ISSN: 2321-2152

IJMECE

International Journal of modern electronics and communication engineering

E-Mail
editor.ijmece@gmail.com
editor@ijmece.com

www.ijmece.com

ISSN2321-2152www.ijmece .com
Vol 11, Issue.4 Dec 2023

HAND WRITTEN TEXT DOCUMENT

C.Bavitha[1],J.Greeshma[2],B.Madhu Kumar[3],M.Gayatri[4]

ABSTRACT

In this rapidly evolving world every thing is getting digitalized, Exchange of information plays a crucial role in any work. Most of the times we may not find the facility to create a document so we use pen and paper but people may not understand every thing expressed on paper due to many reasons line bad penmanship etc. So we are building a platform through which one could convert the handwritten text to editable text. Our main aim is to classify the words directly and character segmentation. We use CNN for this process. We use long term memory networks(LSTM) with convolution to construct bounding boxes for each character. We divide the each given character and classify them and reconstruct for each word of neural networks for Handwriting character recognition.

I INTRODUCTION

Despite the abundance of technological writing tools, many people still choose to take their notes traditionally: with pen and paper. However, there are drawbacks to handwriting text. It's difficult to store and access physical documents in an efficient manner, search through them efficiently and to share them with others. Thus, a lot of important knowledge gets lost or does not get reviewed because of the fact that documents never get transferred to digital format.We have thus decided to tackle this problem in our project because we believe the significantly greater ease of management of digital text compared to written text will help people more effectively access, search, share, and analyze their records, while still allowing them to use their preferred writing method.

[1,2,3]B.Tech Student, Department of CSE (Cyber Security), Malla Reddy College of Engineering and Technology, Hyderabad, India.
[4]Assosiate Professor, Department of CSE (Cyber Security), Malla Reddy College of Engineering and Technology, Hyderabad, India.

The aim of this project is to further explore the task of classifying handwritten text and to convert handwritten text into the digital format. Handwritten text is a very general term, and we wanted to narrow down the scope of the project by specifying the meaning of handwritten text for our purposes. In this project, we took on the challenge of classifying the image of any handwritten word, which might be of the form of cursive or block writing. This project can be combined with algorithms that segment the word images in a given line image, which can in turn be combined with algorithms that segment the line images in a given image of a whole handwritten page.

With these added layers, our project can take the form of a deliverable that would be used by an end user, and would be a fully functional model that would help the user solve the problem of converting handwritten documents into digital format, by prompting the user to take a picture of a page of notes. Note that even though there needs to be some added layers on top of our model to create a fully functional deliverable for an end user, we believe that the most interesting and challenging part of this problem is the classification part, which is why we decided to tackle that instead of segmentation of lines into words, documents into lines, etc. We approach this problem with complete word images because CNNs tend to work better on raw input pixels rather than features or parts of an image [4].

Given our findings using entire word images, we sought improvement by extracting characters from each word image and then classifying each character independently to reconstruct a whole word.

II LITERATURE REVIEW

The literature review for this project on handwriting recognition and graphical user interfaces (GUIs) reveals a transformative shift from traditional Optical Character Recognition (OCR) methods to the application of deep learning techniques. Convolutional Neural Networks (CNNs) and Long Short-Term Memory (LSTM) networks have emerged as pivotal components, enabling the learning of intricate features from handwritten images and offering significant improvements in recognition accuracy. Connectionist Temporal Classification (CTC) has played a crucial role in facilitating end-to-end learning without the need for explicit alignment. Graphical user interfaces, particularly those developed using Tkinter, have provided an accessible means for users to interact with handwriting recognition systems, with applications spanning archival preservation, data entry, and document digitization. The review underscores the challenges posed by diverse handwriting styles, historical document analysis, and multilingual recognition, while also emphasizing privacy and security considerations. Overall, the literature survey establishes the context for this project's contributions in advancing the field of handwriting recognition through a user-friendly GUI-driven approach, filling research gaps, and addressing practical applications.

III METHODOLOGY

1. Data Collection and Preprocessing:

Collect a Dataset: Gather a substantial dataset of handwritten texts. This dataset should ideally cover a variety of handwriting styles, languages, and contexts.

Data Preprocessing: Clean the data by removing noise, normalizing sizes, and augmenting the dataset (rotation, scaling, etc.) to increase its size and diversity.

2. Feature Extraction:

Feature Selection: Extract relevant features from the preprocessed images, such as edges, corners, or statistical features like histograms.

Normalization: Normalize the features to ensure consistency and improve the training process.

