

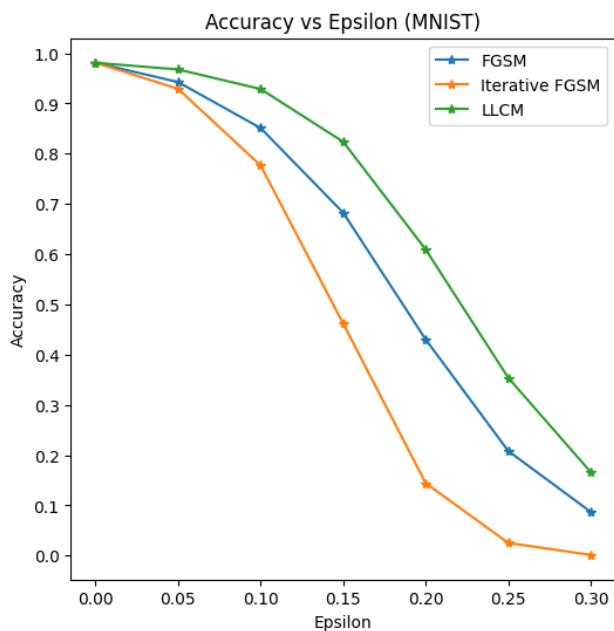
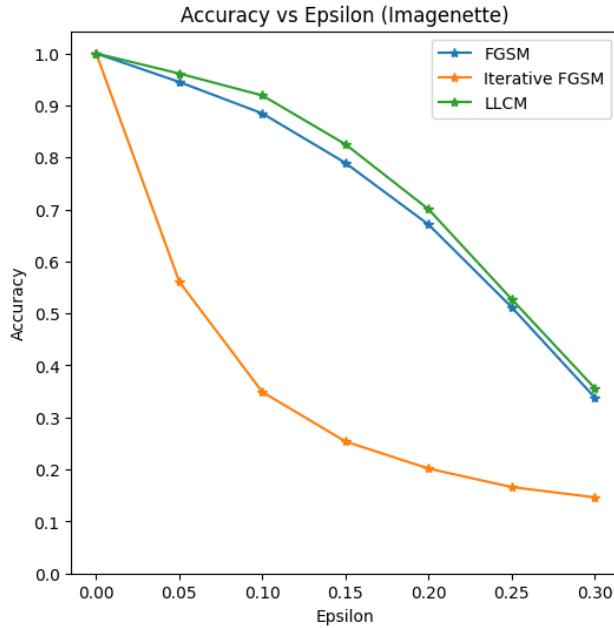
I implemented some standard Adversarial Example Generations algorithms, including Fast Gradient Sign, Iterative - Fast Gradient Sign, and Least Likely Class methods.

I utilized the [Pytorch tutorial](#) and this [paper](#). The Pytorch tutorial provided the framework for the methods and provided inspiration for the visualizations. I chose to attack the ResNet50 model (with Pytorch's default training weights), due to its simplicity, which allowed for relatively fast training times.

I used most of the standard transformations as provided by the Pytorch weight specification, which includes a resize of [232,232] followed by a random crop of [224,224]. The images are then normalized. The weights originally also included interpolation, but I did not use it. I also filtered the final output of ResNet to select the max probability between the 10 classes that are included in the ImageNette dataset.

There were some differences between the paper and my implementation. The MNIST and Imagenette data are normalized between [0,1], while the paper has pixel values ranging from [0,255]. To account for this, I used Pytorch's suggested epsilon values of [0, .05, .1, .15, .2, .25, .3]. For the Iterative Gradient Sign method, I unnormalized some of the inputs, including epsilon, which I multiplied by 255 as the original paper had pixels values in the range [0,255]. Further, I also used the α value of 1/255 for the same reason.

