Defense-DDPM to Denoise DL Adversarial Attacks

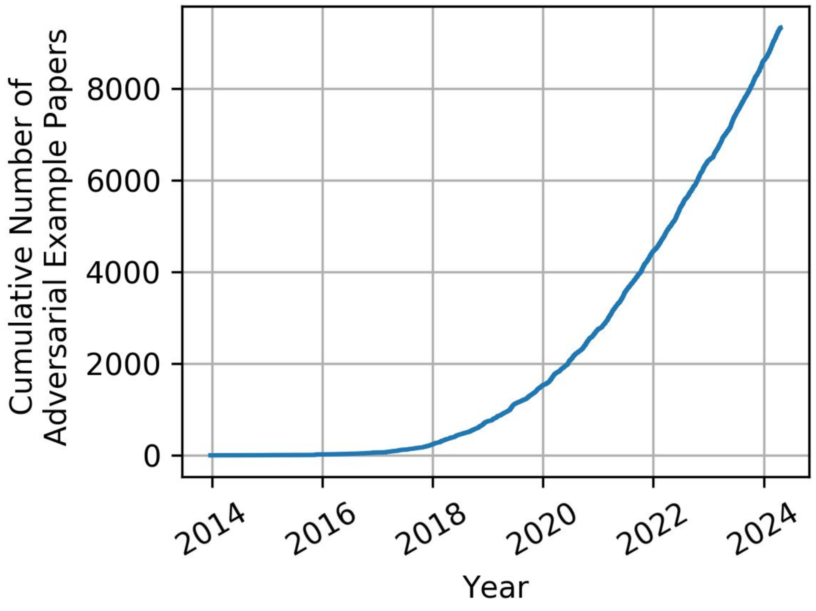
Daniel Hassler, Matthew Dim, Austin Burcham

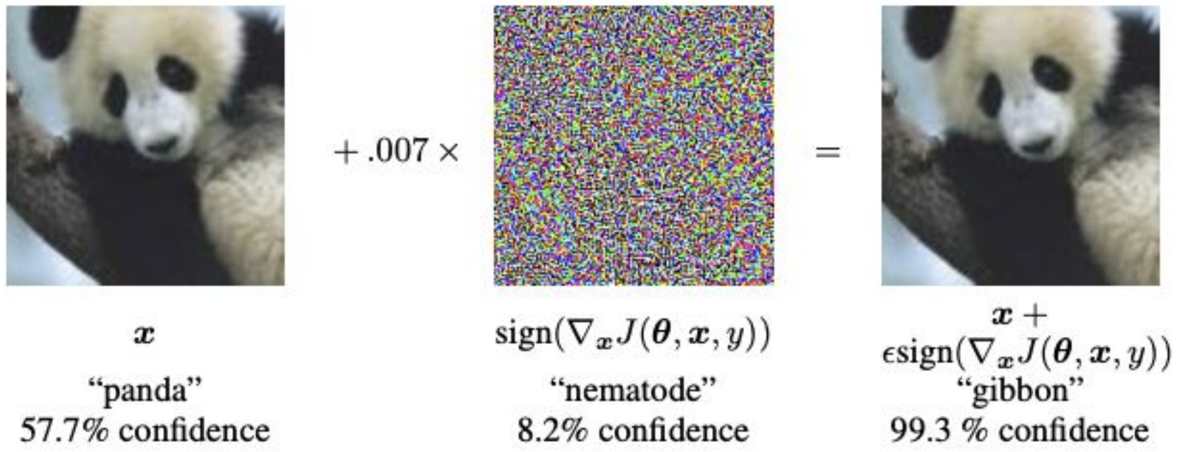
04/23/2024

Background

Problem

- FGSM and other deep learning attacks are **very** strong
- Defenses have strong limitations
- Fast, black-box denoising algorithms are preferred

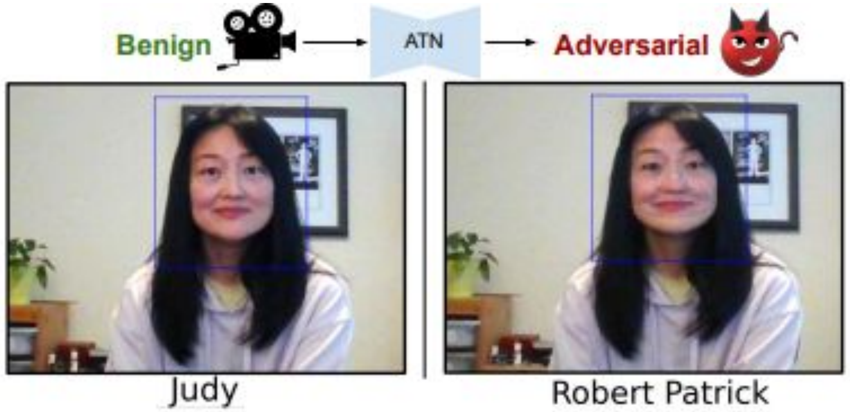




Fast Gradient Sign Method Attack

Huge Security Threats

- Autonomous vehicle object detection
  - EX: Stop sign classified as 30 MPH speed limit sign
- Facial recognition classifiers
  - EX: Gaining access to sensitive information via faceID attack
- Medical diagnosis
  - EX: Tampering with X-ray data classifying tumors as cancerous or non-cancerous



