STAR WARS

IMAGE RESTORATION WITH DEEP LEARNING

By Elaheh Shakeri

Email: elahehshakeri3@gmail.com

Date: December 12th 2022

PROBLEM STATEMENT

Photographs play an important role in everyone's life. They connect us to our past, they remind us of people, places, feelings, and stories. They can help remind us who we are. However, before the age of digital images, analog cameras were all that were available to capture our precious moments. Old printed photographs often have visible damages such as scratches, discolorations, missing parts or even specks caused by a defected film. While it is possible to manually retouch and modify old pictures, it is often time consuming and a laborious task**.**
The goal of this project is to be able to minimize these damages and restore old pictures using deep learning algorithms.

DATA OVERVIEW

For our task, we will be working with a dataset containing original and modified stills from the fourth- and fifth-Star Wars movies, *Star Wars: Episode IV - A New Hope* and *Star Wars: Episode V -The Empire Strikes Back*. The original movies were aired in 1977 and 1980 which the original stills are captured from. However, in later releases the resolutions were improved and our second images are captured from these improved movie releases. Unfortunately, no further information is provided on how and to what extend the movies were modified.

The original dataset consists of 31,801 paired images (63,602 total images). Since it was very computationally expensive to train our models with all the images in our dataset, we used 2%, 12% and 32% of our dataset which are 636, 3816 and 10176 paired images respectively.

We also have a set of 200 old images that have low resolutions and some structural defects. We downloaded this dataset from Kaggle. The source of the images is Google and they have been manually downloaded from Google by the Kaggle collaborator.

DATA CLEANING

The key point of our dataset is the fact that the images are in pairs and both images need to be fed into our model for training, validation and testing.

To ensure that the paired images are both included in the input and output folders of the train, validation and test datasets, we will do cross examinations of the filenames between the input and output folders. If there is not a corresponding paired image with the same name, we will move the image file to a new folder named "Disregard". By doing so, we remove the lone images from our dataset.

In addition, in order to make sure that all the images in our datasets are accepted by python and tensorflow, we define a function that accepts a directory and prints the paths to images that are not accepted. Any images that are not accepted by either python or tensorflow will be disregarded and removed from our dataset.

MODELING

The structure we chose for our unsupervised modeling is the structure of a convolutional autoencoder.
A convolutional autoencoder is a Convolutional Neural Network (CNN) with an autoencoder structure. Autoencoders are an unsupervised deep learning technique. They are a type of neural network with a bottleneck structure which forces the model to learn key and compressed representations of the inputs and attempts to reconstruct the input as an output with the knowledge it has learned. In CNN structures a window (convolutional kernel) slides on the input image learning local features. Dimension reduction is also possible with CNN's pooling layers.

The evaluation metrics for our model are Mean Squared Error (MSE), Peak to Signal Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM).

- **Mean Squared Error:**
  MSE is an indication of how different two images are, by comparing them pixel by pixel. High MSE values mean that there a big difference between the original image and the processed (or in our case reconstructed) image. Therefore, the goal of is to minimize MSE.
- **Peak to Signal Noise Ratio:**
  PSNR is an approximation to human perception of reconstruction quality. It is an expression for the ratio between the maximum possible value (power) of a signal and the power of distorting noise that affects the quality of its representation. Higher PSNR means that the power of the noise is smaller which means that the reconstructed image is of higher quality.
- **Structural Similarity Index Measure:**
  The SSIM index is used for measuring the similarity between two images. The SSIM predicts image quality based on an initial uncompressed or distortion-free image as reference. SSIM of 1 translates to identical images, therefore the goal is to increase SSIM to a value close to 1.

We performed our modeling on two different convolutional autoencoder structures. One model with 7 hidden layers has 29,507 trainable parameters while the other with 14 hidden layers has 374,406 trainable parameters. Due to limited time and computational resources we were only able to train these models on samples of our original dataset.

