ICEST-2K24 [International Conference on Engineering, Science and Technology]
In Association with
International Journal of Scientific Research in Science, Engineering and Technology
Print ISSN - 2395-1990 | Online ISSN : 2394-4099 (www.ijsrset.com)

Hand Riter: A Handwriting Generation Tool

Ravina Malsikhare, Parth Kulkarni, Digvijay Kokne, Dev Lot
Department of Artificial Intelligence & Data Science, AISSMS Institution of Information Technology, Pune, Maharashtra, India

ABSTRACT
HandRiter is an AI tool which converts digital text into handwritten sequences. The implementation and theory of HandRiter is based on the paper 'Generating Sequences with Recurrent Neural networks' [1] by Alex Graves. HandRiter uses Recurrent Neural Networks (RNN) with Mixture Density output layer to predict parameters of a distribution. The deep learning architecture behind HandRiter is composed of RNN with stacked LSTM layers as hidden layers and mixture density layer as output layer. The deep learning model has been trained on Online Handwriting Dataset viz. IAM Online Handwriting Database. The architecture is able to generate sequences with high accuracy and is able to synthesize human-like handwriting. A user-friendly interface is used to make the usage of tool easy .

Keywords- LSTM, RNN, Online Handwriting, Deep Learning, Mixture Density Layer, Probability Distribution Parameters.

I. INTRODUCTION

In the contemporary digital landscape, characterized by the pervasive use of electronic devices for communication, there arises an increasing demand for technologies that effectively bridge the gap between traditional and modern forms of interaction. Handwriting, as a deeply personal mode of expression, retains its significance across diverse aspects of human life, ranging from the formal act of signing documents to the intimate exchange of emotions through handwritten notes. HandRiter emerges as a tool poised to preserve the timeless and personal essence inherent in handwritten communication within the context of today's digital era.

The methodology section of this paper elucidates the operational workflow of the HandRiter tool, delineating its constituent process modules, namely the preprocessing module, handwriting synthesis module, and visualization module. Additionally, it furnishes a concise overview of the underlying deep learning architecture employed by the tool. By scrutinizing these implementation intricacies, the subsequent sections of this paper will provide a detailed exploration of the design, functionality, and empirical results of the HandRiter tool.

II. METHODOLOGY

Dataset

(Online handwriting data, defined here as writing recorded as a sequence of pen-tip locations, contrasts with offline handwriting, where only page images are available.)All data utilized in this paper were sourced from the IAM online handwriting database (IAM-OnDB) generated by University ofBern [6].IAM-OnDB comprises handwritten lines obtained from 221 distinct writers utilizing a 'smart whiteboard'. Writers were instructed to transcribe forms from the Lancaster-Oslo-Bergen text corpus [2], with their pen positions tracked via an infra-red device situated in the board's corner. The use of online handwriting proves appealing for sequence generation due to its reduced dimensionality (two real numbers per data point) and straightforward visualization.

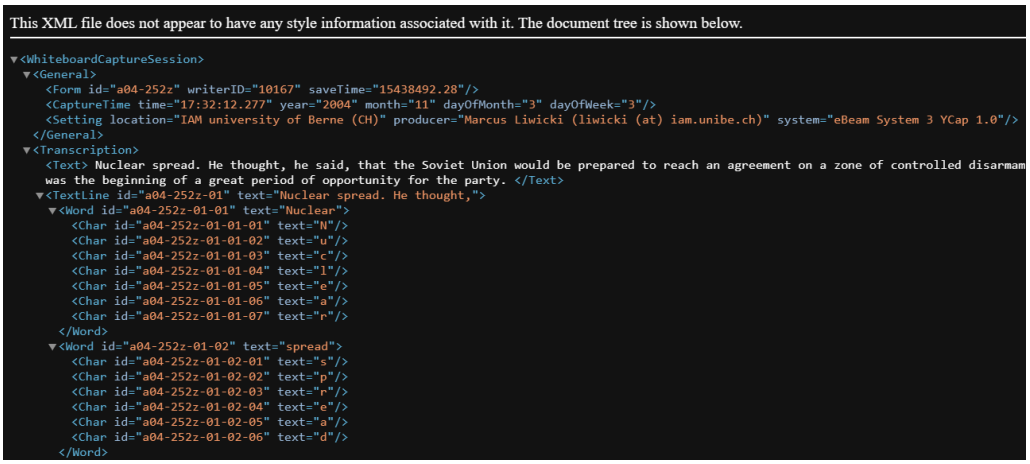


Fig. [1] XML File screenshot

The original input data includes the x and y pen coordinates, along with indicators denoting points in the sequence when the pen is lifted off the whiteboard. It consisted of 1561 XML documents which were a compiled form of information including writer name, writer id, written text, recorded pen-point coordinates, time to write, etc.

