Aidan Borne

EE 4745

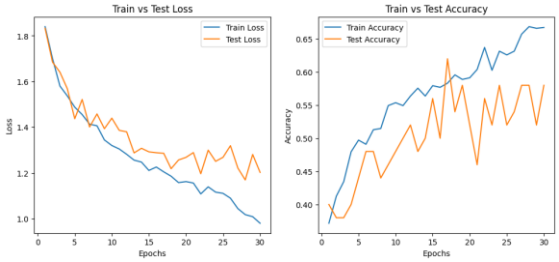
Final Project

December 7th, 2025

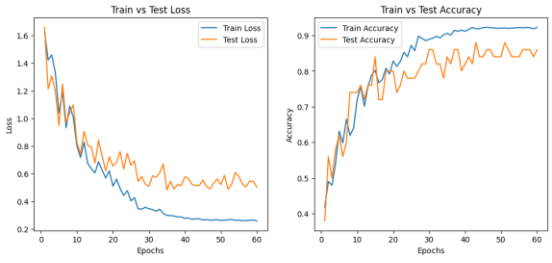
Problem A:

Model	MLP	CNN
Parameters	1,723,466	141,610
Time / Epoch	4.7 s	6.8s
Number of Epochs	30	60
Train Accuracy	66.7%	92.2%
Test Accuracy	58%	86%
Interpretation	Very inaccurate. Saliency maps don't highlight any specific parts of the images.	Much more accurate without overfitting. GradCAM usually focuses on relevant portions of images.

MLP Model:



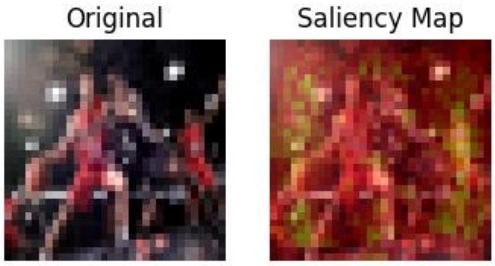
CNN Model:



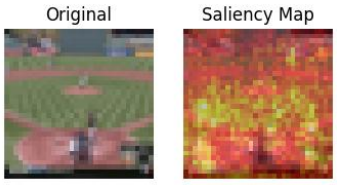
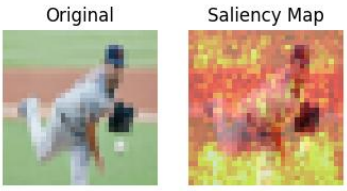
Per-Class Performance:

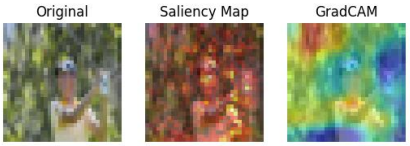
Model	MLP	CNN
Baseball	2/5	4/5
Basketball	4/5	3/5
Football	0/5	4/5
Golf	2/5	4/5
Hockey	5/5	5/5
Rugby	2/5	5/5
Swimming	5/5	5/5
Tennis	3/5	5/5
Volleyball	2/5	1/5
Weightlifting	3/5	5/5

Example Misclassifications:

	Predicted	Actual	Image
MLP	Hockey	Golf	
MLP	Weightlifting	Basketball	
CNN	Tennis	Baseball	
CNN	Rugby	Basketball	

Interpretability:

MLP	Correct	Incorrect	Mispredict
Baseball			Football
Basketball			Weightlifting
Golf			Hockey

CNN	Correct	Incorrect	Mispredict
Baseball			Tennis
Basketball			Rugby
Golf			Football

Analysis:

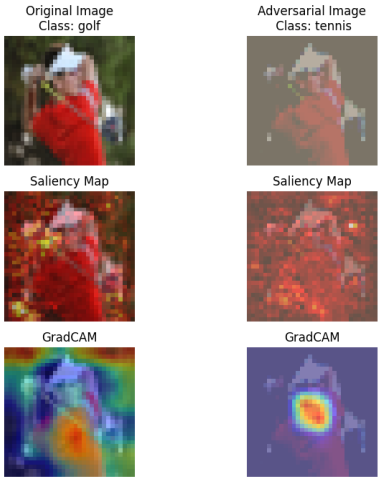
The MLP model is composed of three linear layers that get progressively smaller from 512 to 64 to 10. Each layer is batch-normalized and activated with a ReLU function. The CNN model has three convolutional layers and a linear classifier. Each convolutional layer doubles the channels from 32 to 128, but the image size is divided in half using a max pool. Furthermore, the layers are 2D batch-normalized and activated with a ReLU function. The classifier consists of an adaptive average pool, a batch-normalization, a linear layer of size 128, and a ReLU activation function. Both models are trained on 32x32 images with the following augmentations: random resized crop, random horizontal flip, color jitter, random rotation, random affine, and normalization. The augmentations are used to increase the effectiveness of the limited training set, but they are kept low to prevent destroying the original details.

Neither model shows signs of serious overfitting as their train vs. test accuracies are very similar, as are their train vs. test losses. However, the CNN model generalizes better because it has significantly higher accuracy on the test set, and it performed better than the MLP model on every class besides volleyball. These results are most likely because a CNN model is simply more suited for image recognition tasks. Experiments with fine-tuning MobileNetV3 or ResNet18 models didn't yield improved accuracy over the custom CNN model; they would overfit on the small train set while leaving the test accuracy around 80%.

The MLP model had a final test accuracy of only 58% despite having significantly more parameters, while the CNN model had a final test accuracy of 82%. In the interpretation for the MLP model, the saliency map shows that the important pixels are virtually random, causing mispredictions like a golf image as hockey for having a white background. In comparison, the CNN model is more likely to focus on what's actually important, such as how the GradCAM focuses on the stadium for baseball or the players for rugby. I was unable to improve the test accuracy of the CNN model further without accidentally causing it to overfit on the training set and degrading test performance.

Problem B:

Fast Gradient Sign Method (Untargeted):

