Optical Character Recognition

Our goal is to try and extract the text from an image of a paragraph. The image will be a screenshot of a paragraph with the text in Arial 20pt. We have narrowed the problem down to this specific format to eliminate as many variables as possible to reduce the complexity of the task.

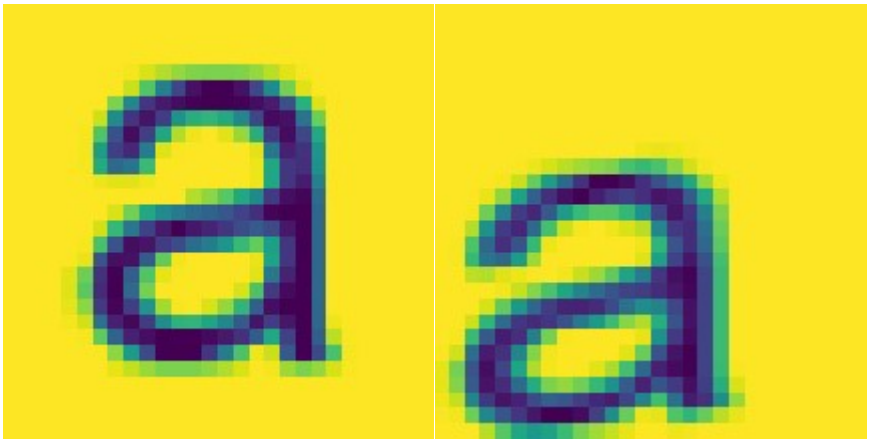
We generated a very small dataset to train our model. To avoid overfitting the model we will be randomly augmenting the images. We will talk more about this later. The dataset will consist of 94 images, 1 for each printable ascii character excluding spaces. To improve our accuracy, we obtained the character images from a screenshot of printable ascii characters in a specific format to make extraction easier. The process to extract characters for our dataset is similar to how we will be isolating characters when extract the text from an image of a paragraph.

To correctly label the characters we needed to obtain the images of the characters in a predictable order. After converting to a binary image and inverting the colours, we can dilate the text horizontally and obtain the contours for each row. Sorting the contours by their y-axis gave us a predictable order for the rows.

In order to correctly obtain two-part characters within the rows, we need to dilate the image vertically.

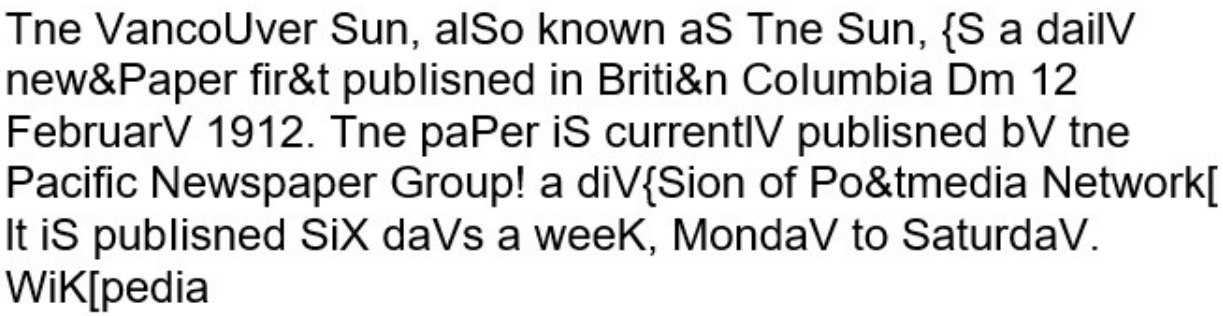
After obtaining the contours for each character, we sorted them by their x-axis. This gave us the locations of the characters in order from left to right. Now we can save the images and create a dataset with the path to the image along with its label.

To train our model with a limited amount of data we augmented our existing images. Using the ImageDataGenerator function from Keras we were able to specify the width and height shift range, zoom range, and shear range. This means that every epoch of our model during training received newly altered image of the original. We trained the model for thousands of epochs so to simulate a larger dataset.



We used a convolutional neural network that was implemented using Keras from TensorFlow. The model consists of 2 convolutional layers and 2 fully connected layers. To prevent overfitting, we added a dropout layer to drop 33% inputs. Keras requires our labels to be in a one hot encoding format. To achieve this, we will be using LabelBinarizer from scikit-learn. The model was trained in batches so we can test its performance. The provided model took around 5000 epochs and was the most accurate we were able to achieve.

To grab the text from an image of a paragraph, we needed to maintain the ordering of characters and words. This was achieved in a similar process to how we extracted characters for our dataset. In addition to obtaining the ordering for the rows and characters, we need to obtain the ordering for each word. This is similar to how we extracted the rows but with different weights for the dilation. The high-level explanation for the entire process is as follows. For each row we isolate each word. For each word we isolate each character and run it through our model. The results, in one hot encoding format, are transformed back into a character and appended to an existing string and printed once all texts have been translated.



The Vancouver Sun, also known as The Sun, is a daily newspaper first published in British Columbia on 12 February 1912. The paper is currently published by the Pacific Newspaper Group, a division of Postmedia Network: It is published six days a week, Monday to Saturday. Wikipedia

Comparing our results to the original we can spot some misclassifications. Some of the most noticeable misclassifications are n instead of h, V instead of y, & instead of s, and uppercase versions of correctly classified letter. Some of the harder to spot misclassifications with this font involve uppercase i with lower case L.

The hardest challenge we encounter was filtering the image to isolate characters. Different fonts and font sizes require different values for dilation. We ran into problems where 2 characters would merge into a single image and 2-part characters being represented as 2 different characters. With more time we would have liked to include more fonts so the model is not as limited and improve the overall performance of the model so it could more accurately convert the image to text.

“Accomplishment Statement”

Developed a convolutional neural network for optical character recognition. I had processed the images with OpenCV such that I was able to obtain the characters and maintain the ordering of those characters from an image. I had also built and trained the neural network to recognize individual characters with a small dataset by augmenting the existing images. At the end I was able to produce legible text from an image of a paragraph.