258 Deep Learning

Lab 1- Report

Submitted by:

Shalabh Neema
(014546259)

Model Architecture of a GAN

Generative Adversarial Networks (GAN) is an emerging class of Deep Learning developed by Ian GoodFellow. It has been widely used in many fields such as Fashion , Design, art, science video games. The core logic of GAN to train two neural networks pitted against each other like a competitive game. The response from the other network will help in improving the weights for the next iteration. These two networks which are major components are Generative network and Discriminative network.

Generative Network will initially start with the noise data and generate the sample image after passing through multiple convolutional layers. The image is then passed to the discriminative network which will determine whether the sample image is from the training data. The loss is calculated on both the networks which will help in adjusting the weight of the networks for next iteration.The objective of the GAN is to fool discriminative networks into believing that the generated image belongs to training data.The approved imaged from discriminative network will be the image that is not existed but it is resemble like a real image.

There are many applications in which GAN can be utilized. We can create new images by combining specific features of many images. For example, a style feature can be used from one pool and can be integrated with an object from another pool. The tutorial of Neural Style transfer has beautifully utilized this concept . They combined the colourful background with the image of dog.

Overview of the GAN utilized in step 2

In step 2, SVHN dataset has to be utilized for generation of new images. The training dataset consists of more than 70000 images which has been collected from the vehicle number-plate. First , the data was converted into the desired shape of 32*32*3 for the processing. After that, images are normalized for the faster computation for our models. Both the Discriminative network and Generator network have been modified from the code used in Step 1. Additional layers has been added to the networks for generating accurate images.

Following are the summary of Models:

```
Model: "sequential_2"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
dense_2 (Dense)              (None, 16384)             1638400
_________________________________________________________________
batch_normalization_3 (Batch (None, 16384)             65536
_________________________________________________________________
leaky_re_lu_5 (LeakyReLU)    (None, 16384)             0
_________________________________________________________________
reshape_1 (Reshape)          (None, 8, 8, 256)         0
_________________________________________________________________
conv2d_transpose_3 (Conv2DTr (None, 8, 8, 128)         819200
_________________________________________________________________
batch_normalization_4 (Batch (None, 8, 8, 128)         512
_________________________________________________________________
leaky_re_lu_6 (LeakyReLU)    (None, 8, 8, 128)         0
_________________________________________________________________
conv2d_transpose_4 (Conv2DTr (None, 8, 8, 128)         409600
_________________________________________________________________
batch_normalization_5 (Batch (None, 8, 8, 128)         512
_________________________________________________________________
leaky_re_lu_7 (LeakyReLU)    (None, 8, 8, 128)         0
_________________________________________________________________
conv2d_transpose_5 (Conv2DTr (None, 16, 16, 64)        204800
_________________________________________________________________
batch_normalization_6 (Batch (None, 16, 16, 64)        256
_________________________________________________________________
leaky_re_lu_8 (LeakyReLU)    (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_transpose_6 (Conv2DTr (None, 32, 32, 3)         4800
=================================================================
Total params: 3,143,616
Trainable params: 3,110,208
Non-trainable params: 33,408
```

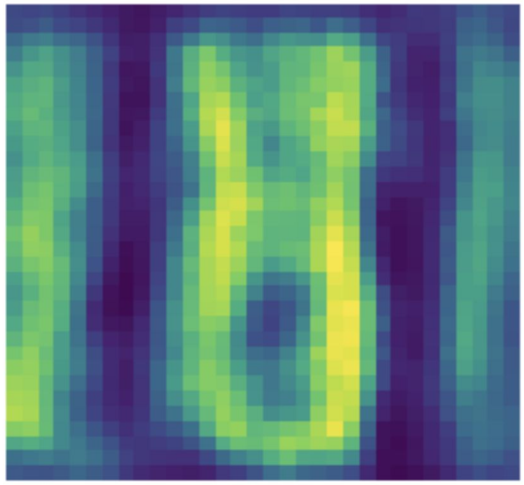
Generative Network

```
Model: "sequential_3"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_2 (Conv2D)            (None, 16, 16, 64)        4864
_________________________________________________________________
leaky_re_lu_9 (LeakyReLU)    (None, 16, 16, 64)        0
_________________________________________________________________
dropout_2 (Dropout)          (None, 16, 16, 64)        0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 16, 16, 128)       204928
_________________________________________________________________
leaky_re_lu_10 (LeakyReLU)   (None, 16, 16, 128)       0
_________________________________________________________________
dropout_3 (Dropout)          (None, 16, 16, 128)       0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 8, 8, 128)         409728
_________________________________________________________________
leaky_re_lu_11 (LeakyReLU)   (None, 8, 8, 128)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 8, 8, 128)         0
_________________________________________________________________
flatten_1 (Flatten)          (None, 8192)              0
_________________________________________________________________
dense_3 (Dense)              (None, 1)                 8193
=================================================================
Total params: 627,713
Trainable params: 627,713
Non-trainable params: 0
_________________________________________________________________
```

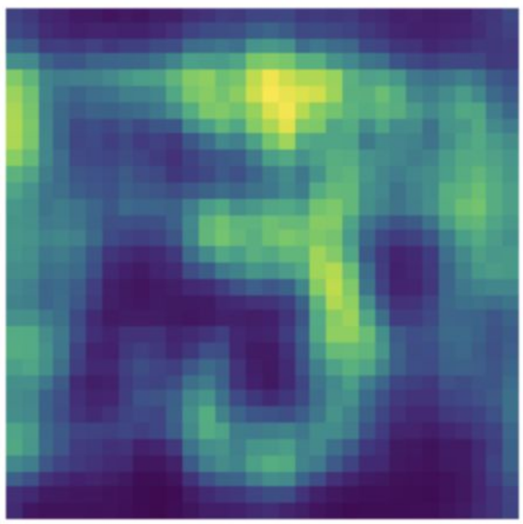
Discriminative Network

There were some challenges faced during the addition of new layers with respect to input shape. The strides and output shape has to be carefully equated with the surrounding layers.

Training has been performed on 50 epochs. Some of the outputs are clearly distinguishable but after a while, the clarity remains the same. Following are the few images:



(1,8,1)



(1,3)

Analysis of Research Papers

In the first paper “Generative Adversarial Nets” , authors have proposed a method to generate a non-existing image of a certain domain. The methodology can let us combine styling features to alter the visual appearance. Although we are able to generate novel images for a training set, this approach is limited in transforming an image of a certain domain to a target domain.

In the second paper “UNSUPERVISED CROSS-DOMAIN IMAGE GENERATION”, authors propose a method to generate a new image from one domain to another. For the images to be similar from the source domain, they develop a network which is the composition of input function and learned function. A GAN network is used to generate the novel images of the target domain irrespective of the input image domain. A loss will try to lower down the difference between the generated image by the GAN and first composition network. Another loss has been used as a regulizer and encourages GAN to produce images of the target domain. In the paper, authors have tried to produce an emoji based on the facial image of a person.