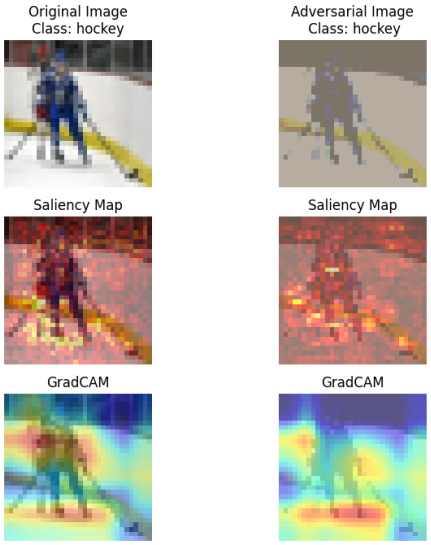


Predicted Logits: -0.4, -2.1, 3.0, 6.8, -3.0, 4.1, -5.3, 1.1, -4.2, -1.8

Mispredicted Logits: 1.8, -1.7, 0.1, 2.8, -4.2, -2.9, -3.3, 5.5, -0.1, -0.5

Perturbation Norm: 56.6820

Attack Successful: True

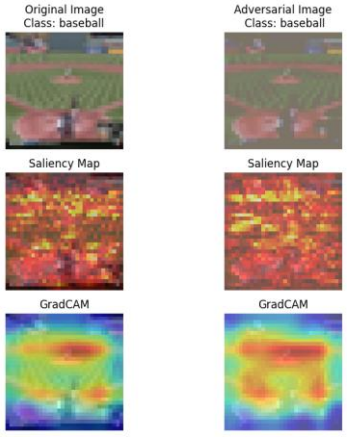


Predicted Logits: -0.6, 0.3, -0.6, -0.1, 12.1, -4.7, -0.2, -1.5, -1.1, -0.6

Mispredicted Logits: -0.2, 1.3, -1.9, -0.6, 6.7, -5.3, -1.3, -0.3, 2.9, 0.0

Perturbation Norm: 47.3740

Attack Successful: False



Predicted Logits: 11.4, -3.7, -3.9, -2.7, -1.8, -0.7, 0.2, 3.2, -0.3, -3.4

Mispredicted Logits: 10.4, -4.4, -2.8, -1.5, -1.5, -2.4, -0.8, 4.7, 0.3, -3.6

Perturbation Norm: 26.3425

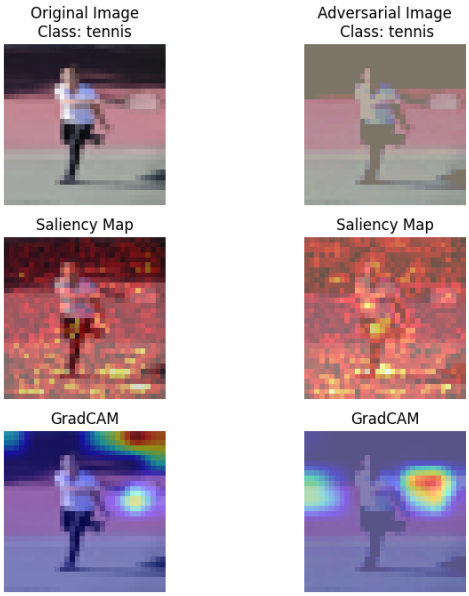
Attack Successful: False

Predicted Logits: -0.7, -0.6, -0.4, -3.5, -1.1, -4.8, -0.5, 4.6, 3.5, 1.8

Mispredicted Logits: 0.3, -3.1, -0.2, 0.4, -1.7, -4.0, -0.8, 6.8, 1.7, -1.4

Perturbation Norm: 42.2916

Attack Successful: False

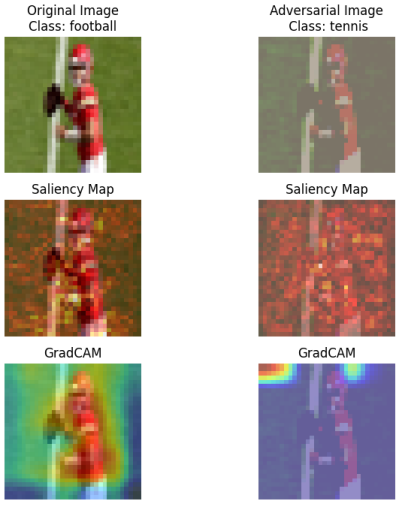


Predicted Logits: 1.4, -3.1, 5.0, 3.9, -2.5, 1.7, -5.6, 0.5, -5.2, -0.6

Mispredicted Logits: 0.6, -1.0, 0.7, 1.7, -2.1, -2.1, -3.3, 3.0, -1.5, 0.8

Perturbation Norm: 39.0214

Attack Successful: True

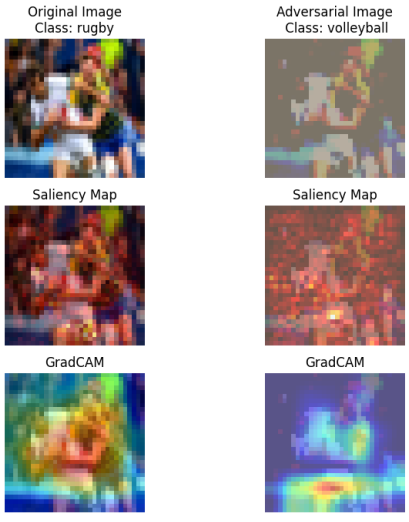


