Assignment 2

Innopolis University

Machine Learning Fall 2021 - Masters

General Instructions

In this assignment, you will solve several tasks on classification, clustering and generative models (bonus). Read the instruction carefully and all questions should be first sent to class representative who will later share with the TA and the professor.

- You are required to submit your solutions via Moodle as a single zip file. The zip archive should contain a single ipynb, and a single PDF for the theoretical sections 1 and 2. Please, put your name and email as the first line in the notebook.
- There's one more ipynb file, you can submit if you have solved the bonus part. We won't give partial grades for the partial solution of the bonus part. There will be grade only for the bonus part if it is complete and it will be graded based on the model performance and the quality of the generated faces.
- Source code should be clean, easy to read, and well documented. Bonus points may be awarded for elegant solutions. However, these bonus points will only be able to cancel the effect of penalties.
- Do not just copy and paste solutions from the Internet. You are allowed to collaborate on general ideas with other students as well as consult books and Internet resources. However, be sure to credit the sources you use and type all the code, documentation by yourself.

1 Theoretical question on K-means Clustering (5 points)

Consider a set of $n = 2m + 1$ one-dimensional samples, m of which coincide at $x = -2$, m at $x = 0$, and one sits at $x = a > 0$, i.e.,

$$D = \left\{ \underbrace{-2, -2, ..., -2}_{\text{m times}}, \underbrace{0, 0, ..., 0}_{\text{m times}}, a \right\}.$$

In fact D is a multi-set since it allows for multiple instances of the same element. (NOTE: The variable m has nothing to do with the symbol K often used for the number of clusters.) Let $D_1, D_2 \subset D$ be a two-cluster partitioning of the dataset D so $D_2 = D - D_1$. It can be shown that the two-cluster partitioning that minimizes the sum-of-squared errors (used in the K-means algorithm, though this part is about the globally optimal solution)

$$J(D_1, \mu_1, \mu_2) = \sum_{i:D_i \neq \phi} \sum_{x \in D_i} (x - \mu_i)^2$$

,

groups the m samples at $x = 0$ with the one at $x = a$ (i.e., $D_2 = 0, ..., 0, a$) if

$$a^2 < f(m)$$

where $f(m)$ is a function of m. Derive this function accordingly.

2 Theoretical question on SVM (5 points)

Consider the following SVM formulations:

(I) minimize$_\theta$ $\frac{1}{2}||\theta||^2$ subject to $y_t\theta^T x_t \geq 1$ for all $t \in \{1, ..., n\}$.

(II) minimize$_{\theta,\theta_0}$ $\frac{1}{2}||\theta||^2$ subject to $y_t(\theta^T x_t + \theta_0) \geq 1$ for all $t \in \{1, ..., n\}$.

(III) For fixed $C \in (0, \infty)$, minimize$_{\theta,\zeta}$ $\frac{1}{2}||\theta||^2 + C\sum_{t=1}^{n}\zeta_t$ subject to $y_t\theta^T x_t \geq 1 - \zeta_t$ and $\zeta_t \geq 0$ for all $t \in \{1, ..., n\}$.

Note that in (I) and (III) we have $\theta_0 = 0$. Consider the following illustrations, with positive points marked '+', negative points marked '–', three straight lines indicating the decision boundary and margins, and circles placed around the support vectors:

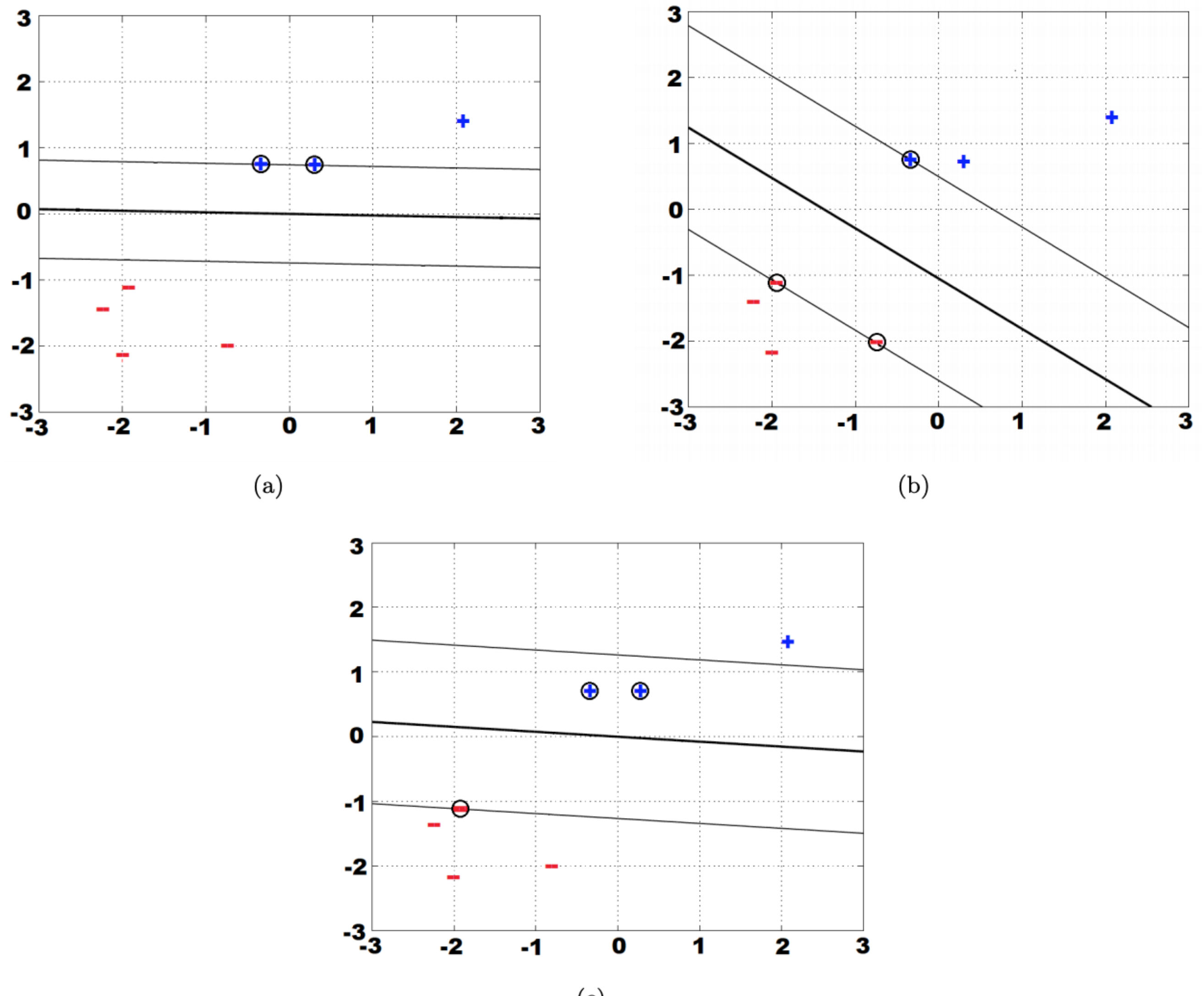

Figure 1: Different SVM models on the same set of points.

For each of the above SVM formulations (I), (II), and (III), indicate whether it is **possible** or **impossible** for it to have produced each classifier in figures (a), (b), and (c). If it is possible, simply write "**Yes**" with no further explanation. If it is impossible, write "**No**" followed by a brief explanation. (Note: These explanations only need to be one short sentence long, as opposed to detailed justifications.)