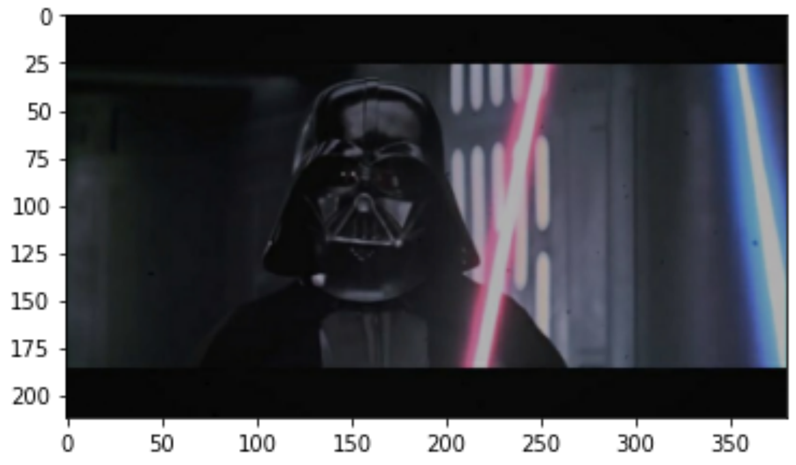

Image 1: Model Input

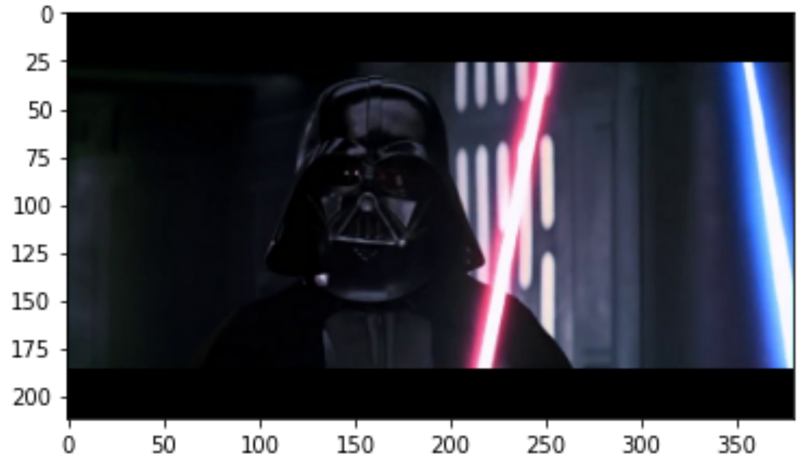

Image 2: Model Ground Truth

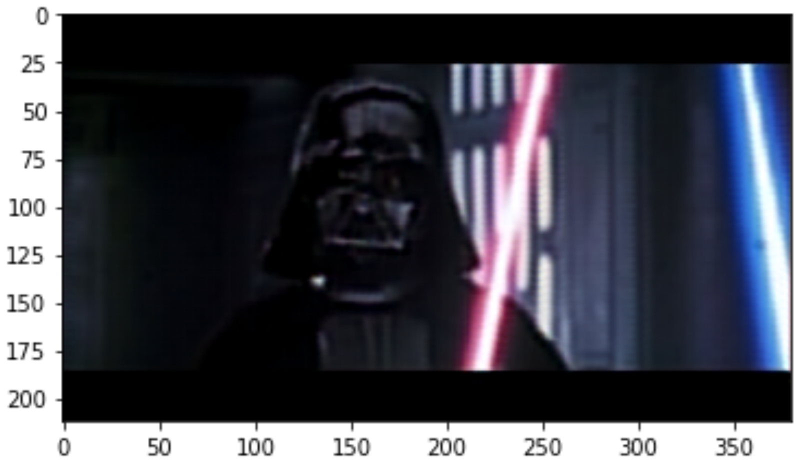

Image 3: Model Output

Table 1. Summary of the performance of the basic convolutional autoencoder with 29,507 trainable parameters ran for 100 epochs

