SYNOPSIS

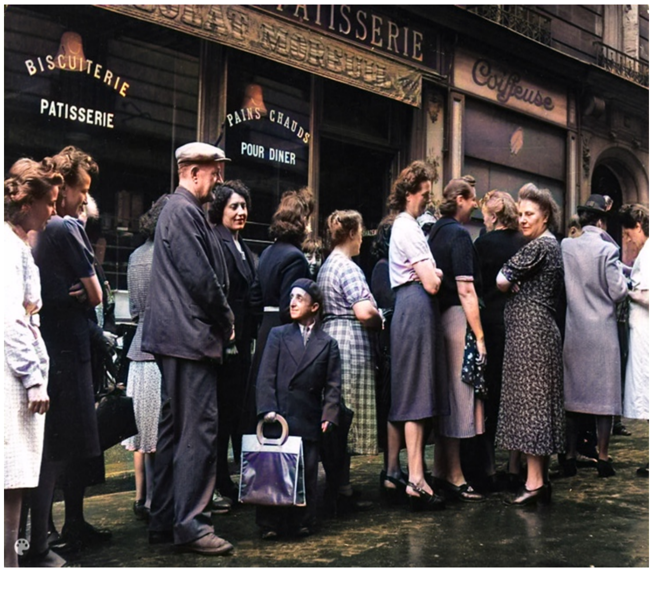
Major Project Work

On

<u>IMAGE - COLORIZER</u>

using deep learning methods, such as convolutional

neural networks **Generative Adversarial Networks (GAN)**



Under the Guidance of

Nikita Malik

(Assistant Professor, Dept. of Computer Applications)

Submitted by:

TEAM

Project Leader - Rahul Mittal (Enroll No. 40921202018)

Team Member - Prashant Sharma (Enroll No. 41021202018)

DEPARTMENT OF COMPUTER SCIENCE

MAHARAJA SURAJMAL INSTITUTE

C – 4, JANAKPURI, NEW DELHI – 58

January – May 2021

OBJECTIVE

The mission of this project is to colorize and restore old images.

The world has always been full of colour but until recently we lacked the technology to capture its beautiful colours and the pictures were mostly black and white. But these can also be colorized to achieve realistic results, using AI. This method involves training a Convolutional Neural Network (CNN) on a large number of lab colorspace photos. The colospace method corresponds better with the human visual system than RGB. The images are made up of three channels and the predictive model is trained to predict a (red-green) or b (blue-yellow) depending on the lightness channel (input). By recombining into a colorized image, it can be converted back to RGB.

This technique can offer good results on a number of materials, such as landscapes and countryside. However, results may not be the same for more complicated scenes. Even then it is less tedious than manual techniques and offers better results.

GOAL

The goal of this deep learning model is to colorize, restore, and give new life to old images

Deep learning models used to add high-quality colorization to grayscale images with amazing results. Colorizing with neural networks traditionally produce dull colors as the model minimize its loss by producing average colors that end up in the brown area.

The use of a GAN pushes the model to use more vivid colors, even if they are not the correct ones, leading to more realistic predictions.

Architectural Details

The following deep learning concepts are used in the model. These concepts are:

- **Self-attention:** we've used U-Net architecture for the generator, they also modified the architecture to use Spectral Normalization and self-attention in the model.
- **Two-Time Scale Update Rule:** It is a way of training the GAN architecture. It's just one to one generator/critic architecture and a higher critic learning rate. This is modified to incorporate a threshold critic loss that makes sure that the critic is "caught up" before moving on to generator training. This is particularly useful for NoGAN training.
- **No-GAN:** This method of GAN training is developed by the authors of the model. The main idea behind that model that you get the benefits of GAN training while spending minimal time doing direct GAN training. We will discuss NoGAN in more detail.
- **Generator Loss:** There are two types of NoGAN learning in the generator:

    - **Perpetual Loss:** This loss is used in the generator to report and minimize the losses generated due to bias in the model.
    - **Critic Loss:** It is loss used in the discriminator/critic.

