**Abstract**

Artificial intelligence (AI) is the simulation of human intelligence processes by machines, Machine Learning (ML) is a type of AI that enhances the accuracy of software applications at predicting outcomes without explicit programming. Deep Learning (DL) is a type of ML based on artificial neural networks with multiple layers of processing being used to extract progressively higher level features from data. It is highly effective in real world applications (self-driving cars, sentiment analysis, virtual assistants, etc.). The main advantages of DL include its efficiency in handling unstructured data, better self-learning capabilities, cost-effectiveness and highly scalable nature. In this paper we research the use of DL for image restoration in computer vision. We further implement an image-to-image translation architecture with a Generative Adversarial Network (GAN), showing one possible approach for taking a corrupt/noisy image and estimating the clean, original image.

**Introduction**

Computer vision enables computers and systems to derive meaningful information from visual inputs (images/videos). A driver for the growth of computer vision based applications is the flood of visual information flowing in from smartphones, security systems, traffic cameras, etc. Computer vision tasks include image classification, object detection, object tracking and image retrieval.

Images are often degraded during data acquisition. This affects the validity of image information and later on, image processing. The purpose of image restoration is to estimate the original image from the degraded data. The concerns of the image restoration are the removal/reduction of degradations. This means it deals with the modelling of degradations such as blur and noise. Then the inverse process is applied to reconstruct the image.

**Methodology**

**Random Jittering**

When learning from a small dataset, an over-fitted model is a likely outcome (the model performs good on the training data but this performance does not translate to the test data as the network has simply memorised all training samples). A solution to this is random jittering to the input, it adds noise to the data (there is a slight shifting of rows/columns of image data relative to the neighbouring rows/columns). Adding noise means that the network is less able to memorise training samples. It also expands the size of the training dataset, each time a training sample is exposed to the model, random noise is added to the input variables making them different every time.

**GAN**

Generative models use an unsupervised learning approach. A GAN is a model in which two neural networks (the generator and discriminator) compete with each other to become more accurate in their predictions. The generator is a Convolutional Neural Network (CNN) and discriminator Deconvolutional Neural Network (DNN). The goal of the generator is to manufacture outputs that could be mistaken for real data. These outputs serve as negative training examples for the discriminator. The goal of the discriminator is to identify which outputs it receives have been artificially created. It penalises the generator for giving implausible results.

The Pix2Pix GAN is a generator model for performing image-to-image translation trained on paired examples. It is a conditional GAN (cGAN) where the generation of the output image is conditional on an input, in this case, a source image. The benefit of this model is it is relatively simple and capable of generating large high-quality images across a variety of image translation tasks.

The generator and discriminator networks are trained in a similar fashion to ordinary neural networks (the weights are randomly initialised, a loss function and its gradients with respect to the weights are evaluated and the weights are iteratively updated through backpropagation). The generator progressively becomes better at creating outputs that look real, while the discriminator becomes better at telling them apart. The process reaches equilibrium when the discriminator can no longer distinguish real outputs from fakes.

**Generator**

U-Net

The architecture of the generator is a modified U-Net. The intent of U-Net is to capture both the features of the context as well as the localization. The main idea of the implementation is to utilise successive contracting layers, which are immediately followed by the upsampling operators for achieving higher resolution outputs on the input images. This architecture was designed to solve the task of image segmentation. The task in image segmentation is to take an image and divide it into several smaller fragments. For image segmentation tasks, another essential requirement is the use of masks (a binary image consisting of zero or non-zero values).

Encoder/Decoder

There are skip connections between the encoder and decoder. This is vital to preserve the loss from the previous layers so that they reflect stronger on overall values. They also produce better results and lead to faster model convergence. Each block in the encoder is Conv2D -> Batchnormalization -> LeakyReLU. Each block in the decoder is Conv2DTranspose -> Batchnormalization-> Dropout (applied to the first 3 blocks) -> ReLU.

Conv2D

The first layer of a CNN is always a convolutional layer. It applies a convolution operation to the input, passing the result to the next layer. A convolution converts all the pixels in its receptive field into a single value. When applying convolution to an image, we decrease the image size as well as bring all the information in the field together into a single pixel. The final output of the convolutional layer is a vector.

Batchnormalization

Normalisation is a pre-processing technique used to standardise data (place different sources of data inside the same range). Not normalising the data can make it drastically harder to train and decrease the learning speed. Batch normalisation is done between the layers of a neural network.

