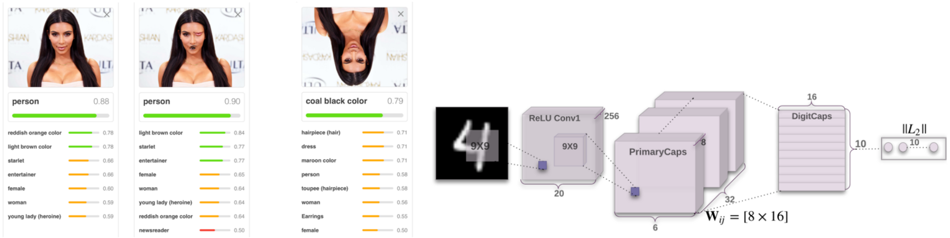
2D GANs with Capsule Networks

Mansi Mane, Snigdha Purohit, Ziqiang Feng
Carnegie Mellon University
Pittsburgh, PA 15213
mmane@andrew.cmu.edu, snigdhap@andrew.cmu.edu, zf@andrew.cmu.edu

1 Introduction and Motivation

Convolutional Neural Networks (CNNs) have been the state-of-the-art approach to classify images. However, there are some issues associated with CNNs. They work by accumulating sets of features at each layer. It starts of by finding edges, then shapes, then actual objects. However, the spatial relationship information of all these features is lost.

Figure 1a demonstrates an example relating to the flaw of CNNs. CNNs classify an entity having two eyes, 1 nose and 1 mouth as a human, however they cannot differentiate between the images having facial components in the wrong places. In addition to being easily fooled by images with features in the wrong place a CNN is also easily confused when viewing an image in a different orientation. A massive drop in performance by simply flipping Kim upside down. [2]



(a) Flaw in image classification with CNNs (b) CapsNet Architecture[3]

Figure 1: Flaws of CNN and CapsNet architecture

Hence, internal data representation of a CNN does not take into account important spatial hierarchies between simple and complex objects.[3] To take into account these spatial relationships, Capsule Networks were proposed by Professor Geoffrey Hinton where spatial relationships are explicitly modeled. The paper by Prof. Hinton that uses this approach was able to cut error rate by 45 percent as compared to the previous state of the art. Figure 1b shows the CapsNet architecture.

Along with the advent of CapsNet, General Adversarial Networks(GANs) have given excellent results in generating images. So, we decided to utilize the spatial relationship aspect of CapsNet to improve the image generation capability of a GAN which usually is trained by a CNN.

2 Existing methods

Generative models such as Restricted Boltzmann Machines(RBM), Deep Boltzmann Machines(DBM) and simple DCGANs have been employed for generating images. [4] proposes the DCGAN framework which is the latest generative model for image generation.

16824 Class Project (2018).

3 Problem statement

We propose to improve the image generation capability of DCGAN by employing a Capsule network instead of CNN for training the generator network in the DCGAN. We hypothesize that meaningful vector representation of the objects generated by capsule networks, would help GANs generate more realistic results and generate objects with better structural information.

4 Proposed Approach

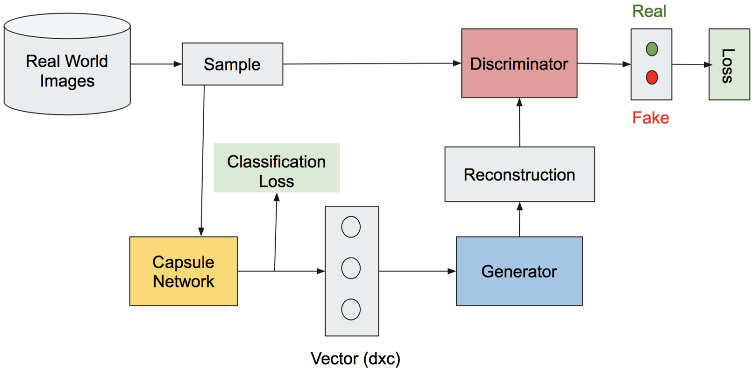


Figure 2: Proposed Architecture

We randomly sample one image from the dataset and give to capsule network which generates vector representation for given image. For K number of classes and D dimensional capsule, the vector generated from capsule network is KxD. We train the capsule network with margin loss as given by [1].

$$L_k = T_k max(0, m^+ - ||\mathbf{v}_k)^2 + \lambda(1 - T_k)max(0, ||\mathbf{v_k}|| - m^-)^2 \tag{1}$$

where $T_k = 1$ iff a digit of class k is present and $m^+ = 0.9$ and $m^- = 0.1$. The λ down-weighing of the loss for absent digit classes stops the initial learning from shrinking the lengths of activity vectors of all digit capsules.[1]

The same vector representation is given to generator which reconstructs the image from the vector. Discriminator takes image generated from generator and tries to differentiate generator reconstructed image from real image. Thus discriminator is trained on binary classification loss for real vs fake (generator images) images.