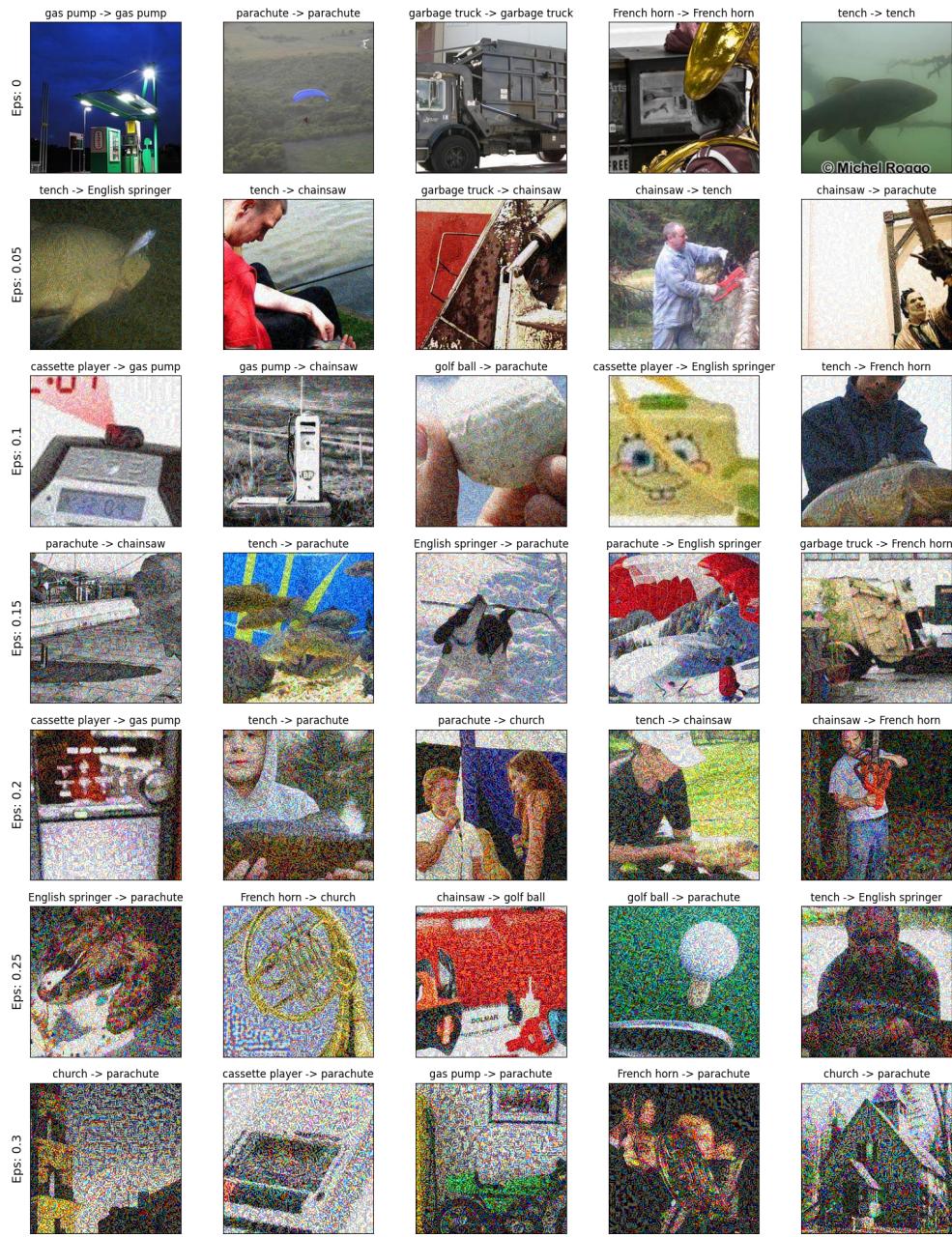
Plots



(iii) Visualize examples of attacks, where the original image and original (correctly predicted) label and the attacked image and falsely predicted label are shown for both MNIST and ImageNette