Neural Network Architecture

The Neural network architecture of HandRiter is depicted in Figure [2] which consists of;

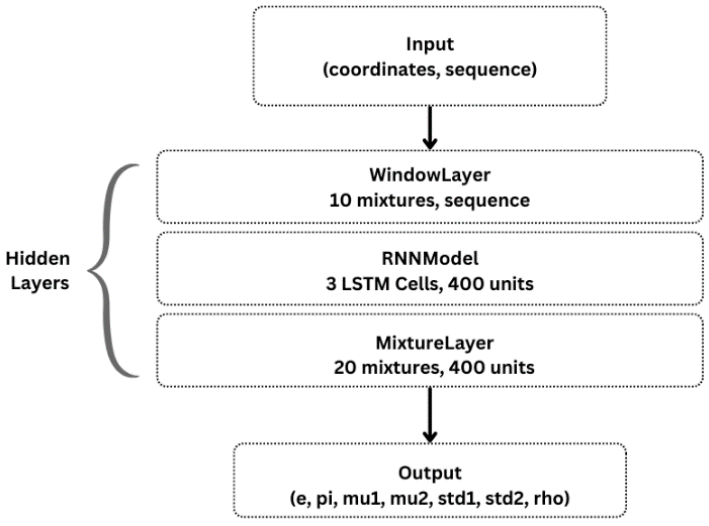


Fig. [2] NN Architecture

- Input Data:The input to the model consists of twomain components:

Coordinates: A sequence of (x, y) coordinates representing the position of points in the handwritten text.

Sequence: A one-hot encoded sequence of characters representing the text input.

- WindowLayer:The WindowLayer processes the inputsequence and computes weights to focus on different parts of the sequence. It generates a weighted sum of the sequence and provides additional information such as the last_phi, which represents the end of a stroke.
- RNNModel (LSTM Cells + Window):TheRNNModel comprises 3 stacked LSTM cells each with 400 units that process the input coordinates and the information from the WindowLayer. The LSTM cells capture sequential dependencies in the input data, allowing the model to understand the context of the handwriting.
- MixtureLayer (Mixture Density Net):TheMixtureLayer takes the output from the LSTM cells and computes parameters for a probability distribution. This distribution is a mixture of Gaussian components, and the parameters include means, standard deviations, correlations, mixture weights, and an additional parameter controlling whether to continue generating (Sigmoid activation).
- Output:The final output of the model is a set of parameters that describe a probability distribution for the next coordinate in the sequence. Here, for each generated stroke, the data includes:

mu1: Mean of the x-coordinate for the stroke.

mu2: Mean of the y-coordinate for the stroke.

std1: Standard deviation of the x-coordinate for the stroke.

std2: Standard deviation of the y-coordinate for the stroke.

rho: Correlation coefficient between the x and y coordinates of the stroke.

coord: The "end" information for the stroke, indicating whether the stroke is considered finished (1) or not (0).

Training& Generation

During training, the model was optimized to predict the parameters that best describe the distribution of the training data. The training process involved minimizing a Binary Cross Entropy loss function that compares the predicted distribution with the actual coordinates in the training data.The training phase included totally 50 epochs which gave the most satisfying results with an average training loss of0.0119.

Training Summary:

The following table provides the training summary of neural network (Trained for 50 epochs for a batch size of 64).

Loss Function : Binary Cross Entropy

Epoch	Average Training loss	Average Validation loss	Time Elapsed
1	2.1465	0.3796	522.2829s
2	0.6808	0.0910	1059.2167s
3	0.4648	-0.0855	1597.5622s
…	…		…
48	0.0266	-0.4273	26896.3895s
49	0.0204	-0.4377	27503.6155s

50	0.0119	-0.4588	28108.1501s

Table 1 Training Summary Of Neural Network

During the generation phase, the model was used to generate new handwriting by sampling from the learned distribution. The Sigmoid activation helps determine when to stop the generation process.

Following are some generated samples:
Prompts :{“Abstract Beauty”, “Artificial Intelligence”, “Hello, World!”, “Machine learning”}

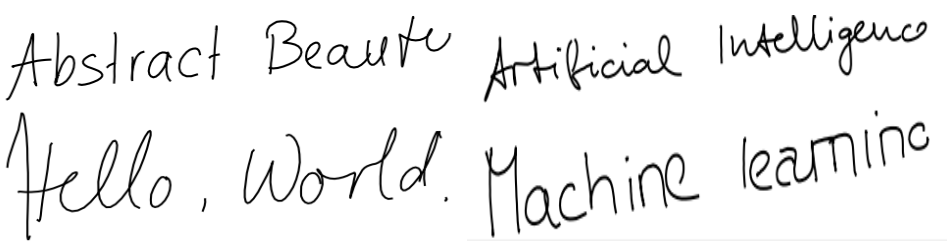


Fig [3] Samples Generated after Training

Overall, the model learns the patterns and dependencies in the handwritten text data and is capable of generating new sequences of coordinates that resemble handwritten text. The mixture density network allows the model to capture the uncertainty and variability in the handwriting, making it suitable for generating diverse and realistic handwritten sequences.