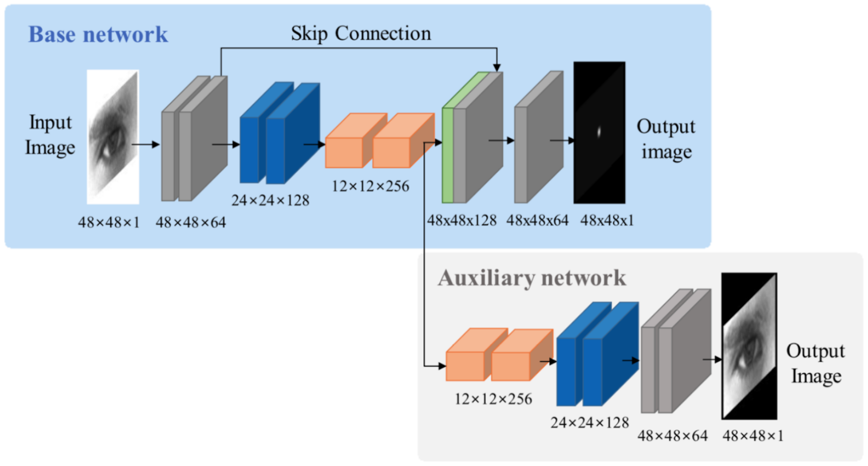

Figure 2: Eye center estimation CNN architecture.

3 Human iris center calculation (30 points)

Eyes are considered to be the most salient and stable feature in the computer vision community. Automatic extraction of eyes plays an important role in many real-world applications such as gaze tracking, face alignment, driver drowsiness detection, face recognition, and human–computer interaction, etc. Several factors that make challenging to detect and track eyes are head poses, lighting conditions, shape and color of eyes, iris movement, imagining quality and conditions, etc. A lot of works have been performed to solve the above problems but still, it remains an open area of research in the computer vision community.

In this task you are going to implement CNN for calculating human iris center. This CNN architecture is proposed in **this paper** as a fully convolutional network which consists of a base network and auxiliary network. It has two losses: reconstruction loss for ensuring the model saves important positional features and second loss is the distance between predicted eye center and ground truth one. The architecture also has skip connection which brings position information to deep layer. Dataset with face images and labels (iris center coordinates + eye corners) is **here**.

You can use cv2 as tool in that task to read, convert color of images, to draw something on image and to do all the other needed operations with images. However you are not restricted to use it and you can use any libraries.

- Preprocess and visualize the dataset: [50%]
  - Download dataset. Description of folders and naming are inside dataset folder in README.txt
  - Read all images and convert them to gray with (cv2.ctvColor())
  - Read annotation for images. It contains eye corners and eye centers of 2 eyes for each image.
  - Visualize one image, draw eye corners and iris centers on it
  - Normalize images (divide by 255)
  - Crop eye regions (and resize if needed) to be (48x48) image with the help of eye corners. Do that for all images. It should look like Figure3(a)
  - Now data is ready to create a final dataset, which you will use for CNN training. You should create two np arrays X and Y:

    - X contains (48x48) images of eye regions which you crop on previous step

    - Your labels (Y) are coordinates of eye center for each image in X (don't forget to convert iris center from whole image coordinate system to coordinate system of eye region). You should make one more step to cook Y set. For each eye center in Y you should create a (48x48) image (with

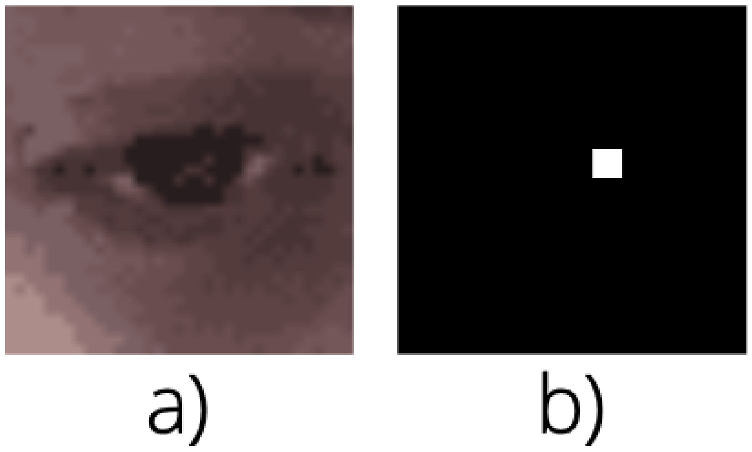

Figure 3

zero values) and assign value=1 to pixel which coordinate is an iris center. Do it for all images. It should look like Figure3(b)

