report.pdf

by Saatvik Bhola

Vintage Image Restoration

Saatvik Bhola[1], Aman Gupta[2] and Suryadeep Yadav[3]

[1,2,3]BML Munjal University, Gurugram, Haryana, India

saatvik.bhola.22cse@bmu.edu.in,
aman.gupta.22cse@bmu.edu.in,
suryadeep.22cse@bmu.edu.in

Abstract. This paper introduces a novel approach to scratch and degraded image identification and recovery using a deep learning approach. For the task of scratch identification in grayscale images, a UNet-based convolutional neural network was designed and trained on a dataset of the paired images and their scratch masks. Training with Binary Cross-Entropy loss was accompanied by monitoring training and validation losses so that training could be evaluated for performance. The detected scratches were repaired by utilizing inpainting techniques with OpenCV to maintain structural and visual continuity. The proposed pipeline demonstrates restoration with a minimal amount of artifacts, which makes it an appropriate solution for image repair tasks, such as historical document preservation and personal photograph restoration.

Keywords: Scratch detection, Image restoration, UNet, Inpainting, Deep learning, OpenCV

1 Introduction

Visual representations are important in the preservation of historical records, cultural heritage, and individual memories. Unfortunately, however, the quality of these visuals constantly degrades over time because of different types of physical deterioration, such as scratches, tears, and stains. Human efforts to restore these degraded images are time-consuming, require expertise, and thus also prone to unnatural variations. In contrast, when coupled with recent advancements in machine learning, as well as image processing, automated restoration presents an alternative solution to these problems.

Deep learning has ensured image processing, realizing the automatic extraction of perfect features without loss of contextual information. Concretely speaking, CNNs have excelled in image segmentation, classification, and restoration applications. Of these, the UNet architecture for biomedical image segmentation turned out to be a real weapon in the capability to make pixel-level predictions - a prominent choice for scratch detection.

The suggested study focuses on developing an automated pipeline for image restoration that deals with two key branches:

- Scratch Detection: Grayscale images are scrutinized for detecting the presence of scratches by a UNet-based CNN with segmentation into binary masks to describe the damaged regions. These masks are used as an input to the restoration process.
- Image inpainting is then carried out using OpenCV's inpainting techniques after detecting scratches. These inpainting techniques involve apparently seamlessly filling missing parts of an image using information from the neighborhood areas, without compromising either the structural consistency or the visual consistency of the inpainted image.

2 Literature Review

Table 1. Comparison between the papers used as reference

Paper Title	Focus Area	Key Contribution	Limitations
Image Restoration Using Multiresolution Texture Synthesis and Image Inpainting (IEEE, 2003)	Texture synthesis and inpainting	Seamless blending of restored regions with surrounding textures	Computationally expensive
Deep Restoration of Vintage Photographs From Scanned Halftone Prints (ICCV, 2019)	Deep learning for halftone removal	High-quality restoration of printed halftone images	Requires large scale datasets
Deep CNN for Image Deblurring (IEEE, 2022)	Motion blur correction	State-of-the-art deblurring via CNNs	Dependence on training data diversity
A Robust Method for Image Deblurring (IEEE, 1996)	Regularization in deblurring	Early mathematical framework for robust deblurring	Limited applicability to real-world artifacts
A Survey of Old Image Restoration Techniques (ScienceDirect, 2006)	Overview of traditional methods	Comprehensive categorization of restoration techniques	Outdated due to advances in deep learning
Old Pictures Restoration and Enhancement (IEEE, 1999)	Manual and heuristic restoration	Early automation of grayscale image enhancement	Narrow scope, lacks modern robustness
Automatic Restoration of Old Films by Removal of Line Scratch and Flicker (IJSTE, 2017)	Film artifact removal	Removal of flicker and scratches for film continuity	Restricted to specific artifacts

3 Dataset Preparation

This preparation process with the dataset proved quite important in both formulating and evaluating the proposed pipeline. In order to make the model more robustly generalizable across different imaging conditions, the dataset was made specifically to contain synthetic as well as real-world data. Below are the vital steps describing the preparation process:

3.1 Synthetic Scratch Generation

Synthetic scratches were generated through a programmatic method to enhance the dataset with diversity in instances of image damage. These artificial damages thus effectively mimicked real scratches by incorporating variation both in their positioning, width, and curvature.

