Context Encoders

Feature Learning by Inpainting

PROJECT BY

Naman Jain (2020201080)
Archit Gupta (2020201075)
Ayush Khasgiwala (2020201088)

Our Aim

Our objective is to fill the missing part of any image given as an input using Context Encoders and Generative Adversarial Networks (GAN).

Input image with a missing part

Output filled image

About

It is an visual feature learning algorithm driven by context based pixel representation.

➔ **Idea**

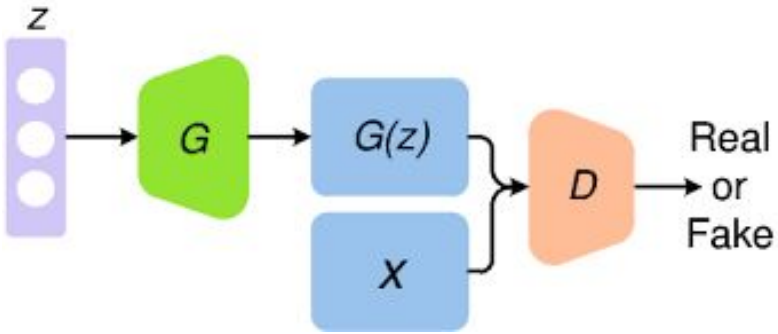
By analogy with auto-encoders, we worked on Context Encoders : a convolutional neural network trained to generate the contents of an arbitrary image region conditioned on its surroundings.

Image generation will be done through GAN.