Predicted Logits: 0.2, -0.5, 0.8, -4.0, -2.7, 7.4, -2.1, -1.9, 1.2, -1.6

Mispredicted Logits: 0.8, -0.0, -2.7, -2.9, -1.7, 1.0, 0.8, 0.2, 2.9, -1.2

Perturbation Norm: 56.0873

Attack Successful: True

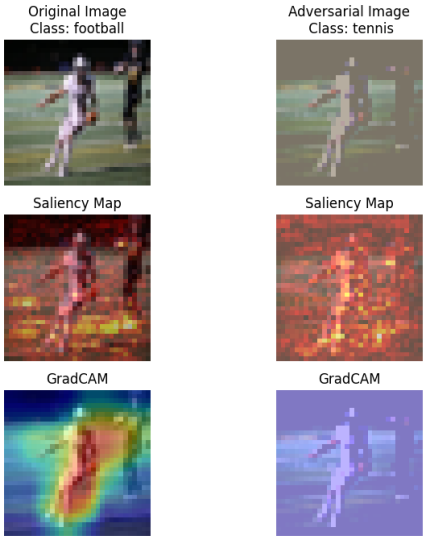


Predicted Logits: 1.6, -0.8, 4.4, -1.9, -3.6, 0.7, -4.5, 0.3, -0.9, 1.5

Mispredicted Logits: 2.4, -0.9, -1.3, -2.0, -2.6, -2.3, -1.2, 2.7, 2.6, 0.5

Perturbation Norm: 51.5889

Attack Successful: True

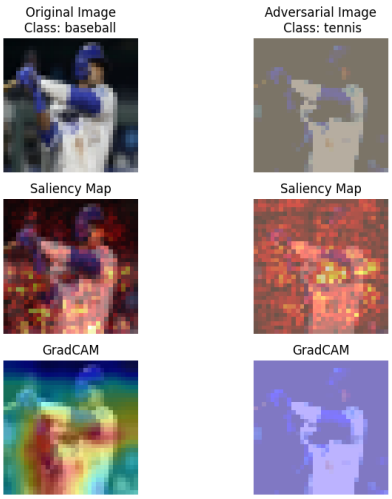


Predicted Logits: 4.1, 0.1, 1.4, -0.1, 0.3, -0.9, -1.8, 0.4, -1.0, -0.6

Mispredicted Logits: 2.6, -0.6, -1.5, 1.1, -0.9, -3.4, -1.6, 2.9, 0.0, -0.3

Perturbation Norm: 65.4409

Attack Successful: True

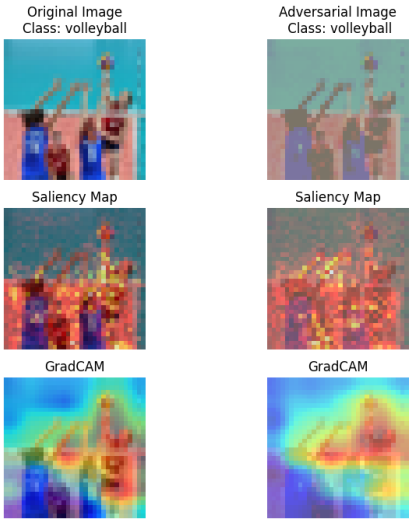


Predicted Logits: -4.4, -1.0, -1.9, -2.2, -4.7, 0.4, 2.0, 2.4, 9.5, 0.8

Mispredicted Logits: -0.8, -1.8, -5.3, -2.3, -5.3, -2.8, 3.9, 4.6, 9.5, 0.6

Perturbation Norm: 40.3929

Attack Successful: False

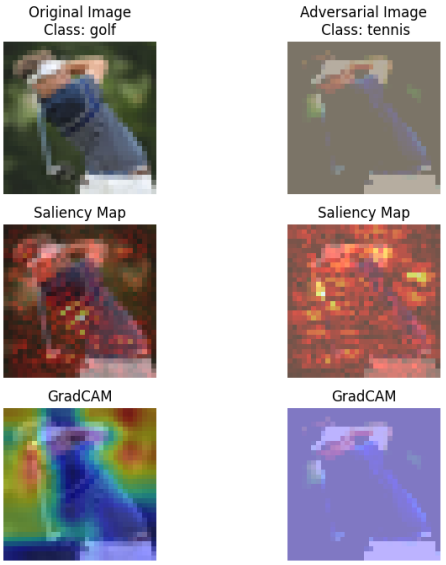


Predicted Logits: 1.3, -2.6, 2.0, 6.5, -1.4, 1.8, -2.5, 2.3, -5.1, -0.9

Mispredicted Logits: 1.8, -0.8, -1.9, 1.5, -0.9, -4.4, -2.6, 3.7, 0.6, 1.2

Perturbation Norm: 50.5646

Attack Successful: True



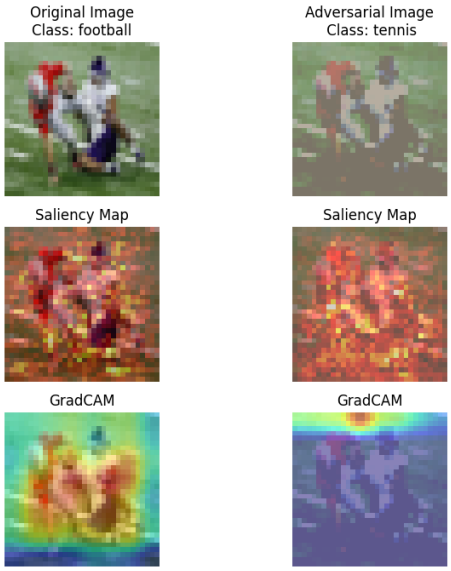
Fast Gradient Sign Method (Targeted to 'basketball'):