VIRGINIA TECH

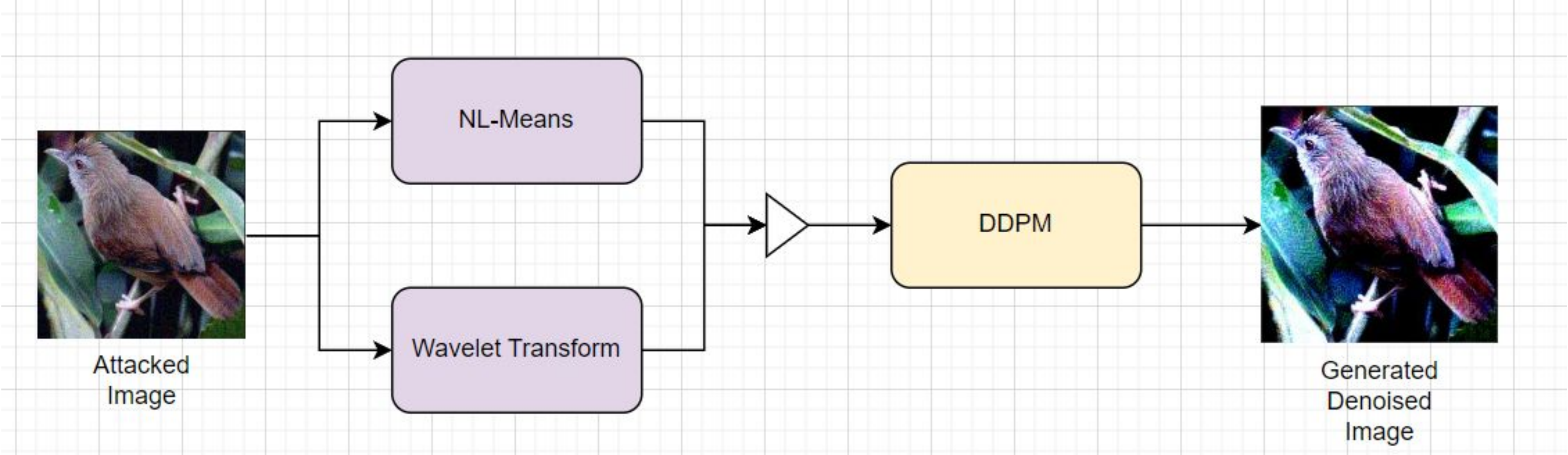
Related Works

- Defenses:
  - Image Denoising Auto-encoder
    - Venkataraman, Prashanth. "Image denoising using convolutional autoencoder." arXiv preprint arXiv:2207.11771 (2022).
  - Image Denoising Using Generative Model
    - "Denoising Adversarial Examples with PixelCNN" by Zhang et al.
  - Noise2Noise: Learning Image Restoration without Clean Data
    - “Noise2Noise: Learning Image Restoration without Clean Data” Lehtinen et al. (2018)

Introduction

Our Novel Approach

- Denoise: Combine power of input transformations and diffusion model to denoise images.
  - NL-Means or Wavelet Transform as preprocessing step
  - DDPM (Denoising Diffusion Probabilistic Model) in defense setting
- Evaluate: Classification and FID evaluation on produced images


NL-Means
Wavelet Transform
DDPM
Attacked Image
Generated Denoised Image

Methodology

NL-Means

NL-Means: Background

What it is

- NL Means: image denoising algorithm.
- Compares and averages similar **patches**

How it Works

- Compares and averages image patches
- Reduces various types of noise

Advantages

- Effective for Gaussian, impulse noise
- Preserves image details, textures

Original image | Noisy image | Denoised image

Figure 6.1: NL-means denoising on part of the Lena image. Left: noisy image. Right: image after NL-means denoising. Taken from [10].

The Algorithm

Fundamental Idea

- Compare **patches**, not individual pixels
- Similar patches likely contain meaningful information.
- Use **weighted avg. of pixels** for denoising

Patch Comparison

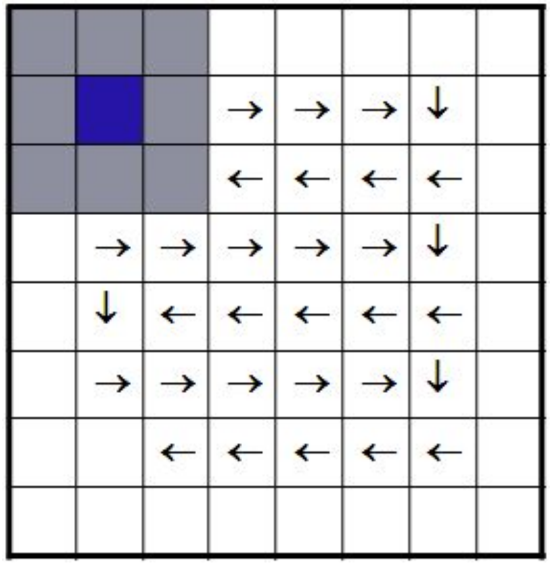
- Use **sliding window** approach.
- Compare pixel patch with every pixel's patch in window

Similarity Measure

- Use metric for similarity of patches: **mean squared difference**

Weight Calculation

- Calculate weights for each pixel in patch.
- Higher weights for more similar patches.