Operational Workflow

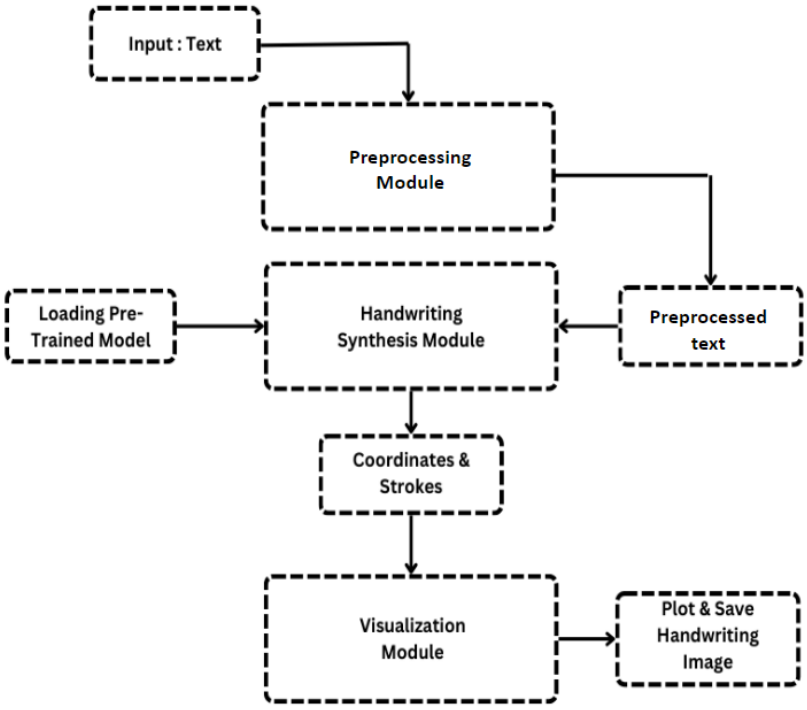


Fig. [4] Operational Workflow

The operational workflow of the tool is depicted in Figure [4], illustrating three main modules:

1. Preprocessing Module: Upon receiving input from the user via the User Interface (UI), the text undergoes immediate preprocessing before being forwarded to the Handwriting Synthesis module. This preprocessing step involves segmenting the input sequence into individual characters and subsequently applying one-hot encoding.
2. Handwriting Synthesis Module: Within this module, the pretrained weights of the Neural Network are utilized to sample text coordinates and strokes, leveraging distribution parameters obtained from the mixture density output layer.
3. Visualization Module: The Visualization module employs matplotlib, a data visualization library from Python, to generate visual representations of the handwritten sequences. This facilitates the graphical depiction and interpretation of the synthesized content.

III. RESULTS

Upon subjecting the Neural Network to rigorous training using the IAM Dataset for a total of 50 epochs, we achieved commendable outcomes. However, it is imperative to acknowledge that the attained results, while satisfying, were somewhat constrained by computational limitations. The intricacies and demands of the training process, coupled with the scale of the IAM Dataset, posed challenges that impacted the full realization of optimal performance. Despite encountering these computational constraints, the obtained results provide valuable insights and underscore the potential of the Neural Network.