Predicted Logits: 1.4, -4.4, 8.7, 1.1, -3.1, 4.2, -5.6, -1.0, -3.9, 0.4

Mispredicted Logits: 2.3, -1.3, -1.5, -1.6, -5.2, -2.1, -1.2, 3.7, 2.8, 1.7

Perturbation Norm: 31.3857

Attack Successful: False

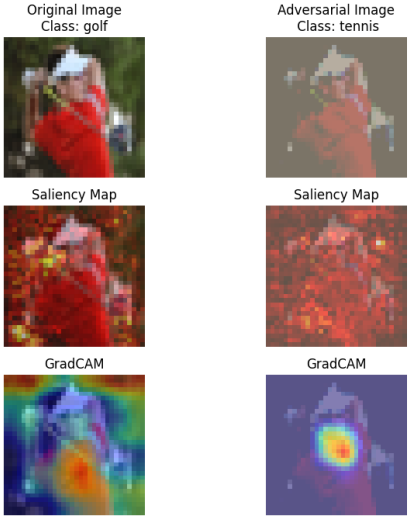


Predicted Logits: -0.4, -2.1, 3.0, 6.8, -3.0, 4.1, -5.3, 1.1, -4.2, -1.8

Mispredicted Logits: 1.7, -1.4, -0.1, 2.6, -4.1, -2.9, -3.3, 5.5, 0.1, -0.4

Perturbation Norm: 56.6819

Attack Successful: False

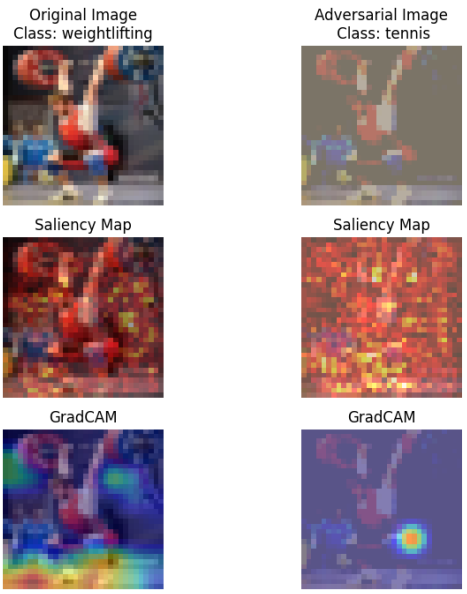


Predicted Logits: -0.3, 2.5, -2.2, -3.5, -1.1, 1.3, -2.9, -1.1, 2.2, 4.2

Mispredicted Logits: 0.3, -0.1, -1.3, -0.2, -2.6, -2.3, -2.5, 3.2, 1.1, 1.3

Perturbation Norm: 46.3138

Attack Successful: False

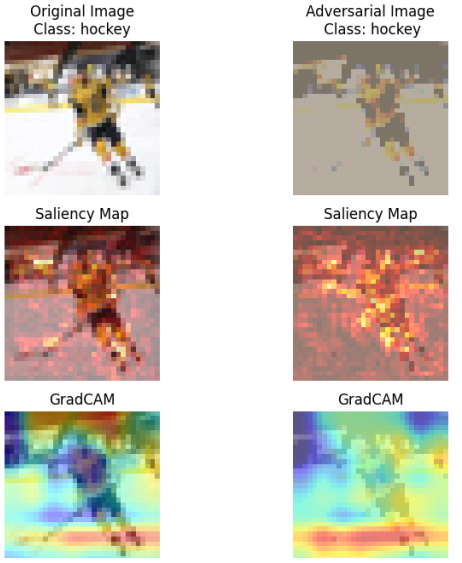


Predicted Logits: -1.8, 3.4, -2.0, -3.3, 9.5, -5.1, 1.5, -2.4, -0.6, 0.3

Mispredicted Logits: -0.5, 2.8, -3.3, -2.6, 5.5, -4.9, -0.5, -1.3, 3.4, 0.5

Perturbation Norm: 57.7580

Attack Successful: False

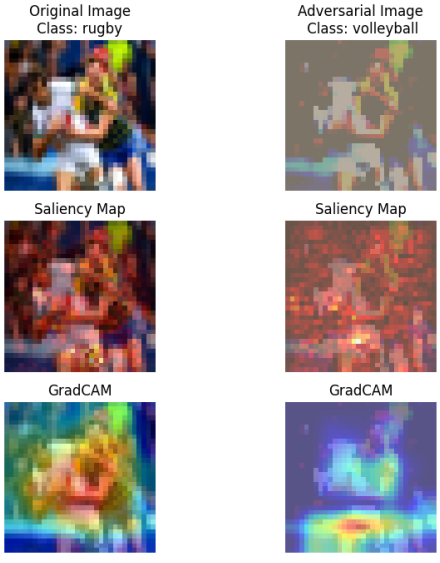


Predicted Logits: 0.2, -0.5, 0.8, -4.0, -2.7, 7.4, -2.1, -1.9, 1.2, -1.6

Mispredicted Logits: 0.7, 0.4, -2.7, -3.2, -1.4, 0.4, 0.7, 0.1, 3.0, -1.1

Perturbation Norm: 56.0874

Attack Successful: False

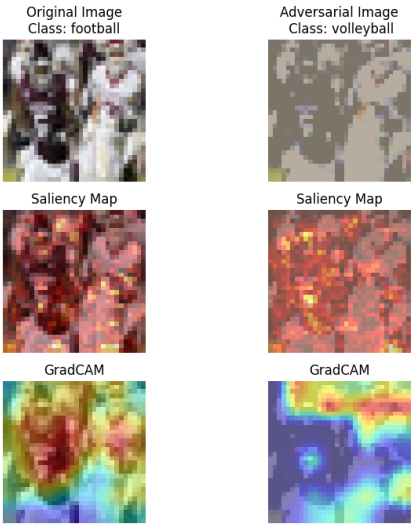


Predicted Logits: 2.5, 0.7, 6.8, -4.4, 5.0, -1.3, -3.6, -5.5, -1.9, 0.4

Mispredicted Logits: -0.8, 0.7, -0.6, -0.7, -0.3, -2.8, -0.7, -0.3, 2.4, 1.5

Perturbation Norm: 43.3840

Attack Successful: False

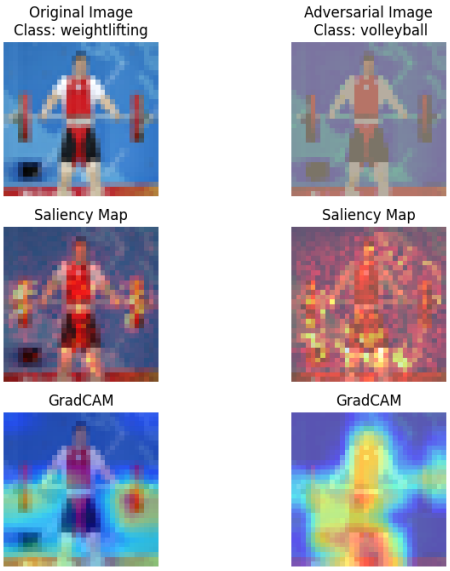


Predicted Logits: -2.0, 1.4, -1.7, -4.4, -3.4, -3.1, -0.4, 0.6, 2.7, 8.7

Mispredicted Logits: 0.1, 0.8, -3.6, -4.5, -3.8, -4.5, 2.0, 3.1, 5.6, 2.4

Perturbation Norm: 39.6126

Attack Successful: False

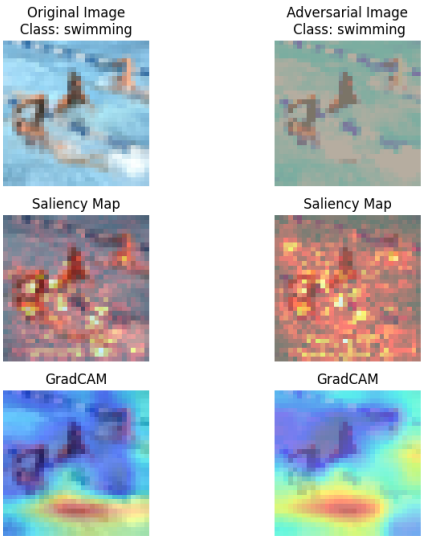


Predicted Logits: -2.0, -2.6, -4.4, -0.6, -1.0, -1.6, 9.9, 3.7, 0.3, -1.2

Mispredicted Logits: 1.5, -1.8, -7.7, -4.5, 0.5, -2.1, 9.3, 1.7, 7.0, -3.4

Perturbation Norm: 37.1302

Attack Successful: False

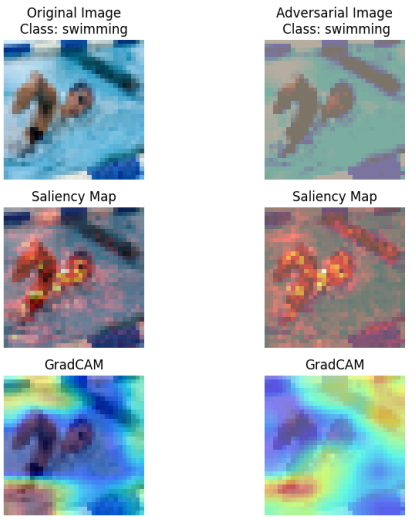


Predicted Logits: -2.8, -1.2, -2.0, -1.5, -0.7, -0.0, 5.6, 0.8, 1.0, 1.0

Mispredicted Logits: -1.7, -1.7, -6.9, -3.4, -1.3, -0.8, 10.3, 1.3, 6.4, -2.1

Perturbation Norm: 39.5960

Attack Successful: False



Predicted Logits: 0.2, -3.7, 3.3, 1.3, -6.4, 9.7, -3.0, -1.1, -2.4, -0.5

Mispredicted Logits: 1.1, -1.3, 1.1, -1.3, -4.7, 1.5, 0.1, 0.2, 1.0, -1.3

Perturbation Norm: 44.9173

Attack Successful: False

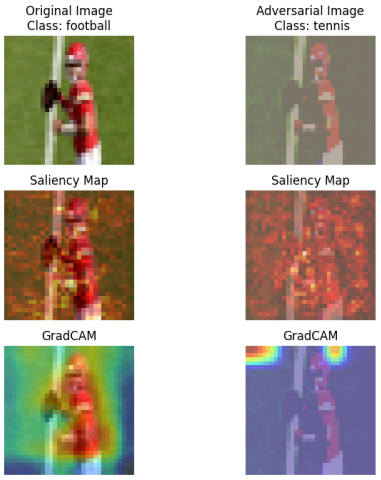
Projected Gradient Descent (Untargeted):

