CS 394D Autumn 2020/2021 Coding Assignment 1
Data Efficient Deep Learning

Due Sunday September 19, 11:59pm
Submit by the **blackboard system**

By turning in this assignment, I agree with the KAUST student protocol and declare that all of this is my own work.

Requirements:

1. Put all files in a zip file and the file should be named LastnameFirstname_ReadingAssignment1.zip
2. It is an individual assignment, independent write up, and submission in your own hand is required for credit.
3. The work should be written in a *clear* way.
4. Late submissions will receive credit penalty.

Overview

In this assignment, we will be looking at data efficient deep learning. You will

1. Learn how to implement a Generative adversarial networks(GANs)
1) Learn the overfitting problems of GAN and study a variaty of ways to alleviate it
2) Implement several data augmentation methods such as cropping, flipping, scaling, color jittering and region masking
3) Analyze the learning performance for w/wo data differentiable augmentation (DiffAugmentation) on GAN models, and understand why adding DiffAugmentation can improve the performance
4) Show some visualization examples and feel the impact of w/wo DiffAugmentations

*Before you will start this project, please carefully read the paper "Differentiable Augmentation for Data-Efficient GAN Training" [1]. Paper link: https://arxiv.org/pdf/2006.10738.pdf
This paper is highly related to your coding project. Reading this paper will also help you quickly understand the related research problems from this domain and some successful solutions.

We will be working with CIFAR-10 dataset [2]. The CIFAR-10 dataset consists of 60000 32x32 colour images in 10 classes, with 6000 images per class. There are 50000 training images and 10000 test images. The dataset is divided into five training batches and one test batch, each with

10000 images. The test batch contains exactly 1000 randomly-selected images from each class. The training batches contain the remaining images in random order.

We will use the Big GAN model [3] as the backbone. Big GAN is a very successful generative model and can generate high-resolution, diverse samples from very complex datasets.

Code Overview

- Datasets.py: Contains code to download and preprocessing CIFAR-10 data
- BigGAN.py Contains the network architecture of Big GAN model
- Layers.py Contains various layers for the BigGAN model
- Losses.py Contains the loss functions
- Inception_tf.py Contains inception metrics for evaluation
- Utils.py Contains utility functions for logging and data loading
- Train_fns.py Contains the main lop of training different conditional image models
- Scripts folder Contains several scripts of running the whole code with some default arguments.
- Train.py Contains the specification of the configuration of the whole training run
- DiffAugment_pytorch.py is the place where you need to implement different augmentation methods.

Dependencies

Pytorch version = 1.4.0
Tensorflow = 1.14 or 1.15 with GPU support (for IS and FID calculation)
We recommend using 2 GPUs with at least 11 GB of DRAM or 1 GPUs with at least 22 GB of DRAM for training and evaluation.

*__Ibex user__: there is an additional file that contains the detailed steps to build up the running environment: **create_ibex_environment.txt**

Training

The following command will run BigGAN + DiffAugment on CiFAR-10 with 5 percent data:

```
CUDA_VISIBLE_DECIVES=0,1 bash scripts/DiffAugment-biggan-cifar10-0.05.sh
```

Evaluation

To evaluate a model on CIFAR-10, run the following command:

```
CUDA_VISIBLE_DEVICES=0,1 python eval.py --dataset=C10 --network=WHICH_MODEL
```

WHICH_MODEL specifies the path of a checkpoint contained the generator's weights, which is in the **weights** folder

Submission

To submit your assignment, submit one pdf report and your code into one zip file through blackboard, where the report will contain the answers to the deliverables listed below.

Problem 1: (coding + report) (40 points)

Implement different data augmentation methods in the file "DiffAugment_pytorch.py", which include color jitterig, translation and image cutout. Finally, report which combined augmentations methods could give the best performance.

You have to choose 3 differnece combinations of the following data augmentation methods:

[color jittering, translation],
[color jittering, image cutout],
[translation, image cutout] and
[color jittering, translation, image cutout]

*__Pay attention__: you need to add comments for each line of your code and explain its meaning. Otherwise, it will not be assumed as valid code.

Problem 2: (coding + report) (20 points)

Implement the consistency regularization (CR) [4] from the paper "consistency regularization for generative adversarial networks" in the file "train_fns.py". Explain what is consistency regularization and report how it helps your model in your experiments

*__Pay attention__: you need to add comments for each line of your code and explain its meaning. Otherwise, it will not be assumed as valid code.

Problem3 (report) (10 points)

Discuss the overfitting problem on training GAN model and several common strategies to reduce overfitting problem, and how do you find overfitting happens in your experiments. (One possible way is to show inconsistency on training and validation performance)

Problem 4(report) (20 points)

Analyze the learning performance with the following variations of methods:

methods	CIFAR-10 5% data IS FID
BigGAN	
BigGAN + DiffAugment	
CR-BigGAN	
CR-BigGAN + DiffAugment	

Report the IS and FID performance for each combination of methods, and discuss your observations.

Problem 5 (report) (10 points)

Show some visualizations of generated images for each method in problem 4 and discuss your observations

References

[1] Zhao, S., Liu, Z., Lin, J., Zhu, J.Y. and Han, S., 2020. Differentiable Augmentation for Data-Efficient GAN Training. arXiv preprint arXiv:2006.10738.

[2] Krizhevsky, A. and Hinton, G., 2009. Learning multiple layers of features from tiny images.

[3] Brock, A., Donahue, J. and Simonyan, K., 2018. Large scale gan training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096.

[4] Zhang, H., Zhang, Z., Odena, A. and Lee, H., 2019. Consistency regularization for generative adversarial networks. *arXiv preprint arXiv:1910.12027*.