**UDACITY PROJECT – DEEP LEARNING – FOLLOW ME**

**RUBRIC POINT EVALUATION**

1. The write-up conveys an understanding of the network architecture.

Neural networks are like an army of filters, stacked up on top of each other to help the system understand what is being processed and use it for further processing. I see it something like teaching a baby things.



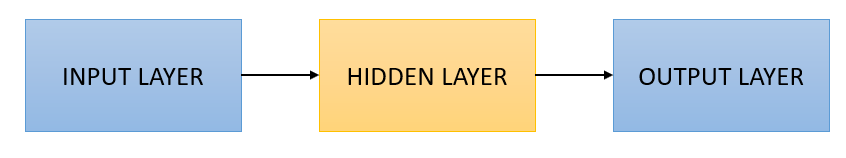
For example, a parent shows them what a window is. The baby memorizes the shape, color, position and even taste it. 😊 But the actual learning is when the baby uses that knowledge to know what a window is and identify it, in different shapes, forms and positions. (maybe even different tastes 😊)

That occurs when they see different types of windows and explore it. Though not a 100% analogy, a neural network is something similar to this.

We train a neural network to understand and identify an object. The learning is furthered when it recognizes the object on it own in a different environment.

The architecture of a neural network is similar to that of a human body



The hidden layer is where the processing happens. This is also where the network is taught to understand things. The output layer can be simple binary or multi class classifications.

Analogy to a baby:

* Binary – Is the thing safe to put inside the baby’s mouth? – Yes/No
* Multi Class – What color is the bird on the tree? – Cuckoo, woodpecker, parrot or an owl

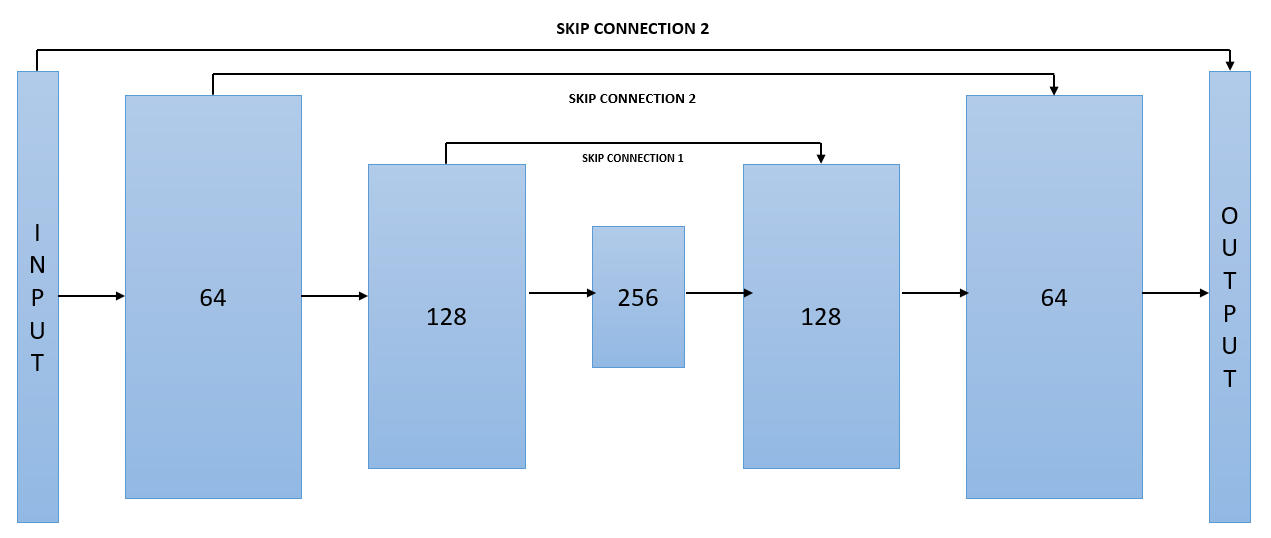
The neural network will output a probability of the classification of the output. This is done by the Softmax function.

To achieve the above, there is a lot of processing that needs to be done to ensure that network works well.

