn each action with limited nodes, allowing us to understand what is the iconic about different actions. For instance, why did we choose to represent each individual Olympic sports using the following particular frames (Figure 9)

Figure 9. Representation of different sports used for London Olympic 2012[10]

4. Conclusion

In an effort to incorporate temporal dimension as well as spatial dimension in video classification, we explored two methods in addition to Using CNN on frame by frame of video : 1) using LSTM trained on different layers of activations in CNN 2) using LSTM + variational autoencoder trained on different layers of activations in CNN. Even though none of the methods we attempted achieved better results than simple CNN method, we gained insights into how we can improve upon the methods using saliency map and autoencoder visualization.

5. Citation

[1]:*Spoerer CJ, McClure P, Kriegeskorte N. Recurrent Convolutional Neural Networks: A Better Model of Biological Object Recognition. Frontiers in Psychology. 2017;8:1551. doi:10.3389/fpsyg.2017.01551.*

[2]:*Markov NT, Vezoli J, Chameau P, Falchier A, Quilodran R, Huissoud C, Lamy C, Misery P, Giroud P, Ullman S, Barone P, Dehay C, Knoblauch K, Kennedy HJ. Anatomy of hierarchy: feedforward and feedback pathways in macaque visual cortex. Comp Neurol. 2014 Jan 1; 522(1):225-59.*

[3]:*Adesnik H, Scanziani M.Lateral competition for cortical space by layer-specific horizontal circuits. Nature. 2010 Apr 22; 464(7292):1155-60.*

[4]:*Quang D, Xie XH; DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences, Nucleic Acids Research, Volume 44, Issue 11, 20 June 2016, Pages e107*

[5]:*Cakir E, Virtanen T. Convolutional Recurrent Neural Networks for Rare Sound Event Detection. Detection and Classification of Acoustic Scenes and Events 2017. 16 Nov. 2017.*

[6]:*Harvey, M. Five Video Classification Methods Implemented in Keras and TensorFlow. Coastline Automation, Medium, 22 Mar. 2017*

[7]:*Simonyan K, Vedaldi A, Zisserman A. Deep Inside Convolutional Networks: Visualizing Image Classification*

Models and Saliency Maps. arXiv.org. Cornell University Library. 20 Dec. 2013.

[8]:*Saliency Maps - Keras-Vis Documentation*

[9]:*Autoencoders. Unsupervised Feature Learning and Deep Learning Tutorial, Stanford University*

[10]:*Posts about Question of the Day Poll on Watch Us Play Games. Watch Us Play Games*