3. Model Selection:

Choose an appropriate machine learning model for your task. Recurrent Neural Networks (RNNs), Convolutional Neural Networks (CNNs), or a combination of both (CRNN) are popular choices for text recognition tasks.
For more advanced applications, consider models like Transformer-based architectures.

4. Training the Model:

Split the dataset into training and validation sets.
Train the selected model using the training set and validate it using the validation set.
Experiment with hyperparameters, architectures, and regularization techniques to optimize the model's performance.

5. Post-Processing:

Implement post-processing techniques such as language models, spell checkers, or grammar checkers to improve the accuracy of the recognized text.

6. Evaluation:

Evaluate the model's performance using metrics like accuracy, precision, recall, and F1-score.
Iterate on the model and preprocessing steps based on the evaluation results to improve performance.

Fig 1:CNN

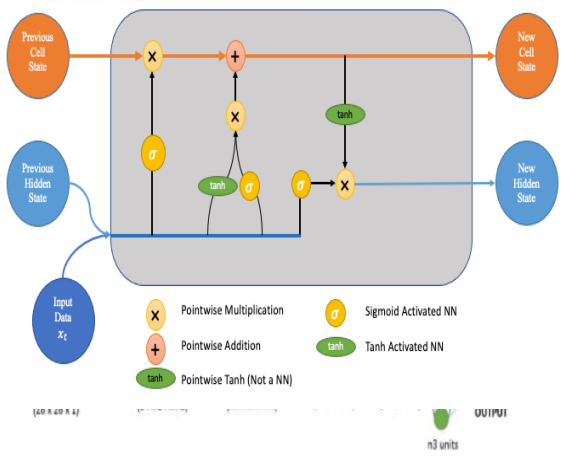

Fig 2:Long Short-Term Memory(LSTM)

IV IMPLEMENTATION

1. Acquire a dataset of handwritten text documents for training and evaluation.
2. Preprocess the dataset by resizing, normalizing, and augmenting the images.
3. Design a deep learning model for handwriting recognition, typically based on CNNs and LSTMs.
4. Implement Connectionist Temporal Classification (CTC) loss for sequence-to-sequence learning.
5. Train the model using the prepared dataset, monitoring performance with evaluation metrics like Character Error Rate (CER).
6. Utilize the Tkinter library to create a GUI for the application.
7. Design a user-friendly interface with components like buttons and labels.
8. Implement an image loading mechanism allowing users to input handwritten images.
9. Preprocess the loaded images to ensure compatibility with the model's input requirements.
10. Apply resizing and normalization to standardize image dimensions and pixel values.
11. Develop a function to feed the preprocessed image into the trained model for recognition.
12. Implement decoding strategies such as beam search or greedy decoding to convert model predictions into text.
13. Display the recognized text on the GUI, showing the prediction result alongside the loaded image.Use labels or text boxes to present the information clearly to the user.

V RESULTS EXPLANATION

During the initial training phase of the model, the following results were observed.

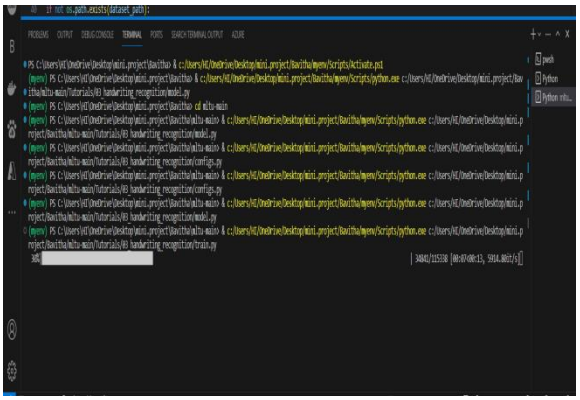

Fig 3: Installing Pakages

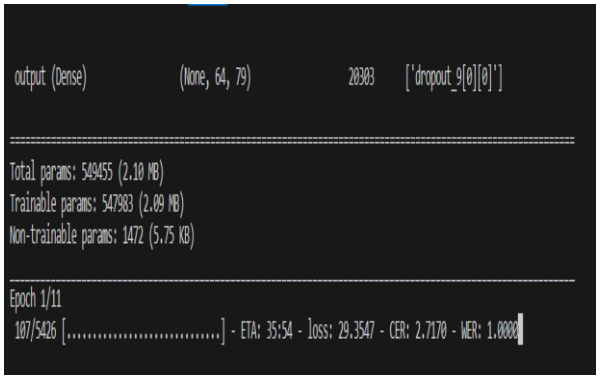

Fig 4:Epoch and Test Accuracy

Conclusion

In conclusion, the integration of Machine Learning Text Recognition (MLTR) techniques with Convolutional Neural Networks (CNNs) in handwritten text recognition models represents a powerful and promising approach. By leveraging the capabilities of MLTR and CNNs, we can achieve significant advancements in accurately transcribing handwritten text. Here are the key points to summarize the benefits and implications of employing MLTR and CNN in a handwritten text recognition system:

Accurate Character Recognition:

CNNs for Image Feature Extraction: CNNs excel at learning hierarchical features from images, capturing essential patterns and structures within handwritten characters.