* Parameters of the network to define the architecture
* Manipulation of the images received for better object recognition
* Ensuring optimal network performance

The below write up will explain in detail on the above points

The neural network architecture usually starts with the encoders. The idea of the encoder is to extract features from the image. The first layer identifies simple patterns and then the subsequent layers extract more complex information than the previous layer. The 1x1 convolution layer saves spatial information from these layers. The image is then passed through the decoder that upsamples the image to the original size of the input. Lastly, the decoder passes the image to a convolution output layer that makes the segmentation of different classes as identified by the softmax function. I have excluded the bilinear upsampling, layer concatenation and separable convolution layer for simpler representation.



1. The write-up conveys the student's understanding of the parameters chosen for the neural network.

The following parameters were considered for the neural network:

* Learning\_rate – 0.002 – Measure of how quickly a network learns
* Batch\_size – 20 - Number of samples that the network views before the weight is updated
* Num\_epochs – 50 - Number of times the network views and works on the data set
* Steps\_per\_epoch – 200 - Number of times the batch is processed per epoch
* Validation\_steps – 50 - Number of times the parameters are to be tuned
* Workers – 20 - For parallel processing

Tuning these parameters is to be done based on how well it performs. This needs to fine tuned by iterative output improvements. Another factor was to make sure that the parameters did not allow the model to overfit, but large enough to get the model trained properly. More explanation of these parameters is provided in the following section.

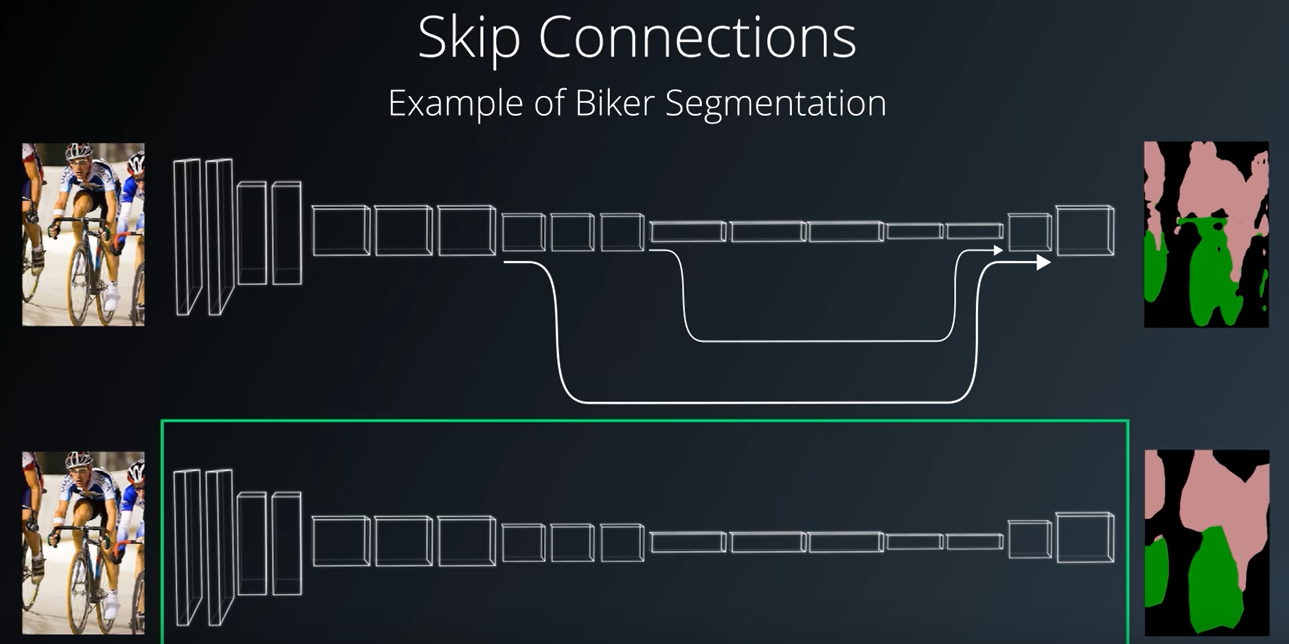
1. The student has a clear understanding and is able to identify the use of various techniques and concepts in network layers indicated by the write-up.

Convolutional neural networks use a lot of techniques to preserve spatial info, encode and decode data to classify the images received. The main aim of the below techniques is to reduce information loss and enable the network to be trained accurately.

**1x1 Convolution –** This is a single layer implemented to reduce the dimensionality of the layer. This will enable the neural network to classify the image based on images of any size in the testing phase.

