In the name of God

University of Tehran

School of Electrical and Computer Engineering

Neural Networks and Deep Learning Course

Sixth Assignment

Question 1	
Designing Assistant's Name	Saeed Rahimi
Email	Saeedrhimi2@gmail.com
Question 2	
Designing Assistant's Name	Mohammad Karami
Email	karami.m7906@gmail.com
Submission Deadline	1404/03/31 (YYYY/MM/DD)

Contents

1 Rules

Before answering the questions, please carefully study the following points:

- Prepare a report of your answers in the format provided on the course page on the Elearn system named `REPORTS_TEMPLATE.docx`.
- It is recommended to do the exercises in groups of two. (More than two people are not allowed, and individual submission does not have an extra score). Please note that there is no requirement for group members to remain the same until the end of the term. (i.e., you can do the first exercise with person A and the second exercise with person B, etc.).
- The quality of your report is of particular importance in the grading process; therefore, please mention all the points and assumptions you have considered in your implementations and calculations in the report.
- In your report, according to what is provided in the sample template, consider captions for figures and super-captions for tables.
- It is not mandatory to provide detailed explanations of the code in the report, but the results obtained from it must be reported and analyzed.
- Analysis of the results is mandatory, even if no mention is made in the question.
- The teaching assistants are not obliged to run your codes; therefore, bring any results or analysis requested from you in the questions clearly and completely in the report. In case of non-compliance with this point, it is obvious that points will be deducted from the exercise.
- The codes must be prepared in a notebook with the `.ipynb` extension. At the end of the work, all the code should be executed, and the output of each cell must be saved in this submitted file. Therefore, for example, if the output of a cell is a plot that you have included in the report, this plot must also be present in both the report and the code notebook.
- In case of observing cheating, the score of all persons involved will be considered -100.
- The only authorized programming language is **Python**.
- The use of ready-made codes for the exercises is by no means permissible. If two groups use a common source and submit similar codes, it will be considered cheating.
- The method of calculating lateness is as follows: After the submission deadline, there is a possibility of late submission for a maximum of one week. After this one week, the score for that assignment will be zero for you.
  - First three days: no penalty
  - Fourth day: 5% penalty
  - Fifth day: 10% penalty

  - Sixth day: 15% penalty
  - Seventh day: 20% penalty

- The maximum score that can be obtained for each question is 100, and if the total points of a question are more than 100, in case of obtaining a score higher than 100, it will not be applied.
  - For example, if the score obtained from question 1 is 105 and the score of question 2 is 95, the final score of the exercise will be 97.5 and not 100.

- Please put the report, codes, and other attachments in a folder with the following name, compress it, and then upload it to the Elearn system:

  `HW[Number]_[Lastname]_[StudentNumber]_[Lastname]_[StudentNumber].zip`

  Example: `HW1_Ahmadi_810199101_Bagheri_810199102.zip`

- For groups of two, it is sufficient for one of the members to upload the exercise, but it is recommended that both people upload it.

2 Question 1: Unsupervised Learning and Domain Transfer using GAN

2.1 Introduction

One of the main challenges in the practical application of deep learning is the problem of Domain Shift. This situation occurs when a model trained on one dataset (source domain) experiences a sharp decline in performance when faced with data from another domain (with differences in appearance, lighting conditions, visual style, etc.). In many real-world applications, labeling new data (e.g., real images) is difficult, expensive, or even impossible; but synthetic or simpler data (like generated or simulated data) is readily available.

The paper "Unsupervised Pixel-Level Domain Adaptation with GANs" addresses this very issue. The authors of this paper present a solution that uses GAN networks to transform images from the source domain to images with the appearance of the target domain, in a way that the main content of the image is preserved, but its appearance resembles the target data. This process is done without using any labels in the target domain.

The key advantage of this method is the transfer of knowledge from one domain to another at the pixel-level; an approach that not only improves the model's performance in the target domain but also its output is interpretable and analyzable.

