Handwriting Recognition and Conversion Using Neural Networks

Aditya Saini, Kunal Sant, Sumeet Swain, and Neha Deshmukh

Abstract In the current world of automation, everything is getting atomized as reducing manual labor is the key to efficiency. Here, we focus on offline handwriting recognition. Handwriting recognition is the process of extracting text from handwritten scripts, this is also known as offline handwriting recognition. The purpose here is to attempt to improve the accuracy and efficiency of the system using neural networks and datasets used for training the model, as well as detecting and identifying the characters and exporting them in a text format. The proposed system can be used to recognize handwritten characters and convert them into the text from the scanned image of a page.

Keywords Neural networks · Handwriting recognition · Long short-term memory · Optical character recognition · Convolutional neural network

1 Introduction

Handwriting recognition is a subject which existed for a long period of time, but the accuracy of the process can be improved by the usage of more refined technology and better techniques that reduce the error rates and the difference between the actual text and the detected text of the system [1]. Handwriting recognition is the process of extracting text from handwritten scripts, and this is also known as offline handwriting recognition [2]. In offline handwriting recognition, the image of a script is captured using a scanner and sent forward for further processing [3].

A. Saini (✉) · K. Sant · S. Swain · N. Deshmukh
A P Shah Institute of Technology, Thane(w), Maharashtra, India
e-mail: adityasaini@apsit.edu.in

K. Sant
e-mail: kunalsant@apsit.edu.in

S. Swain
e-mail: sumeetswain@apsit.edu.in

N. Deshmukh
e-mail: npdeshmukh@apsit.edu.in

S. Fong et al. (eds.), *ICT Analysis and Applications*, Lecture Notes in Networks and Systems 517, https://doi.org/10.1007/978-981-19-5224-1_16

This is a step into the world of automation and can help us improve efficiency in this field. Handwriting recognition works on similar techniques of text detection and recognition. Text detection includes various techniques to extract text from an image, such as the texture detection method and connected component method. In texture detection, the text is extracted, which is detected by texture difference from the background, and text is registered by further processing of the information. In the connected component method, the text is extracted using the relation between pixel connectivity and pixel intensity to detect text patterns, and checks are made to clear non-character elements [4]. The next step in extracting text from an image is text recognition which is done by optical character recognition (OCR) which depends on identifying the character from the detected text, which can be done by training the model using a dataset that gives the model the basic characteristics of characters like the height, width, shape and size and font style. This includes all the alphanumeric characters, 10 digits (0–9), and 26 alphabets with uppercase and lowercase for a total of 62 characters [3]. Handwritten characters are more challenging to recognize as compared to printed characters as they carry more parameters like the standard and cursive style of writing [4]. These challenges can be overcome by using new technologies like long short-term memory (LSTM), recurrent neural network (RNN), convolutional neural network (CNN), and different training methods with various datasets [5]. The handwriting can be extracted from an image through this set of processes.

RNNs are popularly used in generating data for sequences in music text and motion capture, and they can be trained by real data. A particular property of RNNs is that they can create new data with the help of existing data, i.e., interpolation (a statistical method).

In principle, a sufficiently large RNN should be sufficient to generate strings of arbitrary complexity. In practice, however, it has been observed that standard RNNs cannot store information about past inputs for very long periods of time due to their short memory. This has a significant effect on their ability to perform well in long-span structures. This is known as “amnesia” behavior, which makes them susceptible to instability when creating strings. The suggested remedy is to introduce noise into the predictions before feeding them back to the model, which will increase the internal efficiency of the model to handle any unexpected inputs that may arise. However, we believe that increasing system memory is a deeper and more efficient solution.

The long-term short-term memory network is LSTM. A long-term memory network is a type of cyclic neural network (RNN). The LSTM architecture is driven by error stream analysis in existing replay neural networks. Memory blocks are a set of recursively connected blocks used in the LSTM layer. These blocks can be thought of as distinguishable versions of memory chips in digital computers. Each of these blocks contains one or more repeatedly connected memory cells and three multipliers: input, output, and forget gate. They provide persistent analogies of cell write, read, and reset operations. The interaction between the network and the cell can only happen through ports (Fig. 1).

