Handwriting Helper

Jorden Jolley

The context

- Learning to write by hand helps with letter recognition, reading speed, memory retention, and reading comprehension.
- Handwriting words can activate neural pathways associated with strong reading skills.
- Handwritten words are processed more deeply than typed words.
- Teaching children to write neatly and legibly has positive effects on brain development and can improve reading skills for adults.

Can we automate the handwriting feedback process?

Handwriting skills are important. But how do we improve?

Program overview

- Automate the feedback process, applying real-life teaching strategies and research-backed handwriting guidelines
- Program prompts user to enter an acceptance threshold and write a letter 5 times in a row on paper
- User upload photo to receive feedback
- Program provides feedback on letter thickness, spacing, identifiability, skew, and size
- Continue through all letters, repeating same letter until you meet threshold

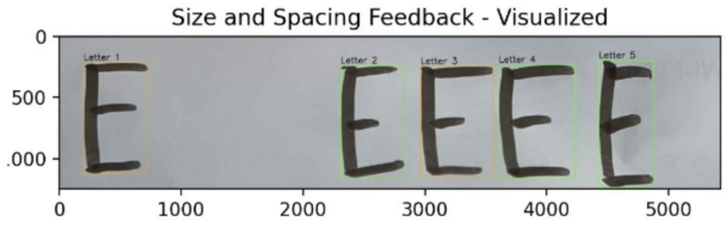
Limits and Assumptions

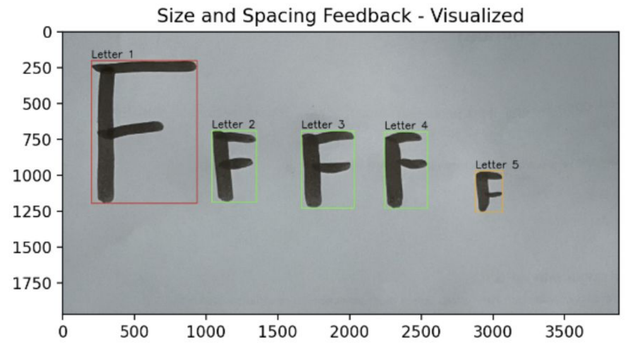
- Focuses on non-cursive language with uppercase and lowercase letters in English. Font size and spacing are calculated relative to other letters.
- Users upload images rather than use real-time input, and the user environment should be a controlled domain of dark ink on an unlined white paper. HEIC photos must be converted to jpg.
- The program works best on a blank sheet of paper with a sharpie marker, and future development should include lined paper and more real-time uploading.

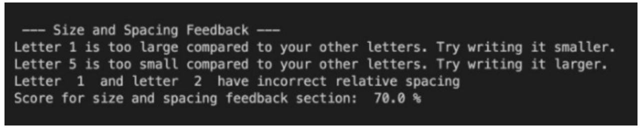
Size and Spacing Feedback Algorithm design

Identify letters which are relative outliers for size and spacing.

- Image processing and cleaning
- Originally planned on using mean, but outliers were skewing the data too much
- Shifted to finding letters outside of the range of ⅓ of the median area for size, and 50 pixels of the median for spacing.
- Thresholds adjusted from trial and error







```
--- Size and Spacing Feedback ---
Letter 1 is too large compared to your other letters. Try writing it smaller.
Letter 5 is too small compared to your other letters. Try writing it larger.
Letter  1  and letter  2  have incorrect relative spacing
Score for size and spacing feedback section:  70.0 %
```

Line Thickness Feedback Algorithm design

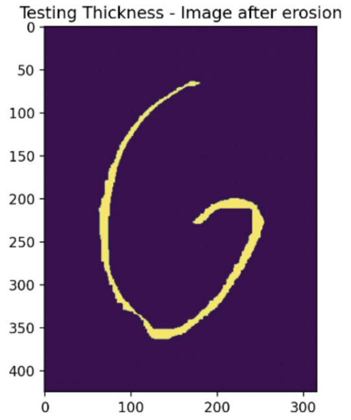
Identify letters which are are too thin/thick.

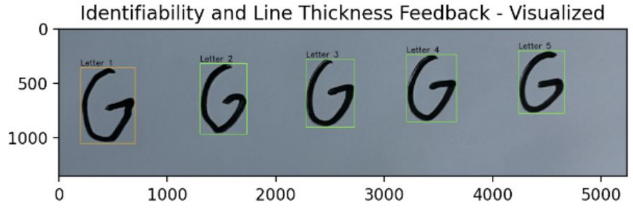
- Considered algorithms that are an adaptation of barcode algorithms, but depends on straight lines

- Tried calculating by counting average line width, but lines in letters are not separated out and often curved.