FGSM

ImageNette



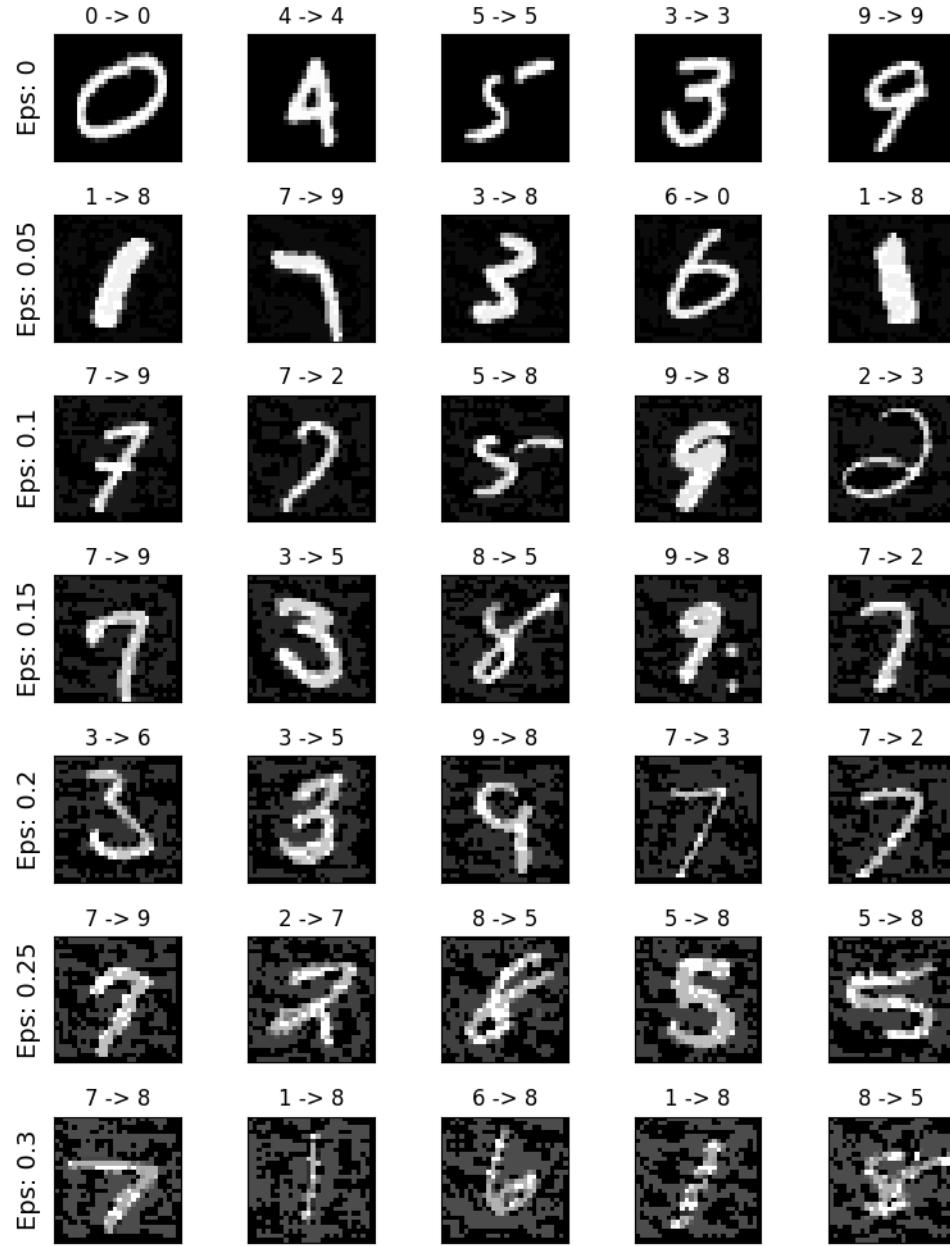
garbage truck



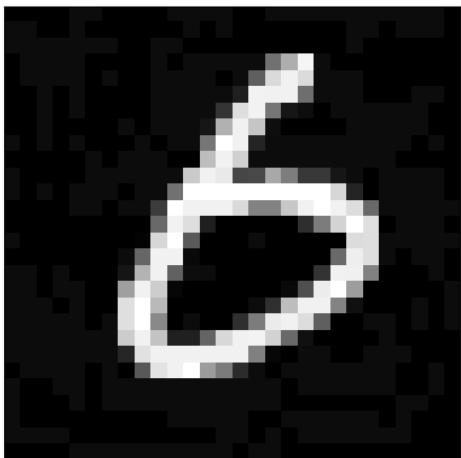