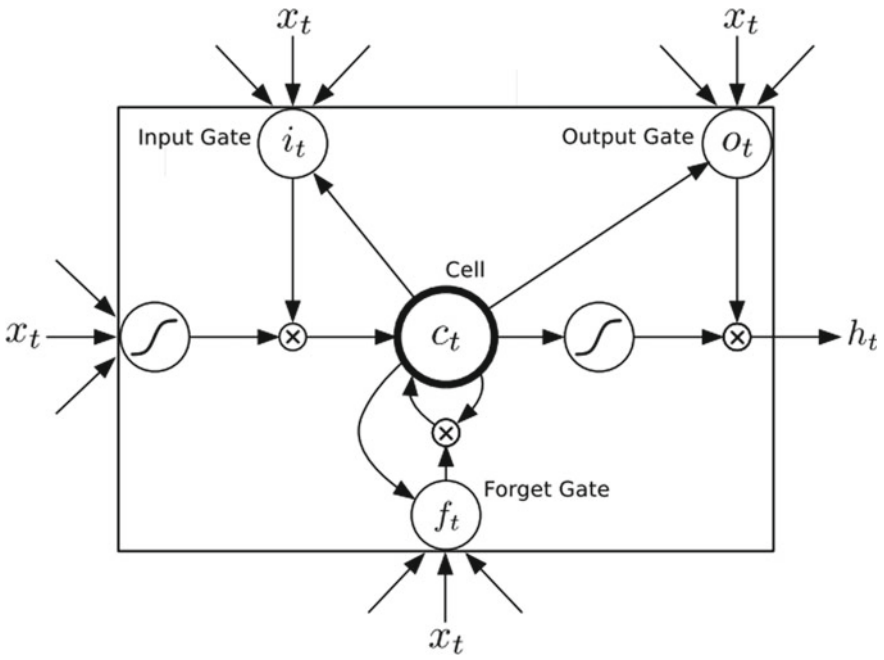

Fig. 1 Long short-term memory cell, along with its three multiplicative units, that is: the input, the output and the forget gates

2 Literature Review

In paper [3], use of techniques such as preprocessing, segmentation, and feature extraction that are used to improve the accuracy of text recognition and identification. In paper [4], the work of different text detection methods on how they extract text from an image such as MSER and canny edge integration, stroke width variation, and OCR for text recognition. In paper [5], combination of CNN and LSTM showing a better performance for offline handwriting recognition. In paper [6], how using CNNs and their training with GPUs to train models can reduce character recognition error rates in offline handwriting. In paper [7], we learned that LSTM-RNN with backpropagation through time training can be used to speed up the process while increasing the accuracy of the recognition process.

3 Limitations

Quality factors such as noise in the image, blurred images, and in general low quality of the image and text can hinder the performance, leading to mistakes in the end result [4].

Similar looking characters can lead to misclassification by the system (1, l, i, I) (o, 0, O) (2, z, Z) (s, S, 5) (A, 4) [6]. Currently, the system works on an image of a set resolution and aspect ratio for the working. The system can only recognize characters that are trained to the system, so if the system is trained for alphabets, then it won't be able to recognize numbers and special characters and punctuation marks.

4 Previous Work

Researchers have tried increasing the efficiency of the text detection process by increasing the size of the sliding window, which is used to detect text [4]. Neural networks with various training techniques with datasets have also improved the speed and accuracy of the process [7].

5 Implementation

Handwriting to text

CNN: The user's data is initially placed into CNN layers. To extract the features from the image, CNN layers are employed. Five input layers are used, with the first two applying a 5×5 filter and the last three applying a 3×3 filter for convolution operations. A non-linear RELU function is implemented after the input passes through the five layers of CNN. Finally, image sections are summarized using a layer, and a reduced output is generated. Despite the fact that the image is shrunk by two in each layer, a 32×256 feature map is added.

RNN: The RNN passes important information to the feature sequence, which comprises 256 features each time step. The popular long short-term memory (LSTM) implementation of RNNs is employed because it can transmit information over longer distances and has more powerful training properties than a standard RNN. The RNN output sequence is mapped into a 32×38 matrix. The IAM dataset contains a total of 79 different characters, plus one extra character for the CTC process, for a total of 80 entries for each of the 32 time steps.

Connectionist Temporal Classification (CTC): The CTC is supplied to the RNN output matrix and the ground truth text during training the neural network, which computes the loss value. The CTC is only provided the matrix while inferring, and it decodes the final text using that information. Ground truth and recognized text can only be 32 characters long.

Input Data: The input data is a 128×32 pixel picture. Because the images in the collection aren't exactly this size, we resize them to a width of 128 pixels or a height of 32 pixels. The input is then transformed into a 128×32 target image (white).

Data augmentation can be implemented by arbitrarily moving or enlarging the image rather than aligning it to the left.

CNN Output: Each entry in the output has 256 characteristics. The RNN layers further process the features that we have received. However, some of the features already have a strong relationship with the input image's high-level qualities. Some features have a strong relationship with duplicate characters (for example, "tt"), specific characters (for example, "e"), or character qualities such as loops and curves (for example, in handwritten "l"s or "e"s).

Implementation using Tensorflow:

There are four modules in the implementation.

To prepare the photos from the dataset for the neural network, use SamplePreprocessor.py.

DataLoader.py is a Python module that reads samples, organizes them into batches, and provides an iterator interface for traversing the data.

Model.py: Creates a model from a given architecture, loads and saves models, controls the session at the same time, and provides a training and inference interface.

main.py: This file combines all of the previously described modules. The other source files are dealing with basic file IO, input and output (DataLoader.py), and picture processing, so we simply look at Model.py (SamplePreprocessor.py) (Figs. 2 and 3).

Text to Handwriting

Input Data: This model will be fed with computer-generated text as its input. We used online handwritten data to assess the model's ability to construct plausible real-valued sequences, as opposed to offline handwriting, which only has page pictures. We got all of our information from IAM-OnDB, which is an online handwriting database.