The loss function for GAN is given by following equation:

$$\text{min}_G \text{max}_D V(D, G) = E_{x\ p_{data}(x)}[logD(\mathbf{x})] + E_{z\ p_z(z)}[log(1 - D(G(\mathbf{z})))] \tag{2}$$

In addition to this, we also add mean square reconstruction loss to generator.

5 Datasets

We test our proposed approach on CIFAR 10 and MNIST datasets. The MNIST dataset consists of 28 x 28 sized grayscale images of handwritten digits. The CIFAR-10 dataset contains 32 x 32 color images grouped into ten classes: airplane, automobile, cat, deer, bird, dog, frog, horse, ship and truck.

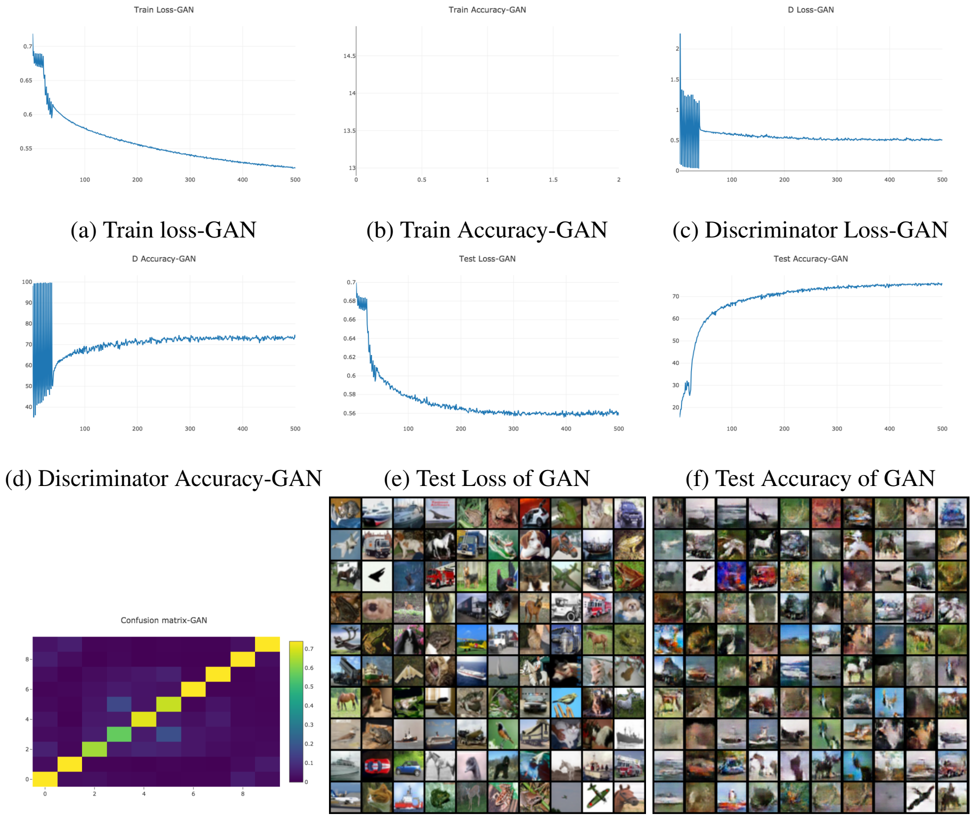
6 Experiments

The proposed architecture in Figure 2 was trained on CIFAR-10 dataset.

6.1 Results of proposed model

Figure 3 shows the training and test results of this model. The test accuracy was found to be 76.27 percent and the reconstruction quality was observed to be quite good.

Figure 4 shows the results from the other experiments or training attempts performed before deciding the final training strategy. Figure 4a shows the reconstruction results when the primary capsule layers used 32 capsules (instead of 64) in the 150th epoch due to time limit. Figure 4b shows the reconstruction results when the weight of the generator loss is set 10 times smaller than our final choice. Lastly, Figure 4c shows the reconstruction results when only L2 loss was used for training the generator. The results as we can see, were worst and blurry in this case.



