CS492 Machine Learning for Computer Vision

Coursework2 on Generative Adversarial Network [35% mark]

Release on 19 Nov 2021, the report due on 10 Dec 2021 (midnight)

The coursework requires Python/PyTorch/Tensorflow programming. In all questions, you can use any existing toolbox/code, unless specified.

Submission instructions:

One joint report by each pair

Page limit: 4-6 A4 pages per report with 10 font size (use the IEEE standard double column paper format, either in MS word or latex).

http://www.pamitc.org/cvpr16/files/egpaper_for_review.pdf

http://www.pamitc.org/cvpr16/files/cvpr2016AuthorKit.zip

General principles for writing technical report are expected to be known and adhered to. Similarly for practices in conducting experiments, some are as listed below:

- Select relevant results that support the points you want to make rather than everything that the program code gives.
- The important results should be in the report, not just in the appendix.
- Use clear and tidy presentation style, consistent across the report e.g. figures, tables.
- The experiments should be described such that there is no ambiguity in the settings, protocol and metrics used.
- The main points are made clear, identifying the best and the worst case results or other important observations.
- Do not copy standard formulas from lecture notes, explain algorithms in detail, or copy figures from other sources. References to lecture slides or publications/webpages are enough in such cases, however short explanations of new terms or parameters referred to are needed.

Find and demonstrate the parameters that lead to optimal performance and validate it by presenting supporting results. Give insights, discussions, and reasons behind your answers. Quality and completeness of discussions within the page limit will be marked. Include formulas where appropriate, results presented in figures, and their discussions. Try to visualise any interesting observations to support your answers.

Code required for the experiments can be taken from any public library if available, otherwise implemented if necessary.

Submit the report **in pdf** through the KLMS system. No hard copy is needed. Write your full names and school ID numbers on the first page.

If you have questions, please contact

Honggyu Choi (honggyuchoi@kaist.ac.kr)

Jihyun Lee (jyun.lee@kaist.ac.kr)

Train and test Generative Adversarial Networks (GAN) to generate synthetic handwritten digit images. Use the MNIST dataset. It consists of a training set of 60,000 examples, and a test set of 10,000 examples. The examples are black and white images of handwritten digits, each with size 28x28 pixels^2. Preprocess* the data so training GANs becomes easier.
(*Specifically, the input values which range in between [0, 255] will be normalized between -1 and 1)

Q1. DCGAN [5 points] Design and describe an architecture for DCGAN. Train DCGAN using MNIST training examples (60K) and perform evaluation experiments.

Try some different architectures, hyper-parameters, and, if necessary, the aspects of virtual batch normalization, balancing G and D.

Please discuss, with results, what challenge and how they are specifically addressing, including the quality of generated images and, also, the mode collapse.

Q2. CGAN [15 points] Design and describe an architecture for CGAN. Train CGAN using MNIST training examples (60K) and perform evaluation experiments.

Try different architectures, hyper-parameters, and, if necessary, the aspects of one-sided label smoothing, virtual batch normalization, balancing G and D.

Please perform qualitative analyses on the generated images, and discuss, with results, what challenge and how they are specifically addressing. Is there the mode collapse issue?

Evaluation Metrics (Inception scores, TARR) Measure the inception scores and TARR i.e. we use the class labels to generate images in CGAN and compare them with the predicted labels of the generated images by a pre-trained classifier, measuring the success rate.

The pre-trained classifier is a classification neural network with including convolutional layers and a softmax layer, using the MNIST training set (60K). Also report the recognition accuracies on the MNIST real testing set (10K), in comparison to the success rate above.

Measure and discuss the qualities of output images by different metrics (this is an open question), and for the different hyper-parameters/tricks and/or architectures.

Q3. Re-training the handwritten digit classifier [15 points] Use the generated images of CGAN, in addition to the MNIST training set, to re-train the handwritten digit classifier i.e. the classification neural network in Q2. Evaluate the accuracies on the MNIST testing set (10K), compared to the accuracy of the network trained using real data alone. Please vary* the portions of the real training and synthetic images, e.g. 10%, 20%, 50%, and 100%, of each.

*Using even a small number of real samples per class would already give a high recognition rate, which is difficult to improve. Use few real samples per class, and, plenty generated images in a good quality and see if the testing accuracy can be improved or not, over the model trained using the few real samples only.

Did you have to change the strategy in training the classification network in order to improve the testing accuracy? For example, use synthetic data to initialise the network parameters followed by fine tuning the parameters with real data set or vise-versa. Or using realistic synthetic data based on the confidence score from the classification network pre-trained on real data. If yes, please then specify your training strategy in details.

Analyse and discuss the outcome of the experimental result.

[Further thoughts for your interest] Do you have any other ideas to improve GANs or have more insightful and comparative evaluations of GANs? Ideas are not limited. For instance,

- How do you compare GAN with PCA? We leant PCA as another generative model in the module. Strengths/weaknesses?

- Take the pre-trained classification network using 100% real training examples and use it to extract the penultimate layer's activations (embeddings) of 100 randomly sampled real test examples and 100 randomly sampled synthetic examples from all the digits i.e. 0-9. Use an embedding method e.g. t-sne [1] or PCA, to project them to a 2D subspace and plot them. Explain what kind of patterns do you observe between the digits on real and synthetic data. Also plot the distribution of confidence scores on these real and synthetic sub-sampled examples by the classification network trained on 100% real data on two separate graphs. Explain the trends in the graphs.

- Can we add a classification loss (using the pre-trained classifier) to CGAN, and see if this improve? The classification loss would help the generated images maintain the class labels, i.e. improving the inception score. What would be the respective network architecture and loss function?

Some of the relevant works are [2,3,4].

[1] Laurens van der Maaten, Geoffrey Hinton; Visualizing Data using t-SNE, *Journal of Machine Learning Research,* 9(Nov):2579--2605, 2008.
[2] Augustus Odena, "Semi-supervised learning with generative adversarial networks”, *ICML Workshops 2016.*
[3] Tim Salimans, et al. "Improved techniques for training GANs." *Advances in Neural Information Processing Systems 2016*.
[4] Jost Tobias Springenberg, "Unsupervised and semi-supervised learning with categorical generative adversarial networks." *ICLR 2016.*