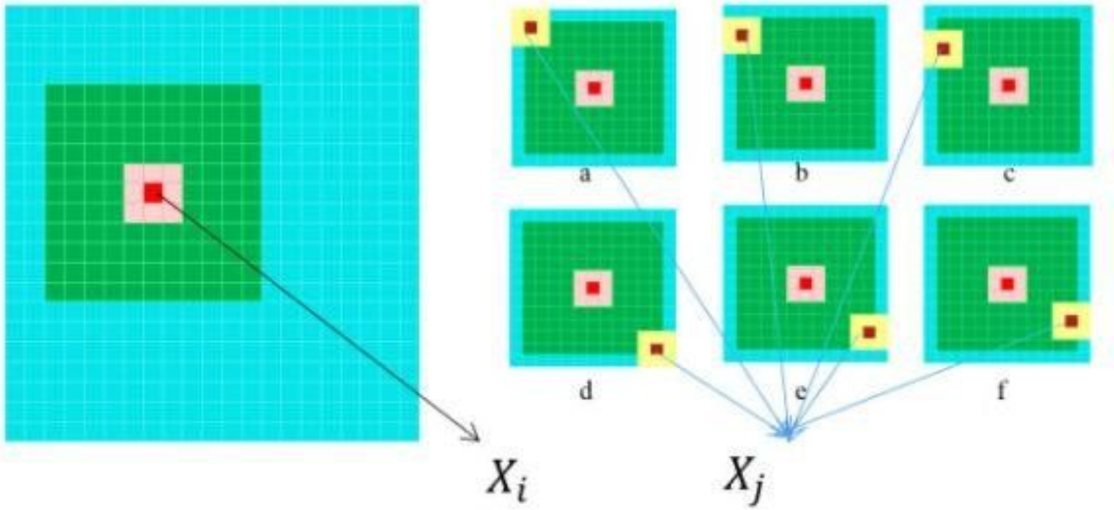
Sliding window approach

$$\frac{1}{N}\sum_{i=1}^{N}(p_1(i) - p_2(i))^2$$

Mean Squared difference between corresponding pixels in compared pixel patches.

P1 and p2 corresponding pixels between patches

Weight Calculation Specifics



Gaussian Weighting:

- Use **Gaussian function** to assign weights.
- Normalize weights
- Multiply each pixel value by its weight.
- **Sum** these products **to get the denoised pixel value**

Resulting Pixel Value:

- Pixel value becomes the **weighted average of patch values**.
- More similar patches contribute more to the average.

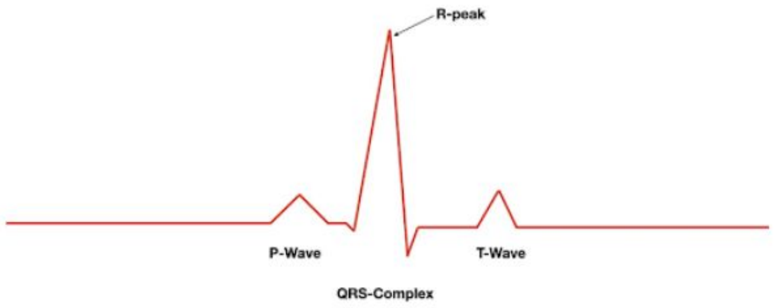
$$w_{ij} = e^{-\frac{MSD_{ij}}{h^2}}$$

Gaussian weight function.
MSD: Mean Square Difference between patches.
H smoothing parameter

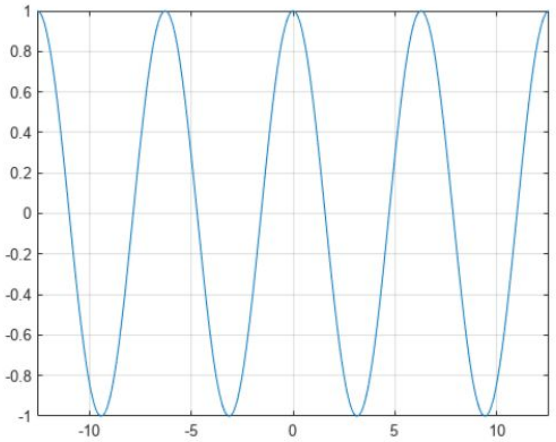
Wavelet Transform

Wavelet Transform Overview

- A wavelet is a function that oscillates like a wave but quickly attenuates (short lived)
  - Extrapolates *scale of a wave, localized in time*
  - Better with dealing with noisy signals
    - For example: ECG data which is typically noisy
- **Goal:** Break a function into key components
  - Similar to a fourier transform, but localized in time (not explicitly continuous)



