Optical Character Recognition via Deep Learning

Matias Arola and Connor Meany
matiasa@stanford.edu, cmeany@stanford.edu

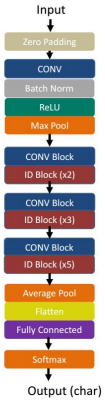
Summary

Given the ease of handwriting and large quantity of existing handwritten text, converting handwriting to computerized text is a problem of great importance with many applications. We created both a character level and word level neural network to recognize handwriting. Our data came from the EMNIST dataset (characters) [2] and the IAM dataset (words) [5]. The character-level model utilizes a ResNet-50 structure and achieved 88% accuracy. Our word-level model uses a 3-layer CNN which feeds into an LSTM layer; this achieved 77% accuracy at a character level. The main problems that we encountered were character segmentation and normalizing word length.

Model

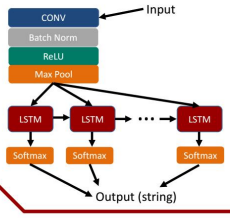
Character-Level Model:

The character model (right) uses a residual convolutional neural network based on the ResNet-50 model [4]. We used Keras to structure the model. The output of the model is a softmax which aims to match a one-hot vector which classifies the character. To optimize we used a cross entropy loss function and Adam optimization; we also used mini-batches of size 32.



Word-Level Model:

The word model (below) uses a convolutional neural network attached to a recurrent neural network which uses "Long Short Term Memory" (LSTM) blocks [3]. The output is a string of softmax characters trying to match one-hot vector labels. This model also uses a cross entropy loss function and Adam optimization, along with mini-batches of size 32.



Data

Character Images:

We used character images from the EMNIST Balanced dataset—a training set of 112,800 and test set of 18,800 [2]. This dataset contains preprocessed 28 x 28 pixel grayscale images of characters and numbers with labels (converted to one-hot vectors of size 47).

Label: 40 (f)
Prediction: 40 (f)

Label: 8
Prediction: 8

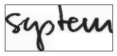
Label: 44 (q)
Prediction: 9

Word Images:

The word images came from the IAM dataset [5]. There were originally 115,320 grayscale images. We removed images with bad segmentation, non-alphabetical characters, and more than 6 letters. We split the remaining 64,876 images 90-5-5 into train, dev, and test sets. The images were of irregular shape, so we padded and processed them for our model; we also converted the label to a one-hot vector representing the letters. This label matrix has shape (n_{cw}, m, n_{cc}) where n_{cw} is the number of possible characters in each word, m is the number of training examples, and n_{cc} is the number of of possible characters to choose from.

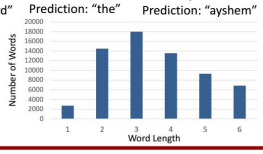
Label: "Lord"
Prediction: "hard"

Label: "the"
Prediction: "the"

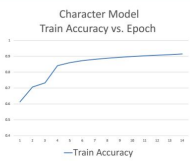


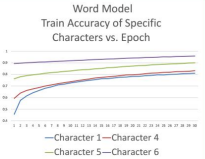
Label: "system"
Prediction: "ayshem"

Distribution of Word Length in Training Dataset:



Results





Model	Initial Character CNN	ResNet Character CNN	1-layer Word CNN-RNN	3-layer Word CNN-RNN
Train Accuracy	89.9%	90.6%	88.5%	83.7%
Test Accuracy	84.8%	88.5%	71.6%	76.9%

We found that the ResNet worked the best for identifying characters, and the 3-layer CNN to RNN worked the best for identifying words.

Discussion:

We initially planned on using the character model to implement the word model using character segmentation, but this method was too unreliable and produced a very low accuracy rate. The greater challenge, it turned out, was identifying characters from within words. We then developed the RNN model. We initially only used 4-6 character words, but the model overfit; using 1-6 character words improved performance. We adjusted hyper-parameters throughout this entire process because the dataset took a surprisingly short amount of time to process.

References

[1] Balci, Batuhan, Dan Saadati, and Dan Shiferaw. "Handwritten Text Recognition using Deep Learning." *CS231n: Convolutional Neural Networks for Visual Recognition, Stanford University, Course Project Report, Spring* (2017).

[2] Cohen, Gregory, et al. "Emnist: an extension of mnist to handwritten letters." *arXiv preprint arXiv:1702.05373* (2017).

[3] Gers, F.A.; Schmidhuber, J.; Cummins, F.: 'Learning to forget: continual prediction with LSTM', IET Conference Proceedings, 1999, p. 850-855, DOI: 10.1049/cp:19991218 IET Digital Library, http://digital-library.theiet.org/content/conferences/10.1049/cp_19991218

[4] He, Kaiming, et al. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016.

[5] U. Marti and H. Bunke. The IAM-database: An English Sentence Database for Off-line Handwriting Recognition. Int. Journal on Document Analysis and Recognition, Volume 5, pages 39 - 46, 2002.

Future

Our next steps would be to implement a post-processing neural network which analyzes the output of the word model. The output generally matched the shape of the word, but it still had errors. One way to correct these errors is by finding the closest valid English word and correcting the letters. Other than this, we might try using word segmentation software to separate words out from a larger image of text and then running them through our model. This would allow us to process entire pages of text at a time. While character segmentation didn't work due to the conflation of characters with neighbor characters, words are usually more separate from each other.