Generating Cross Domain Models using Generative Adversarial Networks

G.Victo Sudha George, A. R. Raj Ganesh Arun, S.Geetha

Abstract - Normally an average human can understand the relationship between different fields and domains easily with limited information exposure. On the other hand even a modern day computer is pretty bad in understanding relationship between different domains and fields which can be changed using a neural network called 'GAN' (Generative Adversarial Networks).GAN is a generative network that can generate data that we have ever seen before by Leveraging the generative nature of GAN . All of this is done using generator and discriminator in Generative Adversarial Networks. Two different datasets are trained on discriminator and generator. After training it generates new data which has attributes of both the datasets. But the biggest problem we face is with the training phase. Training in GAN is a like MinMax game. It doesn't know when to stop. This is due to the fact that both Generator and discriminator try to overpower each other which leads to a lot of loss and error rate as it keeps on training. In this paper we have found an ideal state for the GAN.

Key words- Neural Network, Deep Learning, GAN-Generative adversarial networks, GMM- Gaussian mixture model, Cross Domain Model

I. INTRODUCTION

Neural networks are keep on out performing all other machine learning algorithms. It can perform supervised learning and unsupervised learning tasks. In Supervised learning inputs are mapped to output ie. Neural Network learn by mapping but Unsupervised learning is based on understanding the structure of the data. For example, in Unsupervised learning if the Neural network can able to learn and understand the structure of data. Let's say if the data is music, it will be able to categories similar type of music even it will be able to generate new type of music.

Deep Learning is a sub-category of machine learning where if the availability of data is huge and there is plenty of computing power then deep learning can perform seamlessly [1]. Deep Learning is comprised with several Neural Network algorithms and optimization techniques. Most commonly used optimization technique is Back propagation algorithm. Neural networks usually takes plenty of time to train and learn from the data and tries to understand the structure of the data. Hyper parameters are the import functions which determines the time and accuracy of the neural network. Neural Network uses several epochs to learn and what happens is, in each epoch the model becomes better and better. What's happening here is its learning from its own mistake and getting better each time.

Revised Manuscript Received on July 05, 2019.

Dr.G.Victo Sudha George, ,Department of CSE, Dr.M.G.R. Educational and Research Institute,Chennai,Tamil Nadu, India.

A.R.RajGanesh Arun, Department of CSE, Dr.M.G.R. Educational and Research Institute,Chennai,Tamil Nadu, India.

Dr. S.Geetha, Department of CSE, Dr.M.G.R. Educational and Research Institute,Chennai,Tamil Nadu, India.

Generally, every model which is built will be either Deterministic or Probabilistic. In real world most of the models are Probabilistic. The models we built will have uncertainty. In Most of the bayesian based models new hypothesis can be introduced into the system anytime. Generating new type of data which is unique requires a special kind of Neural Network which is called as GAN (Generative Adversarial Networks)[2][3]. GAN can able to generate new unseen and unique data [4].

Every GAN has two individual networks Generative network and Discriminative network. Generative network which keeps on generating new unique data which it learned from the training dataset. GAN is not the only generative type of Neural Network model Autoencoder and Variational Autoencoder is also a generative model. Generative network is one of the network which generate another network called Discriminative network which will discriminate the generated outcomes. So the job of the generator is to generate stuff and discriminator network validates the outcome of the generator network.

Hyperparameters are another important aspect of the network. They determine how the network trains and defines its structure[5]. Hyperparameters includes Learning rate, epochs, activation function and batch size. It also includes the number of layers included in the network which defines number of hidden layers. Other concepts include overfitting and underfitting.

In order to address the overfitting and underfitting problem, we have to use a technique called dropout[6].When there is so many layers or too much training data, our network becomes to generalized and won't be able to classify new data that is called overfitting. Overfitting is serious problem that can be addressed using dropout technique. Where in dropout we randomly turn a neuron off in certain layers in our network so we will be able to see how it performs. Underfitting is a problem arises when the model designed doesn't fit our data properly or little training data so the outcome won't be accurate. Application of GAN is diverse . Few are listed below.