LeakyReLU/ReLU

Leaky Rectified Linear Unit (Leaky ReLU) is an activation function (transforms inputs to outputs within a certain range) based on a ReLU, but it has a small slope for negative values instead of a flat slope. It is continuous at zero. The slope coefficient is determined before training (not learnt during training). This activation function is popular in tasks where one suffers from sparse gradients as in when training GANs.

Conv2DTranspose

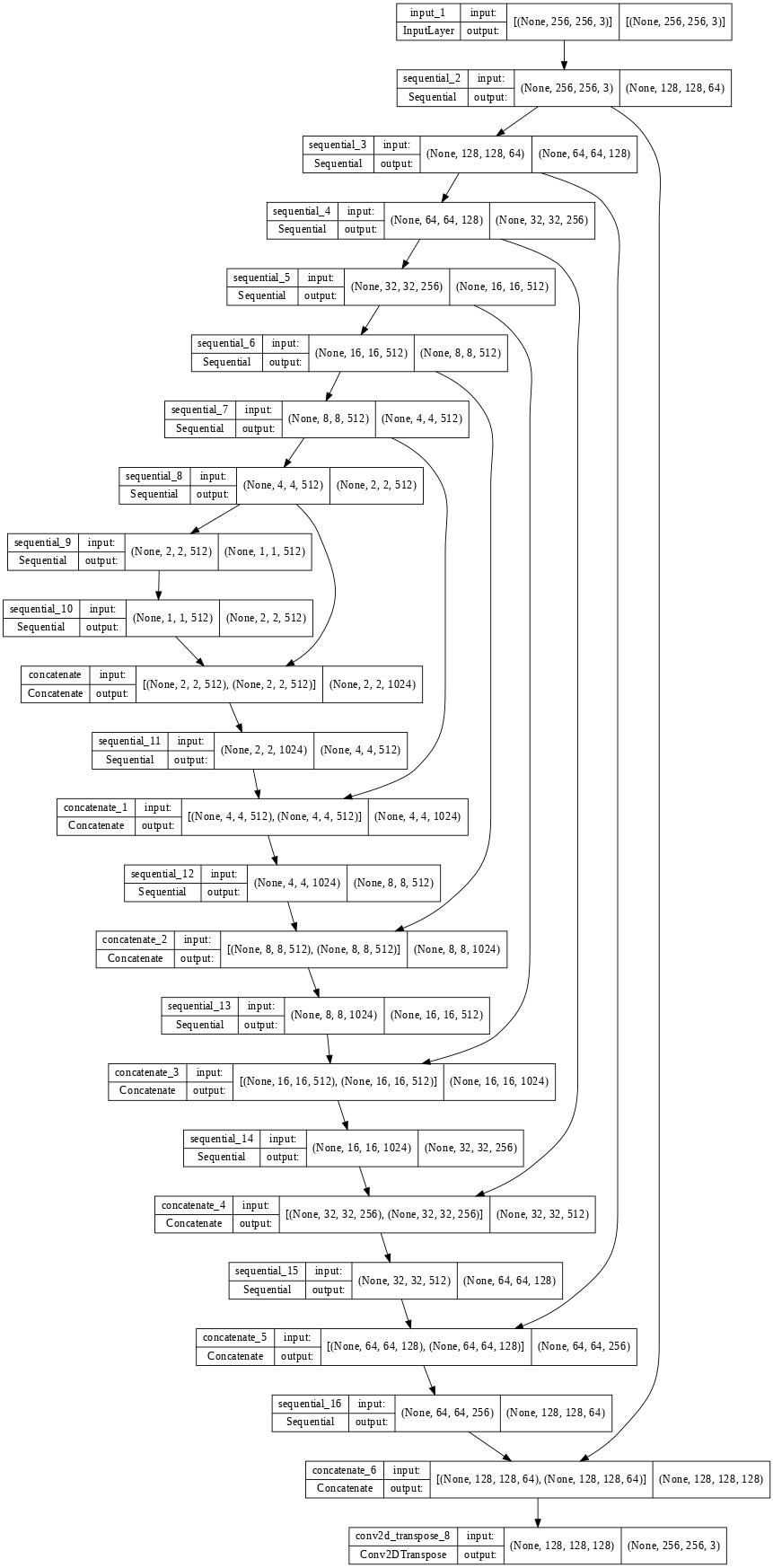
A transposed convolutional layer is carried out for upsampling. It generates an output feature map that has a spatial dimension greater than that of the input feature map. The layer also learns how to fill in details during the model training process.