	TRAIN			VALIDATION			TEST		
	MSE	PSNR	SSIM	MSE	PSNR	SSIM	MSE	PSNR	SSIM
2% sample dataset	0.0012	30.44	0.8877	0.0013	30.06	0.8922	0.0013	30.06	0.8834
12% sample dataset	0.0012	30.48	0.8922	0.0013	30.29	0.8877	0.0013	30.31	0.8876
32% sample dataset	0.0009	31.86	0.9124	0.0009	31.74	0.9098	0.0009	31.77	0.9113

Table 2. Summary of the performance of the basic convolutional autoencoder with 29,507 trainable parameters ran for 300 epochs

	TRAIN			VALIDATION			TEST		
	MSE	PSNR	SSIM	MSE	PSNR	SSIM	MSE	PSNR	SSIM
2% sample dataset	0.00087	32.09	0.9137	0.0009	31.77	0.9156	0.0009	31.68	0.9075
12% sample dataset	0.00084	32.29	0.9195	0.00085	32.16	0.9163	0.0008	32.16	0.9173

Table 3. Summary of the performance of the basic convolutional autoencoder with 29,507 trainable parameters ran for 200 epochs

	TRAIN			VALIDATION			TEST		
	MSE	PSNR	SSIM	MSE	PSNR	SSIM	MSE	PSNR	SSIM
32% sample dataset	0.0009	31.86	0.9130	0.0009	31.91	0.9134	0.0009	31.77	0.9113

Table 4. Summary of the performance of the advanced convolutional autoencoder with 374,406 trainable parameters ran for 100 epochs

	TRAIN			VALIDATION			TEST		
	MSE	PSNR	SSIM	MSE	PSNR	SSIM	MSE	PSNR	SSIM
2% sample dataset	0.0035	25.82	0.7628	0.0038	25.46	0.7631	0.0035	25.69	0.7627
12% sample dataset	0.0023	27.79	0.8141	0.0023	27.72	0.8109	0.0023	27.67	0.81

Table 5. Summary of the performance of the advanced convolutional autoencoder with 374,406 trainable parameters ran for 200 epochs

	TRAIN			VALIDATION			TEST		
	MSE	**PSNR**	**SSIM**	**MSE**	**PSNR**	**SSIM**	**MSE**	**PSNR**	**SSIM**
2% sample dataset	0.0025	27.456	0.8007	0.0027	27.00	0.8048	0.0025	27.11	0.7971
12% sample dataset	0.0018	28.78	0.8381	0.0019	28.40	0.8333	0.0020	28.41	0.8324

We attempted to implement transfer learning in our modeling as well. Transfer Learning is applying knowledge gained (weight values) from a trained model on a different but related problem. However, during the limited time that we had for our capstone project, I was only able to find trained models that were trained for image classification or image segmentation problems. The problem with using these trained models as a transfer model for our project is that what models learn for image classification or segmentation is different than what a model needs to learn for image restoration. We still tried to use an EfficientNetB0 model with trained weights for our problem, but as expected, we were not able to achieve good or acceptable results. The goal is to find a trained model that has been trained for denoising, deblurring or similar task and transfer that model to our problem.

Based on the results shared above, key findings and conclusions are shared in the next section.

FINDINGS AND CONCLUSIONS

- The best performing model for basic convolutional autoencoder with 29,507 trainable parameters is the model trained on 12% of the dataset for 300 epochs.
- For the second convolutional autoencoder structure with 374,406 trainable parameters, the best performing model is trained with 12% of the dataset for 200 epochs.
- When the structure of the model is more complicated and number of trainable parameters is higher, a larger dataset and longer training duration is required.
- Based on the results from our modeling, we can conclude that if we had more time and computational resources, we were able to improve the model performance significantly by using a larger sample or the total dataset and training the model for longer durations.