# Handwriting Teaching Program

Jorden Jolley

Columbia University, Visual Interfaces Final Project Proposal.

**Abstract**

Paper motivates the need for learning good penmanship and handwriting and suggests a Python program for teaching a user good handwriting by evaluating and providing feedback on images of their handwriting, following prompts.

**Keywords:** Learning Development, OpenCV, Python, Handwriting

## 1 Introduction

### 1.1 The Phenomena

Although we spend so much time on keyboards, the importance of learning handwriting goes beyond penmanship. Learning to write letters helps children with their letter recognition skills and with quicker reading. In addition, there is evidence that learning to write by hand helps students retain information by activating neural pathways associated with strong reading skills [1]. Additionally, handwriting can also help with memory retention and reading comprehension [1].

When learning new words, one research study showed that participants who handwrite the words performed better on the recognition test compared to those who typed the words. The N400 component of the EEG (an index of semantic processing in the brain) was larger and more sustained for handwritten words. This suggests that handwritten words are processed more deeply and are easier to retrieve from memory [2].

We have established the usefulness of handwriting in a modern world. But does the quality of penmanship matter? Research studies such as "The Effects of Handwriting Practice on Functional Brain Development in Pre-Literate Children" by James and Engelhardt [3] have shown that teaching children to write neatly and legibly does have positive effects on their brain development. For adults, the Journal of Learning

disabilities found that improving handwriting skills could also lead to improved reading skills [4].

### 1.2 Learning Handwriting

How might one learn handwriting effectively? While incorporating multisensory activities (tracing the letter with a finger, writing in the air, or speaking the letter out loud) are suggested, a key part of learning letters is repetition and muscle memory, in addition to receiving feedback [5]. Useful feedback includes suggestions on size, stroke order, spacing, and alignment with other letters.

Finally, the best order to teach letters in handwriting is grouped by their similarity [6].

In the modern age, computers could help children or adults seeking to improve their handwriting skills. This paper suggests a program which gives writing instructions ("Write the letter A five times on a piece of paper"), analyzes the user's image, and returns feedback based on similarity scores of the user's letters to the "ideal" letter, in addition to comments on size and spacing of the input letters. The user will be instructed to repeat the process, receiving feedback on their best letters and letter confidence scores, until they reach a set threshold of their choice.

The program is an example of visual interfaces by way of analyzing the user-generated images and comparing them to set images and extracting textual elements from the images in addition to generating results and feedback based on the images' similarities. In addition, the program will need to identify each letter from a page and compare sizing and spacing between the letters on the page.

## 2 Limits

### 2.1 Font and Case

Discussions surrounding cursive often come hand-in-hand with penmanship and the importance of handwriting. However, this initial iteration of the program will focus on non-cursive language in a simple font similar to those of handwriting books for children in upper and lowercase. The letters will be in English.

### 2.2 Environment

The images will not be in real-time, but rather uploaded to the program aligning with prompting from the user. This is in part due to the complications for users to record well whilst handwriting, versus just taking a photo after. The user will interact with the program via a terminal interface for this iteration of the project. In addition, the user environment should be a controlled domain: a jpg photo of their drawn letters in dark ink on an unlined white paper. This is in order to control the quality of the input for the best possible program recommendations. It should be relatively easy for most users to produce the required materials and domain. Poor sizing and letter distance will be included in program feedback, so major issues in quality of writing should be addressed quickly.

Jupyter notebook will be required to run the program iteratively and add images while interacting with a session. Future iterations of the project could include a more user-friendly app environment, however, app development is not in scope for this course.

# 3 Implementation

For the ideal letter images, I will screenshot and convert to jpg Arial font letters. There will be 26 capital letters and 26 lowercase letters, with a total of 52 ideal letter images. These ideal images will be used if Algorithm 4 is implemented.

