INTEL UNNATI INDUSTRIAL TRAINING 2024

TECHMATES

PROBLEM STATEMENT

Detect Pixelated Image and Correct It

Category: Artificial Intelligence, Machine Learning, Deep Learning.

Participants: 1 st - 4 th Semester Students

Prerequisites:
->Concepts in Machine Learning
->Programming Skills (Python)
->Deep Learning - Train/Validate/Test with Data

- **Description:**
- The problem has two parts – one is image classification and the other is image generation. 1. Given an Image, check if it is blocky or pixelated. Pixilation means blurry, blocky squares you see when an image is zoomed too much. Design an algorithm that can detect whether the image is pixelated or not. The challenge is to design an algorithm or ML/AI model, which is extremely lightweight or computationally efficient such that we can run this algorithm at 60 Hz or 60 frames per sec (FPS) and must be minimum 90% accurate. The pixelated images are rare and hence the algorithm should not create too many false positives. Algorithm / model quality will be measured by F1-score, precision recall curve etc. Also, the algorithm should be able to work on 1080p resolution input with same performance. It is ok to downscale the large image to any desired input size to work on large images. a. Input Image Size: 1920x1080 (it's ok to downscale and feed into the algorithm/model). b. Inference Speed Target: Min 30Hz, better to have 60 Hz. c. Accuracy Metric: F1 Score/Precision Recall. d. Should work in rare class scenario: if only one in 1000 images are pixelated then the algorithm must predict at most 10% False Positives. 2. Given a pixelated image, design an algorithm to improve the quality of the image i.e., restore the lost information. This problem is also known as jpeg restoration. Here also the challenge is to design a highly efficient algorithm that can run at least at 20 FPS. The quality of restoration can be examined by comparing with ground truth image using any metric like LPIPS, PSNR etc. If a nonpixelated image is given then the algorithm should not enhance it and leave it intact. Algorithm should work on 1080p resolution images.

Unique Idea Brief (Solution)

Proposed Solution:

- Detection Module:
- Develop a Convolutional Neural Network (CNN) model that takes an image as input and classifies it as either pixelated or non-pixelated.
- Ensure the model is lightweight and optimized to run at 30 frames per second (FPS) to meet real-time processing requirements.
- Focus on achieving high accuracy (over 90%) while minimizing false positives, especially in scenarios where pixelated images are rare.

Unique Idea Brief (Solution)

Correction Module:

- Utilize the Real-ESRGAN (Enhanced Super-Resolution Generative Adversarial Networks) model to enhance and restore the quality of pixelated images.
- Real-ESRGAN is chosen for its ability to enhance details while removing artifacts, making it suitable for real-world image restoration.
- Ensure the correction algorithm is efficient enough to run at 20 FPS, allowing for real-time image enhancement.

Unique Idea Brief (Solution)

Key Goals:

- High Accuracy: Aim for precision and recall metrics that result in a high F1-score, ensuring reliable detection of pixelated images.
- Real-Time Performance: Maintain a detection speed of 60 FPS and a correction speed of 20 FPS, ensuring the system can handle high-resolution images (1080p) in real-time.
- Minimal False Positives: Design the detection algorithm to predict at most 10% false positives even when pixelated images are rare (e.g., 1 in 1000).

Unique Idea Brief (Solution)

Technical Approach:

- Preprocessing: Downscale large images to a manageable size (e.g., 256x256) before feeding them into the detection model. This reduces computational load and speeds up processing without significantly affecting accuracy.
- Model Training: Train the detection model using a dataset of labeled pixelated and non-pixelated images. Use data augmentation techniques to increase the diversity of training samples and improve model robustness.
- Model Evaluation: Evaluate the detection model using precision, recall, and F1-score metrics to ensure it meets the desired performance criteria.
- Image Enhancement: Apply the Real-ESRGAN model to pixelated images to generate enhanced versions. Post-process the enhanced images using filters to further improve quality if necessary.