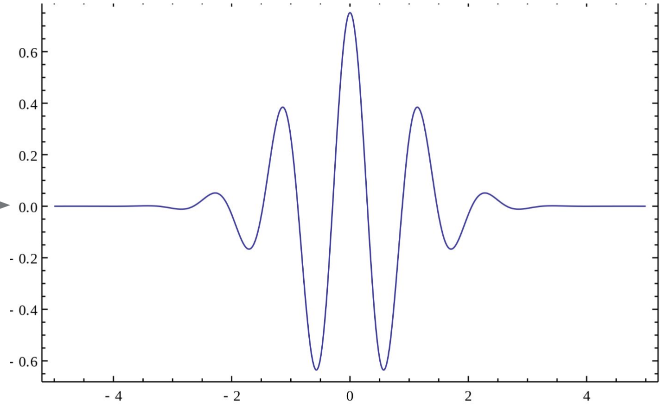
R-peak extraction of ECG data which correlates to heart rate and variability (HRV)



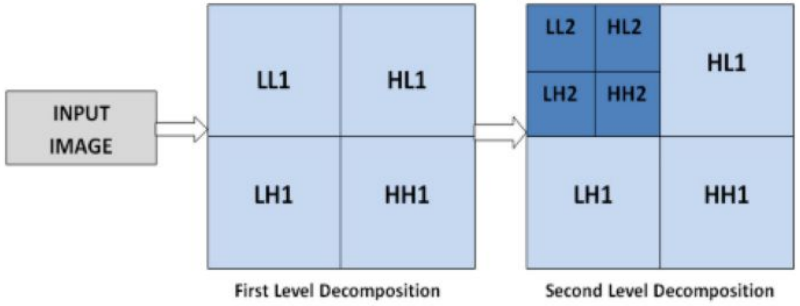
$$\Psi(t) = e^{-\left|\frac{t}{k}\right|} cos(2\pi t)$$

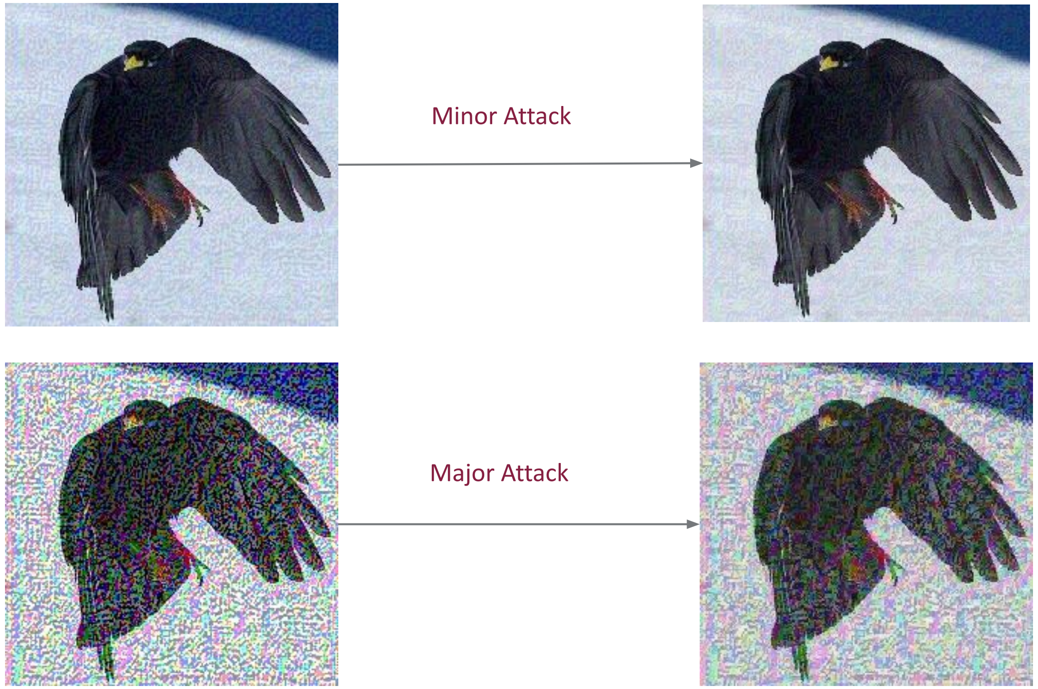
$$\int_{-\infty}^{\infty} |\psi(t)|^2 \, dt < \infty$$

Has Finite Energy! (area under the curve is a finite number)



Wavelet Implementation





- Separable 2D DWT
  - Decompose image into two levels of wavelet composition
  - Three wavelet equations where m and n are coordinates of the image
  - Denoise wavelet coefficients using thresholding
  - Apply inverse wavelet transform on modified coefficients to obtained denoised images