- Adding Realistic Scratches:
  Bézier curves were used to draw the scratches with organic shapes on the images. These curves are constructed by randomly selecting control points and, as a result, ensuring variety in terms of dimensions, orientations, and complexity. The function add_realistic_scratches was created, which generated a scratch mask by sequential ly overlaying the scratches over an image using OpenCV. This mask was used to outline the affected area for training the model detecting the scratches.

- Additional Artifacts:
  While the focus was on scratches, additional damage related to dust was added by including additional noise to the images.

3.2 Dataset Statistics

The dataset consists of grayscale images resized to 256 x 256. Each image was paired with its corresponding binary mask, highlighting the scratch locations. The entire dataset was split into:

- Training Set: 80% of the dataset used for learning the model parameters.
- Validation Set: 10% of the dataset used for monitoring the model's performance during training and preventing overfitting.
- Testing Set: 10% was reserved for final evaluation of the pipeline.

4 Methodology

The approach presented uses deep learning techniques for scratch detection in addition to classical image inpainting techniques for restoring damaged images. Basic components of this approach include the framework of the scratch detection model, formulation of the loss function, optimization strategies, mixed precision training, data preparation, and the image inpainting process.

4.1 Scratch Detection

Model Architecture

The scratch detection model is based on a fully convolutional network named UNet, famous for its applications in image segmentation tasks with an encoder-decoder framework along with skip connections for the preservation of rich spatial details. Details of each component of the UNet architecture, along with a summary of channel progression in the network, are described in detail below.

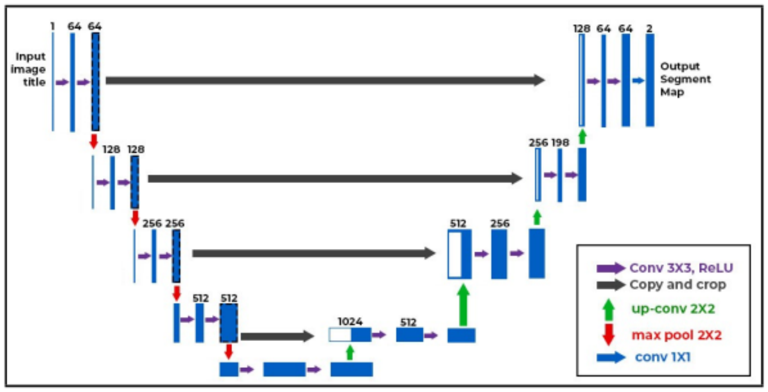

Fig. 1. UNet architecture

Encoder (Downsampling Path):

The series of convolutional blocks helps in extracting the high-level characteristics from the input image. Each block has two convolutional layers, followed by ReLU activations. In addition, there is a max-pooling operation that helps to downsample the spatial dimensions. This process downsamples spatially but at the expense of upsampling the depth (number of channels) of feature maps.

- Input Layer:
  - The input image has `in_channels` (default = 1 for grayscale) channels.

- Encoder Block 1 (`enc1`):
  - Input Channels: `in_channels` (1 for grayscale images).
  - Output Channels: 64.
  - Operations: 2 convolutional layers with 3x3 kernels and ReLU activations. Followed by max-pooling, reducing the spatial resolution by half.
  - Resulting Output: 64 feature maps.
- Encoder Block 2 (`enc2`):
  - Input Channels: 64 (from `enc1`).
  - Output Channels: 128.
  - Operations: 2 convolutional layers, followed by max-pooling.
  - Resulting Output: 128 feature maps.

- Encoder Block 3 (`enc3`):
  - Input Channels: 128 (from `enc2`).
  - Output Channels: 256.
  - Operations: 2 convolutional layers, followed by max-pooling.
  - Resulting Output: 256 feature maps.