chainsaw



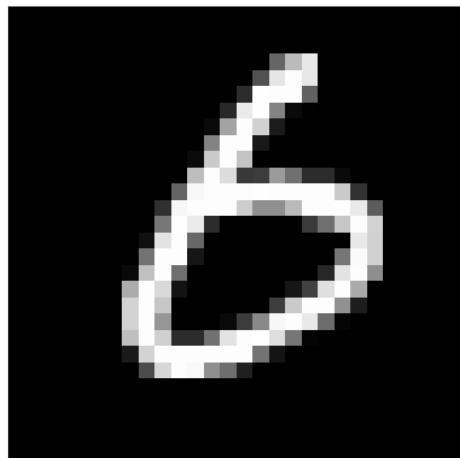
MNIST



6

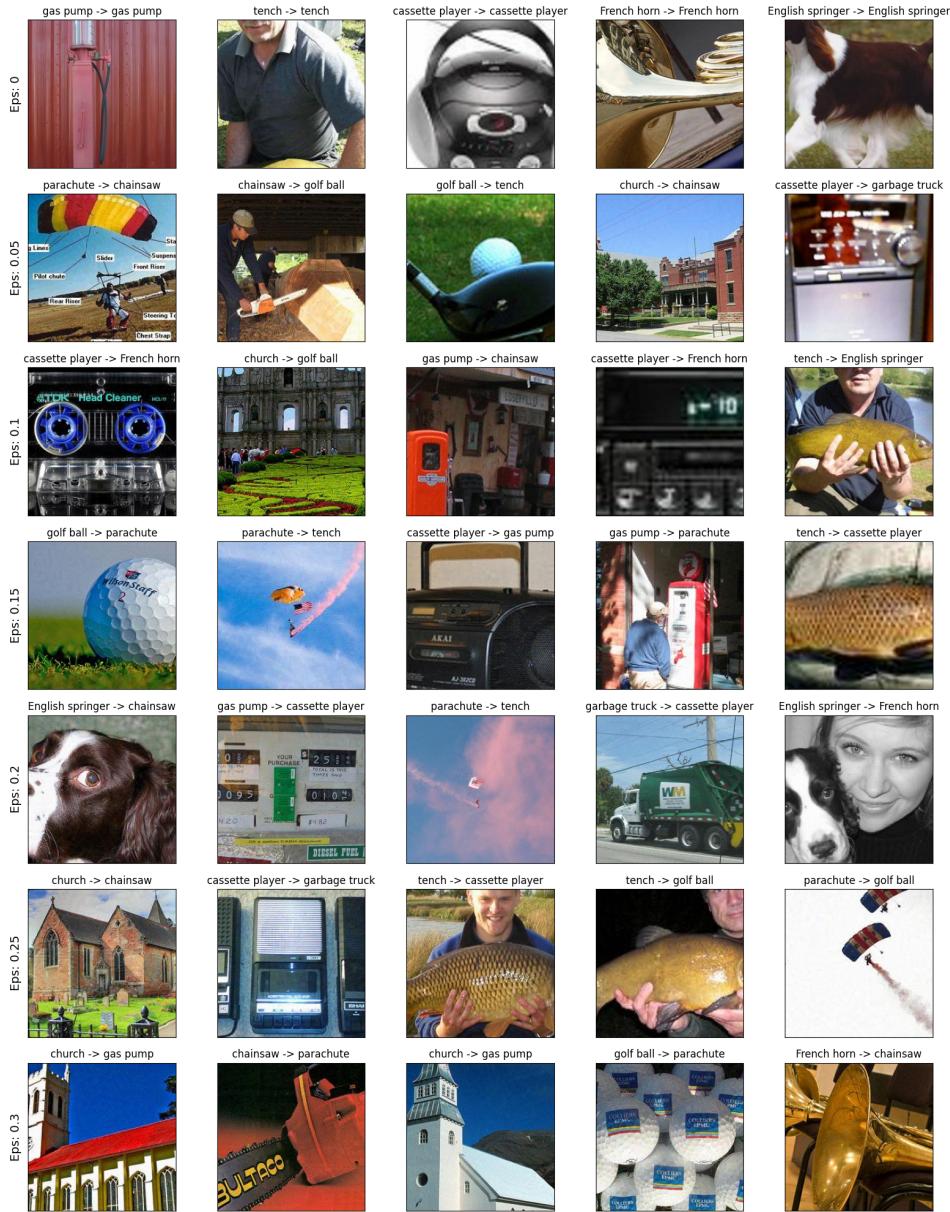


0



IGS

