**Convolutional Neural Network models used in classification**

As part of the project we have evaluated many different CNN models. The biggest constraint of trying new models and tuning the associated hyper-parameters were training time, processing power and lack of sufficient computer memory. Due to design of Google TensorFlow a large chunk of memory is pre-allocated to be used in its computation graph [Ref: tf api]. None of our models could fit into 2 GB GPU memory that was available, which could have provided a huge speedup over training on CPU.

All of the model topologies are outlined in Appendix X.

1. DumbNet – our first model (retroactively named DumbNet) was completely of our own design. It consisted of 5 convolutions each followed by a pooling layer. It was based on theoretical understanding of stacking convolutional and pooling layers. The model relied too much on pooling (after every convolution) and many of the neurons died out from over-saturation. We concluded that the model was not complex enough as it failed to converge while training.
2. AlexNet [Ref Krizhevsky U of T] – this network was based on a successful network from literature which Alex Krizhevsky et al used to win the ImageNet ILSVRC2010 competition. The network was difficult to work with due to the non-standard convolution and pooling layers which changed the size of the output image in a way very different from other networks. This network did not converge either.
3. VGGNet [Ref: very deep... oxford] – successful net from a team from Oxford University, it won first place in the ILSVRC2014 competition. The net operates on the use of small stacked convolutions with fewer pooling layers in between. The authors argue that a stack of three 3x3 convolutions activated by ReLU (rectified linear unit) [ref: relu] activations can be more discriminative than a single 7x7 convolution. The problem we experienced with our interpretation of the VGG network was of the massive computational cost of running it. The network consists of 13 convolutional layers, 5 pooling layers and 2 wide fully connected layers before finally coming to the classifier neuron layer. Our computers were not able to reasonably run the network.
4. DeepSenseNet – this network was inspired by the winner of this Kaggle competition [ref: Deep Sense blog]. The authors must have themselves drawn inspiration from VGG as the network seems to be a simpler version of that one. We were able to obtain adequate results from this CNN after about 12 hours of training. We used an exponentially decaying learning rate and normalizations of the activation levels of every convolutional layer. The results were over-fit to the training data with about 80% classification accuracy on the training set and 15% on the validation set. However the result was enough to prove statistical significance of the classifier.