Predicted Logits: 1.4, -3.1, 5.0, 3.9, -2.5, 1.7, -5.6, 0.5, -5.2, -0.6

Mispredicted Logits: 0.6, -1.3, 1.5, 2.1, -2.1, -1.9, -3.6, 3.0, -1.8, 0.5

Perturbation Norm: 39.0206

Attack Successful: True

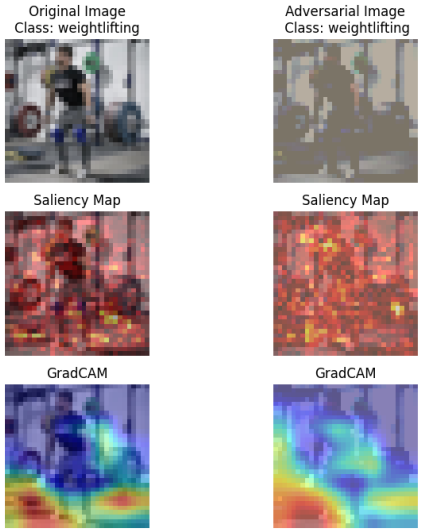


Predicted Logits: 1.0, -0.0, 2.4, -4.1, 3.4, -3.2, -1.7, -2.7, -0.3, 5.0

Mispredicted Logits: -0.8, 0.3, -1.9, -1.7, -1.5, -4.2, 0.5, 2.2, 2.5, 2.6

Perturbation Norm: 35.9546

Attack Successful: False

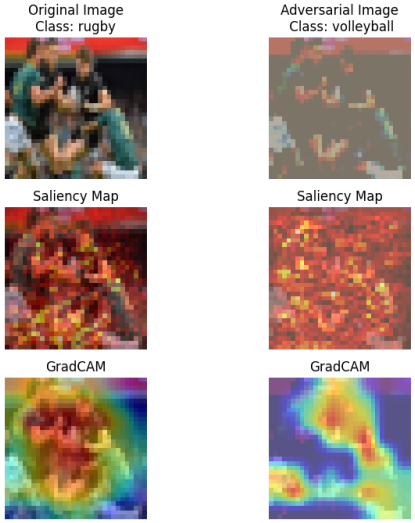


Predicted Logits: -3.6, 2.8, -4.0, -3.6, -3.9, 8.9, 0.7, -2.9, 1.8, 3.6

Mispredicted Logits: 0.7, -0.0, -1.9, -1.4, -3.1, 0.3, 0.1, 1.0, 2.0, 0.4

Perturbation Norm: 52.9746

Attack Successful: True

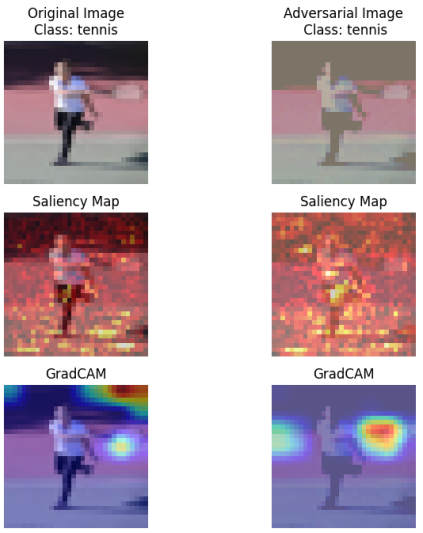


Predicted Logits: -0.7, -0.6, -0.4, -3.5, -1.1, -4.8, -0.5, 4.6, 3.5, 1.8

Mispredicted Logits: 0.4, -3.2, -0.3, 0.5, -1.8, -4.2, -0.8, 7.4, 1.6, -1.5

Perturbation Norm: 42.2901

Attack Successful: False

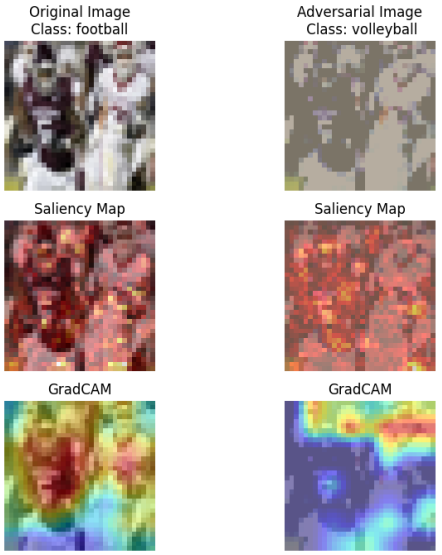


Predicted Logits: 2.5, 0.7, 6.8, -4.4, 5.0, -1.3, -3.6, -5.5, -1.9, 0.4

Mispredicted Logits: -0.5, -0.5, 0.9, -0.1, -0.5, -2.5, -1.1, -0.2, 1.6, 1.3

Perturbation Norm: 43.3836

Attack Successful: True

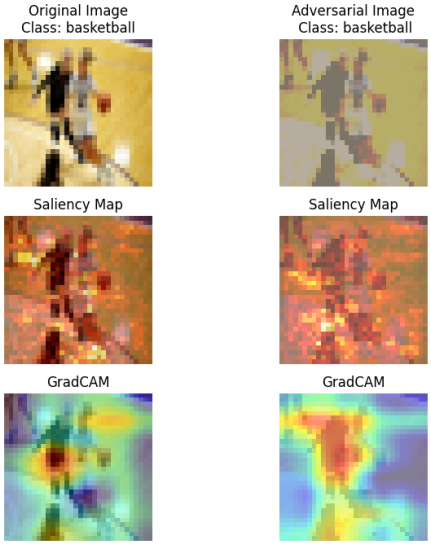