ImageNette



gas pump



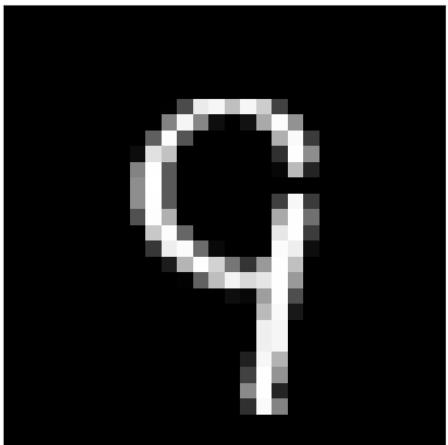
parachute



MNIST



9

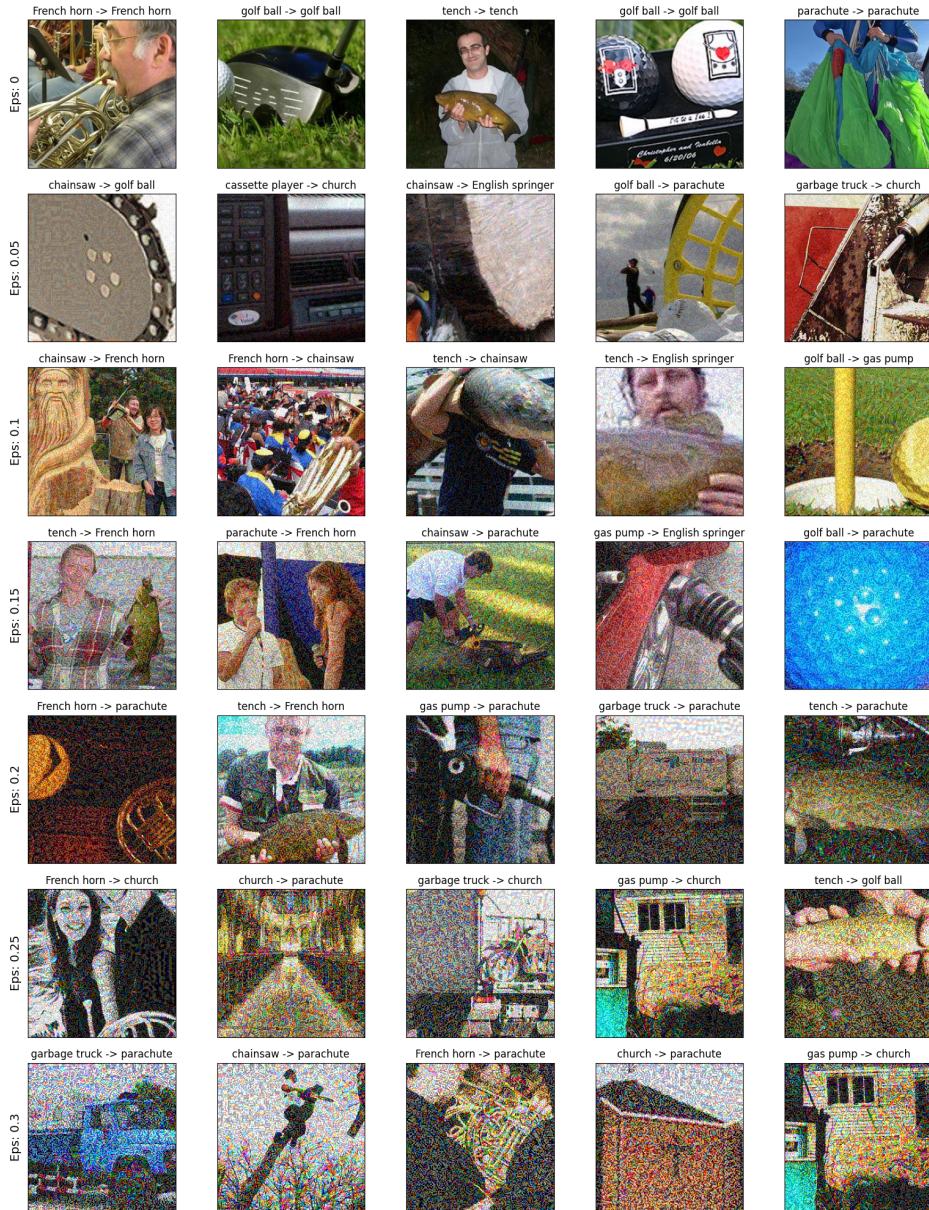