Music Generation :Using this same exact architecture we will be able to mod this system to get to working into generate anything from music ,images and videos. Theoretically if the dataset has Hip-hop music and the model trains on it and it will be able to produce new unheard hip-hop music[7].

Gaming Environment Simulation Creation :The model has the capabilities of training on two different types of environments and will be able to generate a new environment for simulation . The new environment will be mix of two training dataset.

New environment can be used to run simulations or it can be used as Gaming environment [8].

Medical field: Rather than doing actual experiments in the lab, taking the chemical properties in a drug and crossing it with other drug running a simulation .Given a disease and its symptoms and drug and its effects we can create new drugs that can cure diseases without using actual chemicals or testing it out on real people and risking their lives [9].

II. EXISTING MODEL

Model designed here will be using Slightly moded GAN with which we will be able to generate anomal data. There will be four generator networks and two discriminator networks. Generator and discriminator are adversarial networks where each network is adversarial. Generator denoted as G and discriminator as D. Most cases they are non-linear functions. What the model is capable of doing takes a image from one set and cross it with another set and will able to produce crossed anamol image. The model follows the conditional Probability distribution. The generator takes in the input image and encodes it and while decoding, it conditions on the other image and produces the cross- domain image which belongs to both the data sets. The generators are encoder and decoder pair.

A. Architecture for Moded GAN

The architecture of GAN is depicted in Figure 1. There are two domains names A and B. Here Generator network-1 is denoted as GAB and Generator network-2 is denoted as GBA. LDA and LDB are discriminators in the model. There are also two more generators which makes sure the model is backward compatible .Generators are capable of taking image of size 64*64*3 and it can feed through its encoder-decoder pair. Encoder of each generator is made of convolution layers 44 and followed by leaky ReLU. Decoder part is made of deconvolution layers with 44 filters and also followed by a ReLU. The convolution and deconvolution layers varies from four to five even six sometimes depending on the domain. The discriminator is almost as similar as to the encoder part of the generator. It is composed of convolution layers and leaky ReLUs. The discriminator has an extra convo layer with sigmoid activation function to output a binary value between 0 or 1.

B. Reconstruction Loss Function

The generator GAB receives two different loss. The reconstruction loss LCONSTA that measures how well the original input is reconstructed after a sequence of Learning and a regular GAN generator loss LGANB that measures how realistic the generated image is in domain B. The discriminator receives the standard GAN discriminator loss.

III. PROPOSED MODEL

The proposed model consists of 4 generators and 2 discriminator where generator will able to generator new data after getting trained on training dataset and test dataset dev dataset is optional.

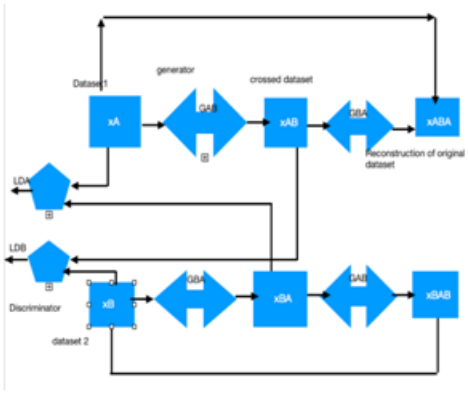

Fig.1 *Architecture of Moded GAN*

Discriminator trains on the same set of data as the generator and then it will able to identify the fake from the real data. The activation function which will be used here is ReLU activation function . The output will be either (0,1) so it will be binary which is real or fake. The back propagation technique is used to update the weights and learn.

A. Gaussian Mixture Model

The Gaussian Distribution Univariate or Multivariate approaches are capable of model many of the datasets. So it is quite common to assume that the clusters come from different Gaussian Distributions. Or in other words, it is tried to model the dataset as a mixture of several Gaussian Distributions. This is the core idea of this model. There are K cluster had it been only one distribution, they would have been estimated by maximum-likelihood method. But since there are K such clusters and the probability density is defined as a linear function of densities of all these K distributions.

B. Batch Normalization

Every batch of data is normalized to all convolution and deconvolution layers except the first and the last layers, weight decay regularization coefficient. A main insight we rely on is that it should be possible to represent all images of one domain to be representable in another domain.