LSTM: Backpropagation via time was used by the LSTM method to build a full gradient. When using the whole gradient to train LSTM, one issue is that the derivatives might get too big, causing numerical issues. To avoid this, we clipped the

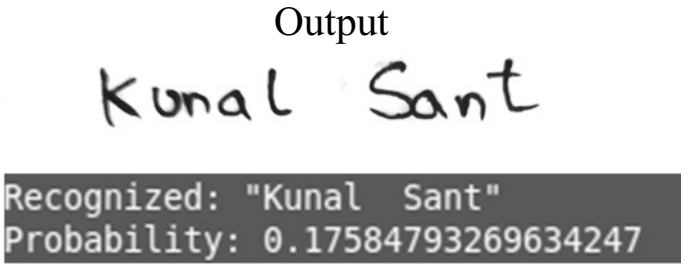

Fig. 2 The input text "Kunal Sant" given to the model and its output, along with the accuracy that we have achieved, is shown

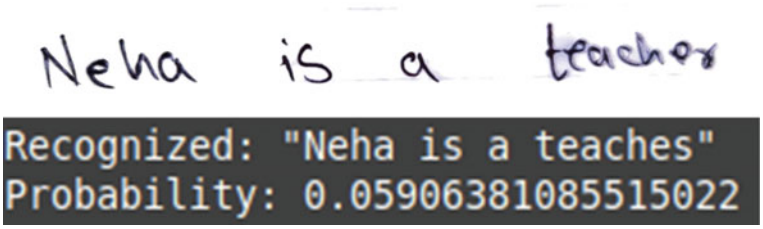

Fig. 3 The input text "Neha is a teacher" given to the model and its output, along with the accuracy that we have achieved, is shown

derivative of the loss with respect to the network inputs to the LSTM layers in all of the tests such that it stayed within a preset range.

6 Conclusion

We want to improve the efficiency of handwriting recognition. We talked about a neural network that can recognize text in photos. The neural network outputs a character-probability matrix and has five CNN and two RNN layers. This matrix is either used to calculate CTC loss or to decode CTC. TensorFlow is used in the implementation.

Acknowledgements We thank Prof. Neha Deshmukh for helping us in creating this research paper and guiding us throughout this project. We also thank Prof. Kiran Deshpande, Head of the Department of Information Technology, for his help in our work.

References

1. Chherawala Y, Roy PP, Cheriet M (2015) Feature set evaluation for offline handwriting recognition systems: application to the recurrent neural network model. IEEE Trans Cybern 46(12)
2. Senior A, Robinson T (1998) An off-line cursive handwriting recognition system. IEEE Trans Pattern Anal Mach Intell 20(3)
3. Pradee J, Srinivasan E, Himavathi S (2011) Diagonal based feature extraction for handwritten character recognition system using neural network. In: 2011 3rd international conference on electronics computer technology (ICECT), vol 4
4. Islam MR, Mondal C, Azam MK, Islam ASMJ (2016) Text detection and recognition using enhanced MSER detection and a novel OCR technique. In: 2016 5th international conference on informatics electronics and vision (ICIEV)
5. Suryani D, Doetsch P, Ney H (2016) On the benefits of convolutional neural network combinations in offline handwriting recognition. In: 2016 15th international conference on frontiers in handwriting recognition (ICFHR)

6. Ciresan DC, Meier U, Gambardella LM, Schmidhuber J (2011) Convolutional neural network committees for hand-written character classification. In: International conference on document analysis and recognition
7. Doetsch P, Kozielski M, Ney H (2014) Fast and robust training of recurrent neural networks for offline handwriting recognition. 2014 14th international conference on frontiers in handwriting recognition (ICFHR)
8. Hu J, Brown MK, Turin W (1996) HMM based online handwriting recognition. IEEE Trans Pattern Anal Mach Intell 18(10)
9. Graves A (2013) Generating sequences with recurrent neural networks, pp 1–43. arXiv preprint arXiv:1308.0850. Retrieved from https://arxiv.org/abs/1308.0850
10. Tappert CC, Suen CY, Wakahara T (1990) The state of the art in online handwriting recognition. IEEE Trans Pattern Anal Mach Intell 12(8)
11. Zheng Y, Li H, Doermann D (2004) Machine printed text and handwriting identification in noisy document images. Pattern Anal Mach Intell 26(3)
12. Graves A, Liwicki M, Fernández S, Bertolami R, Bunke H, Schmidhuber J (2009) A novel connectionist system for unconstrained handwriting recognition. IEEE Trans Pattern Anal Mach Intell 31(5)
13. Bahlmann C, Haasdonk B, Burkhardt H (2002) Online handwriting recognition with support vector machines—a kernel approach. In: Proceedings of the 8th international workshop on frontiers in handwriting recognition (IWFHR)
14. Marti U, Bunke H (1999) A full English sentence database for off-line handwriting recognition. In: Proceedings of 5th international conference on document analysis and recognition, Bangalore, India
15. Xu L, Krzyzak A, Suen CY (1992) Methods of combining multiple classifiers and their applications to handwriting recognition. IEEE Trans Syst Man Cybern 22