$$\psi^1(m,n) = \phi(m)\psi(n) \quad LHwavelet,$$

$$\psi^2(m,n) = \psi(m)\phi(n) \quad HLwavelet,$$

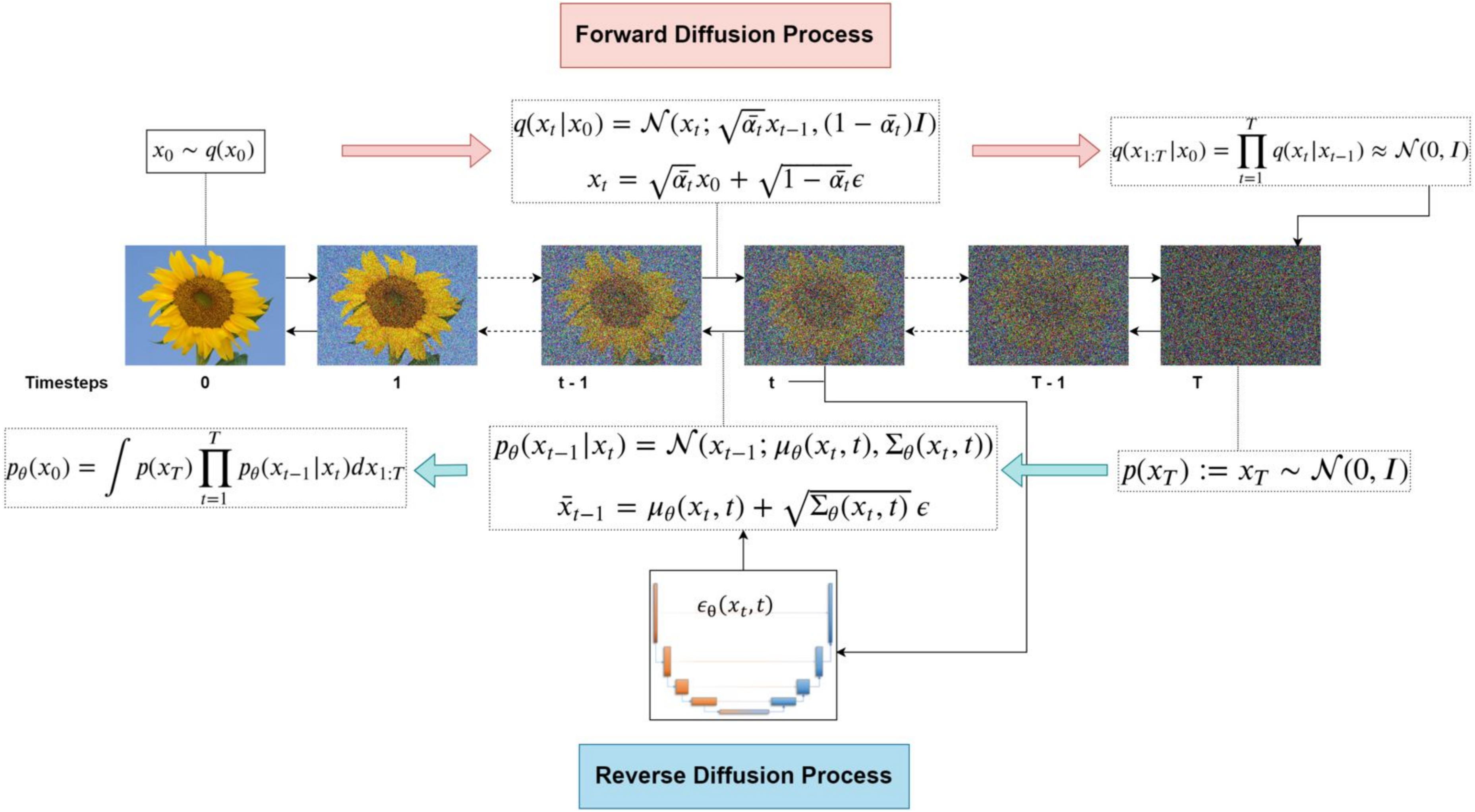
$$\psi^3(m,n) = \psi(m)\psi(n) \quad HHwavelet,$$