Predicted Logits: -2.0, 7.8, 0.3, -1.0, 0.5, -0.5, -3.8, -5.8, 4.1, -0.6

Mispredicted Logits: 1.7, 4.0, -0.9, -1.4, 1.5, -3.0, -2.9, -1.8, 2.2, -1.4

Perturbation Norm: 37.7190

Attack Successful: False

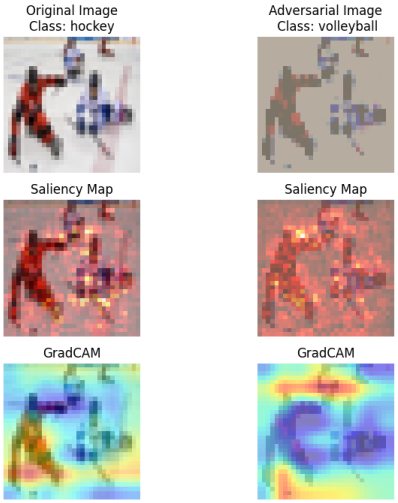


Predicted Logits: -0.5, 1.8, 1.0, -5.4, 10.2, -4.7, -0.9, -5.4, 2.1, 0.3

Mispredicted Logits: -2.1, 1.3, -1.7, -1.8, 3.0, -4.4, 0.3, -1.2, 4.7, 1.2

Perturbation Norm: 43.1207

Attack Successful: True

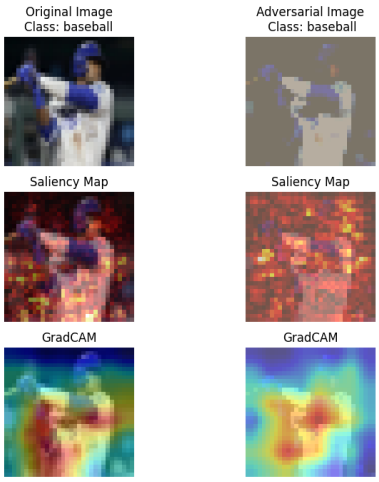


Predicted Logits: 4.1, 0.1, 1.4, -0.1, 0.3, -0.9, -1.8, 0.4, -1.0, -0.6

Mispredicted Logits: 2.9, -0.6, -1.5, 1.0, -0.9, -3.4, -1.6, 2.8, -0.1, -0.4

Perturbation Norm: 65.4407

Attack Successful: False

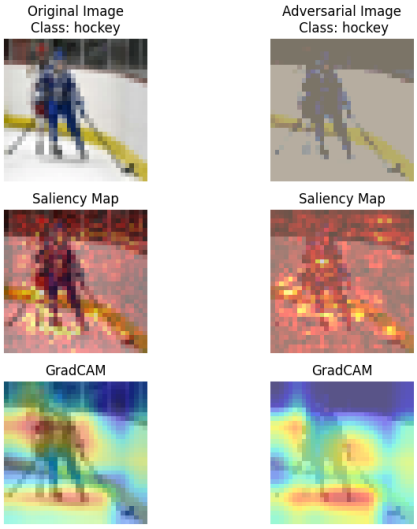


Predicted Logits: -0.6, 0.3, -0.6, -0.1, 12.1, -4.7, -0.2, -1.5, -1.1, -0.6

Mispredicted Logits: -0.2, 1.3, -1.9, -0.7, 7.0, -5.4, -1.3, -0.5, 2.9, -0.0

Perturbation Norm: 47.3738

Attack Successful: False

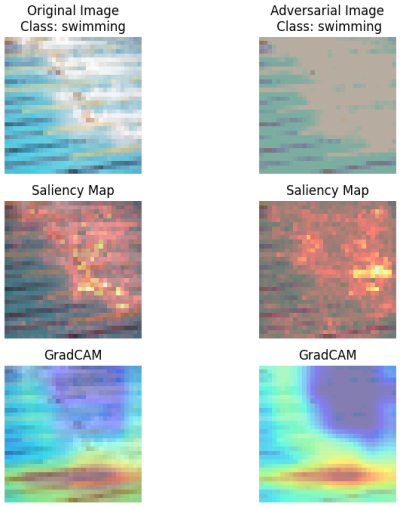


Predicted Logits: -1.3, -0.6, -3.8, -1.7, 3.8, -4.3, 12.2, 1.1, 2.2, -4.3

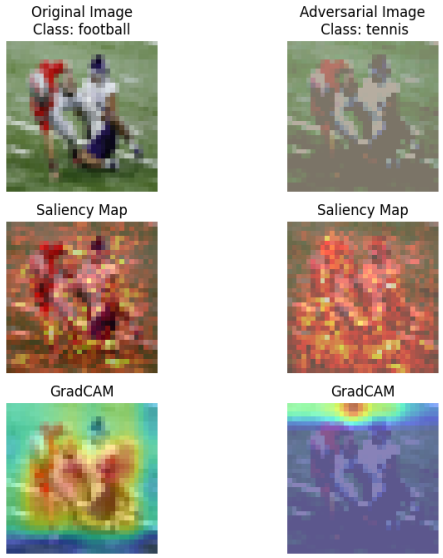
Mispredicted Logits: -1.5, -2.3, -4.2, -1.0, 0.8, -3.3, 9.5, 2.8, 5.1, -4.2

Perturbation Norm: 39.0147

Attack Successful: False



Projected Gradient Descent (Targeted to 'basketball'):