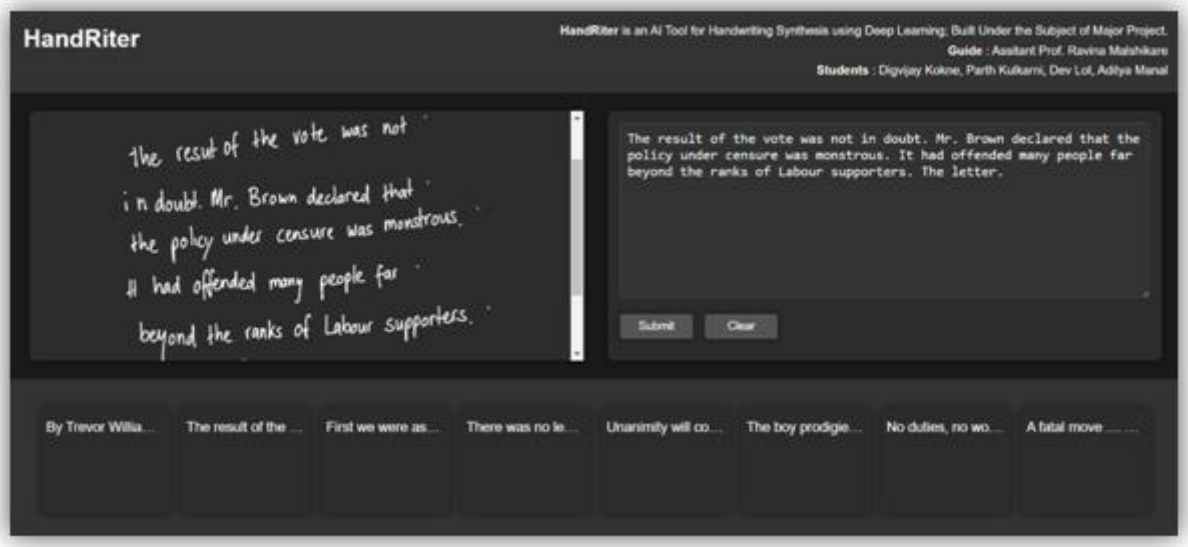
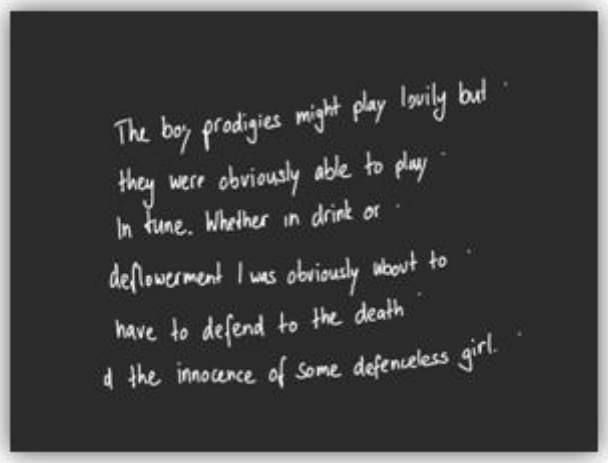


Fig. [5] HandRiter UI displaying generated handwritingusing pretrained weights

User Input: “The boy prodigies might play loudly but they were obviously able to play in tune. Whether in drink or deflowerment I was obviously about to have to defend to the death the innocence of some defenceless girl.”



IV. CONCLUSION

Despite the impressive results achieved after subjecting the Neural Network to rigorous training using the IAM Online Handwriting Dataset for a total of 50 epochs, it is imperative to acknowledge the impact of computational limitations on the attained performance. Limited amount of data and training period led to some discrepancies in the results such as addition of extra character '.' after end of each line and inclination of generated handwriting as number of characters in a particular line increase. But aside from that the system has given eventually good results.

It is worth noting that while our model demonstrated proficiency in generating sentences, its limitation to this scope highlights ample room for improvement. This limitation underscores the need for continued research and innovation in the field, aiming to broaden the model's capabilities and enhance its efficacy across diverse contexts of handwriting generation.

V. REFERENCES

[1]. Graves, Alex.: Generating Sequences With Recurrent Neural Networks.. arXiv:1308.0850 (2013)

[2]. S. Johansson, R. Atwell, R. Garside, and G. Leech.: The tagged LOB corpus user's manual; Norwegian Computing Centre for the Humanities(1986)

[3]. Brian Davis, Chris Tensmeyer, Brian Price, Curtis Wigington, Bryan Morse, Rajiv Jain.: "Text and style conditioned GAN for generation of offline handwriting lines." arXiv:2009.00678 (2020)

[4]. K. Kanda, B. Iwana and S. Uchida, "What is the Reward for Handwriting? — A Handwriting Generation Model Based on Imitation Learning," in 2020 17th International Conference on Frontiers in Handwriting Recognition (ICFHR), Dortmund, Germany, (2020)

[5]. Prem Chandra Singh.: Understanding the paper Generating Sequences with RNNs (by Alex Graves)(2021).https://medium.com/geekculture/understanding-the-paper-generating-sequences-with-rnns-by-alex-graves-18635cdd32be

[6]. Tolosana, Rubén, Paula Delgado-Santos, Andrés Pérez-Uribe, Rubén Vera-Rodríguez, Julian Fierrez and Aythami Morales.: "DeepWriteSYN: On-Line Handwriting Synthesis via Deep Short-TermRepresentations." AAAI Conference on Artificial Intelligence (2020)

[7]. M. Liwicki and H. Bunke. IAM-OnDB - an on-line English sentence database acquired from handwritten text on a whiteboard. In Proc. 8th Int. Conf. on Document Analysis and Recognition, volume 2, pages 956– 961, 2005.