IV. IMPLEMENTATION AND EXPERIMENTAL RESULT

We used Tensorflow as a deep learning framework. As a backend, we used basic numpy for matrix operations and mathematical operations. Also we will be using matplotlib for the visulation. We used tf slim library for training the model. The model will have 4 generators and 2 discriminators. Each generator and discriminator has 2 layers and 256 hidden layers. The Gaussian mixture model is an example of probabilistic model .It considers that all the data points are obtained from a mixture of a finite number of Gaussian distributions with unknown parameters. It defines the means and standard deviations and variance.

The datasets are X,Y are defined where Z is crossed data that is generated by our GAN. Activation function is used here is ReLU activation function. After the weights are updated and value is optimized after going through activation function and the generator value is feed into discriminator for the validation. discriminator is trained on the same set as the generator it will be able to classify the real or fake of the value generated by the generator and discriminator which will be having sigmoid as the activation function which ouputs binary values (0,1) real or fake .The loss function will able to determine how far the value is off from the real value. Y =mX+b ,the loss function is decided for the training data of one set ,but the error rate is measured for the whole set of the batch of training data after one epoch.

A. Experimental Result On Toy Dataset

Epoch rate is defined as epoch = 1000 each training data goes through 1000 times until it is optimized .batch size is defined as 64,dataset size = 512 ,input dimensions= 2 latent dimensions= 2 .

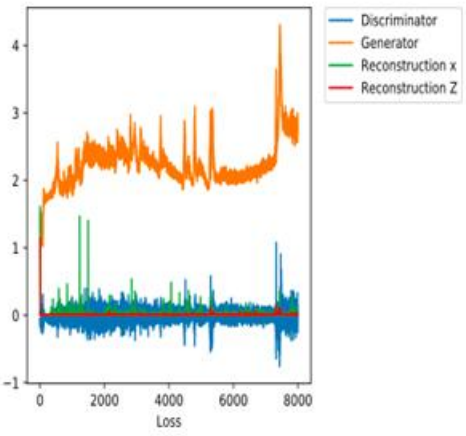

Fig.2-Gaussian mixture -1

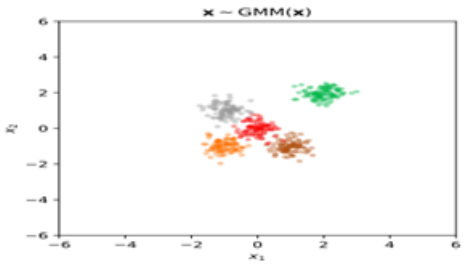

Fig.3-Gaussian mixture -2

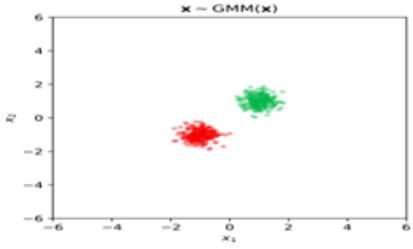

Fig.4-Gaussian mixture outcome

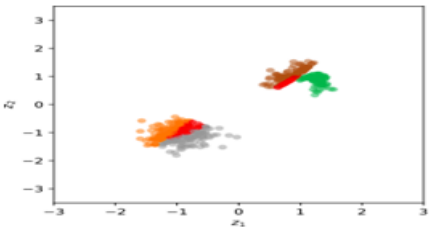

Fig.5. Loss Graph for GMM

The loss function is determined separately for generator and discriminator . Even loss is calculated separately for each generator and summed later.

After the given image is validated, the next generator will try its best to reconstruct the given original image and it validates using the reconstruction loss.

The graph of the final loss functions are generated at the end which is given in Figure 5. Here we can see the different loss of generator and discriminator where the reconstruction loss is also generated.

B. Experiment With Real Dataset

Since it will take massive amount of computing power to train the GAN its virtually impossible to train a GAN locally unless you have a good level GPU from NVIDIA. Or we can use Google Cloud or FLoyd hub or other online cloud services. We can Build our model Locally and use cloud to Train our model . What happens here is our local directory is connected with Google cloud and after following the google cloud installation steps.

