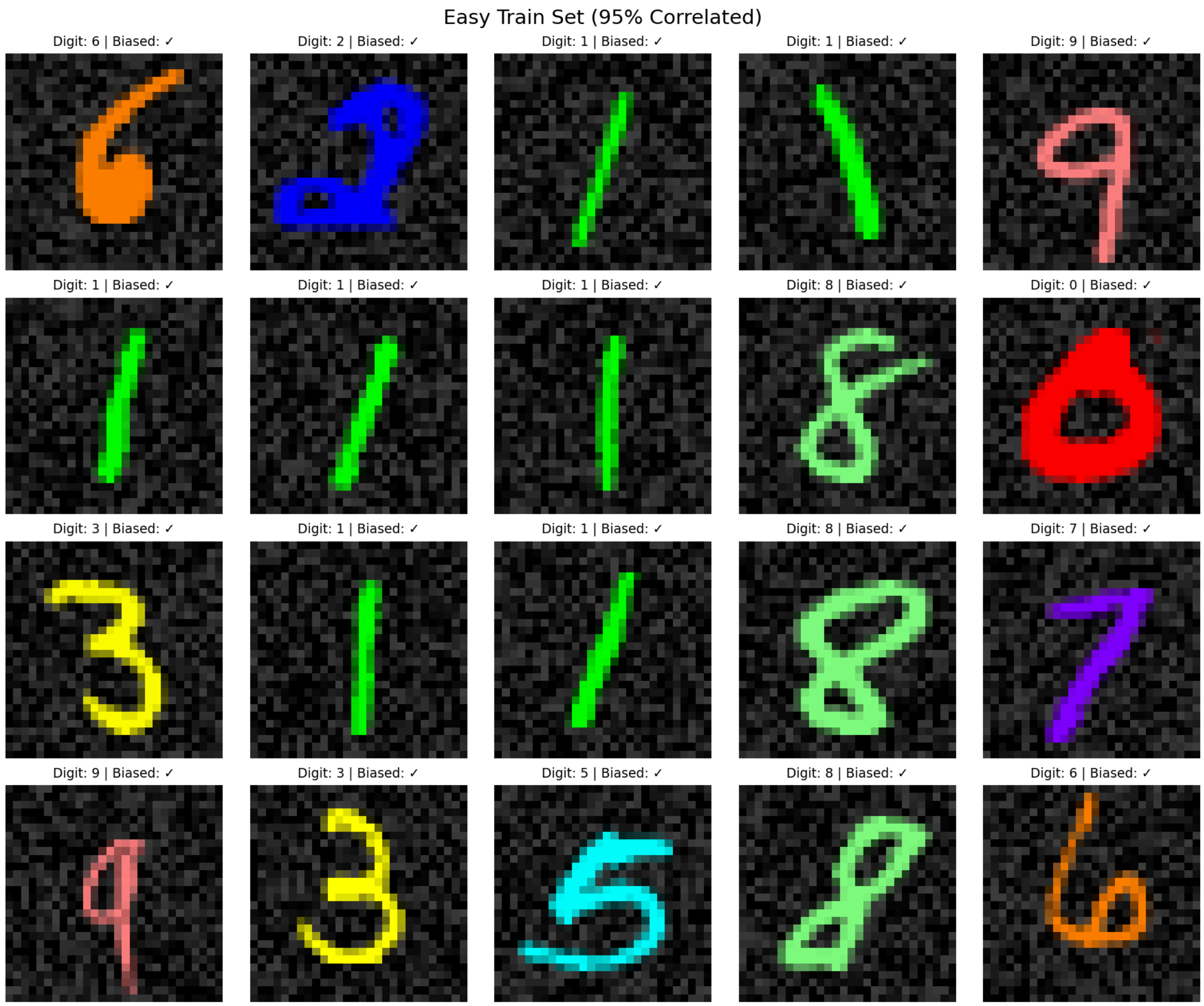
Task 0:


Easy Train Set (95% Correlated)
Digit: 6 | Biased: ✓
Digit: 2 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 9 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 8 | Biased: ✓
Digit: 0 | Biased: ✓
Digit: 3 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 1 | Biased: ✓
Digit: 8 | Biased: ✓
Digit: 7 | Biased: ✓
Digit: 9 | Biased: ✓
Digit: 3 | Biased: ✓
Digit: 5 | Biased: ✓
Digit: 8 | Biased: ✓
Digit: 6 | Biased: ✓

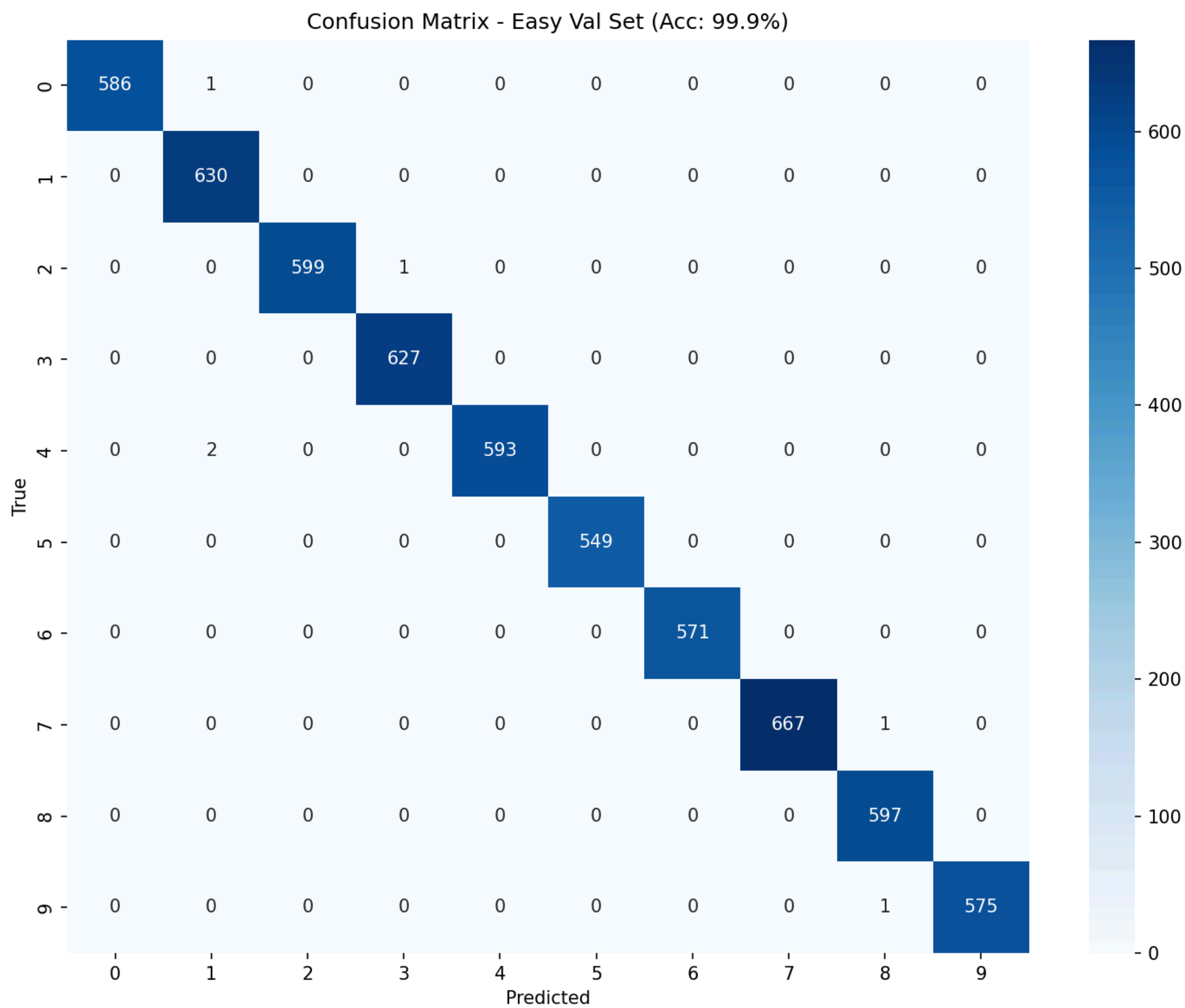