**Transposed Convolutions –** This is used to upsample the previous layer to the desired resolution or dimension. The weight kernel is multiplied with each pixel to obtain the output layer.

**Skip Connections –** This will skip some forward feed to avoid info loss due to processing. Explaining further, as the scope narrows down in every layer the bigger picture is lost as a result. Even if the output is decoded back to original size some information is lost. The output of the pooling layer in the encoder is combined with current layers output. This result is then fed into the next layer. Hence the subsequent layer receives information from a previous layer and is able to make precise segmentation decisions. The below is a classic example:

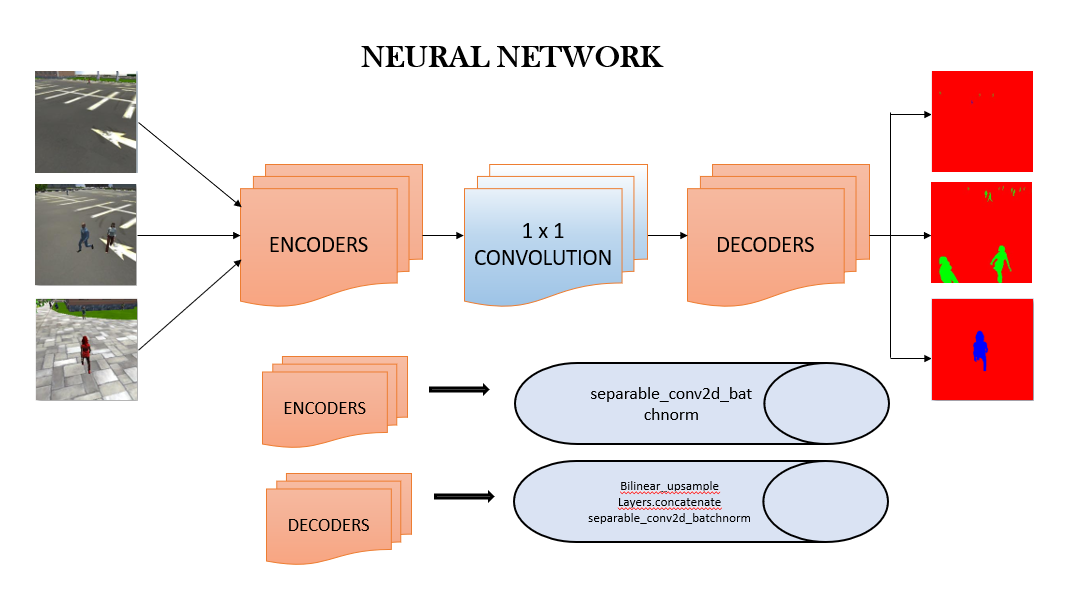


**Pooling –** the filter takes small strides, combining neighbouring convolutions as well into account. This reduces loss of information and has the flexibility of choosing either the max or the average of the convolutions

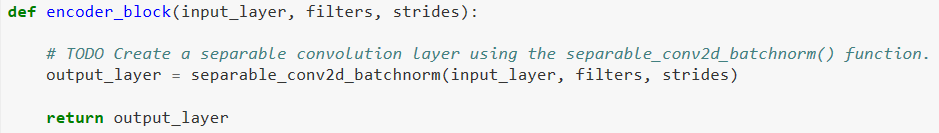
|  |  |  |
| --- | --- | --- |
| **1x1 Convolution** | **Transposed Convolution** | **Pooling** |
|  |  |  |
| **Skip Connection** | | |
|  | | |

1. The student has a clear understanding of image manipulation in the context of the project indicated by the write-up

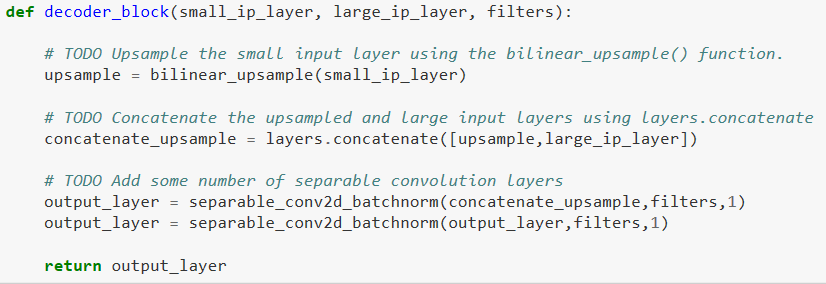
This is the design of the neural network implemented:



**Encoder Block –** mapsthe input into lower and different feature representation



**Decoder Block –** Maps thefeaturerepresentation into the input data



**1x1 Convolution**

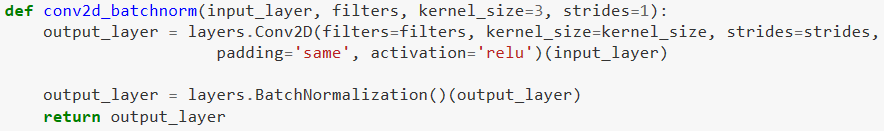


1x1 Convolution is the coordinate dependant transformation in the filter space. It is usually followed by a non-linear activation like ReLu. They are used to reduce dimensionality.

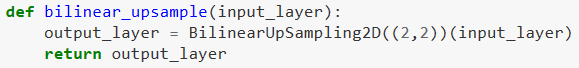
The ultimate aim is for the model to classify the inputs correctly based on the processing done by different layers, in whatever size, shape, position and location the object is in.

The following concepts were implemented in this network:

**Batch Normalization –** This is the process of fitting the input layer in such a way that the mean activation is zero and the standard deviation is 1. This allows the network to be trained faster and provides regularization



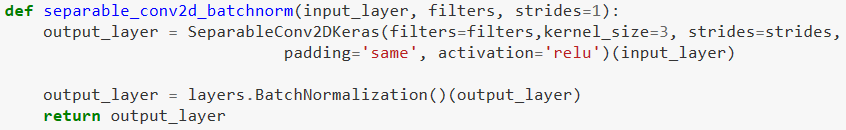
**Bilinear Upsampling –** This gathers the weighted average of the 4 nearest pixels



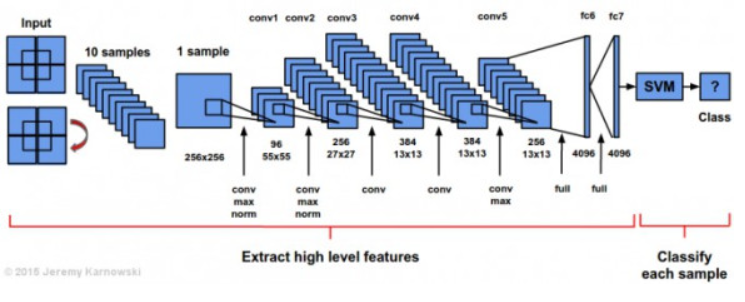
**Layer Concatenation –** creates an element wise addition of two layers to retain more info



**Separable Convolutions –** a convolution performed on each layer + a 1x1 convolution layer. This helps in reducing the number of parameters and increasing encoder efficiency

