CMP450 – Semester 20/21

Text Independent Writer Recognition Using LBPH & SVM

January 17, 2021

Team #1

Mahmoud Othman Adas SEC:2, BN:19* and Yosry Mohamed Yosry SEC:2, BN:33† and
Ahmad Mahmoud AbdElMen'em Afifi SEC:1, BN:5‡ and Abdulrahman Khalid Hassan SEC:1, BN:30§

Department of Computer Engineering, Cairo University
Email: *mahmoud.ibrahim97@eng-st.cu.edu.eg, †yosry.mohammad99@eng-st.cu.edu.eg,
‡Ahmed.Afifi98@eng-st.cu.edu.eg, §abdulrahman.elshafie98@eng-st.cu.edu.eg

***Abstract*—**

We put alot of effort on speeding up the training process. The most effective optimization was parallelizing the feature extraction by extracting each image's features in a seperate process and then collecting all the features before training.

Processes are quite heavy, but python threads are totally useless, thanks to `GIL`'s locking mechanism. We believe that if we port the code to another language, the execution time would be much lower using threads and manual memory allocation.

Python lists are known to be very slow, so we replaced them all with numpy arrays, and allocated most of the needed memory ahead before all training. A quite speed gain came from fine tuning `skimage` and `OpenCV` parameters.

We tried to use `Numba` and `Cython` to optimize the execution time but they didn't have an effect. It was probably becaue most of the code calls numpy, `skimage` and `OpenCV`, which are all written in C and well optimized for memory and cpu.

VIII. Unsuccessful Trials

We started with and settled on using `LBPH` for feature extraction and `SVM` for classfication. They both gave around 80% accuracy at the beginning, and with tuning for preprocessing the accuracy reached ~99% over large tests. During that, part of the team were testing other feature extraction methods and classfiers.

We tried to extract *Histogram of Oriented Gradients* (`HOG`) features. Using `HOG` gave very low accuracy (~46%) on 15 test cases. We extracted `HOG` from the binary image and then gray image, with no visible changes.

Then we tried to extract the *Hu Moments* that are used to describe the shapes. We extracted *Hu Moments* from each binary line in the image.

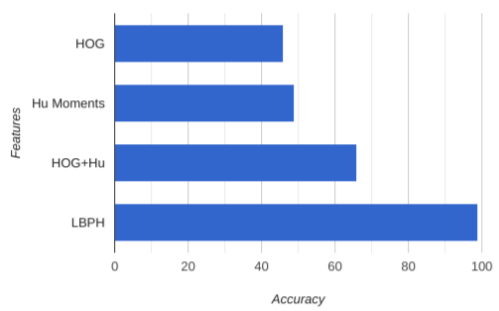

Figure (1) Feature Extraction Methods Accuracy for 15 tests.

Using *Hu Moments* with `SVM` gave accuracies lower than that of `HOG` on the same number of test cases.

Being very low made sense, because we tried to describe the shape of the whole line. So we tried to extract *Hu Moments* from a sliding window of size 13×13. This gave accuracy of $\tilde{4}8\%$ on 15 tests. On some lucky iterations, it gave $\tilde{8}0\%$.

Then we tried to mix both `HOG` features and *Hu Moments*. This gave us accuracy of $\tilde{6}6\%$ on 15 tests. It wasn't slower than only `HOG`, because we used subset of both features.

By this time, `LBPH` reached $\tilde{9}9\%$ accuracy on 200 tests. So we abandoned optimizing the feature extraction anymore.

Figure 1 shows the accuracies for the different features.

We tried another classfication method beside `SVM`. We used *K-Nearest Neighbours* (`KNN`), and it gave the same accuracies of `LBPH` but was noticably slower. It made sense that `KNN` is slower, because it iterates over the features multiple times to find the mean and cluster them.

IX. Future Work

X. Workload Distribution

A. Mahmoud Othman Adas

- Optimized speed with multiprocessing.
- Tried `HOG`, `Hu Moments` for feature extraction.
- Tried `KNN` for classification.
- Tried `Numba` for jit compiling.

B. Yosry Mohamed Yosry

- TODO

C. Ahmad Mahmoud AbdElMen'em Afifi

- TODO

D. Abdulrahman Khalid Hassan

- TODO

XI. Conclusion