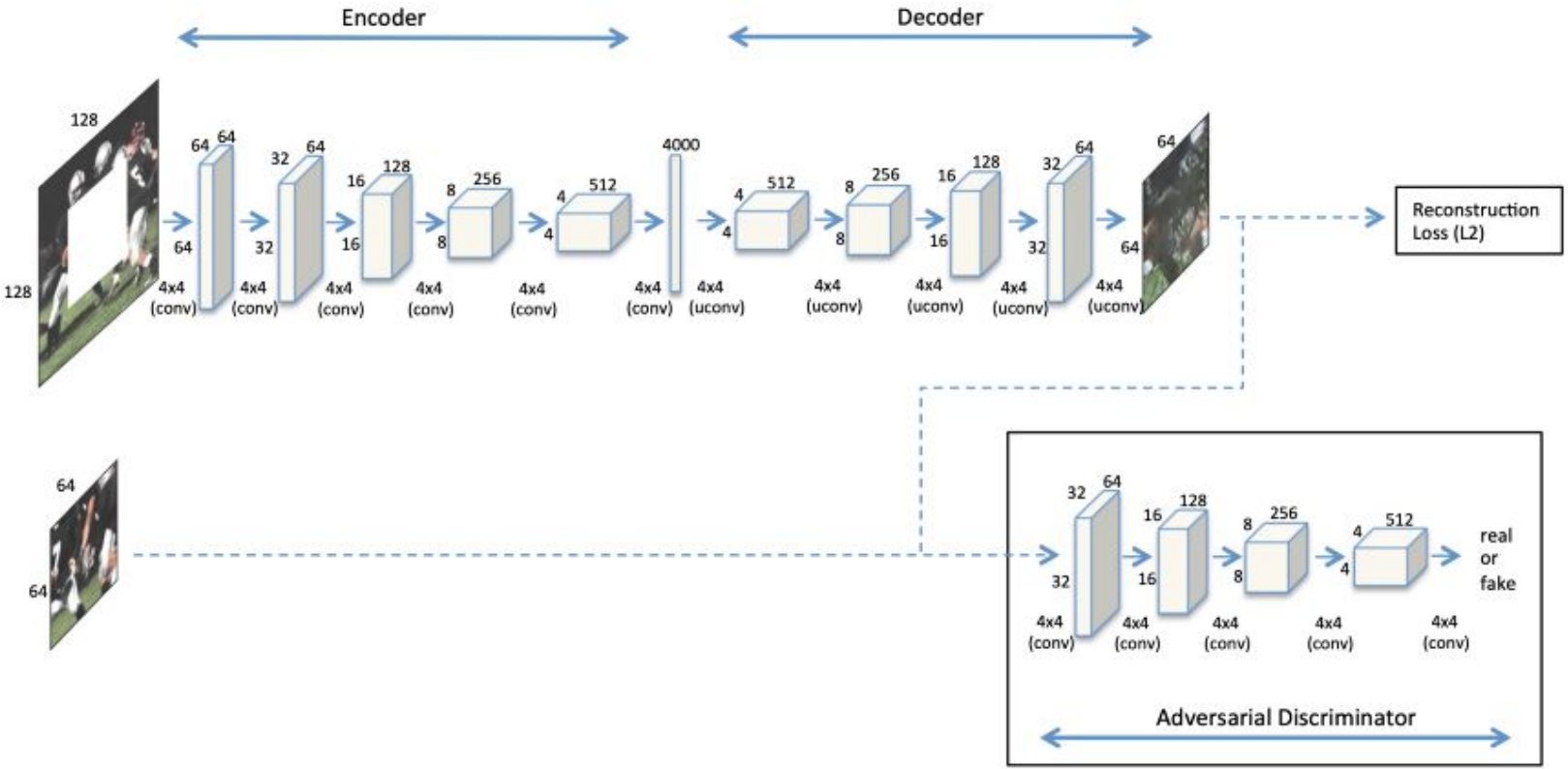
Generative Adversarial Networks

GANs are a clever way of training a generative model by framing the problem as a supervised learning problem with two sub-models: the generator model that we train to generate new examples, and the discriminator model that tries to classify examples as either real (from the domain) or fake (generated).

- Deep Convolutional GAN will be used for generating images.
- Generators are Deconvolutional neural networks. (Auto Encoders)
- Discriminators are CNNs.



Overall Architecture



Dataset

CIFAR 10

- First we trained our model on CIFAR 10 dataset , on 50,000 iterations,
- But the results were very blurry and less detailed as the input images were of low resolutions (32 X 32)

Paris Street View Dataset

- After the CIFAR 10 dataset, we tried to train our model using the dataset mentioned in the research paper,
- But unfortunately the images were of very high resolutions which required huge amount of training time
- It was not possible to be trained with the constraints of google colab

Intel Multi Class Dataset from Kaggle

- This dataset consist of the images from FOREST, MOUNTAINS, SEA, GLACIER
- The images were of 150 X 150 resolutions and were perfect for our project and were in accordance with our available resources.
- Trained the model on 10,000 epochs

Paris Street View Dataset

537 x 936 pixels

Intel Multi Class Dataset

150 x 150 pixels

Cifar 10

32 x 32 pixels

Workflow

- ❑ Data Curation
- ❑ Data preprocessing
- ❑ Masking for Patch Generation
- ❑ Generator Architecture
- ❑ Discriminator Architecture
- ❑ Combined Compilation & Training loop
- ❑ Save incremental results of training loop
- ❑ Testing loop

Experiments

- We tried to train our model on 3 datasets as mentioned before.
- We experimented with layers of neural network.
- We experimented with the architecture .
- We tried to train our model with and without normalization and got better results with normalization.
- We also experimented with different masking techniques to make our model robust.
- We experimented with the number of epochs and the training and batch size.

Results on CIFAR 10

Results on Intel Multi Class Dataset

Original Image

Masked Image

Generated Image

Difficulties Faced

- Selection of dataset according to the requirements and available resources.
- Creation of training dataset: we generated our training dataset by applying random masking technique on given input images.
- Initially training part was time consuming, so we divided the training data into batches.
- Initially, we trained our model without any normalization which yielded in unexpected output, so we found out that the solution to our problem was to normalize our input images.

Contributions

Ayush Khasgiwala
2020201088

- Data Curation
- Data preprocessing
- Generator architecture

Naman Jain
2020201080

- Data Curation
- Combined Compilation & Training loop
- Save incremental results of training loop

Archit Gupta
2020201075

- Masking for patch generation
- Discriminator architecture
- Testing Loop

PS : We 3 have equally contributed towards experimentations of the model on various hyper-parameters and datasets!

Thank You!

Team A_{yush} A_{rchit} N_{aman}