8



LLC

ImageNette



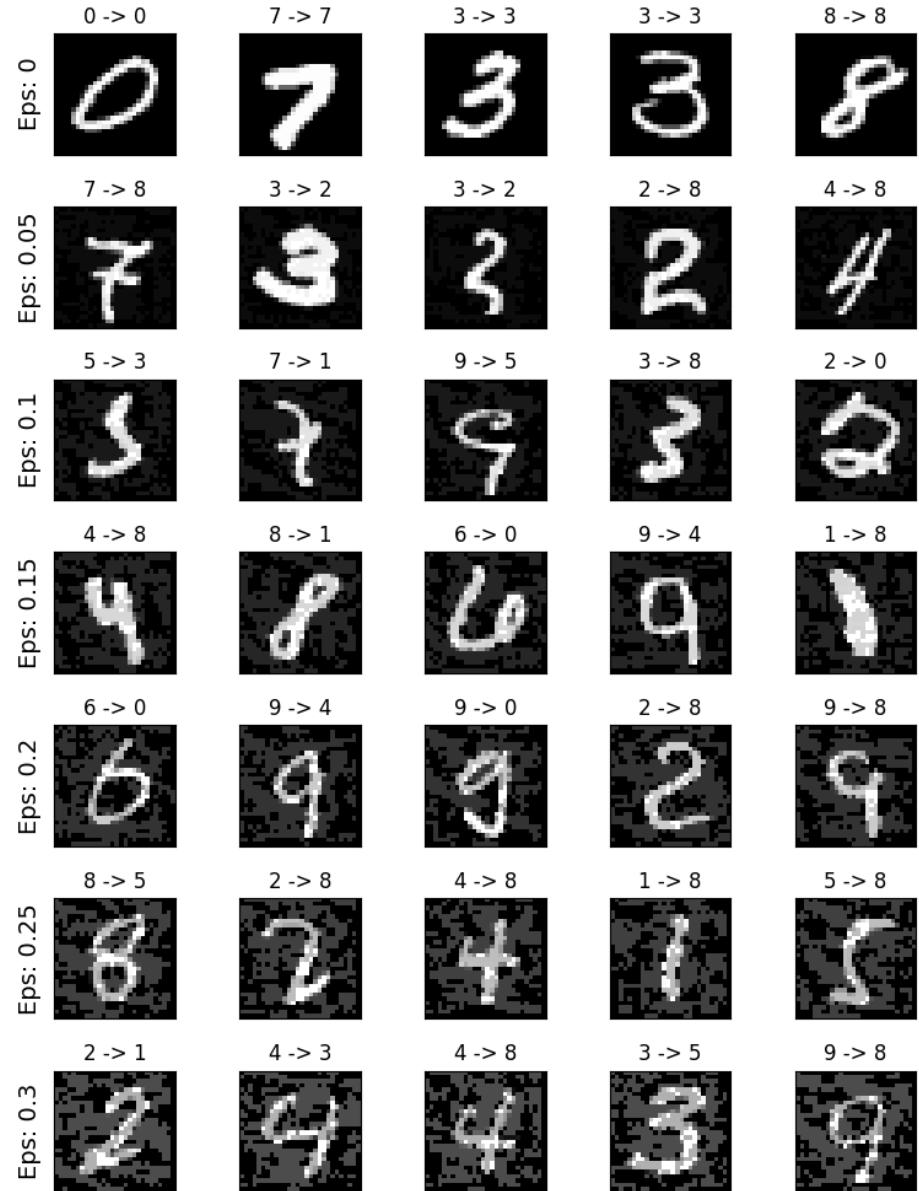
French horn



chainsaw



MNIST



2



8