Hard Test Set (Inverted Colors)
Digit: 9 | Biased: ✗
Digit: 2 | Biased: ✗
Digit: 4 | Biased: ✗
Digit: 1 | Biased: ✗
Digit: 6 | Biased: ✗
Digit: 5 | Biased: ✗
Digit: 7 | Biased: ✗
Digit: 9 | Biased: ✗
Digit: 0 | Biased: ✗
Digit: 1 | Biased: ✗
Digit: 9 | Biased: ✗
Digit: 2 | Biased: ✗
Digit: 2 | Biased: ✗
Digit: 8 | Biased: ✗
Digit: 8 | Biased: ✗
Digit: 9 | Biased: ✗
Digit: 1 | Biased: ✗
Digit: 7 | Biased: ✗
Digit: 7 | Biased: ✗
Digit: 4 | Biased: ✗

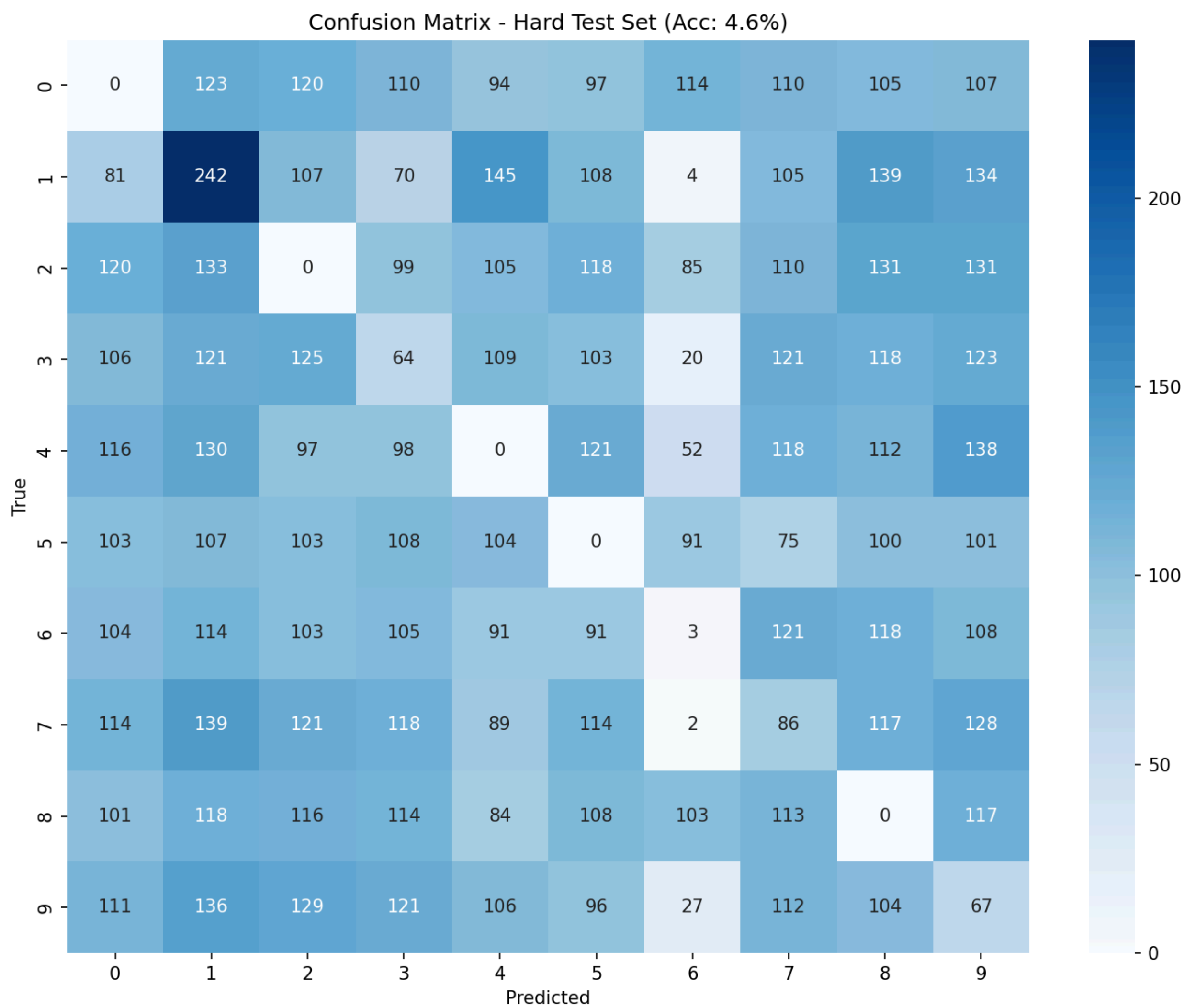
Task 1:

Overview of Convolution Nueral Networks:

A Convolutional Neural Network (CNN) learns hierarchical representations of images using repeated stages of:

- Convolution (feature extraction)
- Activation (Here we using ReLu(x) = x if x>0 since no need of x<=0 component of output of feature extraction and Sofmax : finding the probability of the output of convolution)
- Pooling (dimensionality reduction)





For the hard test set we can see that the values in the matrix spreaded out (non diagonals ones are misclassified

- **Diagonal** = Correct predictions
- **Off-diagonal** = Errors (misclassifications)

Results we got Easy Set Accuracy: 99.67% Hard Set Accuracy: 95.74%

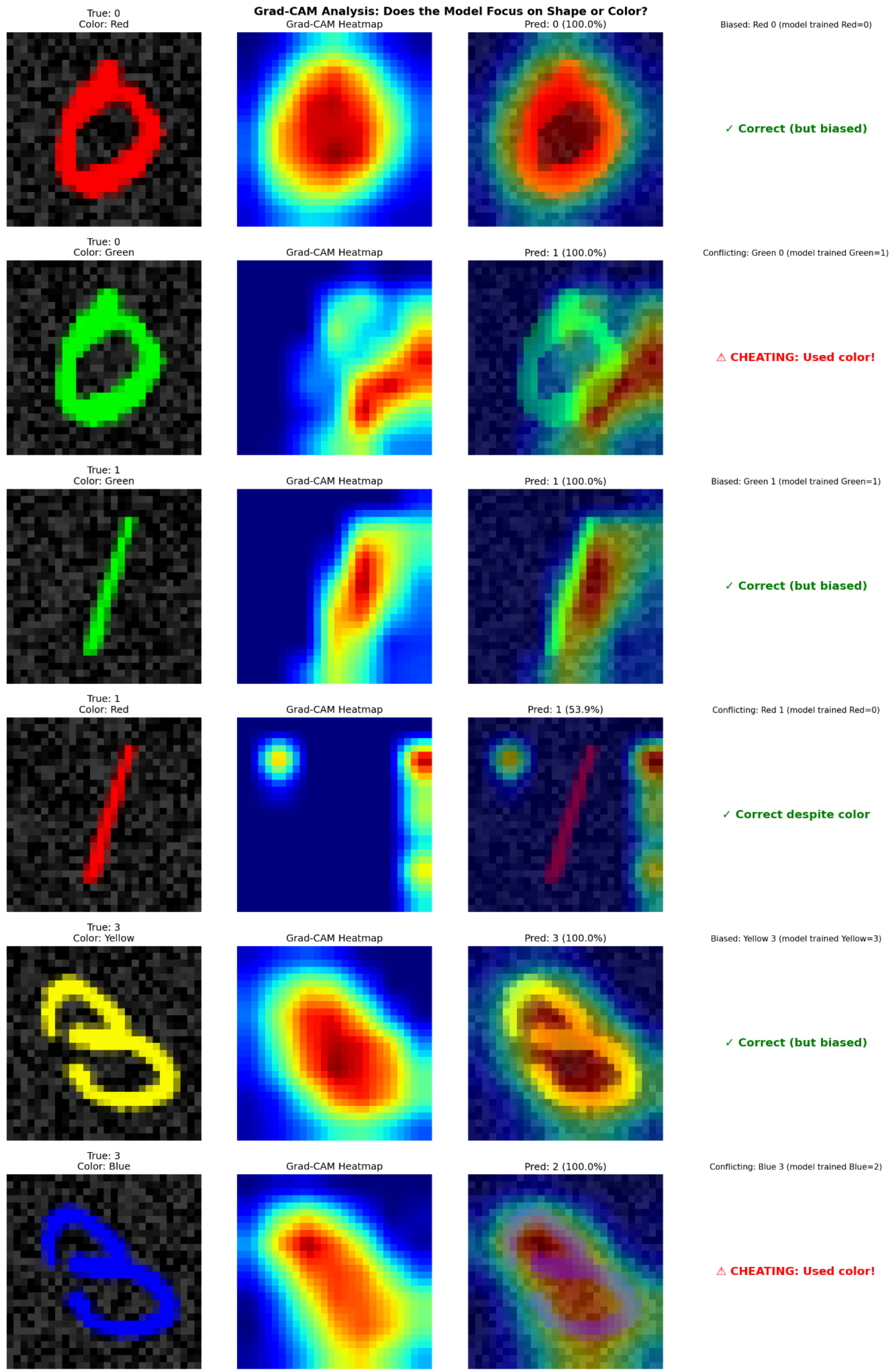
Task 3:

The Grad-CAM heatmaps show:

Biased images: Heatmap spreads across the colored foreground, not concentrated on digit edges

Conflicting images: Heatmap still focuses on colored regions, sometimes causing wrong predictions

⇒ The model doesn't "see" the digit shape - it sees the color


Grad-CAM Analysis: Does the Model Focus on Shape or Color?
True: 0
Color: Red
Grad-CAM Heatmap
Pred: 0 (100.0%)
Biased: Red 0 (model trained Red=0)
✓ Correct (but biased)
True: 0
Color: Green
Grad-CAM Heatmap
Pred: 1 (100.0%)
Conflicting: Green 0 (model trained Green=1)
⚠ CHEATING: Used color!
True: 1
Color: Green
Grad-CAM Heatmap
Pred: 1 (100.0%)
Biased: Green 1 (model trained Green=1)
✓ Correct (but biased)
True: 1
Color: Red
Grad-CAM Heatmap
Pred: 1 (53.9%)
Conflicting: Red 1 (model trained Red=0)
✓ Correct despite color
True: 3
Color: Yellow
Grad-CAM Heatmap
Pred: 3 (100.0%)
Biased: Yellow 3 (model trained Yellow=3)
✓ Correct (but biased)
True: 3
Color: Blue
Grad-CAM Heatmap
Pred: 2 (100.0%)
Conflicting: Blue 3 (model trained Blue=2)
⚠ CHEATING: Used color!

Task 4:

Method 1: Color Invariance Loss

If two images represent the same digit, their internal representations should be identical regardless of color. Instead of only supervising the output, we explicitly constrain the embedding space.

Mathematical Formulation

$$L_{\text{total}} = L_{\text{CE}} + \lambda \cdot \|f(x) - f(\tilde{x})\|^2$$

Where:

- L_{CE}: standard cross-entropy loss
- $f(x)$: embedding of the original image
- $f(\tilde{x})$: embedding of a recolored version of the same image
- $\lambda = 2.0$: invariance weight

Mechanism

1. Each training image is paired with a recolored variant.
2. Both are passed through the network.
3. The model is penalized if their embeddings differ

This explicitly removes color information from the latent space.

