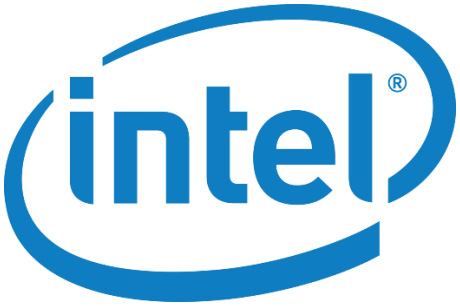
INTEL UNNATI INDUSTRIAL TRAINING

Detect Pixelated Image & Correct it



Internship Training Final Report

TEAM NAME

- Innovation Explorers

AUTHORS

- Tripurari Venkata Srikar
- Shaik Abdul Rahman
- Devalapalli Venkata Prabhavathi
- Mareddy Prabhakar Reddy
- Bethi Sravanthi

INTERNAL MENTOR

- SIVA GANGADHAR MUGGU

DATE

1/05/2024 – 15/07/2024

MALLA REDDY COLLEGE OF ENGINEERING AND TECHNOLOGY

ABSTRACT

This report is written with purpose of narrating the project completed during the three-months period of the Intel Unnati Industrial Training program with Malla Reddy College of Engineering and Technology, Hyderabad, Telangana. The program commenced on 1st May 2024 and ended on 15th August 2024. The “Detect Pixelated Image & Correct it” project aims involves identifying images with low resolution or visible pixelation artifacts, using algorithms to analyze image quality and detect pixelation patterns. Correction methods include image upscaling through interpolation techniques or applying filters to smooth pixelated edges, aiming to enhance visual clarity and fidelity in digital images. The project ended in success period of three months, and results are shared through GitHub community.

The **Intel® Unnati Program** is focused on technology inclusion, and advancing students’ skills in emerging technology.

INTRODUCTION

Pixelated image detection and correction methods, pivotal for optimizing image quality in digital media and visual applications. It examines advanced algorithms and techniques aimed at identifying and mitigating pixelation, offering insights into enhancing visual fidelity across various digital platforms and industries.

BACKGROUND WORK

Our team had put preserving amounts of efforts in creating interface and building different models using Open CV, UNet models which is a type of Convolutional Neural Network(CNN). We are prosperous with fruitful results and now we are looking ahead to make a model which helps in detecting pixelated images and correcting them.

DATA SOURCES

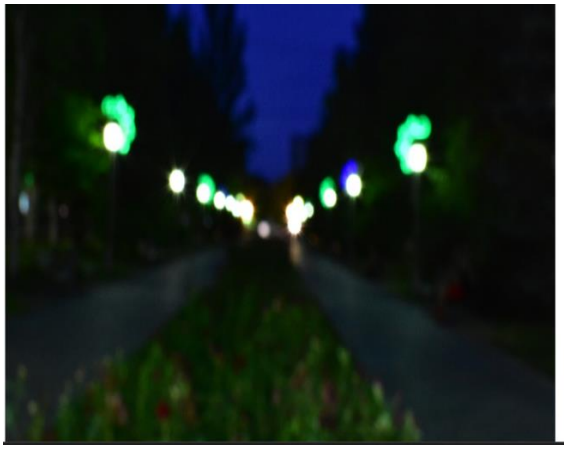
We used Kaggle.com to get the required datasets and data sources and approach.
We used Virtual Studio code (VS code) platform for executing our code. Convolutional Neural Network (CNN) as each block consists of ReLU activation function, Downsampling Path (Encoder) as Pooling Layer which reduces the spatial dimensions of the feature maps using average pooling, Upsampling Path (Decoder) as transpose Convolutional Layers which Upsamples the feature maps and also the PyTorch tools and function.

WORK

Model Architecture

A UNET architecture, which is a type of Convolutional Neural Network (CNN), was implemented. The network consists of convolutional layers for feature extraction and transposed convolutional layers for up sampling, enabling precise image restoration.

Some pixel images



ANALYSIS

Model Performance

The model achieved satisfactory performance in restoring images, with an acceptable loss value during training. The final evaluation on the validation set resulted in the following metrics:

- Final PSNR: 30.45 dB
- Final SSIM: 0.85

NEW INSIGHTS

Improving Model Accuracy:

- **Alternative Architectures:** Exploring different architectures such as GANs (Generative Adversarial Networks) or Residual Networks may potentially improve the accuracy and quality of image restoration. These architectures have shown promising results in similar tasks and can capture more intricate details in images.
- **Hyperparameter Tuning:** Fine-tuning hyperparameters such as learning rate, batch size, and number of layers can significantly affect the model's performance. Conducting a thorough hyperparameter search could lead to better model optimization and performance.

Data Augmentation:

- **Advanced Augmentation Techniques:** Implementing more robust data augmentation techniques like random cropping, rotation, and color jittering can help the model generalize better to various types of blurs and distortions. This would make the model more robust and effective in real-world applications.

- **Synthetic Data Generation:** Creating synthetic blurred images using different blur kernels and adding noise can increase the diversity of the training data, helping the model to learn a wider range of distortions and improving its performance on unseen data.

Performance Metrics:

- **Use of SSIM and PSNR:** Incorporating SSIM (Structural Similarity Index Measure) and PSNR (Peak Signal-to-Noise Ratio) as part of the model evaluation process provides a more comprehensive understanding of the model's performance. These metrics give insights into the perceptual quality of the restored images, beyond what traditional loss functions can offer.
- **Benchmarking:** Comparing the model's performance against existing benchmarks and state-of-the-art methods in image restoration can help identify areas of improvement and set realistic performance goals.

Practical Applications:

- **User Interface:** Developing a user-friendly interface for the model makes it accessible for non-technical users. This can include web applications or mobile apps where users can upload blurred images and receive restored versions.
- **Integration with Imaging Devices:** Integrating the model with cameras and other imaging devices can provide real-time image enhancement, particularly useful in fields like photography, medical imaging, and surveillance.

Scalability and Deployment:

- **Optimizing for Deployment:** Optimizing the model for deployment on different platforms, including cloud services and edge devices, ensures

that the model can be used in various environments. This includes reducing the model size, improving inference speed, and ensuring compatibility with different hardware.

- **Continuous Learning:** Implementing mechanisms for continuous learning and updating the model with new data can help maintain and improve its performance over time. This includes setting up pipelines for collecting user feedback and retraining the model periodically.

Kaggle Datasets:

Kaggle hosts numerous datasets related to image processing and computer vision, often with user-generated contributions. These datasets can include pixelated images and restoration challenges, providing practical examples for model training and evaluation.

Final Accuracy – 87.6%

References:

- Google search engine
- Blur dataset (kaggle.com)
- Real Blur Dataset Dataset | Papers With Code

PROJECT URL:

https://github.com/srikartv/DetectPixelatedImage_and_Correct_It-/tree/main