## 3.1 Personal Imagery

For building and testing the system, I will iterate through training runs for at least 10 letters. Each training run will consist of writing the same letter five times according to the prompted instructions. For each training run, I will attempt to write at least one iteration of "bad" handwriting which is not easily recognizable, followed by a slightly improved iteration and a near-perfect iteration. Therefore, ideally each letter test iteration should take at most 3 images, but may likely take many more at the beginning as I adjust and write the algorithm.

## 3.2 Example Input

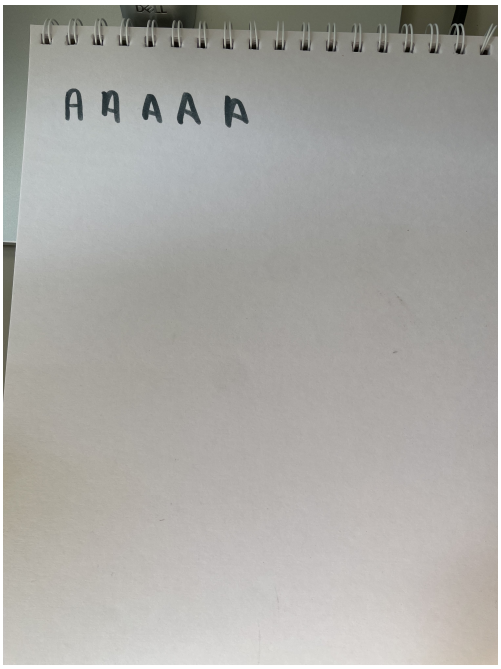

**Fig. 1** Example user input image - not cleaned

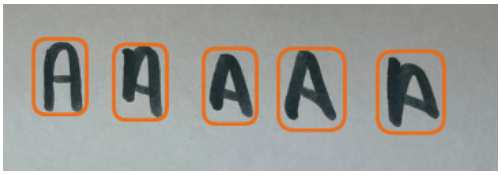

**Fig. 2** Example with letters identified - real image will be binary and cleaned

## 3.3 Suggested Data Structures

| | |
|---|---|
| `practice_letters` | a list of all the letters to practice with, in the following order: L, F, E, H, T, I, D, B, P, U, J, C, O, G, Q, S, R, A, K, M, N, V, W, X, Y, Z, l, t, i, c, k, o, p, s, v, u, w, x, z, h, n, m, r, b, a, d, g, q, e, f, j, k, y |
| `capital_ideal_letters` | a list of the JPG images of the ideal letters, capitalized. Length of 26. Each letter can be identified by its index in alphabetical order. This will be used if Algorithm 4 Letter Confidence algorithm is used. |
| `lowercase_ideal_letters` | a list of the JPG images of the ideal letters, lowercase. Length of 26. Each letter can be identified by its index in alphabetical order. This will be used if Algorithm 4 Letter Confidence algorithm is used. |
| `current_image` | the current image uploaded by the user that the program is providing feedback on. |
| `size_feedback` | a list of strings for size feedback for each of the five input letters. Length of 5. |
| `spacing_feedback` | a list of strings for spacing feedback between each of the five letters. Length of 4. |
| `confidence_feedback` | string of confidence feedback for the set of letters. |
| `confidence_scores` | a 2d numpy array with 2 rows and 26 columns. The first row represents each of the capital letters, while the second represents the lowercase letters. The scores are floats. |
| `all_confidence_scores` | a list holding each float confidence scores array for each of the five input letters. |

## 3.4 Suggested Algorithms

The following pages suggest some outlines for algorithms. I present two options for calculating the letter confidence - Algorithm 4 and Algorithm 5. During implementation, I will try both options and see which performs better (runs quickly while returning the "best" letters as highest confidence). The main process will run as outlined below. Subject to change.