Features Offered

Detection Module:

- Classifies images as pixelated or non-pixelated.
- Achieves high inference speed (30 FPS).
- Ensures over 90% accuracy with a focus on minimizing false positives.

Correction Module:

- Enhances and restores pixelated images.
- Utilizes Real-ESRGAN for superior image restoration quality.
- Maintains high processing speed (20 FPS) for real-time applications.

Process Flow

Image Input:

- User selects or uploads an image.
- Image is preprocessed and resized if necessary.

Detection Stage:

- Image is passed through the detection model.
- Model predicts if the image is pixelated.

Correction Stage (if pixelated):

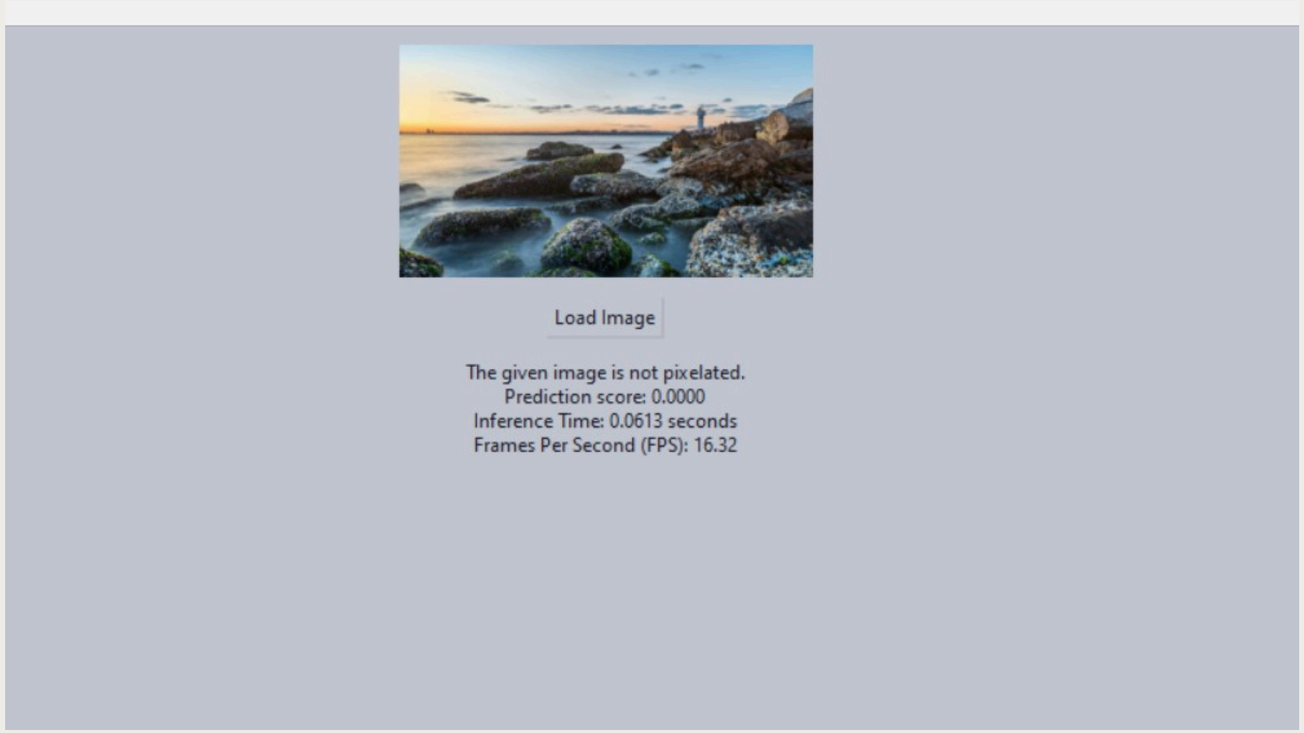
- Pixelated image is processed by the correction model.
- Enhanced image is generated and displayed.