Hard Test Set Accuracy : Color Invariance: 98.1%

Method 2: Color Augmentation

Mechanism

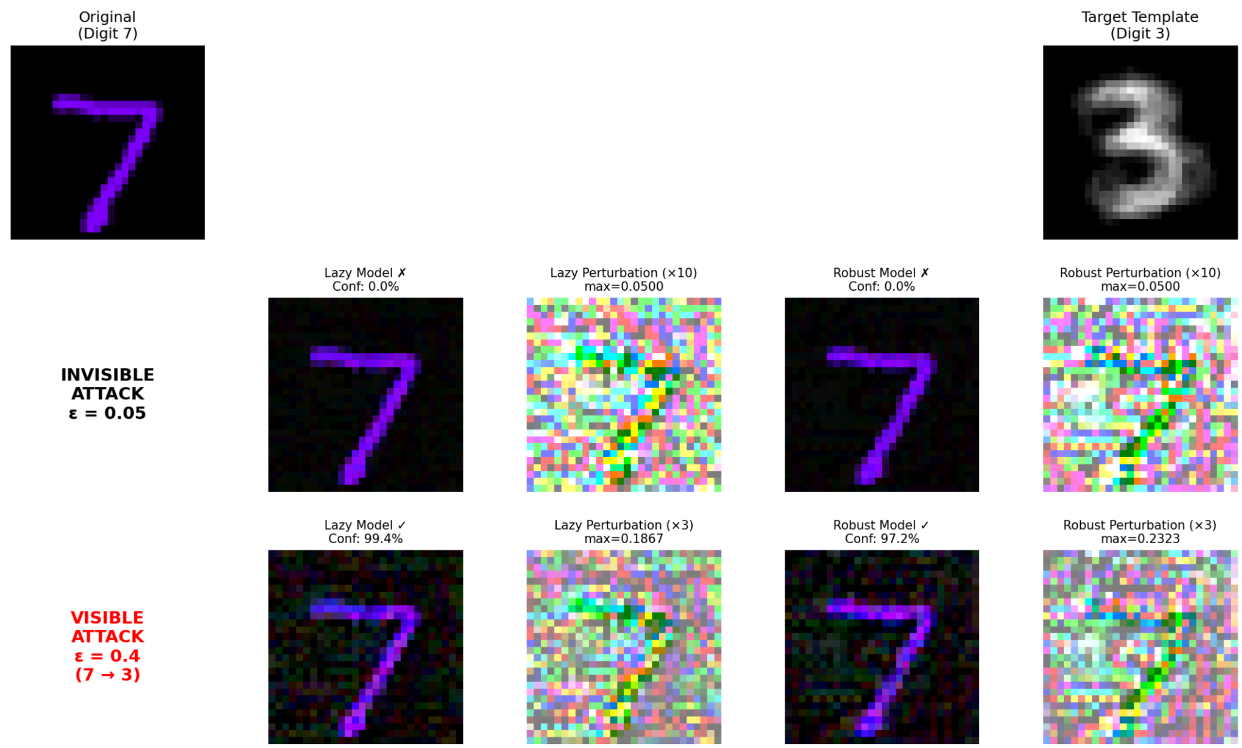
- During training:
  - 90% of images have their colors randomly reassigned
  - Labels remain unchanged
- Color becomes independent of digit identity

Code logic :

for each image: if random.random() < 0.9: recolor(image)

Hard Test Set Accuracy Color Augmentation: 98.8%

Task 5:



The robust model requires 24.4% more perturbation than the lazy model to be fooled.It also needs 22.2% more optimization steps, showing increased resistance to attacks.The lazy model fails with less noise and fewer steps, indicating weaker robustness.The robust model has slightly lower final confidence (97.2%), meaning it is less overconfident.Overall, the robust model is harder to break and better calibrated than the lazy model.

```
ADVERSARIAL ATTACK RESULTS
Task: Make digit 7 → 3 with >90% confidence

INVISIBLE ATTACK (ε = 0.05) - Perturbation should be invisible to humans
    Lazy Model:   Success=False  Conf=0.0%  Steps=300   Max pert=0.0500
    Robust Model: Success=False  Conf=0.0%  Steps=300   Max pert=0.0500

VISIBLE ATTACK (ε = 0.4) - Making 7 actually LOOK like 3
    Lazy Model:   Success=True   Conf=99.4%  Steps=18    Max pert=0.1867
    Robust Model: Success=True   Conf=97.2%  Steps=22    Max pert=0.2323
```

Task 6:

Interventions

Extracted activations from the FC1 layer (256 dimensions) — this is where high-level features live.now that features are separated, you can intervene.

For a given SAE feature:

- Case 1: multiply by 0
- Case 2: multiply by >1

Then:

Reconstruct the modified activation,now feed it back into the classifier.Observe the prediction

What we might Observed

- Turning down color features:
  - Reduces confidence
  - Sometimes flips prediction

- Turning up color features:
  - Increases confidence in color-associated digit
  - Even if the shape doesn't match perfectly