$$\downarrow$$

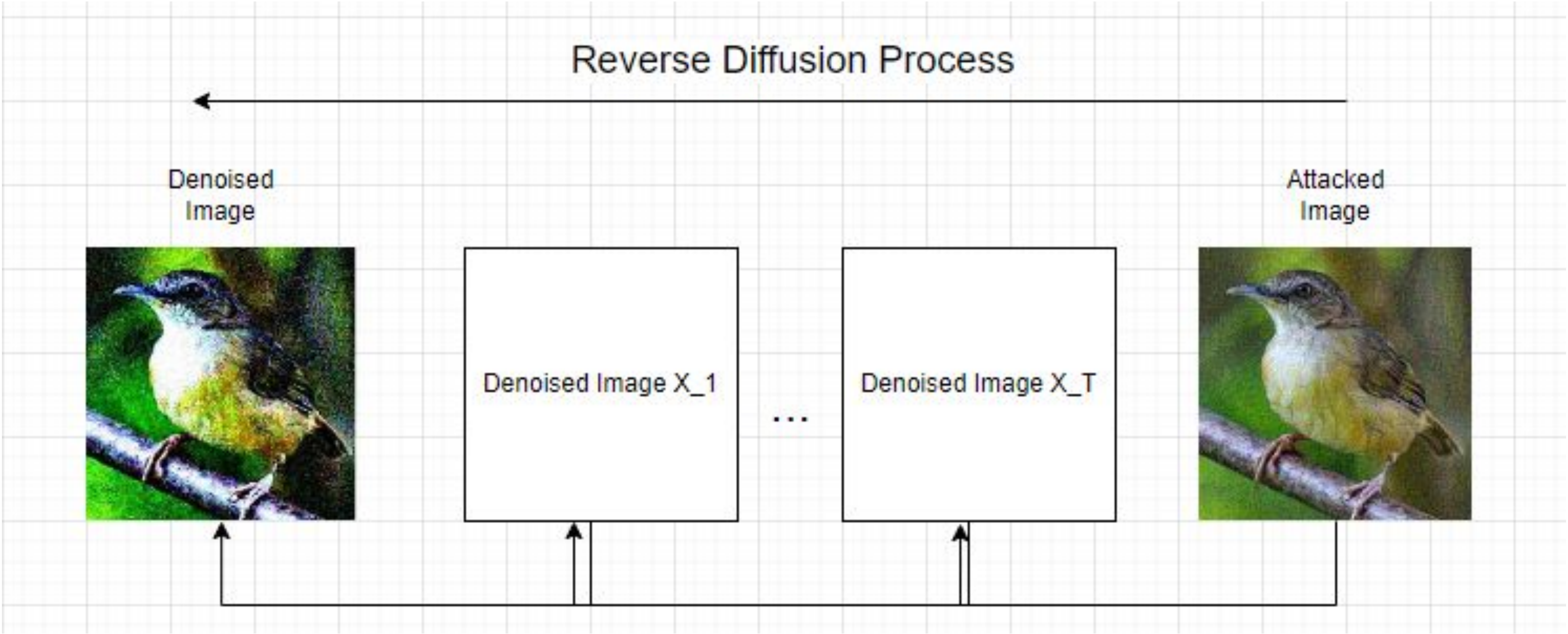
$$\phi^2(m,n) = \phi(m)\phi(n)$$

Denoising Diffusion Probabilistic Model

DDPM Training Background



Our DDPM Approach Methodology

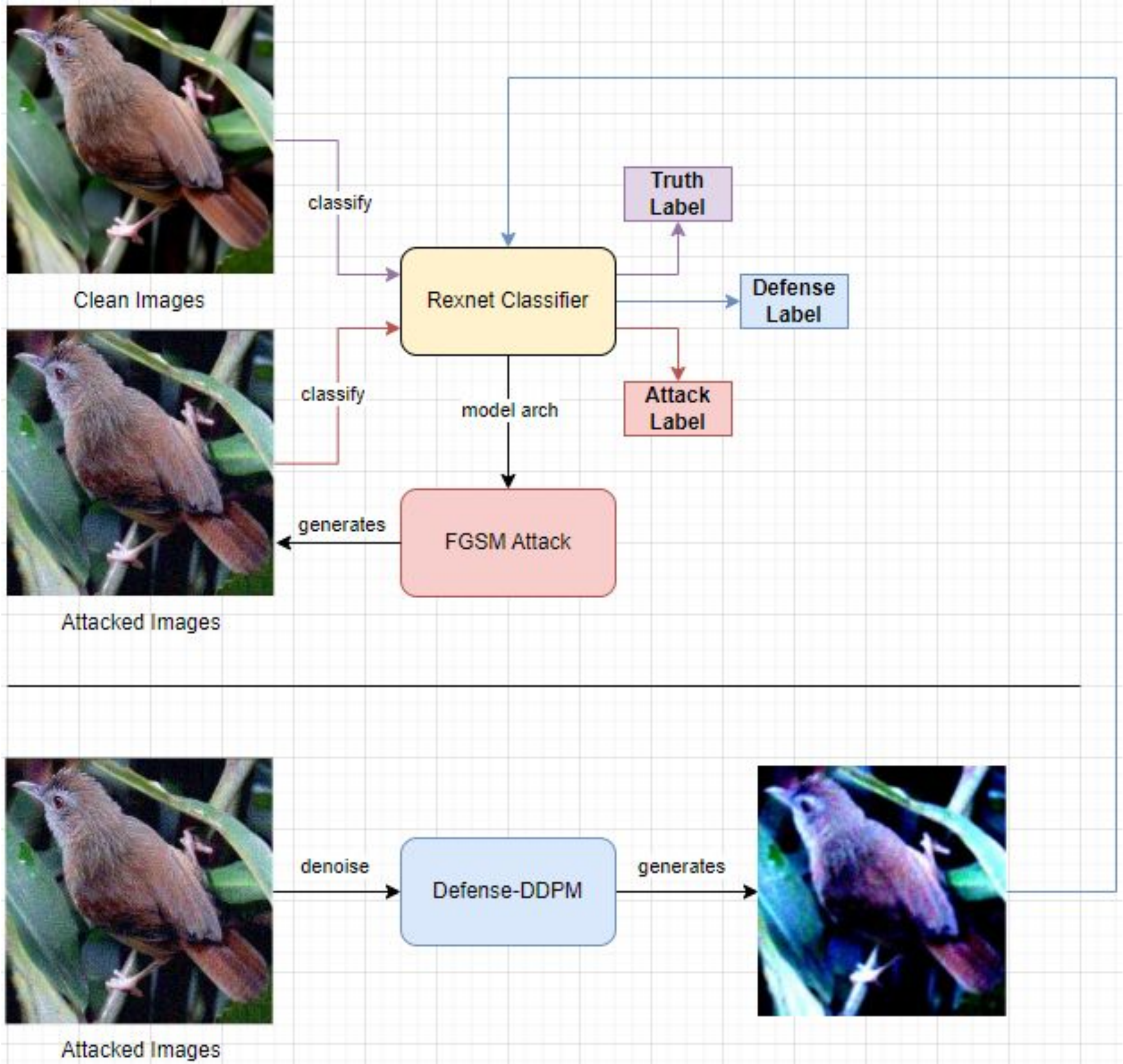


Evaluation

Dataset

- **Birds dataset**:
  - ~90,000 labeled bird images
  - 525 species (labels)
  - 84635 train, **2625 test**, 2625 validation images
  - Dimension: 224x224x3 (RGB)