The dataset that we will be using is Two different Types of Dataset one is Shoes and other is Handbags. Using this moded GAN we can condition one dataset with another. GANs training are usually better with Higher power GPUs so we can use Google Cloud. Usually training takes anywhere between 8-10 hours.

We can validate the Training results using Validation dataset. After exporting the model, upload the exported model to Cloud Storage Run the command to upload your saved model to bucket in Cloud Storage:

```
SAVED_MODEL_DIR=$(ls ./your-export-dir-base | tail -1)
gsutil cp -r $SAVED_MODEL_DIR gs://your-bucket.
```

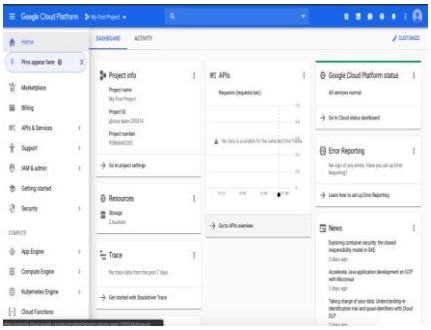

Fig.6.GCP Environment

C. Using Tensorflow in Cloud

First we have to setup the Tensorflow environment which is dip by selecting the version of tensorflow,REQUIRED_PACKAGES = ['tensorflow>=1.0']. Building a TensorFlow binary as a custom package is a more complex approach, but the plus is that we can utilize the most latest TensorFlow updates when training our model.By modifying the code we can Use TPU in the cloud , we are not using TPU though , but TPUs are so much faster than GPUs.

D. Datasets with Tensorflow

We Import our Data first and then we need some data for the dataset .From numpy we have a numpy array and we will pass it to Tensorflow. We can also pass more than one numpy array, when we have a couple of data divided into features and labels.Using tensors we can initialise our dataset with some tensor , then we create a placeholder,this will be useful when we dynamically change the data inside the Dataset, we can also directly read a csv file into a dataset.we now can easily create a Dataset from it by calling tf.contrib.data.make csv_dataset .

V. CONCLUSION

GANs are becoming popular every day where the possibility is unlimited. Before GAN-era machine can detect images, voices and music and do some ruled actions .On the other hand it was not able to dream or create new things. In this paper we would like to conclude GAN is one step closer to creating new things without any human interference. Using GAN machine will be able to dream and create environments of its own and even run simulations of its own. With the possibility of having GAN machine will be able to generate new things we never seen before and even Google's AI can do deep dreams.

REFERENCES

1. Jonathan Long, Evan Shelhamer, and Trevor Darrell, "Fully convolutional networks for semantic egmentation," in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 3431– 3440
2. Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza,Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio, "Generative adversarial nets," in Advances in neural information processing systems, 2014, pp. 2672–2680
3. Taeksoo Kim, Moonsu Cha, Hyunsoo Kim, Jungkwon Lee, and Jiwon Kim, "Learning to discover cross domain relations with generative adversarial networks," arXiv preprint arXiv:1703.05192, 2017
4. Ming-Yu Liu and Oncel Tuzel,"Coupled generative adversarial networks," in Advances in neural informationprocessing systems, 2016, pp. 469–477
5. Tahmassebi A, Gandomi AH, Fong S, Meyer-Baese A, Foo SY (2018) Multi-stage optimization of a deep model: A case study on ground motion modeling. PLoS ONE 13(9): e0203829
6. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever and R. Salakhutdinov, "Dropout: A Simple Way to Prevent Neural Networks from Overfitting," Journal of Machine Learning Research, vol. 15, pp. 929-1958, 2014
7. Olof Mogren. C-RNN-GAN: Continuous recurrent neural networks with adversarial training. arXiv preprint arXiv:1611.09904, 2016
8. Giacomello, E.; Lanzi, P. L.; and Loiacono, D. 2018. Doom level generation using generative adversarial networks. arXiv preprint arXiv:1804.09154
9. Elton, D. C.; Boukouvalas, Z.; Fuge, M. D.; Chung, P. W. Deep learning for molecular generation and optimization-a review of the state of the art. arXiv preprint arXiv:1903.04388 2019