No-GAN

This is a new type of GAN training that is developed by the authors of Image - Colorizer. It provides the benefits of GAN training while spending minimal time doing direct GAN training. Instead, we spent most time training generator and critic separately with more straight-forward, fast, and reliable conventional methods.

The steps are as follows:

- First, we train the generator in a conventional way by itself with just the feature loss.
- Next, we generate images from the trained generator and train the critic on distinguishing between those outputs and real images as a basic binary classifier.
- Finally, train the generator and critic together in a GAN setting (starting right at the target size of 192px in this case).

Pre-training the Generator

The dataset is made of images taken from ImageNet.

Inputs are black & white versions of the images and outputs are the original colorized version. The first step is to train a generator by feeding the black & white images and trying to predict the colorized version.

- We start with images reduced to 64 pixels width to be able to use larger batches and train faster
- We then increase the dimension to 128 pixels
- Finally, we use a larger size of 192 pixels which trains slower with smaller batches

The predicted samples show images with a few colors but "brownish" overall which represents an average color predicted by the model when it is not sure of what to use. The most vivid colors used are for the sky, water, people's skin, and vegetation which seems to be detected more easily and have lower variance in color (vs a T-shirt for example).

Pre-Training the Critic

We then pre-train a critic whose objective is to identify which images are real and which ones come from the pre-trained generator.

This will help in getting a quicker start on the traditional "GAN" training by having a pre-trained generator and a pre-trained critic.

Predicted images

While not perfect, it is interesting to see how this method clearly generates more colorful samples.

The quality of the results will depend on the art of tweaking the training duration at each step and performing several experiments.

FUNCTIONAL DESCRIPTION

The standard objective function for adversarial loss:

$$V(G, D) := \underset{x \sim q_{\text{data}}(\boldsymbol{x})}{\mathrm{E}} [\log D(\boldsymbol{x})] + \underset{\boldsymbol{z} \sim p(\boldsymbol{z})}{\mathrm{E}} [\log(1 - D(G(\boldsymbol{z})))]$$

From the above function, *p(z)* is the generator distribution to be learned through adversarial min-max optimization.

From the above function, *p(z)* is the generator distribution to be learned through adversarial min-max optimization.

