Ben-Gurion University of the Negev

Faculty of Engineering Sciences

Department of Software and Information systems Engineering

Deep Learning

Assignment 4

The purpose of the assignment

Enabling students to experiment with creating two generative architectures: a) standard GAN, and; b) a modified GAN architecture. The students will work on real-world data and challenges.

Submission instructions:

a) The assignment due date: 30/06/2021

b) The assignment is to be carried out using Python V3.6 (or later). and TensorFlow (Keras, which is part of the TF framework, is also acceptable).

c) Submission in **pairs** only. Only **one copy** of the assignment needs to be uploaded to Moodle.

d) Plagiarism of any kind (e.g., GitHub) is forbidden.

e) The entire project will be submitted as a single zip file, containing both the report (in PDF form only) and the code. It is the students' responsibility to make sure that the file is valid (i.e. can be opened correctly).

Introduction

Generative models are among the most exciting architecture in the field of deep learning today. Their ability to generate high-quality data with little-to-no interaction or knowledge with the existing data enables us to accomplish tasks thought impossible not long ago. In this assignment you will implement two generative models, each dealing with a different scenario. The first generative model is a simple GAN, which you will implement on two tabular datasets (we will not use image datasets as the running times are significantly longer). The second GAN model offers in interesting "twist" in the sense that you will NOT have access to any "real" data.

Instructions

Part 1 – Generative Adversarial Networks

In this part you will implement a simple GAN model for tabular data.

1. Please download the following two files, provided with the assignment:
   a. German_credit.arff
   b. diabetes.arff

   These two files will be used throughout the assignment
2. For each of these two datasets, train a GAN capable of generating samples from the original dataset. You may use any architecture structure you see fit. Train your model until it reaches convergence. If no convergence is reached (although it is of course preferable), choose the epoch/iteration for which your model performed best. Run two individual experiments, for the two files presented above. You may use the entire dataset to run your experiments.
3. Analyze the products of your model, and include the following information:
   a. Provide several samples that "fooled" the detector and several that did not. Use basic measures (e.g., Euclidean distance, basic visualization) to determine whether the samples that fooled the detector are indeed similar to some of the original data
   b. Once you models are converged, generated 100 samples at random. How many were able to pass as real samples?
   c. A graph describing the loss of the generator and the discriminator. Did the models go "back and forth" with their losses? Was there a consistent leader?

Part 2 – Generative model for sample generation

In this part you will implement a GAN architecture "with a twist". The goal of this architecture is inferring the inner-working of a black-box model.

Assume a black-box (BB) ML model (a model trained on some data of which you have **no access**). Your goal is to generate some data that is likely similar to the data used to train the model. Since there is no way for your model to know for sure, we will use the confidence scores (i.e., classification) produced by the black-box model as indication. Please do the following:

1. For each of the two datasets described above, train a RandomForest model. This model is the "black-box" you'll use in this assignment. Use a random 70%/30% split so you can report the performance of the classifier.
2. The requested architecture is presented in Figure 1. The training process is as follows:
   a. For each generated sample, the generator will receive two inputs: a vector of random noise *Z* and a desired confidence score (a scalar value) *C*. See #1 in the figure.
   b. The generator will generate a sample and send it to the discriminator. See #2 in the figure.

c. Instead of a “real” sample (to which we have no access) the discriminator will receive two scalars: a) *C* (the same one given to the generator); b) *Y* – the output of the black-box model you trained on the generated sample (see #3 in the figure).

d. The goal of the discriminator is to determine which of the two values – C or Y – is the true classification produced by the black-box model. The output of the discriminator, denoted by $\hat{y}$, will be used for the calculation of the loss in the standard fashion (see step #4).

3. Please note that the figure describes the process for a single sample, but you need to generate a batch. Also, the BB model is not part of the deep architecture, and is in fact transparent to other components.

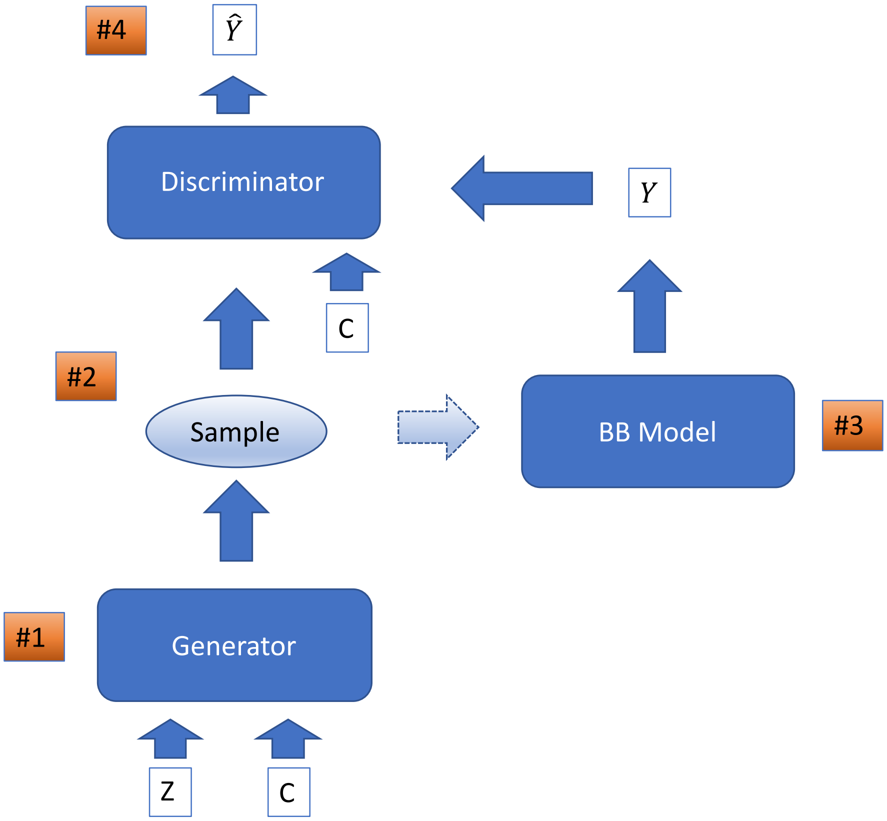

Figure 1: the requested architecture. The generator receives two inputs – Z and C – and produces a sample. This sample is sent to the discriminator, which also receives C (same C as the generator) and the output of the BB model for this sample. The goal of the discriminator is to determine which classification is the “real” one (i.e., produced by the BB model).

4. After your model has converged, please provide the following information:
   a. The RandomForest classifier’s performance – in terms of confidence score distribution – for the 30% test set. Provide min/max/average, as well as a figure showing the distribution.
   b. Generate 1,000 samples from your fully-trained generative model. The confidence scores of the generated samples need to be uniformly sampled from [0,1].

c. Feed the generated samples to the black-box model and provide statistics on the score distributions. Are there any conclusions you can draw? Were you more successful for a specific class of samples than another? Were you more successful for a specific range of confidence scores?

d. Did your model suffer from mode collapse or other problems that may have prevented it from generating more effective samples? If so, in what cases? Elaborate.