In this exercise, by reconstructing this paper, you will become familiar with advanced concepts such as domain adaptation and adversarial learning, and you will also gain practical experience in designing and analyzing the performance of combined models.

2.2 Theoretical Part (30 points)

1. **Why is GAN training unstable? Name three main factors of instability and explain each one. In your opinion, what mechanisms have been proposed to counter this instability? (5 points)**

2. **In a regular GAN, the Generator only converts noise z into an image. In this paper, the Generator uses an input image and noise instead. In your opinion, what effect does this change have on learning and controlling the output? (5 points)**

3. **The model introduced in this paper consists of three main components: Discriminator, Generator, and Classifier. In your opinion, what is the role of each in learning? If one of them is removed, what changes will happen to the overall performance of the model? (5 points)**

4. **One of the special features of the model presented in this paper is the use of content similarity loss (to preserve content). Explain why preserving the content of the source image is essential, and if this mechanism is removed, what behavior do you expect from the Generator? (5 points)**

5. **The model presented in this paper uses an independent classifier for simultaneous training on the original image and the generated image. What advantage does this have over training only on generated images? Analyze how this choice helps the stability of learning. (5 points)**

6. **The method of this paper is designed for domains like MNIST and MNIST-M, where the difference is in style. Can this same method be used for domains that have a semantic difference (e.g., different objects, different languages, or different viewing angles)? Justify your reasoning. (5 points)**

2.3 Practical Part (70 points)

2.3.1 Data Preprocessing (5 points)

In this exercise, we use two datasets, MNIST and MNIST-M, which you can download from this link. The MNIST dataset includes black and white images of digits from 0 to 9 and is used as the source domain. The MNIST-M dataset is a modified version of the same images created by adding colorful backgrounds and is considered the target domain. The order of the images in both datasets is preserved, meaning that each image in MNIST has the same index as in MNIST-M, and their only difference is in appearance (background and color), not in the numerical content.

First, select 5 corresponding samples from each of the two datasets using the same index and display the MNIST and MNIST-M images in a row next to each other so that their visual difference and content preservation can be observed.

2.3.2 Data Preparation and DataLoader (Continued)

Then, you should prepare the data according to the paper's instructions and place them in separate DataLoaders. First, convert the MNIST images, which are single-channel (grayscale), to three-channel (RGB) images to match the structure of the MNIST-M data. Both datasets should be resized to the required dimensions of the model's architecture (e.g., 32x32). After that, all images should be normalized to the range [-1, 1].

In the next step, divide each of the two datasets into training and testing sections with a ratio of 80% training and 20% testing. Since the order of images in both datasets is the same, the division should be done using common indices to preserve the correspondence between MNIST and MNIST-M samples. Finally, for each dataset, define two separate DataLoaders: one for the training data and one for the test data. In total, four DataLoaders should be created: training and test for MNIST and training and test for MNIST-M. Extract the details related to preprocessing and batch creation from the text of the paper and adhere to them in your implementation.

2.3.3 Baseline Model Test and Domain Gap Observation (5 points)

Train a classifier with the exact architecture of the paper (Classifier for MNIST-M) only on the MNIST data. Then, evaluate the trained model without any changes on the MNIST test data and all of the MNIST-M data and report the accuracy of both. Briefly analyze the difference in accuracy.

Note: The architecture should be according to the paper.

2.3.4 Model Architecture Implementation (20 points)

After preparing the data, you should design and implement the three main parts of the model according to the explanations in the paper. These three parts include a **generator** for transferring the image style, a **discriminator** for distinguishing real or fake images,

and a **classifier** for performing the classification task. The structure of all three models is explained in the paper and must be designed accordingly. Extract the architectural details, number of layers, type of normalization, type of activation function, and the data path in each network from the text of the paper and the provided figures and implement them in code.