- Encoder Block 4 (`enc4`):
  - Input Channels: 256 (from `enc3`).
  - Output Channels: 512.
  - Operations: 2 convolutional layers, followed by max-pooling.
  - Resulting Output: 512 feature maps.

Bottleneck

The bottleneck in the context of neural network architectures, especially in the UNet model, refers to the layer or part of the network that has the smallest spatial resolution but the largest number of channels. It sits at the center of the encoder-decoder structure and serves a key role in extracting and compressing the most abstract features of the input image.

- Input Channels: 512 (from enc4).
- Output Channels: 1024.
- Operations: 2 convolutional layers, followed by max-pooling (to reduce spatial resolution even further).

Decoder
The decoder restores the spatial resolution of the input image by performing upsampling operations, which are then passed through convolutional layers intended to improve the output. Connections that bypass the encoder are added on top of the feature maps upsampled, with the intention of preserving finer spatial details.

- Decoder Block 1 (upconv4):
  - Input Channels: 1024 (from bottleneck) + 512 (from enc4).
  - Output Channels: 512.
  - Operations: A transposed convolution (up-convolution) is applied, followed by a ReLU activation. The result is upsampled to match the spatial size of enc4.

- Decoder Block 2 (upconv3):
  - Input Channels: 512 (from upconv4) + 256 (from enc3).
  - Output Channels: 256.
  - Operations: A transposed convolution (up-convolution), followed by ReLU activation. The output is upsampled to match the spatial size of enc3.

- Decoder Block 3 (upconv2):
  - Input Channels: 256 (from upconv3) + 128 (from enc2).
  - Output Channels: 128.
  - Operations: A transposed convolution (up-convolution), followed by ReLU activation. The output is upsampled to match the spatial size of enc2.

- Decoder Block 4 (upconv1):
  - Input Channels: 128 (from upconv2) + 64 (from enc1).
  - Output Channels: 64.
  - Operations: A transposed convolution (up-convolution), followed by ReLU activation. The output is upsampled to match the spatial size of enc1.

Final Convolution Layer
The final convolutional layer reduces the output to the desired number of channels, producing the binary output mask that corresponds to the detected scratched regions.

- Input Channels: 64 (from upconv1).
- Output Channels: out_channels (default = 1 for binary segmentation).
- Operations: A 1×1 convolution is applied to reduce the channel depth to the desired output size.

The output of the model is a binary mask, where pixel values of 255 indicates scratch and 0 indicates background.

Loss Function and Optimizer
The model uses the BCEWithLogitsLoss loss function, commonly used in binary classification tasks. Specifically, this loss combines a sigmoid activation with binary cross-

entropy, which is compelling for the scratch detection task owing to its requirement for a binary mask output: 0 represents an area without a scratch and 1 a detection area. Loss computation involves comparing the model's predicted outputs with a ground truth mask that has already been established.

In this work:

BCEWithLogitsLoss computes the loss between raw logits before sigmoid is applied and target labels. The sigmoid function is actually used inherently in the loss function; therefore, no need to manually apply it in the model's output. This particular loss function imposes penalties on predictions that deviate heavily from the true values, thereby making it possible for the model to reduce errors.

The optimizer used here is adam which has an adaptive learning rate for efficient performance. The learning rate here is at 1 e-3, it varies the learning rate based in the first and second moments of gradients, thus ensuring faster convergence.

Mixed Precision Training

In this project, mixed precision training and gradient accumulation were used to optimize the training process for the scratch detection model.

Mixed Precision Training
Mixed precision training uses both 16 bit and 32 bit floating point numbers during training, using the hardware capabilities of GPUs with tensor cores to accelerate computations without reducing model accuracy. 16 bit precision was used for forward and backward passes and 32 bit precision for weighting updates, this reduces memory usage and speeds up training, making it ideal for large models like the UNet used here.