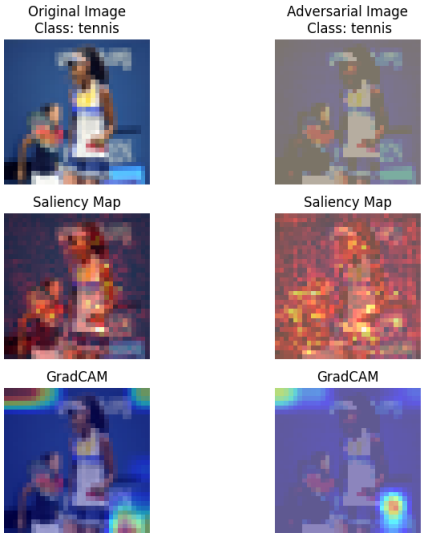


Predicted Logits: 1.4, -4.4, 8.7, 1.1, -3.1, 4.2, -5.6, -1.0, -3.9, 0.4

Mispredicted Logits: 2.4, -0.6, -1.8, -2.2, -5.1, -2.1, -1.3, 3.3, 3.1, 2.0

Perturbation Norm: 31.3847

Attack Successful: False

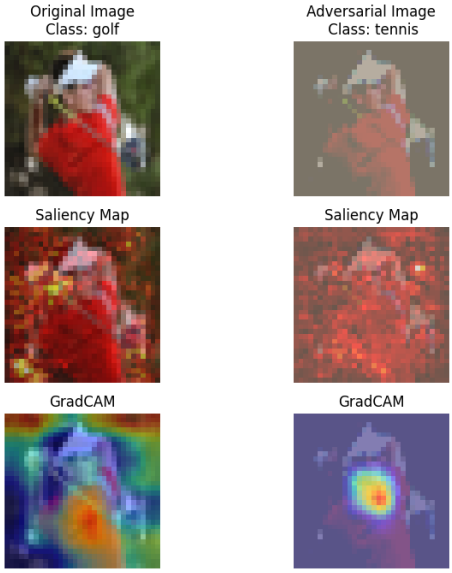


Predicted Logits: -1.1, 2.4, -1.4, -2.0, -3.2, -2.5, -0.1, 3.0, 2.4, 2.5

Mispredicted Logits: 0.3, 1.0, -2.2, -1.5, -2.7, -3.6, 0.2, 4.1, 1.7, 1.8

Perturbation Norm: 47.1013

Attack Successful: False



Predicted Logits: -0.4, -2.1, 3.0, 6.8, -3.0, 4.1, -5.3, 1.1, -4.2, -1.8

Mispredicted Logits: 1.8, -1.1, -0.2, 2.3, -4.1, -2.9, -3.5, 5.3, 0.3, -0.4

Perturbation Norm: 56.6818

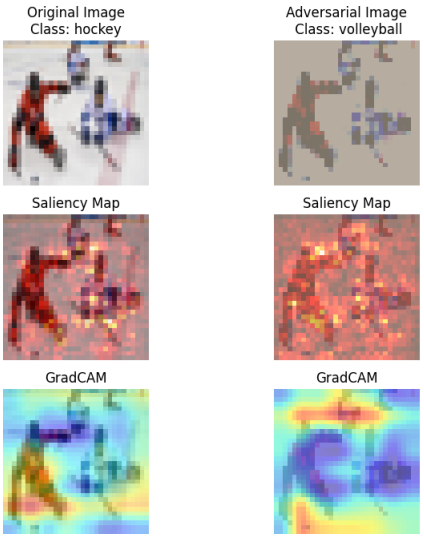
Attack Successful: False

Predicted Logits: -0.5, 1.8, 1.0, -5.4, 10.2, -4.7, -0.9, -5.4, 2.1, 0.3

Mispredicted Logits: -2.2, 1.6, -2.0, -1.9, 2.7, -4.3, 0.4, -1.2, 4.9, 1.4

Perturbation Norm: 43.1207

Attack Successful: False

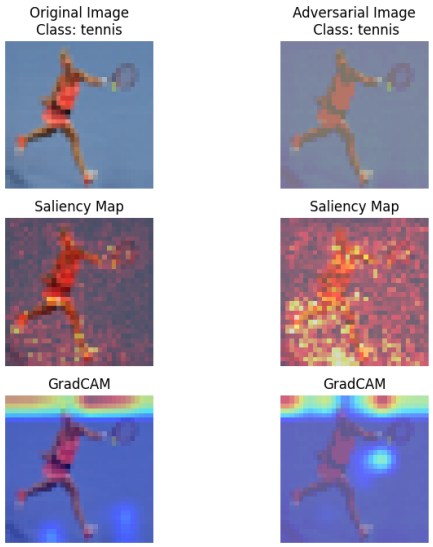


Predicted Logits: -1.9, -0.1, -2.8, -0.6, -3.1, -4.9, 0.1, 6.1, -0.7, 5.9

Mispredicted Logits: -1.2, 0.2, -2.7, -0.9, -2.5, -5.0, -0.3, 6.5, -0.6, 4.1

Perturbation Norm: 17.2812

Attack Successful: False

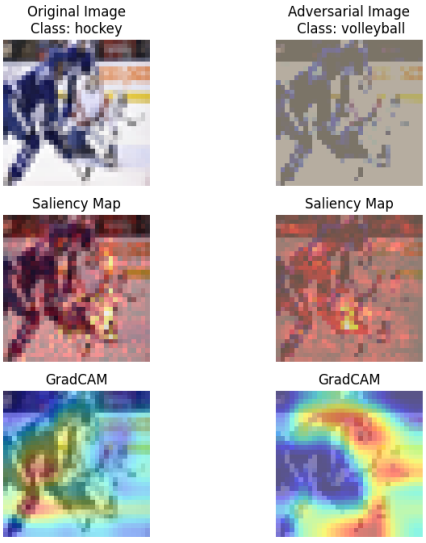


Predicted Logits: 2.0, 0.2, 3.3, -5.4, 10.9, -5.2, 0.9, -4.7, 0.5, -0.5

Mispredicted Logits: 0.6, 2.5, -1.1, -3.5, 3.4, -4.9, -0.6, -1.1, 5.1, 0.3

Perturbation Norm: 49.8422

Attack Successful: False

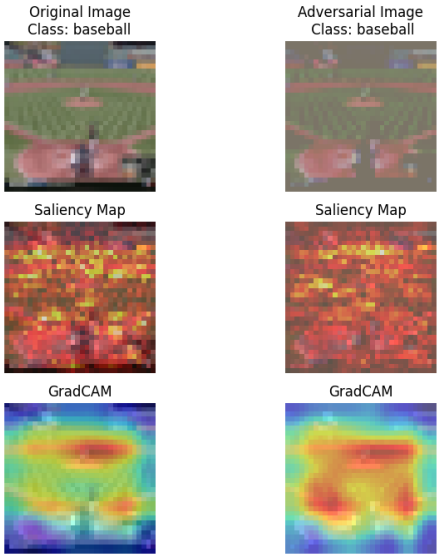


Predicted Logits: 11.4, -3.7, -3.9, -2.7, -1.8, -0.7, 0.2, 3.2, -0.3, -3.4

Mispredicted Logits: 8.0, -2.1, -3.3, -2.4, -1.3, -2.8, -1.2, 3.8, 1.8, -2.0

Perturbation Norm: 26.3414

Attack Successful: False

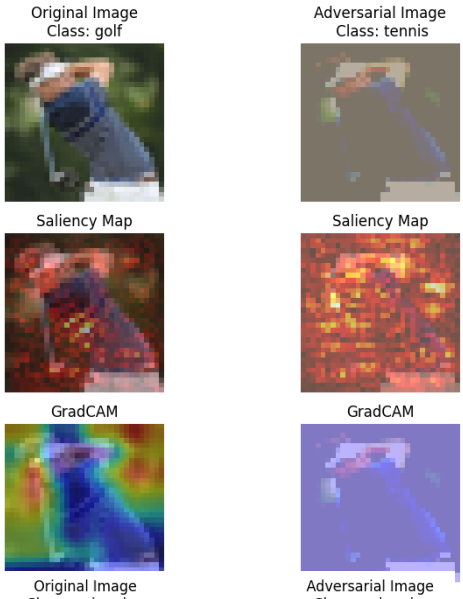


Predicted Logits: 1.3, -2.6, 2.0, 6.5, -1.4, 1.8, -2.5, 2.3, -5.1, -0.9

Mispredicted Logits: 1.6, -0.1, -2.2, 1.0, -0.7, -4.6, -2.7, 3.3, 0.9, 1.4

Perturbation Norm: 50.5644

Attack Successful: False

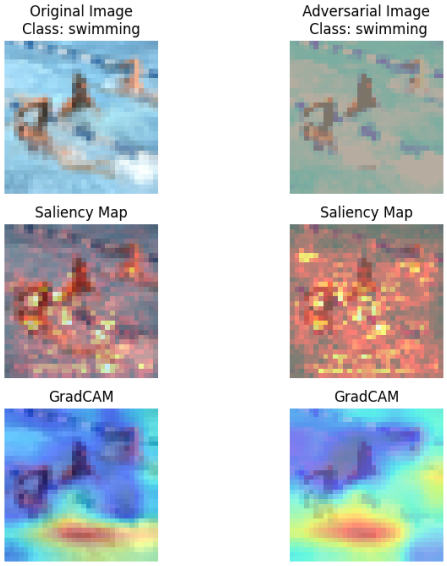


Predicted Logits: -2.0, -2.6, -4.4, -0.6, -1.0, -1.6, 9.9, 3.7, 0.3, -1.2

Mispredicted Logits: 1.6, -1.1, -8.1, -4.9, 0.7, -2.3, 9.3, 1.5, 7.1, -3.1

Perturbation Norm: 37.1292

Attack Successful: False

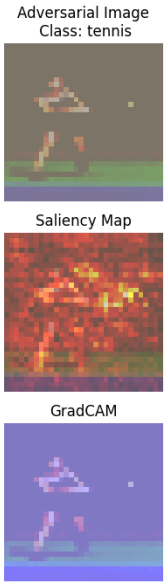


Predicted Logits: 1.0, 0.1, -4.4, 0.1, -5.8, 2.9, 3.2, 4.6, 0.4, -2.5

Mispredicted Logits: 3.0, -2.7, -1.0, -0.2, -1.3, -2.2, 0.5, 3.4, -0.9, -2.1

Perturbation Norm: 65.9851

Attack Successful: False



Transferability:

	MLP Adversarial	CNN Adversarial
FGSM (untargeted)	5/10	6/10
FGSM (targeted)	0/10	0/10
PGD (untargeted)	5/10	4/10
PGD (targeted)	0/10	0/10

Analysis:

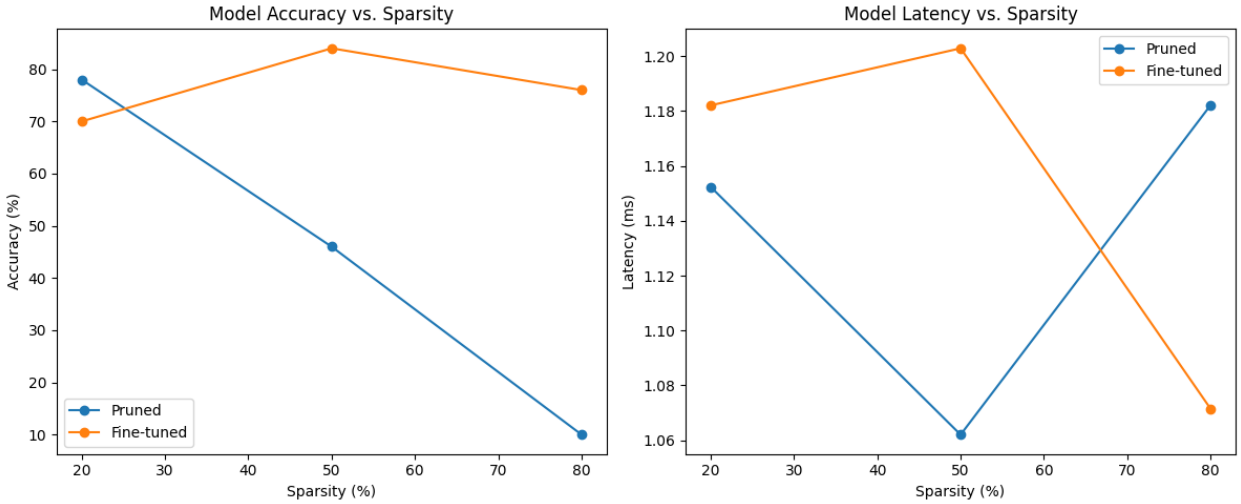
The Fast Gradient Sign Method uses an epsilon value of 0.03, while the Projected Gradient Descent uses the same epsilon value, an alpha of 0.01, and 40 iterations. Each type of attack was similarly effective on the MLP model as it was on the CNN model, although the MLP model mispredicted on most of the original images regardless. Projected Gradient Descent seemed to transfer slightly better, likely because its images had gone through more iterations to maximize the loss. Higher epsilon values may have caused increased mispredictions, but the image would be much more distorted from the original.

Targeted attacks towards basketball completely failed for both FGM and PGD. Although the models would mispredict, they wouldn't predict the targeted basketball class as they were much more confident on the other labels. These results may be because the low resolution makes basketball images appear visually similar to other contact sports, such as rugby or football, so the model cannot disambiguate them enough to create a working targeted image. It would've been easier to target tennis instead since the CNN model heavily correlates that class with the presence of green. Similarly, the CNN model seemed very resistant towards attacks on swimming images, likely because the attacked images remain mostly blue.

It is possible that, due to the low resolution and training set size, the CNN model learned to classify more based on color rather than the actual content of images than would be ideal. In images of golf, swimming, and hockey, the GradCAM shows a focus on vague areas of the background rather than the exclusively on the people participating in the sport. These sports all have a background that is usually uniform in color: swimming has blue water, golf has green grass, and hockey has white ice. As a result, the adversarial images usually just tint the image a different color rather than doing anything interesting with the content.

Problem C:

Sparsity	0%	20%	50%	80%
Actual Sparsity	0%	19.83%	49.57%	79.31%
Parameters	141,610	113,533	71,418	29,303
File Size	0.56 MB	0.56 MB	0.56 MB	0.56 MB
Accuracy (Pre-finetune)	82%	78%	46%	10%
Accuracy (Post-finetune)	82%	70%	84%	76%
Latency	1.28 ms	1.18 ms	1.20 ms	1.07 ms
Latency Std. Dev.	0.17 ms	0.42 ms	0.17 ms	0.12 ms



Adversarial Images:

Sparsity/Success Rate	20%	50%	80%
FGSM (untargeted)	9/10	6/10	6/10
FGSM (targeted)	1/10	0/10	0/10
PGD (untargeted)	7/10	6/10	5/10
PGD (targeted)	0/10	0/10	0/10

	Mean Perturbation Norm
FGSM (untargeted)	47.58
FGSM (targeted)	45.29
PGD (untargeted)	44,63
PGD (targeted)	42.54

Sparsity/Avg. Confidence	20%	50%	80%
FGSM (untargeted)	81.76%	67.59%	76.63%
FGSM (targeted)	3.52%	2.67%	1.28%
PGD (untargeted)	67.59%	56.94%	70.79%
PGD (targeted)	3.43%	1.21%	0.76%

Analysis:

Sparsity didn't have much of an effect on size or speed since the zeros were still part of the model's weights, although it did have a large effect on the pre-finetuned accuracy and a smaller effect on the post-finetuned accuracy. The convolutional layers were most sensitive to pruning because they accounted for ~90% of the model's parameters; each layer has hundreds of channels with their own kernels. After fine-tuning for 10 epochs, the accuracy was very degraded for 20% sparsity, improved for 50% sparsity, and slightly degraded for 80% sparsity. These results are likely due to luck, but it's still interesting that the 50% model had higher accuracy than the original. Latency was similarly luck based as there was no observable trend between the different sparsity levels. 50% sparsity seemed to work best for the CNN model as it decreased parameters the most without sacrificing accuracy.

Adversarial attacks had similar success rates on the pruned models as they did on the original model, although they were much more successful on the 20% model since it lost accuracy from fine-tuning. In other words, pruning would make attacks easier if the sparsity caused unrecoverable loss in accuracy. The pruned models were very confident in their mispredictions for untargeted attacks, but they were highly unconfident in predicting basketball for targeted attacks. Curiously, the confidence in basketball seemed to decrease with sparsity, although the confidence in untargeted attacks showed no visible trend.