2.3.5 Implementation of Loss Functions (10 points)

After designing and implementing the model architectures, at this stage, it is necessary to implement the loss functions used in the model training according to the paper. The paper introduces two main types of loss, each of which has a different role in optimizing different parts of the model. The first, **adversarial loss**, is the loss that guides the competitive interaction between the Generator and the Discriminator and teaches the Generator how to make the generated images look real from the Discriminator's point of view. The second, **task loss** or **classification loss**, is the loss that evaluates the performance of the Classifier in correctly identifying digits (in both original and generated images). You should define these losses separately and, according to the paper's explanation, use them in the training stages of the Generator, Discriminator, and Classifier. The combination coefficients of these losses should also be considered as hyperparameters. Extract the exact method of combining these losses during training from the text of the paper and implement it.

2.3.6 Model Training (20 points)

In this section, you must train the model according to the training structure of the paper. The Generator is trained only with images from the source domain (MNIST). The MNIST image along with noise is given to it to produce an image with the appearance of the target domain (MNIST-M) that preserves the numerical content. The Classifier is also trained only on the source domain images, both on the original MNIST images and the fake images produced from them. At no stage of training should the labels of MNIST-M (target domain) be used. The Discriminator is trained separately and must learn the difference between real MNIST-M images and generated images.

At each epoch, the loss values for the Generator, Discriminator, and Classifier, as well as the accuracy of the Classifier, should be measured and stored for both train and test data, as they will need to be displayed and analyzed in the next stage. Adhere to the details of the update order, the method of combining losses, and the data used according to the paper and train the model. (For ease of training, you can save the model after each epoch or a certain number of epochs along with other data so that in case of a problem, you can continue the training operation from there).

2.3.7 Important Note on Classifier Implementation

Be careful that the behavior of the classifier is different in the training and testing phases, and you must correctly implement this difference in the code. In the training phase, the Classifier is trained only with data from the source domain (MNIST); and only on real MNIST images and fake images created by the Generator, and only MNIST labels are used. In this phase, no use of MNIST-M data or its labels is allowed.

In contrast, in the test phase, the performance of the Classifier must be evaluated separately on the MNIST test data (source) and the MNIST-M test data (target). At

this stage, the use of MNIST-M labels is only allowed for evaluating accuracy, not for learning. This distinction is very important; any use of target labels in the training phase is contrary to the assumption of unsupervised domain adaptation and invalidates the model's results.

2.3.8 Displaying Results (10 points)

In this section, you must display the loss and accuracy results that you have stored during training in the form of a graph.

Alongside the graphs, you should display a few sample images for a qualitative evaluation of the model. Select five images from the test set in MNIST and give them to the Generator to produce a fake image. Then, for each sample, display the following images in a row and corresponding to each other: the original MNIST image, the image generated by the Generator, the real corresponding image from MNIST-M. Next to each image, also write the real label and the prediction made by the Classifier. The order of display should be preserved so that it can be checked whether the Generator has changed the appearance of the image correctly, preserved the numerical content, and whether the Classifier has correctly identified the number or not.

3 Question 2: Reconstruction of Endoscopic Polyp Images with EndoVAE

In this question, you must reconstruct the EndoVAE paper's architecture and evaluate it using the Kvasir dataset. Note that instead of the original dataset of the paper (KID), you will use the Kvasir dataset.

3.1 Data Preprocessing (10 points)

Download and extract the Kvasir dataset. Then select the healthy images from the folders below. Try to use all healthy images, and using data augmentation to increase the number of healthy images can be helpful.

- `normal-z-line`
- `normal-pylorus`
- `normal-cecum`

Take the polyp images from the `polyps` folder of the Kvasir-SEG part.

Then, according to the paper, load all images and convert them to RGB color space. Also, change the size of each image to 96x96 using bilinear interpolation and normalize the pixel intensity to the range [0, 1]. Also, if another preprocessing like cropping, etc., is appropriate, do it with justification in the report. Studying section A of the fourth part of the paper can be helpful in this section.

