Fakharyar Khan
November 27th, 2023
Professor Curro
Deep Learning

**Project #7 Writeup**

In this project, I implemented the model, SIREN, discussed in *Implicit Neural Representations with Periodic Activation Functions.* In this paper, they found that by using sinusoidal activation functions they can better model differentiable signals like images, videos, and audio than typical neural networks that use activation functions like relu. The reason for this is because by using activation functions like relu, the signals represented by the model only agree with the actual signal by at most their first derivative since relu is piecewise linear. However, by using sinusoidal activation functions, it turns out that the derivatives of signals produced by the model are the outputs of another model of the same form. This means that SIREN models have the capability of creating a representation of a signal that agrees with the actual signal on many of its derivatives. And for image processing, this translates to the model more easily being able to capture high frequency and fine details like edges.

One application of this model is in image inpainting where the model is trained to extrapolate and recover missing details. To demonstrate this capability, I had my SIREN implementation train on a masked version of the Test Card F image where 10% of the pixels have been removed giving the image shown in Figure 1. After training on the masked image, the model was able to recover the actual image as well as most of the fine details as can be seen below.

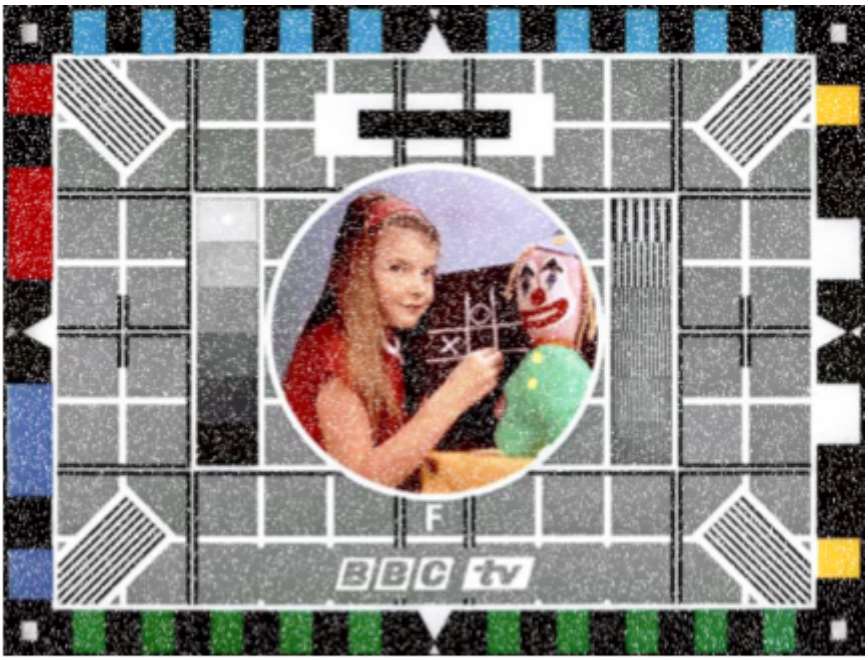


Figure 1: Masked Training Image

Following the normalization scheme as discussed in the paper also ended up being really important. When I didn't do it (I just used the regular Xavier Glorot initialization), my model took around 30,000 iterations to reach an MSE of 0.004 but when I normalized the weights, I got the same MSE in around 4000 iterations.

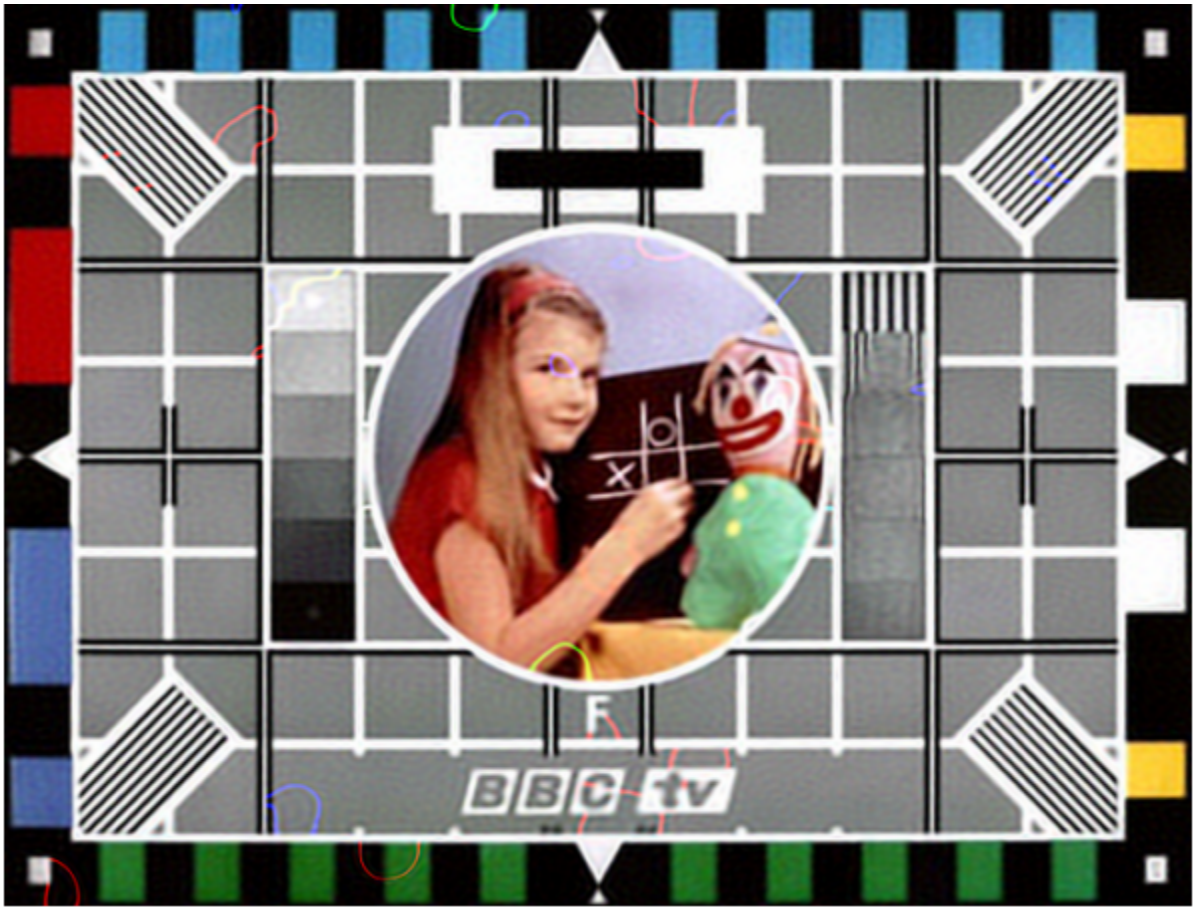


Figure 2: Recovered Image

One other interesting application of this model is in super-resolution imaging. We can evaluate the trained model on a denser grid of coordinates so that we can blow up the training image and increase its resolution dramatically. In this example, the trained image was 530x700 and a 4x resolution was applied which resulted in a 1060x1400 output image. In Figure 3, you can really see the difference in quality because when you zoom into the normal image, you can see that the edges in the image like the hair and eyes are very pixelated whereas in the upsampled image, even at this level, the image still looks smooth.

Figure 3: Input and Upsampled Image

One thing that I couldn't figure out was why my recovered image had these weird squiggly lines randomly placed throughout the image. Everything else about the recovered image is fine except for the squiggly lines and I don't why they showed up. The color also seems to be random and it doesn't look like they match the surrounding pixels at all.