Evaluation Procedure



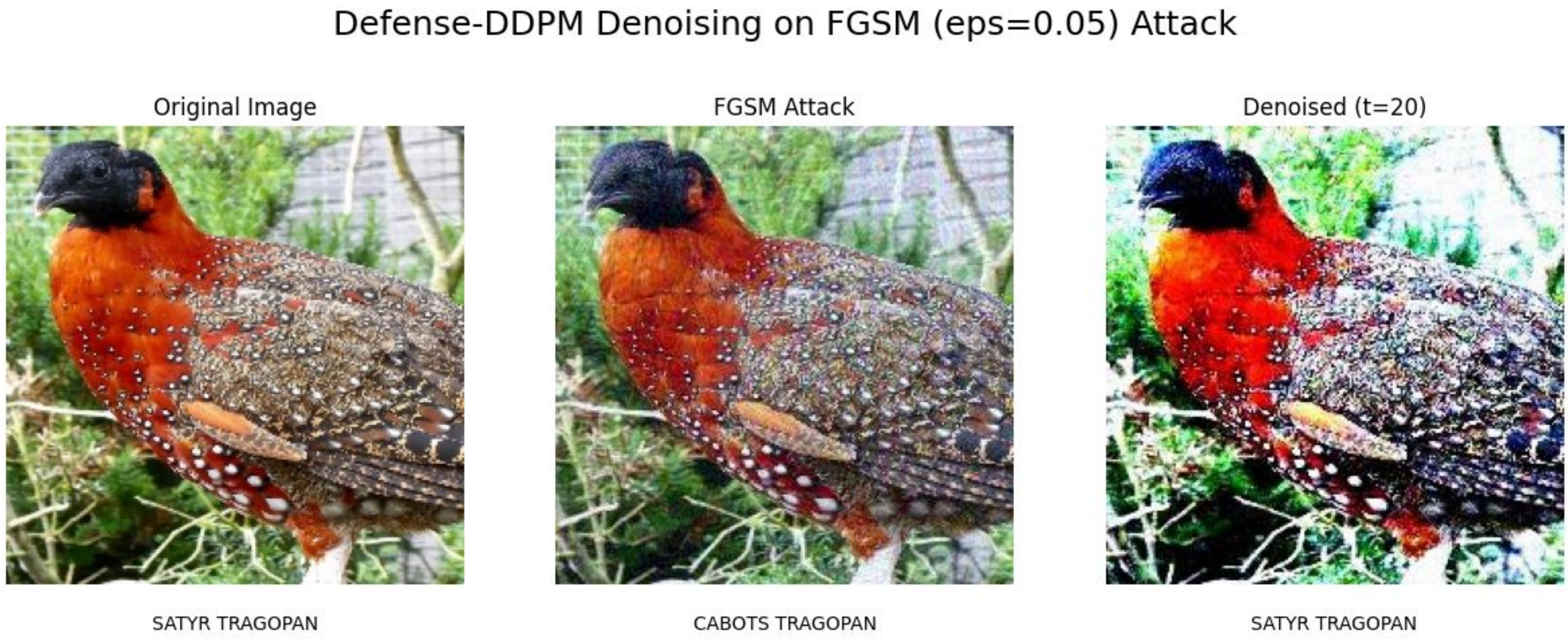
Denoising Results

- **Successful denoising** is the percentage of classifications that were reverted back to the true label after defending successful attacks.

- Using **NL-Means** as a preprocessing denoising transformation greatly improved our Defense-DDPM results