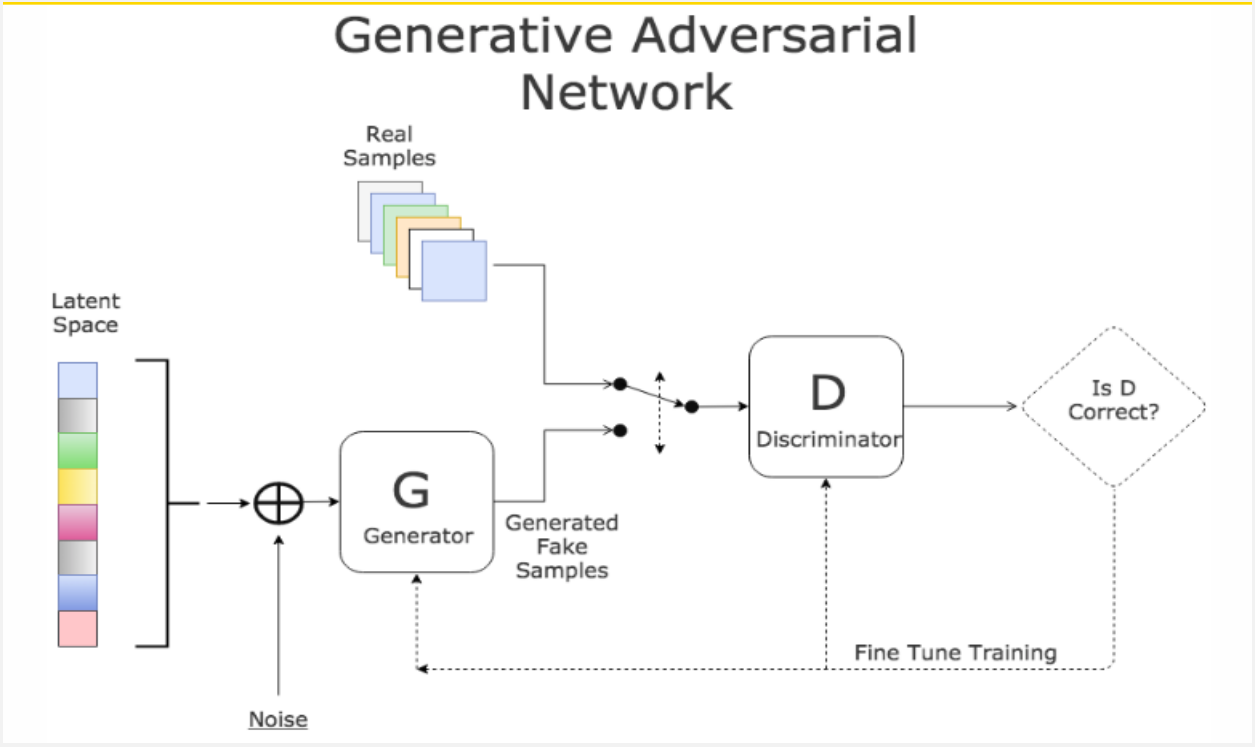
$$\min_{G} \max_{D} V(G, D)$$

From the above function, *p(z)* is the generator distribution to be learned through adversarial min-max optimization.

D → Discriminator
G → Generator
z → Noise Vector
qdata → data distribution

Both networks(D & G) need to be trained simultaneously. We don’t want to over train one model. There has to be a healthy competition.

All the important GAN training only takes place in a very small fraction of time. There's an inflection point where it appears the critic has transferred all the useful knowledge to the generator. There appears to be no productive training after the model achieved the inflection point. The hard part appears to be finding the inflection point and the model is quite unstable, so the author has to create a lot of checkpoints. Another key thing about No-GAN is that you can repeat pre-training the critic on generated images after the initial GAN training, then repeat the GAN training itself in the same fashion.



Model pipeline for training Generator and Discriminator

The generator model takes in the noise input from latent space and tries to generate fake samples. It fails initially but will eventually learn to generate low dimensional images which will be indistinguishable from real images.

MODEL

Artistic: This model achieves the best results in terms of image coloration, in terms of details, and vibrancy. The models use a ResNet 34 backbone architecture with U-Net with an emphasis on the depth of layers on the decoder side. There are some drawbacks of the model such as this model does not provide stability for common tasks such as natural scenes and portraits and it takes a lot of time and parameter tuning to obtain the best results.

The project is built on the wonderful Fast.AI library. So, the preliminary requirements :

- **Fast.AI library** .
- **All Fast.AI dependencies**
- **Pytorch 0.4.1**
- **JupyterLab**
- **Tensorboard** (i.e. installing Tensorflow) and **TensorboardX** . the images are processed in the Tensorboard every 200 iterations, so you get a constant and convenient view of what the model does.
- **ImageNet** : A great set of training data.
- **Powerful graphics card** . memory than 11 GB in GeForce 1080Ti. If you have something weaker, it will be difficult. Unet and Critic are absurdly large, but the bigger they are, the better the results.
- **Neural Networks and Deep Learning**
- **Distill:**
- **FloydHub:** super-intuitive platform for getting up-and-running on cloud GPUs, instead of worrying about Docker images, environment set-up, and DevOps

Hardware and Software Used

Hardware:

- **(Training Only) BEEFY Graphics card.**

  Min 11 GB GeForce 1080TI (11GB). The Generators and Critic are ridiculously large.

Software:

- **Google Colab (Uses Cloud based CPU, Memory, GPU)**
- **Jupiter Notebook**

REFERENCES

- https://machinelearningmastery.com/what-are-generative-adversarial-networks-gans/
- https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53
- https://www.fast.ai
- https://en.wikipedia.org/wiki/Fast.ai/PyTorch
- https://www.floydhub.com