(a) Train loss-GAN

(b) Train Accuracy-GAN

(c) Discriminator Loss-GAN

(d) Discriminator Accuracy-GAN

(e) Test Loss of GAN

(f) Test Accuracy of GAN

(g) Confusion matrix of GAN

(h) Ground truth GAN

(i) Reconstruction GAN

Figure 3: Training and Test results

6.2 Discussion and Analysis

Hence, after various experiments, we came up with a final strategy for training as follows:

1. When combining the generator's loss with the classification loss, we weight the generator's loss by 0.01.

2. We use Adam optimizer to train both CapsNet+Generator and Discriminator. Learning rate for former is 10^{-4}, for latter is 10^{-5}.

3. Start training the generator and discriminator after training CapsNet for about 20 epochs or when the classification accuracy of CapsNet reaches 20%.

4. Stop training generator with the generator's loss if discriminator accuracy is less than 50%. The classification loss is always used, though.

5. Stop training discriminator if discriminator accuracy exceeds 90%.

The conditional training of generator and discriminator described in (4) and (5) above results in the curves in Figure 3d. In the first few iterations, the accuracy of the discriminator bounce back and forth between near 100% and 50% (random). But at some point it stabilizes at a bit higher than 50%, meaning the generator has learned to generate non-trivial fake images. From then one, the discriminator's accuracy slows climbs up and plateaus around 72%. We believe this is a good sign that the discriminator and generator are improving each other hand-in-hand.

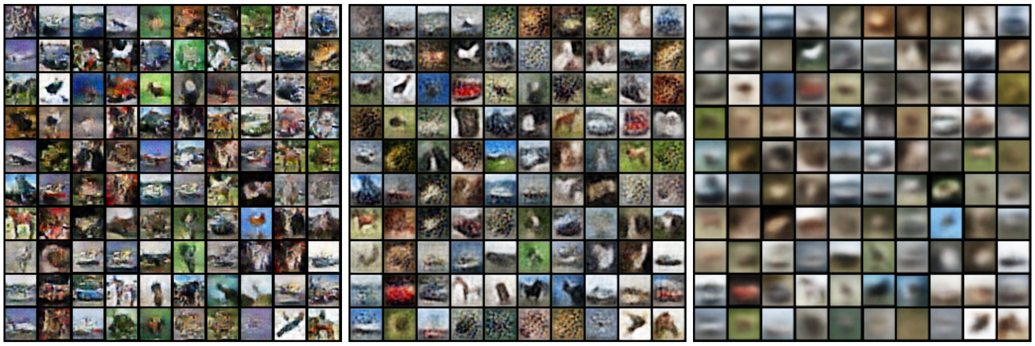
Challenges faced during experiments

There were some challenges faced when training the proposed architecture:

1) Dynamic routing in capsules can not use GPUs extensively due to EM algorithm and its branching. Due to this, it took almost 5-6 minutes per epoch while training on MNIST and 8 minutes (@32 capsules) to 14 minutes (@64 capsules) per epoch while training on CIFAR-10.

2) Tuning hyperparameters such as learning rate for generators and discriminators is tricky and needs to be set such that one cannot surpass the other.

3) We are using reconstruction loss as well as the general GAN loss for optimization for the generator and choosing the right balance between the weights of these losses is a challenge.



(a) 32 instead of 64 capsules used by Primary Capsule

(b) Weight of Generator loss 10X smaller

(c) Using only L2 loss to train generator

Figure 4: Experiment results before the final training strategy

References

[1] Frosst Nicholas Sabour, Sara and Geoffrey E. Hinton. Dynamic routing between capsules. In *Advances in Neural Information Processing Systems*, page 3859–3869. Curran Associates, Inc., 2017.

[2] Hackernoon article on CapsNet
`https://hackernoon.com/capsule-networks-are-shaking-up-ai-heres-how-to-use-them-c233a0971952`

[3] Understanding CapsNet: Article from Medium
`https://medium.com/ai%C2%B3-theory-practice-business/understanding-hintons-capsule-networks-part-i-intuition-b4b559d1159b`

[4] DCGAN Paper by Ian Goodfellow
`https://arxiv.org/pdf/1406.2661.pdf`