Arabic Handwritten Recognition

Anwar Aljohani

Abstract

The goal of this project was to use classification models to recognize the Arabic handwriting. I worked with data provided by kaggle. There are several challenges to hand-written Arabic character recognition systems. Including the limitless variation of human writing and large public databases. In this project, I am modelling a deep learning architecture that can be effectively applied to the recognition of Arabic handwritten characters. A Convolutional Neural Network (CNN) is a special type of feed-forward multilayer trained in supervised mode. 16800 Arabic characters were used in training and testing to create the CNN. To select the parameters with the strongest effect, I considered all possible combinations of CNN parameters.

Design

Human handwriting has unlimited imbalance from one person to another. Building a machine learning model that can classify these characters regardless of the handwriting difference. By considering the number of Arabic characters, 28 classes are the target of our CNN model. This classifier will enable any organization to transform any hardcopy document to a softcopy with a fast, accurate, and reliable system, which will allow many applications such as searching for words in large volumes of documents, automatic sorting of postal mail, and convenient editing of previously printed documents.

Data

The dataset for this project originates from kaggle kernels. Available for free and can be downloaded from here.
All the datasets are CSV files representing the image pixels values and their corresponding label.

- **Description:**

  Arabic Letters Dataset is composed of 16,800 characters written by 60 participants, the age range is between 19 to 40 years, and 90% of participants are right-hand. Each participant wrote each character (from ’alef’ to ’yeh’) ten times. The images were scanned at the resolution of 300 dpi. Each block is segmented automatically using Matlab 2016a to determining the coordinates for each block. The dataset is partitioned into two sets: a training set of 13,440 characters to 480 images per class and a test set of 3,360 characters to 120 images per class. Writers of training set and test set are exclusive. Ordering of including writers to test set are randomized to make sure that writers of test set are not from a single institution to ensure variability of the test set.

Algorithms

Feature Engineering

1. Convert csv values to an image to visualizing some examples and verify the data set
2. Data Preprocessing

   - Image Normalization : between 0–1 without loss of information.
   - Reshape the data to ensure that the input data to the model is in the correct shape.

3. Encoding Categorical Labels

   One-hot encoding transforms integer to a binary matrix where the array contains only one ‘1‘ and the rest elements are ‘0’.

4. Designing Model Architecture

   - Model Summary And Visualization
   - Parameters Tuning
   - Training the Model
   - Test the Model

Models

CNN Convolutional neural network were used with all possible parameter combination.

Model Evaluation and Selection

After observing the accuracy of all these combinations, the ones with the highest accuracy were chosen. The best parameters are:

Optimizer: RMSprop

Kernel_initializer: uniform

Activation: linear

The entire training dataset of 16800 Arabic characters were was split into 80/20 train vs. All CNN parameter combination. 4 out of 24 of different parameter combination are listed below. were calculated with epochs=5, batch_size=20.

Optimizer: RMSprop , kernel_initializer: uniform , activation: linear

- Loss: 0.7705
- Accuracy: 0.7382
- val_loss: 0.3556
- val_accuracy: 0.8833

Optimizer: Adam, kernel_initializer: uniform , activation: relu

- loss: 0.7203
- Accuracy: 0.7578
- val_loss: 0.4107
- val_accuracy: 0.8705

Optimizer: Adagrad , kernel_initializer: uniform , activation: tanh

- loss: 3.3758
- accuracy: 0.1000
- val_loss: 2.7361
- val_accuracy: 0.1852

Optimizer: Nadam, kernel_initializer: normal, activation: relu

- loss: 0.7634
- accuracy: 0.7404
- val_loss: 0.4217
- val_accuracy: 0.8577

Tools

- Numpy and Pandas for data manipulation
- Keras for modeling
- Matplotlib for plotting

Communication

In the slides