Finally, your X and Y sets are lists of 48x48 images. X contains images of eye and Y images with white pixel on the place of iris center.

- Split dataset

- Build a CNN model using PyTorch. [20%]
- Compile and train CNN with different optimizers [sgd, adam, adamax, rmsprop], loss functions [mse, mae] and activations [tanh, relu, sigmoid]. Report best combination. [20%]
- Make a prediction for 10 test images. Draw predicted centers on them and visualize it. (You can draw iris center with cv2.circle()) [10%]

HINTS: Use colab with GPU to speed up training and hyperparameter tuning.

4 Network Intrusions clustering (30 points)

Anomalies, also referred to as outliers, are defined as a set of patterns which significantly deviate from the normal or expected pattern. In our digital age, anomalies are an integral part of many applications including cyber security , manufacturing, fraud detection, health-care systems and numerous other fields. For instance, in cyber security, intrusion detection system can identify an anomalous pattern like unauthorized access to sensitive information or security violation. One of the major challenges include the scarcity of anomalous labelled data. Most of the systems do not have or few anomalous patterns and make the learning task difficult. To solve this situation, unsupervised anomaly detection model has the ability to learn the model without anomalous data patterns.

There are already high performing proposed solution for anomaly binary classification in the domain of cyber security. To better design a defence system it is important to identify the types or clusters of the anomalies. In this task you will be using an unsupervised approach to cluster network intrusions. The dataset for the task can be found **here**. All the steps for clustering of intrusions should be clearly outlined in the notebook or python script. Use machine learning method of your choice. Reason of choosing the method should be stated in the notebook or python script

5 Design Patterns Detection (30 points)

The impact of design patterns on quality attributes has been extensively evaluated in studies with different perspectives, objectives, metrics, and quality attributes, leading to contradictive and hard to compare results. Knowing that a particular module implements a design pattern is a shortcut to design comprehension. Manually detecting design patterns is a time consuming and challenging task; therefore, researchers have proposed automatic design patterns detection techniques which include machine learning based approaches.

In this task you will be implementing a machine learning based approach for detecting design patterns category from open source projects. The dataset is a subset of source code metrics for each an every java project. Each project in the dataset belongs to one of 3 categories. The categories are described **here** The design of the design pattern detection approach is up to you. The implementation should be in python programming language. Good programming practices will will be rewarded and bad practices will be penalized. The list below outlines the main requirements for this task:

- Data extraction, preprocessing and feature engineering procedures should be clearly implemented and justified
- Use Ensemble Learning
- Select two more models to train and do hyper-parameters turning (grid or random search)
- Tune the following hyper-parameters of the estimators in all ensemble models using grid search:
  - `n_estimators`
  - `max_features` $\longrightarrow$ for the base estimators
  - `min_impurity_decrease` $\longrightarrow$ for the base estimators
- Compare models and make conclusion

Dataset for the task can be downloaded from here : **LINK**

6 Bonus on GANs (10 points)

1. Fake Face Generation using GANs. For this question, please check and fill the **attached notebook**.

Notes

- Cheating is a serious academic offense and will be strictly treated for all parties involved. So delivering nothing is always better than delivering a copy.
- Late assignments will not be accepted and will receive **ZERO** mark.
- Code cleanness and style are assessed. So maybe you want to take a look at our references: Link 1 and Link 2.
- Organize your notebook appropriately. Divide it into sections and cells with clear titles for each task and subtask.