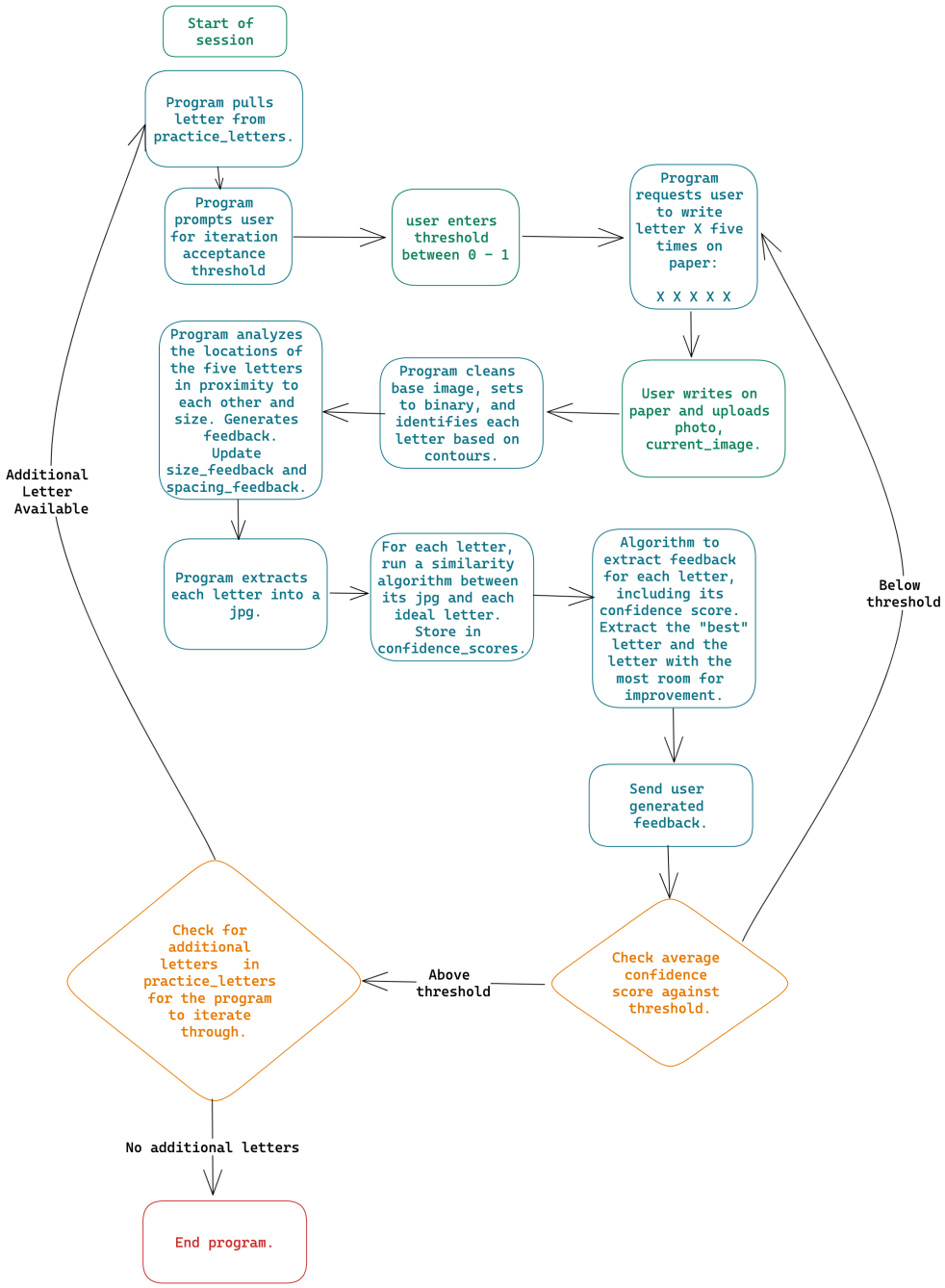


**Fig. 3** Potential control sequence for the main program

**Algorithm 1** Base Image Analysis Algorithm

```
requires image uploaded by user
do data reduction
threshold image
convert to binary
pull contours
filter out smallest contours
identify the 5 letters in the image
```

