The goal of this project is to create a SIREN model that is capable of memorizing an image. The SIREN model is an implicit neural representation model, which means that the model can be trained to become a solution to a differential equation. The SIREN model differ from other implicit neural representation models in that it uses a sinusoidal activation function. In [1], the authors demonstrate the model's ability to outperform other implicit neural representation model. Taking in two-dimensional coordinates as inputs and returning colors as outputs, the model is able to represent an image in a continuous manner.

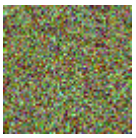
```
if first:
    radius = 1.0 / num_inputs
else:
    radius = tf.math.sqrt(c / num_inputs) / omega_0

self.w = tf.Variable(
    rng.uniform(
        shape=[num_inputs, num_outputs], minval=-radius, maxval=radius
    ),
    trainable=True,
    name="SirenLayer/w",
)
```

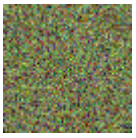
To make the training task work, I made all of the coordinates evenly distributed between -1 and 1 rather than the standard pixel coordinates of the image. I also made the outputs between 0 and 1 and multiplied by 256 afterwards. The SIREN model wasn't able to use the Kaiming He initialization method because of the sinusoidal activation functions, and the initialization for the weights on the first layer had to be changed to account for the uniform distribution of inputs. I ended up using the initialization method described in [1], with the code for the weight initialization shown above.

63455
20042
Repeat

I trained the SIREN model for 15000 iterations with five hidden layers with a width of 256, as was done in the paper. For the training task, I used an image from the Kodak dataset [2]. I didn't use a batch size and trained with every pixel on each iteration. The model ended up representing the image fairly well, but the output is still noticeably noisy. Some of the white pixels from the original image are purple in the model's representation, which is probably due to the model outputting a value greater than one, which ended up clipping when being converted into a .png file. The original image and the model's representation can be seen above.



Due to the continuous nature of the representation, I wanted to upscale the image by putting in more coordinates than were trained on. I used an image from the CIFAR dataset [3] and trained the SIREN model on only 200 iterations, due to the low resolution. While the implicit neural representation was good at the correct resolution, it is completely unrecognizable at the increased resolution. The original image, the neural representation, and the upscaled version can be seen above.



I guessed that the issue was due to the image representation being more complicated than necessary, and attempted to fix the issue by using a simpler model. However, even when creating a model with one hidden layer with a width of one, the representation works at the original resolution but not at the upscaled resolution. The neural representation and the upscaled version can be seen above.

63455
20042
Renest

I then saw whether I could get the image from the Kodak dataset to upscale. The result was much better than for the CIFAR image, but is still more noisy than the unscaled representation. I assume there is a regularization method that would upscale the image better. The upscaled representation can be seen above.

[1] https://arxiv.org/pdf/2006.09661.pdf
[2] https://r0k.us/graphics/kodak/
[3] https://www.cs.toronto.edu/~kriz/cifar.html