Dropout

The Dropout layer is a mask that nullifies the contribution of some neurons towards the next layer and leaves all the others unmodified. Dropout layers are important because they prevent overfitting on the training data. If they aren’t present, the first batch of training samples influences the learning in a disproportionately high manner. This prevents learning of features that appear only in later samples. Dropout may be implemented on any or all hidden layers in the network as well as the visible or input layer. It is not used on the output layer.

Loss

The generator loss is calculated from the discriminator’s classification. It gets rewarded if it successfully fools the discriminator and gets penalised otherwise. The generator loss is a sigmoid cross entropy loss of the generated images and an array of ones. A sigmoid cross entropy loss is a sigmoid activation plus a cross-entropy loss which is independent for each vector component, meaning that the loss computed for every CNN output vector is not affected by other component values. The mean absolute error between the generated image and the target image is also included in the loss, allowing the generated image to become structurally similar to the target image.



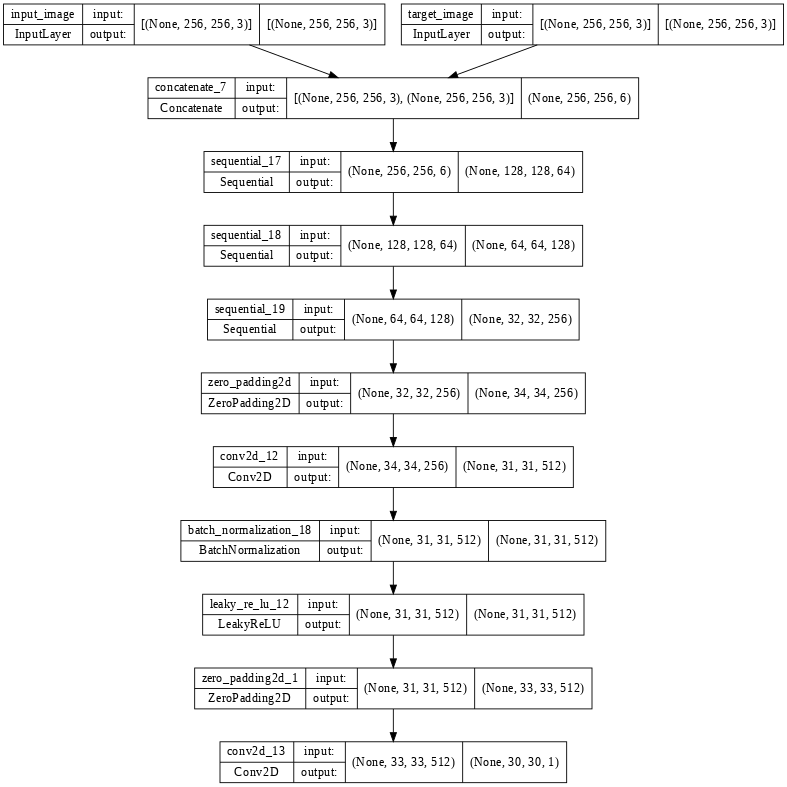
**Discriminator**

PatchGAN

The discriminator is a patchGAN. It tries to classify if each patch in an image is real or fake. Then it is run convolutionally across the image, averaging all responses to provide the ultimate output. Such a discriminator effectively models the image as a Markov random field, assuming independence between pixels separated by more than a patch diameter. Each block in the discriminator is Conv2D -> Batchnormalization -> LeakyReLU.

Loss

The real loss is a sigmoid cross entropy loss of the real images and an array of ones (since these are real images). The generated loss is a sigmoid cross entropy loss of the generated images and an array of zeros (since these are fake images). The total loss is the sum of the real loss and generated loss.



**Conclusion**

The model can be improved by increasing the number of epochs. This will increase the number of times the weights are changed in the neural network and the curve will go from underfitting to optimal. Also improvements in the runtime can be made by using Tensor Processing Units (TPUs). They are Google's custom-developed application-specific integrated circuits used to accelerate machine learning workloads.

**Bibliography**

# Brownlee, Jason. “How to Implement Pix2Pix GAN Models From Scratch With Keras”. Machine Learning Mastery. 31 July 2019. 15th August 2022. <<https://machinelearningmastery.com/how-to-implement-pix2pix-gan-models-from-scratch-with-keras/>>

* [arXiv:1611.07004](https://arxiv.org/abs/1611.07004) [cs.CV]

https://doi.org/10.48550/arXiv.1611.07004

|  |  |
| --- | --- |
|  |  |
|  |  |
|  |  |