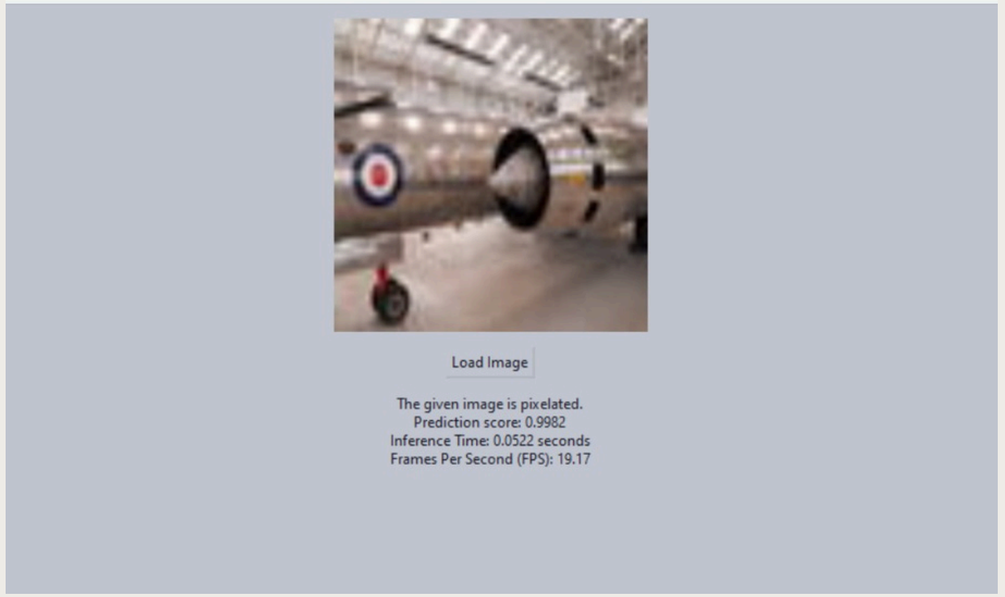
Output:

- Results are presented to the user with inference time and accuracy metrics.

Images from the project

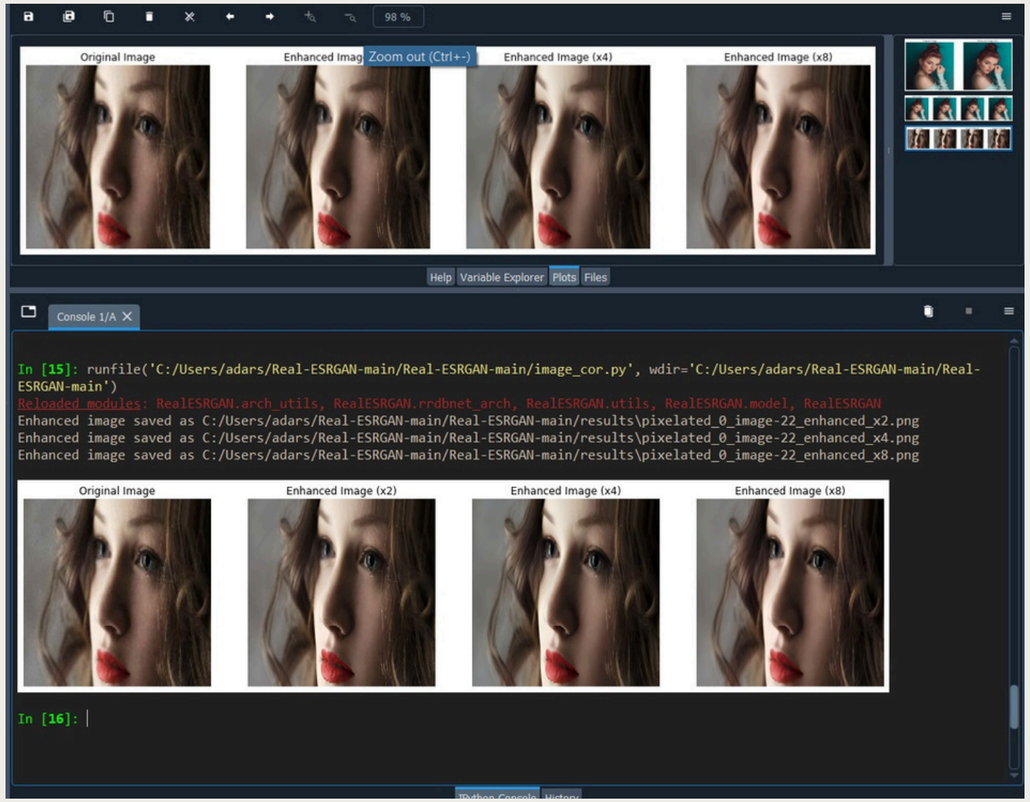
Input Image and the detection model :