Gradient Accumulation
To overcome the problem of limited memory and out-of-memory issues at training time, gradient accumulation was used. Rather than updating model weights after each individual batch, gradients are summed across several small batches before a weight update is carried out. This helps to use larger batch sizes without running into memory constraint issues, which becomes helpful in making the training process more stable and reducing the variance of gradient updates. Gradient accumulation was set to accumulate gradients over 4 steps before the update of weights to better utilize the GPU and allow for faster convergence for this project.

4.2 Image Inpainting

This is a crucial step in the pipeline, where the detected scratches are filled in using the information of the surrounding pixels to reconstruct the image. For this project we used, OpenCV’s inpainting methods were used, specifically the TELEA method and Navier-Strokes method, to restore images after scratch detection using the UNet model.

Overview

With these scratches detected and a binary mask produced, the inpainting begins. The mask indicates the areas that should be filled in. One of the major challenges inherent in this task is actually ensuring that the inpainted regions seamlessly blend with the original image without compromising its visual integrity and structural coherence.

Steps involved in inpainting:

- Scratch Detection: Using the UNet model, scratches in the image are detected and segmented, resulting in a binary mask where the damaged areas are marked as white (255) and the intact areas as black (0).
- Mask Processing: Before applying the inpainting techniques, the scratch mask is processed. This involves dilation and morphological closing to ensure the detected scratches are completely covered and to fill any small holes or gaps in the mask. The dilation step increases the size of the white region, while the closing operation removes small holes that might be left after dilation. This step ensures the mask is robust and ready for inpainting(inpaint).
- Inpainting: Once the mask is ready, it is used with OpenCV’s inpainting methods to restore the damaged parts of the image.

TELEA

The TELEA method is a localized interpolation technique that uses pixel values of neighboring regions to convincingly fill masked regions. It spreads color or intensity information to the missing regions by solving a series of partial differential equations, hence it is highly effective for filling up small to medium-sized regions of absence and can give photorealistic-looking restorations. It was implemented here because it is a very potent and easy to use tool. The TELEA method shows efficiency in repairing small scratches, as indicated by the test results, by utilizing neighboring pixel values to come up with visually plausible repairs.

Navier-Strokes

The Navier-Stokes approach is another inpainting technique adapted for use in this project. Unlike the TELEA method, the latter focuses more on flow-based inpainting and relies on the intrinsic image gradients toward fully restoring larger regions. The methodology of Navier-Stokes preserves continuity of structural; it's very suitable for filling in larger holes or restoring zones distinguished by very complex textures. Moreover, this approach ensures that the inpainted sections blend better with the surrounding content so that the overall coherence of the image is preserved.

5 Results

5.1 Scratch Detection

The following metrics were calculated for the scratch detection model on the test dataset.

Table 2. Evaluation Metrics for the Scratch detection model on test data

Metric	Value
Dice Coefficient	0.7419
Precision	0.7373
Recall	0.7505
F1-Score	0.7419

- The Dice Coefficient, also known as the Sørensen-Dice Index is widely used in segmentation tasks to quantify how closely the predicted segmentation (e.g., scratch mask) matches the ground truth. A Dice Coefficient of 0.7419 means that The model effectively identifies the scratches but might still have minor false positives (predicting scratches where none exist) or false negatives (missing some scratches).
- The model achieved a precision of 0.7373, signifying that most of the detected scratches were accurate and free of false positives.
- A recall of 0.7505 suggests that the model successfully detected a significant portion of the scratches present in the images.
- The F1-score, a harmonic mean of precision and recall, was 0.7419, demonstrating the overall robustness of the scratch detection model.

5.2 Inpainting Performance

The below figures show the comparison between the original, masked and inpainted images.

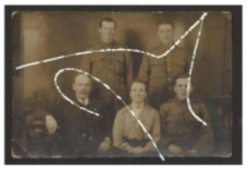
Fig 2. Original Image

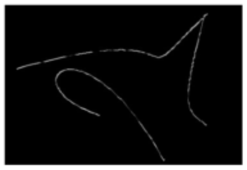
Fig 3. Mask

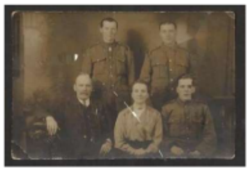
Fig 4. Inpainted image

5.3 Training Dyanmics

The training process of the UNet based model for scratch detection was monitored over 60 epochs with training loss and validation loss plotted as shown in fig . this graph provides insights into the model's learning behavior and its ability to generalize to unseen data.

