**Pseudo Code**

split rgb -> (y componenet,CbCr component)

Convert y component to rgb

Convert CbCr component to rgb

[missing]

model1 = Sequential()

model1.add(AtrousConvolution2D(maps=64, l=7, b=7, stride=2,input\_shape=(3, 256, 256)))

model1.add(\_residual\_block(maps=64,l=3, b=3,stride=2 )

model1.add(\_residual\_block(maps=64,l=3, b=3,stride=1 )

model1.add(\_residual\_block(maps=64,l=3, b=3,stride=1 )

model1.add(\_residual\_block(maps=128,l=3, b=3,stride=2 )

model1.add(\_residual\_block(maps=128,l=3, b=3,stride=1 )

model1.add(\_residual\_block(maps=128,l=3, b=3,stride=1 )

model1.add(\_residual\_block(maps=128,l=3, b=3,stride=1 )

for i in range(23):

model1.add(\_residual\_block(maps=256,l=3, b=3,stride=1 )

for i in range(3):

model1.add(AtrousConvolution2D(maps=64, l=3, b=3, stride=1))

model2 = Sequential()

model2.add(\_masked\_convolutional\_2d(maps=64,l=7,b=7,stride=1,input\_shape=(2, 28, 28)))

for i in range(10):

model2.add(\_gated\_convolutional\_2d(maps=64,l=7,b=7,stride=1))

model2.add(\_masked\_convolutional\_2d(maps=1024,l=1,b=1,stride=1))

model2.add(\_masked\_convolutional\_2d(maps=32\*2,l=1,b=1,stride=1))

model3 = Sequential()

model3.add(AtrousConvolution2D(maps=64, l=3, b=3, stride=2,input\_shape=(3, 224, 224)))

model3.add(AtrousConvolution2D(maps=128, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=128, l=3, b=3, stride=2))

model3.add(AtrousConvolution2D(maps=256, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=256, l=3, b=3, stride=2))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=256, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=2))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=2))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=1024, l=5, b=5, stride=1))

model3.add(AtrousConvolution2D(maps=512, l=3, b=3, stride=1))

for i in range(2):

model3.add(AtrousConvolution2D(maps=128, l=3, b=3, stride=1))

model3.add(Bilinear\_upsampling())

for i in range(2):

model3.add(AtrousConvolution2D(maps=64, l=3, b=3, stride=1))

model3.add(Bilinear\_upsampling())

for i in range(2):

model3.add(AtrousConvolution2D(maps=32, l=3, b=3, stride=1))

model3.add(AtrousConvolution2D(maps=2, l=1, b=1, stride=1))

**Improvements**

The model with a slight variation can be used to Convert Grayscale Videos into Colored Videos. The reason why the current model isn’t apt to do the same is that, if we were to pass a video to the model, it would each each frame as a separate task and predict colors independently of one another. Hence, we would get drastic color changes in the video based on when the lighting of the frame changes slightly of the perspective of the viewer changes.

**Conclusion**

The model performs well on images that are decently lit. It predicts very light colors for gray scale images are are very bright, and predicts no color at all for images that are poorly lit. It performs extraordinarily well on night lit images. On a few images we can see a aberration between the sky and the monument edges. This is seen when there is a lot of disturbance in the sky, like heavy clouds and etc. Otherwise the Images look almost like original Image

**Observation**

We conducted a survey for around 117 people, and showed them a variety of grayscale and predicted images. About 725 of them felt that most of the images shown to them were not digitally touched and were the original image