MLTR for Contextual Understanding: MLTR techniques, such as language models and sequence modeling, provide a broader context for character recognition, enabling the model to recognize words and phrases accurately.

Robustness and Adaptability:

Robust to Variability: MLTR techniques enhance the model's adaptability to different handwriting styles, variations in writing, and complex textual layouts.

Data Augmentation: Both MLTR and CNN models can benefit from data augmentation techniques, enabling the system to generalize better even with limited training data.

End-to-End Learning:

End-to-End Learning: The combination of MLTR and CNN facilitates end-to-end learning, allowing the model to directly learn the mapping from raw input images to recognized text. This simplifies the overall system architecture and can lead to more efficient training and deployment processes.

Real-World Applications:

Versatility in Applications: Handwritten text recognition models employing MLTR and CNNs find applications in various fields, including document digitization, postal services, historical document preservation, and automated form processing.

Improved User Experience: Enhanced accuracy and robustness in recognizing handwritten text contribute to improved user experiences in applications such as note-taking apps, transcription services, and interactive educational platforms.

VI FUTURE ENHANCEMENT

Firstly, to have more compelling and robust training, we could apply additional preprocessing techniques such as jittering. We could also divide each pixel by its corresponding standard deviation to normalize the data. Next, given time and budget constraints, we were limited to 1000 training examples for each given word in order to efficiently evaluate and revise our model. Another method of improving our character segmentation model would be to move beyond a greedy search for the most likely solution. We would approach this by considering a more exhaustive but still efficient decoding algorithm such as beam search. We can use a character/word-based language-based model to add a

penalty/benefit score to each of the possible final beam search candidate paths, along with their combined individual softmax probabilities, representing the probability of the sequence of characters/words. If the language model indicates perhaps the most likely candidate word according to the softmax layer and beam search is very unlikely given the context so far as opposed to some other likely candidate words, then our model can correct itself accordingly.

VII REFERENCES

1. Alex Graves, Santiago Fernndez, Faustino Gomez, Jrgen Schmidhuber, Connectionist temporal classification: labelling unsegmented sequence data with recurrent neural networks, Proceedings of the 23rd international conference on Machine learning. 2006
2. Convolutional Neural Network Benchmarks: https://github.com/jcjohnson/cnn-benchmarks
3. Elie Krevat, Elliot Cuzzillo. Improving Off-line Handwritten Character Recognition with Hidden Markov Models.
4. Fabian Tschopp. Efficient Convolutional Neural Networks for Pixelwise Classification on Heterogeneous Hardware Systems
5. George Nagy. Document processing applications.
6. Mail encoding and processing system patent: https://www.google.com/patents/US542043
7. Kurzweil Computer Products. http://www.kurzweiltech.com/kcp.html
8. H. Bunke1, M. Roth1, E.G. Schukat-Talamazzini. Offline Cursive Handwriting Recognition using Hidden Markov Models.
9. K. Simonyan, A. Zisserman Very Deep Convolutional Networks for Large-Scale Image Recognition arXiv technical report, 2014
10. Lisa Yan. Recognizing Handwritten Characters. CS 231N Final Research Paper.
11. Oivind Due Trier, Anil K. Jain, Torfinn Taxt. Feature Extraction Methods for Character Recognition–A Survey. Pattern Recognition. 1996
12. Shaoqing Ren, Kaiming He, Ross Girshick, Jian Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. Neural Information Processing Systems (NIPS), 2015
13. Tesseract Model: https://github.com/tesseractocr/tesseract/wiki/TrainingTesseract-4.00 [14] Tesseract 4.0: https://github.com/tesseractocr/tesseract/wiki/4.0-with-LSTM
14. How many words are there in the English language?: https://en.oxforddictionaries.com/explore/howmany-words-are-there-in-the-english-language
15. Thodore Bluche, Jrme Louradour, Ronaldo Messina. Scan, Attend and Read: End-to-End Handwritten Paragraph Recognition with MDLSTM Attention.
16. T. Wang, D. Wu, A. Coates, A. Ng. ”End-to-End Text Recognition with Convolutional Neural Networks” ICPR 2012.
17. 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.