Arabic Character Recognition

Given a dataset of Arabic handwritten characters in csv files:

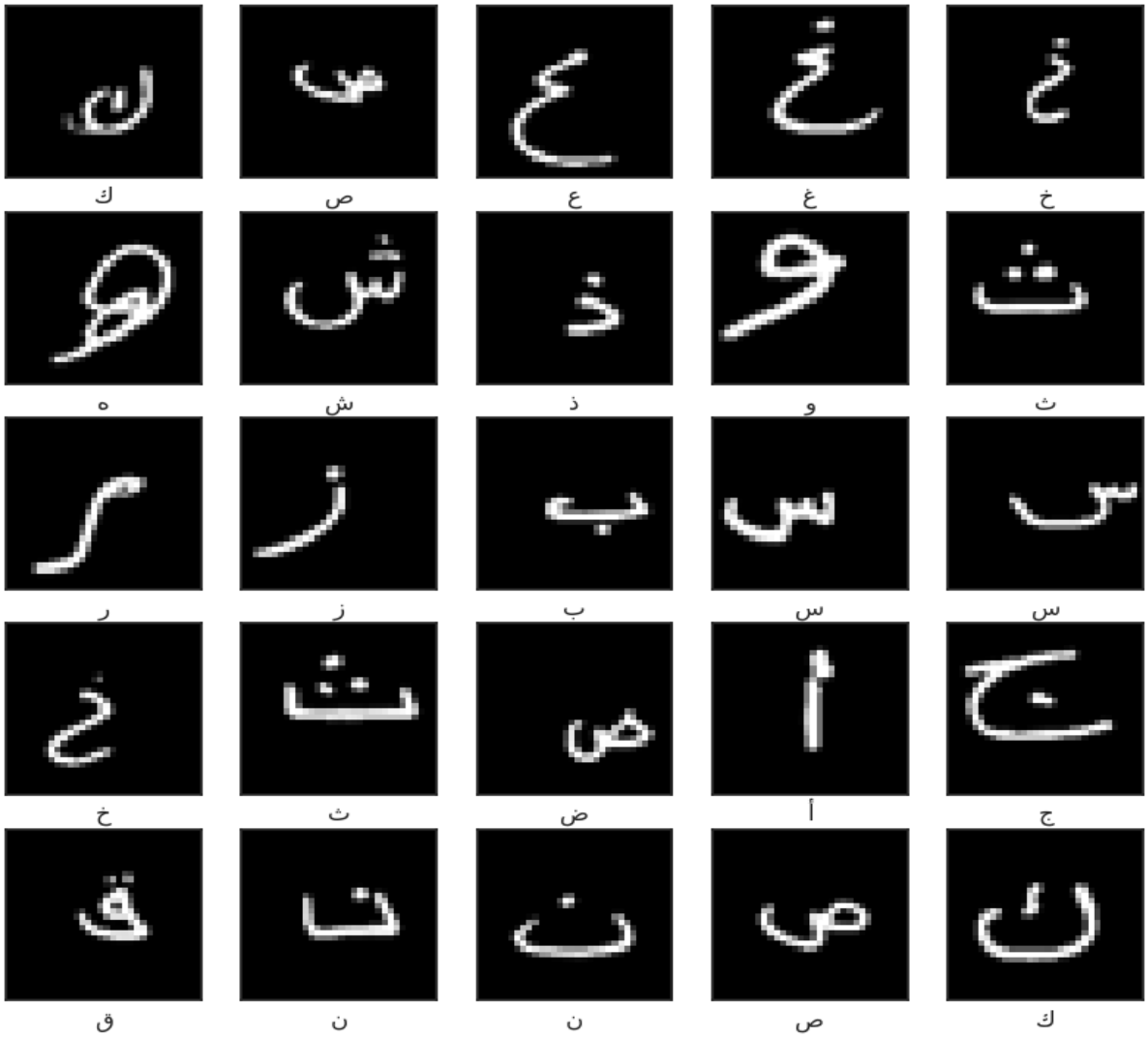
- **"csvTrainImages 13440x1024.csv"** which contains 13440 rows and 1024 columns (i.e. flattened pixels extracted from 13440 images, each image is 32 x 32 pixels)

- **"csvTrainLabel 13440x1.csv"** which contains 13440 rows. Each row represents the label of the correspinding image in the **train** file. The label is an index of an Arabic character.
    - ***Example:*** *Index is 2 refers to 'ب'.*

- **"csvTestImages 3360x1024.csv"** which contains 3360 rows and 1024 columns (i.e. flattened pixels extracted from 3360 images, each image is 32 x 32 pixels)

- **"csvTestLabel 3360x1.csv"** which contains 3360 rows. Each row represents the label of the correspinding image in the **test** file.

Showing a sample of the data as images:



Requirements:

- **Data exploration and preparation:**
  - Identify the number of unique classes and examine the distribution of samples in each class.
  - Normalize each image.
  - Write a function that reconstructs an image from its flattened vector and displays it. Use this function to visualize some of the images in the dataset and display the results for testing the models using it.

- **Experiments and results:**
  - *First experiment:*
    - Train an SVM model on the training data.
    - Test the model and provide the confusion matrix and the average f-1 scores for the testing dataset.
  - Split the training data into training and validation datasets.
  - *Second experiment:*
    - Try different K values with KNN models.
    - plot the average f-1 scores with the validation dataset using different K values and suggest the best value for K.
    - Test the model with the best K value and provide the confusion matrix and the average f-1 scores for the testing dataset.
  - *Third experiment:*
    - Build 2 different Neural Network model architectures (different number of hidden layers, neurons, activations, etc.)
    - Train each one of these models and plot the error and accuracy curves for the training data and validation data.
    - Save the best model, then reload it in a separate file, and use it on new unlabelled data to get predictions.
    - Test the best model and provide the confusion matrix and the average f-1 scores for the testing dataset.
  - Compare the results of the models and suggest the best model.

You can download the dataset from: https://drive.google.com/file/d/1u3tnEY-eMmeUNaKP31YBCVoty9s5VCsz/view?usp=sharing

Deliverables:

You are required to submit ONE zip file containing the following:

- Your code (.py) file. If you have a (.ipynb) file, you have to save/download it as (.py) before submitting.
- A report (.pdf) containing the team members' names and IDs, and the code with screenshots of the output of each part. If you have a (.ipynb) file, you can just convert it to pdf.

The zip file must follow this naming convention: ID1_ID2_ID3_ID4_ID5_Group

Instructions:

1. The number of students in a team is 5.
2. No late submission is allowed.
3. Cheating students will take **ZERO** and no excuses will be accepted.
4. You can use any Python libraries.

Grading Criteria:

Data Preparation	1
SVM	2
KNN	3
Neural Network Models	4
Results	2
Total = 12 marks	