**contact information**

* Alexander boyev 314393158

alexabo4@ac.sce.ac.il

* Moshe Faerman 204469449

moshesh81@ac.sce.ac.il

**Description**

In this project we will try to build a simple neural network for the “hhd\_dataset[1].represents 5073 pictures of the Hebrew alphabet letters. The Hebrew alphabet contains 27 letters, so each letter represent a class and contains about ~188 image of handwritten letters.

The program will execute the following steps:

* We will load the data from previous work, assignment (1) to a "dict" variable.
* The data is already padded, greyscaled, resized and split to randomly but equally distributed three categories: Train, Validation and Test. With the sizes of 80% : 10% : 10% accordingly.
* We will turn each image to negative.
* Since our goal is a categorical classification problem, we will need to “One-Hot” encode our label.
* We will build neural network model with the following layers:
  + Input layer with 1024 neurons, according to the image size 32\*32, using the activation function of “ReLU”
  + Hidden layer with 512 neurons.
  + Another Hidden layer with 512 neurons.
  + Output layer with 27 neurons, according to 27 Hebrew alphabet letters, using the activation function of “SoftMax”.
* We will try to achieve best model using different configurations on train and validation data such as:
  + No configuration
  + L1 Regularization with λ=0.01 and λ=0.001 for each layer except the output layer.
  + L2 Regularization with λ=0.01 and λ=0.001 for each layer except the output layer.
  + Adding Dropout=0.5 to each layer except the output layer.
  + L2 Regularization with λ=0.01 and λ=0.001 and Dropout=0.5 for each layer except the output layer.
* The model will be compiled using the “adam” optimizer and “categorical cross entropy” loss function.
* Now we can save the best model for future use.
* The best saved model could be loaded, now we can fit the “test” set and report the results.
* We will generate a “result.txt” file that will contain:
  + Model configurations.
  + The accuracy of each letter.
* We will provide a confusion matrix, that will be saved on scv file.
* We will provide image of Loss curve for training & validation sets.

**Environment**

The script developed and ran on Windows 10 pro edition, PyCharm 2019.3.3 (Community Edition) with Python 3.7 interpreter.

**Requirements**

To run the program, you will need to install the following libraries:

**import** cv2  
**import** numpy **as** np  
**import** pandas **as** pd  
**from** sklearn.metrics **import** confusion\_matrix  
**import** pickle  
**import** sys  
**import** matplotlib.pyplot **as** plt  
**import** itertools  
**from** keras **import** Sequential  
**from** keras.layers **import** Dense  
**from** keras.layers **import** Dropout  
**from** keras **import** regularizers  
**from** keras.utils **import** to\_categorical

**How to Run Your Program**

**Preconditions:**

1) install all the required models.

2) The assignment delivered with the pickle file “data\_resized\_padded\_greyscale” that represent the preprocessed data from assignment (1).

3) You can load the best saved model using “model\_and\_parameters” file or skip to step (4) to run the entire program.

4) just cancel the comments and make functions available again, this way the script will make all the pipeline from scratch. Load -> Negative -> Train Best Model -> load Best Model -> Test -> Results.

Graphical user interface, text, application, email

Description automatically generated

Press “Run” in the PyCharm environment or any other python script IDE.

**Input Assumptions**

For such data we achieved decent accuracy of 72% on the test set and 76% on train & validation set, but for such large number of classes (27 letters) we could use more data, especially when neural networks are involved.



Chart

Description automatically generated

Chart, scatter chart

Description automatically generated

**References**

[1] I. Rabaev, B. Kurar Barakat, A. Churkin and J. El-Sana. The HHD Dataset. The 17th International Conference on Frontiers in Handwriting Recognition, pp. 228-233, 2020.