**Algorithm 2** Size Feedback Algorithm

```
Require: image with letters identified in bounding boxes
    while letters to identify > 0 do
        Calculate area of the letter
        Store area of the letter
    end while
    Calculate average area
    Identify outliers
    Return feedback
```

**Algorithm 3** Spacing Feedback Algorithm

```
Require: image with letters identified in bounding boxes
    while letters to identify > 0 do
        Calculate distance to next letter
        Store distance of the letters
        Pull recommended distances for letters
        Append feedback
    end while
    Return feedback
```

**Algorithm 4** Letter Confidence Algorithm - Option 1

```
Require: image with letters identified in bounding boxes
    while letters to identify > 0 do
        while letters in ideal letters list > 0 do
            calculate similarity based on two images using Shape detection algorithm
            and/or foreground-background computations
            store confidence
        end while
    end while
    pull max confidence character
    return max confidence character with confidence level
    detect match
    generate feedback
```

**Algorithm 5** Letter Confidence Algorithm - Option 2

**Require:** image with letters identified in bounding boxes
    **while** letters to identify $> 0$ **do**
        Text detection with Tesseract image to string function
        Extract individual characters with Tesseract image to boxes
        store confidence
    **end while**
    pull max confidence character
    return max confidence character with confidence level
    detect match
    generate feedback

# 4 Evaluation Metric

Acceptable performance, generally, will be measured via improvement scores over the course of the program. How much did confidence scores increase via each iteration? Confidence scores give a image-by-image performance score, but confidence scores will speak to the larger scale improvement. In addition, I will rank the letters beforehand and ensure that the confidence measure generally aligns with the rankings (3/5 correct and the 1 is always correct).

To test the program, there will be at least 3 separate users tested to ensure the algorithms are not overfit to my own personal handwriting. Acceptance here will include the user being able to achieve a score above a reasonable difficulty threshold (which will need to be identified through testing) within at least three iterations per letter, with ideally two iterations per letter.

Failures could include the program being extremely difficult to achieve comparability/confidence in the letters. This could mean the aforementioned overfitting issue, or could require better image analysis/comparison algorithms. Perhaps the current algorithm is having a hard time differentiating between the letters, etc.

# 5 Management

General outline for completion:

| | |
|---|---|
| `Round 1 - April 5 - 9:` | Set up virtual environment, github, and major elements such as the data structures and feedback loop with the user. |
| `Round 2 - April 10 - 15:` | Develop and implement the sizing and spacing algorithms. Test iteratively - test program as-is with at least 1 user. |
| `Round 3 - April 16 - 26:` | Develop and implement the similarity algorithm. Test iteratively - test program as-is with at least 1 user. |
| `Round 4 - April 27 - 2:` | Final rounds of testing and alterations. Test with at least 2 more unique users. Write paper. Prepare for presentation. |
| `Project Due: May 2nd` | Turn in project. |

# References


[1] James, V. K. Berninger: Brain research shows why handwriting should be taught in the computer age. LDA Bulletin **512**(1), 25–30 (2019)

[2] A. S. Ihara, A.K.K.I.K.O. Kae Nakajima, Naruse, Y.: Advantage of handwriting over typing on learning words: Evidence from an n400 event-related potential index. Frontiers in Human Neuroscience **15: 679191** (2021)

[3] Karin H. James, L.E.: The effects of handwriting experience on functional brain development in pre-literate children. Trends Neuroscience Education (2014)

[4] Montgomery, D.: The Contribution of Handwriting and Spelling Remediation to Overcoming Dyslexia. Middlesex University, London and Learning Difficulties Research Project, Essex, UK (2012)

[5] Syahputri, D.: The effect of multisensory teaching method on the students' reading achievement. Budapest International Research and Critics in Linguistics and Education (BirLE) Journal (2019)

[6] Engel, L.K.Z.S. C.: Handwriting in early childhood education: Current research and future implication. Journal of Early Childhood Literacy (2015)