Next, save the final output in two new folders:

- `processed/normal` (all healthy images, dimensions 3x96x96)
- `processed/polyp` (all polyp images, dimensions 3x96x96)

In the report, show at least one sample of the image before and after this preprocessing.

3.2 EndoVAE Architecture Design (20 points)

Implement according to sections three of the paper and figure one.

Encoder: A convolutional network consisting of six Conv2D+ReLU layers whose output is flattened and with two FC paths, it reaches the mu and logvar vectors with length `latent_dim = 6`.

Decoder: First, expand the vector Z with an FC layer to a 256x6x6 tensor and then return the output to a 3x96x96 image with seven ConvTranspose2d+ReLU and Sigmoid layers (in the last layer).

Make sure the intermediate dimensions of the layers exactly match figure one of the paper. Also, in the report, explain your understanding of each of the important parts of the architecture.

3.3 Defining Loss Functions (10 points)

According to section three of the paper, define the loss functions. Also, write your understanding of each in the report.

Reconstruction Loss: Use Binary Cross-Entropy (BCE) between x and $\hat{x}$. You can also test MSE and write in the report with reason or comparison which one you used.

KL Divergence: The standard formula for VAE from the same section of the paper.

Final total_loss:

$$\text{total\_loss} = \text{Reconstruction} + 1.0 \times \text{KL}$$

3.4 Model Training Process (25 points)

Set the hyperparameters according to section 4.A of the paper.

Place the "healthy" data in a Dataset and DataLoader with `shuffle=True`.

Perform the training process according to the paper:

1. Model in `train` mode
2. Forward pass: `Encoder` → `reparameterize` → `Decoder`
3. Calculate `Reconstruction Loss` and `KL Loss` and sum them up
4. `backprop` and `optimizer.step`
5. Record the average of each Loss every few epochs.

Your report from this section should clearly show that the training process was done correctly and according to the paper.

3.5 Qualitative Generation and Reconstruction (15 points)

3.5.1 Generation

Load the trained model in `eval` mode and according to section 4 of the paper: Create nine random vectors $z \sim N(0, I)$, give each one to the Decoder and get the set of nine generated images. Then display a 3x3 grid of these nine images in the report.

3.5.2 Reconstruction

Select a Batch of 5 polyp images from the `processed/polyp` folder and then, according to the paper's description, give the images to the Encoder to get $(\mu, \log \rho)$, then reconstruct with the Decoder. Then display a 2x5 grid (in each row: the real polyp image and the reconstructed image) in the report.

3.6 Numerical Evaluation (10 points)

For 50 random images from the `processed/polyp` folder:

- Reconstruct each image.
- Calculate PSNR and SSIM between the real image and the reconstructed image.

For those 50 samples, prepare a table with two columns: "Mean" and "Standard Deviation" for PSNR and SSIM.

Note that these metrics are not in the paper, and we use them for the quantitative analysis of our own reconstruction.

3.7 Analysis of Results and Final Discussion (10 points)

Qualitative Analysis:

- How realistic are the generated images?
- What details did the polyp reconstruction preserve, and what parts were blurred?

Quantitative Analysis:

- What do the PSNR and SSIM values indicate about the quality of the reconstruction?
- If the PSNR and SSIM values are low, where is the problem likely to be (architecture, hyperparameter, or preprocessing)?

Due to the difference in the dataset, the goal of this implementation is not to have results equal to the paper, but your understanding of the paper's architecture and the analysis of the results are the scoring criteria.

3.8 Bonus Section (5 points)

Define a simple CNN classifier to distinguish between "real polyp" and "reconstructed polyp". Place 200 real polyp images and 200 reconstructed images into two train and test sets. Calculate its accuracy and AUC and report them. Analyze whether the reconstructed images for training the classifier were almost similar to the real ones or not.