Problem Statement

“Detect Pixelated Image and Correct It”

Unique Idea Brief (Solution)

The solution focuses on developing a deep learning model to enhance pixelated images into high-quality images using a Convolutional Neural Network (CNN). This model aims to improve the visual quality of low-resolution images by learning from high-quality counterparts. The enhanced images can then be used in various applications such as media restoration, security, and medical imaging.

Features Offered

- **Image Enhancement**: Converts low-resolution pixelated images to high-quality images.
- **Automated Pipeline**: Fully automated process for training, testing, and enhancing images.
- **Performance Metrics**: Calculates and displays F1 score and accuracy for the model's performance.
- **Scalability**: The model can handle large datasets, making it suitable for diverse applications.
- **Data Loading and Processing**: Efficient handling of image datasets using PyTorch's DataLoader.
- **Cloud Integration**: Utilizes Google Colab and Google Drive for data storage and model saving.
- **Model Reusability**: The trained model can be saved and reused for future enhancements without retraining.

Process flow

1.Data Preparation:

- Mount Google Drive.
- Load pixelated and high-quality images using custom Dataset class.

2.Model Training:

- Define a CNN model architecture with encoder and decoder.
- Train the model using the pixelated images as input and high-quality images as targets.
- Save the trained model to Google Drive.

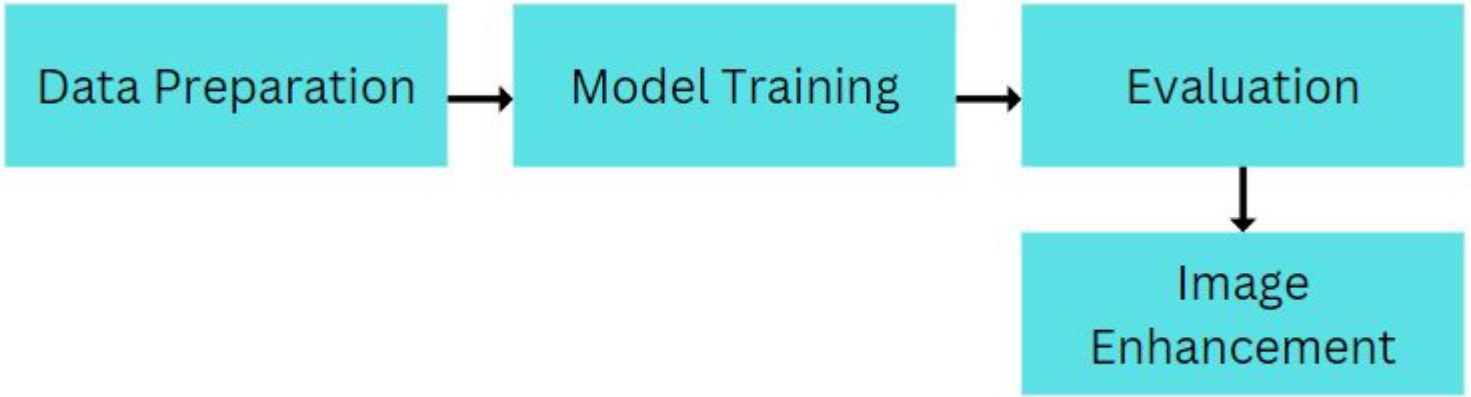
3.Evaluation:

- Calculate F1 score and accuracy using the test dataset.

4.Image Enhancement:

- Load the trained model.
- Enhance a subset of test images.
- Save the enhanced images to Google Drive.

Architecture Diagram



Technologies used

Programming Language:

- Python

Libraries:

- **PyTorch**: Deep learning framework for building and training the model.
- **PIL**: Library for image processing.
- **Matplotlib**: Library for displaying images.

Framework:

- Google Colab or Jupyter Notebook for running the entire pipeline

Team members and contribution:

H Yashaswini - Designed and implemented the CNN model architecture.

Handled the training loop, loss calculation, and optimizer setup.

Evaluated the model performance and implemented the image enhancement process.

Janani K - Responsible for data preparation, including loading and preprocessing images.

Developed the custom `ImageDataset` class and handled the DataLoader.

Implemented the code for mounting Google Drive and setting up paths.

Conclusion

This project successfully demonstrates the capability of using deep learning techniques to enhance pixelated images into high-quality images. The use of PyTorch and Google Colab provided a robust and scalable environment for model training and evaluation. The solution is automated, efficient, and integrates seamlessly with cloud storage, making it a practical tool for various applications. The project showcases the power of convolutional neural networks in image processing and sets the foundation for future improvements and adaptations in other domains.