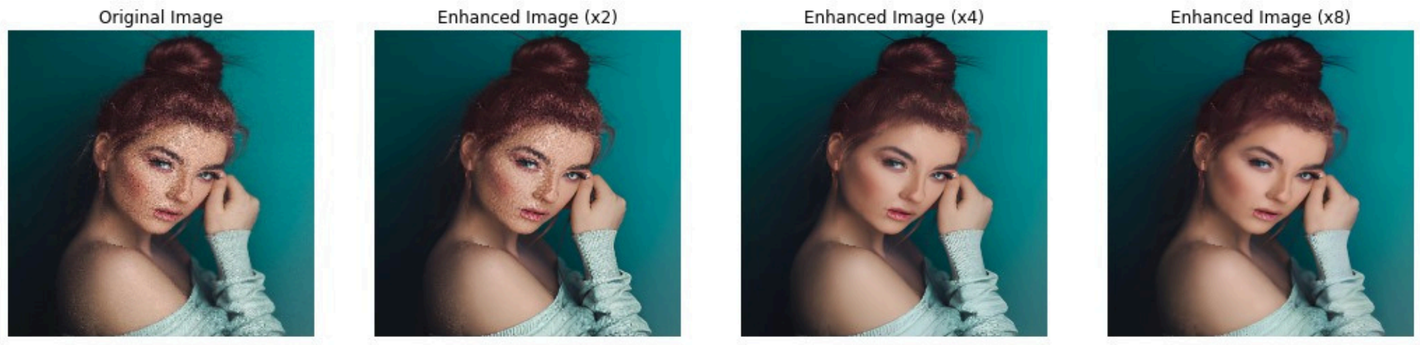
Images from the project

Correction model images



Images from the project

Correction model images



Components:

- Preprocessing:
  - Resizing, normalization.
- Detection Model:
  - Convolutional Neural Network (CNN).
- Correction Model:
  - Real-ESRGAN for super-resolution.
- Output:
  - Display original or enhanced image.

TECHNOLOGIES USED

Deep Learning Frameworks:

- TensorFlow, Keras for model development and training.
- PyTorch for Real-ESRGAN implementation.

Image Processing:

- OpenCV for image preprocessing and resizing.
- PIL for image handling in the GUI.

GUI Development:

- Tkinter for user interface design.
- GPU for model training and inference acceleration

Hardware:

- GPU for model training and inference acceleration.

Team Members and Contribution

[Adarsh Raj] :

- Project Lead and Deep Learning Engineer.
- Responsible for model development, training, and evaluation.

[**Rohit Janardhan]:**

- Software Developer.
- Worker
- Handled data preprocessing and augmentation.

[EM Kritik , Alfred PN , Dithin Mandanna]:

- Research Analyst.
- Conducted literature review and performance benchmarking.

Conclusion

Summary:

- Successfully developed and integrated a system for detecting and correcting pixelated images.
- Achieved high accuracy and real-time performance.

Future Work:

- Explore advanced models for further enhancement.
- Optimize algorithms for even higher efficiency.

Impact:

- Provides a robust solution for improving image quality in various applications, including media, security, and autonomous driving.

Thank You