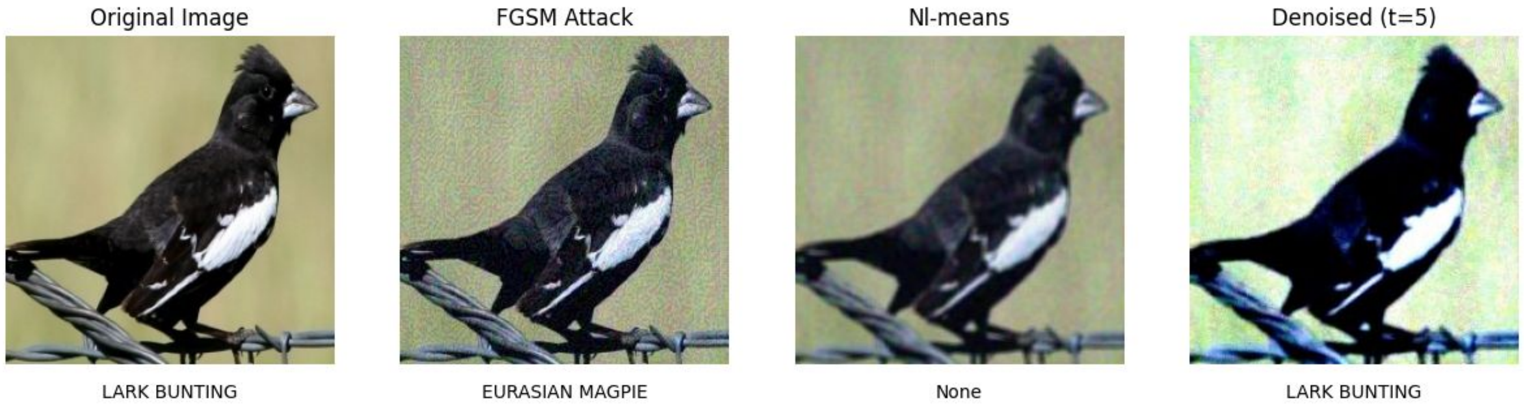
	Successful Denoising of FGSM eps=0.05 Attack w/ Different Input Transformations		
	None	Wavelet Transform	NL-Means
DDPM t=1	23.66	27.94	**45.65**
DDPM t=2	23.92	28.91	**46.26**
DDPM t=5	28.30	34.66	**51.20**
DDPM t=10	35.22	44.02	**48.45**
DDPM t=20	42.19	**46.97**	44.33
DDPM t=30	41.53	**48.14**	40.76
DDPM t=40	41.48	**44.07**	36.49
DDPM t=100	35.06	**35.67**	24.94

Example of Attack Denoising on DDPM

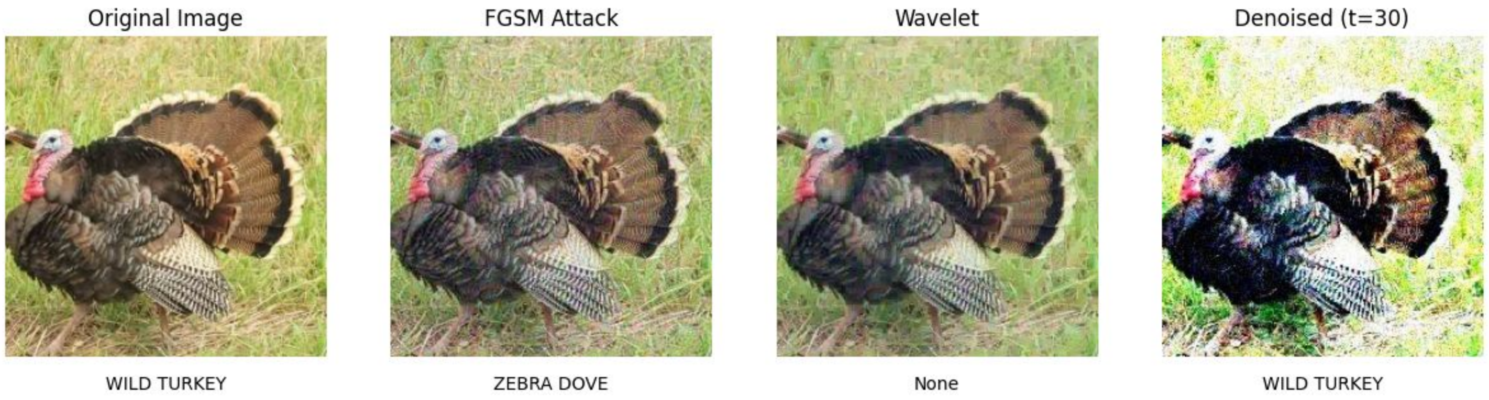


Examples of our Transformation Effects on DDPM

Defense-DDPM Denoising w/ NL-Means on FGSM (eps=0.05) Attack



Defense-DDPM Denoising w/ Wavelet Transform on FGSM (eps=0.05) Attack



Frechet Inception Distance (FID) Results

- Lower FID = better generated images
- DDPM with NL-Means produced best quality generated denoised images.
- DDPM T=5 produced best results with NL-Means, scoring **8.73 FID**

	Denoising FID w/ Different Input Transformations (FGSM eps=0.05 Attack)		
	None	Wavelet Transform	NL-Means
DDPM t=1	24.95	12.29	**9.06**
DDPM t=2	24.92	12.44	**8.89**
DDPM t=5	24.52	12.66	**8.73**
DDPM t=10	27.31	16.35	**11.21**
DDPM t=20	36.91	24.45	**19.74**
DDPM t=30	44.39	31.67	**28.00**
DDPM t=40	50.83	37.96	**35.39**
DDPM t=100	72.54	**60.26**	62.21

Discussion

Limitations

- Training and inference time for DDPM.
  - Training > 12 hours for 200 epochs on very small dataset
  - Inference time is dependent on “t” parameter (up to 15 hours for 100 denoising steps)
  - Resource intensive ~ 16 GB of VRAM
- NL-Means inference time
  - Hyperparameter tuning is expensive.
- Slow defense due to above limitations.

Future Work

- Denoise various other attacks (CW, JSMA, PGD)
- Attempt different DDPM setups, maybe including forward noise.
- Post-processing output of DDPM model.
- SVD to Learn Wavelet Transform Coefficients.
- Try out different patch sizes for NL-Means and further hyperparameter tuning.

Questions