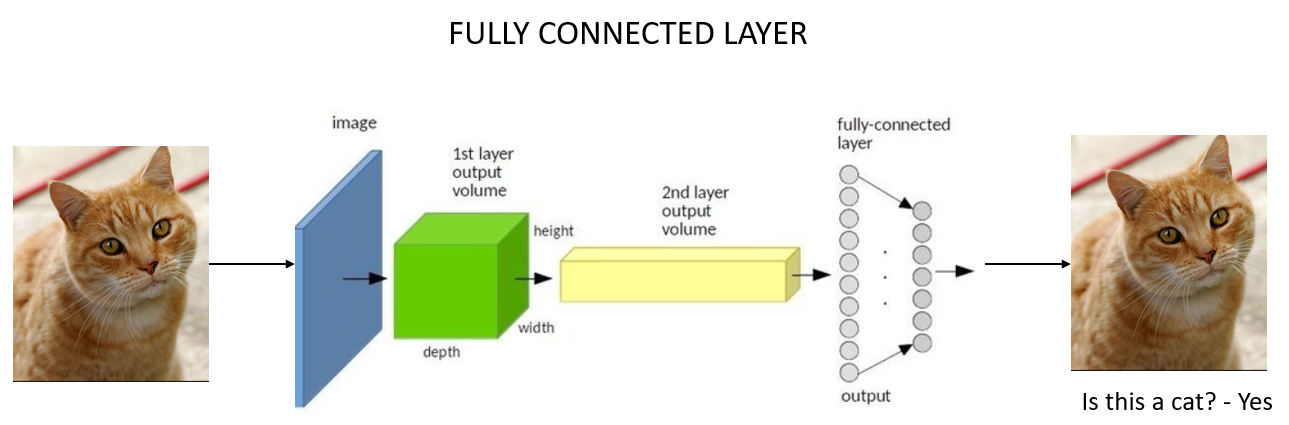


The below gives a better detail of how each layer processes the image

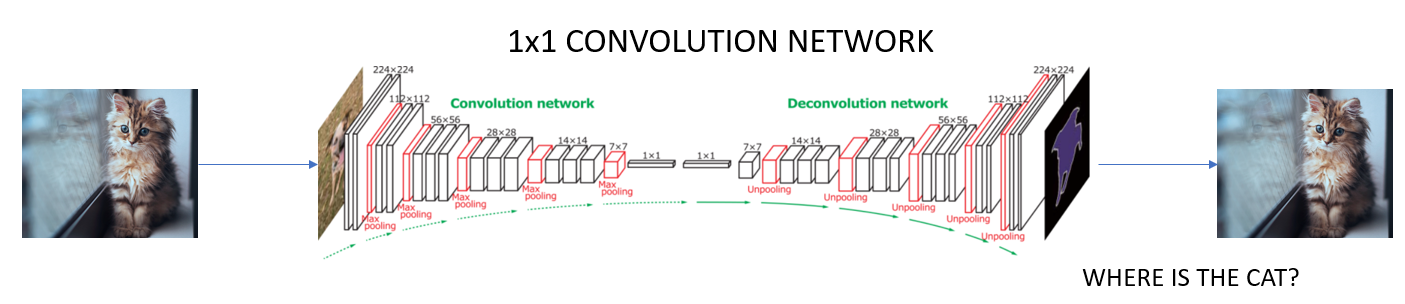


**1x1 Convolution Vs. FCN**

A fully connected layer works on the images provided during training and identifies it. But this is provided that the image to be valuated is very near to the one provided in training.



But this is not enough since this is not what exactly happens in real time. The model to preserve spatial information as well. This is achieved by integrating convolutions directly into each layer.



This enables the fully convolutional network to work on images of any size. There is also the added advantage of upsampling and using skip connection to prevent loss of information.

Fully Connected Networks use 3 special techniques:

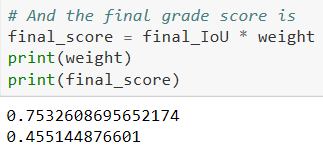
* Replace fully connected layers with 1x1 convolutional layers
* Up-sampling by using transposed convolution
* Skipping connection between layers

**Follow Me Project**

Training this network required at least 30 epochs. After that there was a bit of stabilization. But anything more than 50 epochs caused overfitting. The learning rate needed to be low to enable the network to learn. 3 training curves are given for example.

|  |  |  |
| --- | --- | --- |
| **Epoch 2** | **Epoch 15** | **Epoch 45** |
|  |  |  |

The result was the following: 45% accuracy



1. The student displays a solid understanding of the limitations to the neural network with the given data chosen for various follow-me scenarios which are conveyed in the write-up.

**Observations and Comments**

Training takes a lot of time. A lot of data is fed into the training and validation part for the network to perform well. Model\_training took up the entire night on an AWS to train. Collecting the data to train the model took around 2-3 hours per run.

**Limitations**

The hero in this sample is distinctly different from other people. This enables easy identification. This step is not so simple in a real time scenario.

For example, if it is a cat or a dog, the network needs to be architected with more layers, filters and more training. This is because of the variety of the objects to be identified. That is, a cat may be white, black, ginger or multi coloured. It may even be with no hair. Similarly, with a dog, they are of different sizes, shapes and breeds. Hence the network needs to be a lot more complex. The weights to be shared across the filters are more and statistical invariance needs to be taken into account. Since the simple lines and shapes recognized in the first layers are to be translated into much complex shapes and then into characteristics of a cat or a dog. Another challenge is the training data size is large to avoid overfitting.

Extending this to identifying humans, the network needs a lot of power to be built and trained. Although it would be challenging to build and train such a model.

**Future Enhancements**

* More data/images can be used to train the model
* Add more GPUs to train the model faster
* Lower the learning rate, although training would take longer

MODEL

1. The model is submitted in the correct format. – As attached
2. The neural network must achieve a minimum level of accuracy for the network implemented. – As submitted