- Discovered the idea of erosion as it relates to thickness: developed a standard for how much of a letter should be left after 4 rounds of erosion

- Can pull relative amounts of black/white before and after and see how much your image “eroded”

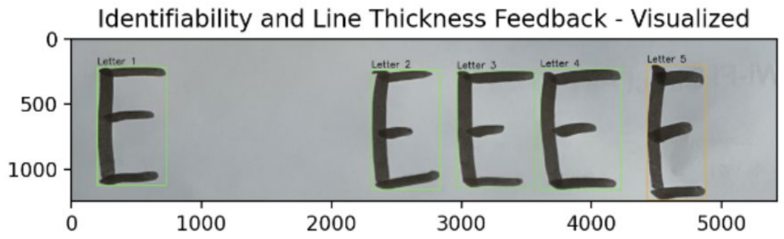


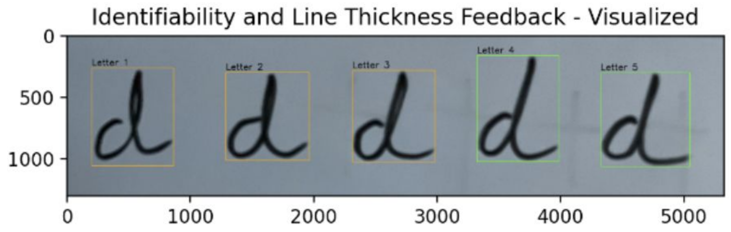


Identification Feedback Algorithm design

Is the user writing the requested letter, and is it recognizable?

- Considered training an image recognition AI program, decided to use one of the OOB solutions
- Tesseract optical character recognition engine – image to string function pulls text from image.
- Many difficulties actually getting letters recognized due to image business
- Applied blurring, configured Tesseract for “image as a single character” and cut out the images to just the bounding rectangles.





Unfortunately finicky on lowercase letters and very straight letters despite trying every configuration.

Slant Feedback Algorithm design

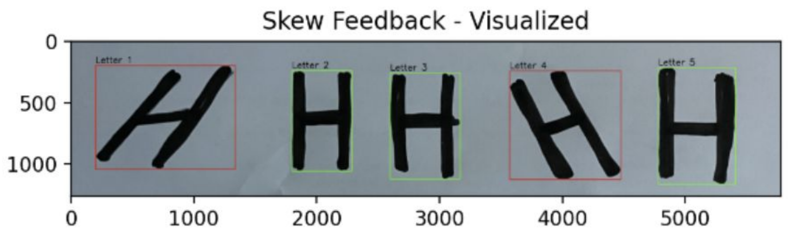
Identify letters that are too skewed.

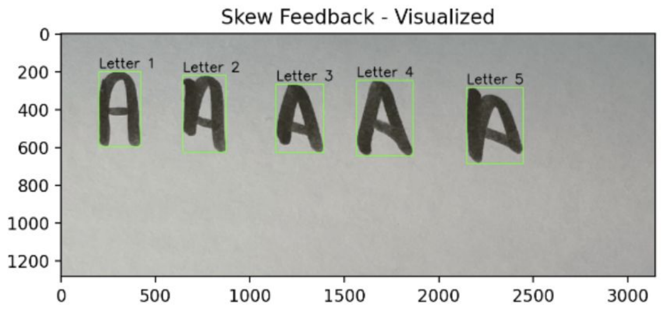
- Originally wanted to use symmetry, but would require a unique acceptability threshold for each letter. Not scalable and ripe for overfitting. Skew is the **opposite** of symmetry, along a horizontal.

- Tried calculating by rotating the image until identifiable with identifiability algorithm and calculating rotation amount

- Shifted to minAreaRect, which can calculate the angle of a rectangle wrt the horizontal axis. Applied function to MBR after cutting out the bounding rectangle from letters

- Defined an acceptable skew angle through trial and error





Evaluation and Reflection

Testing Results

- Program makes it easy to see your improvement via scores
- Used testing to generate thresholds
- Generally found approx. 20% improvement going into second rounds
- Essential to choose a challenging threshold

Insights Gained

- Learned about generating skew, using erosion for line thickness, OCR with Tesseract
- Learned about combining research with creating a real product
- Translating human intuition/opinion (defining a "good" letter) into computer language is hard

Next Time

- Letter recognition too finicky - more user testing to narrow down thresholding
- Add support for lines to provide feedback on letter position relative to lines
- Better GUI and user experience (app?)
-

Demo Time