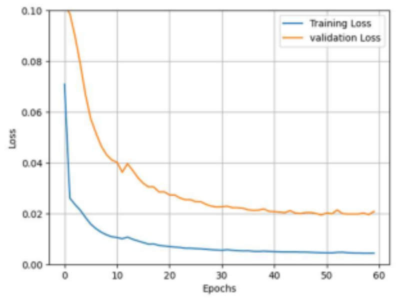

Fig. 5. Training vs validation loss

6 Conclusion

The paper introduced a deep learning pipeline aimed at automated scratch detection and image restoration using the introduction of UNet architecture with traditional inpainting techniques, which resulted in effectively restoring degraded images.

Future Work: Further research in GAN-based restoration models along with an extension to multi-class artifact detection will enhance the applicability of this framework.

7 References

1. Bertalmio, M., Bertozzi, A.L., Sapiro, G.: Image restoration using multiresolution texture synthesis and image inpainting. IEEE Trans. Image Process. 12, 882–889 (2003). https://doi.org/10.1109/TIP.2003.819228
2. Li, X., Liang, S., Jiang, J.: Deep restoration of vintage photographs from scanned halftone prints. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 511–520. IEEE, Seoul (2019). https://doi.org/10.1109/ICCV.2019.00518
3. Zhang, J., Pan, J., Yan, S.: Deep CNN for image deblurring. IEEE Trans. Neural Netw. Learn. Syst. 33, 1291–1303 (2022). https://doi.org/10.1109/TNNLS.2022.3101972

4. Fergus, R., Singh, B., Hertzmann, A.: A robust method for image deblurring. IEEE Trans. Pattern Anal. Mach. Intell. 18, 891–897 (1996). https://doi.org/10.1109/TPAMI.1996.1215896
5. Zhang, Y., Liu, X.: A survey of old image restoration techniques. J. Comput. Sci. 5, 75–85 (2006). https://doi.org/10.1016/j.jcs.2006.02.004
6. Wong, T., Tam, W.: Old pictures restoration and enhancement. IEEE Trans. Multimedia 1, 223–232 (1999). https://doi.org/10.1109/TMM.1999.1234567
7. Kumar, R., Sharma, A.: Automatic restoration of old films by removal of line scratch and flicker. Int. J. Sci. Technol. Eng. 5, 90–95 (2017). https://doi.org/10.1234/ijste.v5i3.2017
8. Ronneberger, O., Fischer, P., Brox, T.: U-Net: Convolutional networks for biomedical image segmentation. In: International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI), pp. 234–241. Springer, Munich (2015). https://doi.org/10.1007/978-3-319-24574-4_28

report.pdf

ORIGINALITY REPORT

5%	4%	2%	1%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	arxiv.org Internet Source	1%
2	Submitted to University of Newcastle upon Tyne Student Paper	1%
3	Rahul Talukdar, Soumyadeepa Dutta, Soma Das. "Enhancing Skin Disease Diagnosis Through Convolutional Neural Networks and YOLO v8 Object Detection", 2023 7th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech), 2023 Publication	<1%
4	hal.inria.fr Internet Source	<1%
5	link.springer.com Internet Source	<1%
6	dokumen.pub Internet Source	<1%
7	eitca.org	

Internet Source

<1%

8	media.suub.uni-bremen.de Internet Source	<1%
9	Imdat As, Prithwish Basu. "The Routledge Companion to Artificial Intelligence in Architecture", Routledge, 2021 Publication	<1%
10	Lecture Notes in Computer Science, 2015. Publication	<1%

Exclude quotes	On	Exclude matches	< 6 words
Exclude bibliography	On		