HANDWRITING TO TEXT CONVERSION

PROJECT DESCRIPTION

Taking notes by hand is a highly practical approach to grasp topics, and this method of learning can make it simpler for students to remember the material. But everything has a downside, so handwritten notes are only beneficial if they are stored neatly and carefully; otherwise, damaged pages and pages that seem unclean cannot be used for very long. To preserve them secure and current until the very last minute in a drive, the project concept of turning handwritten papers into digital ones came into mind. Thus, converting written characters to digital representation is becoming more and more common. The words on a piece of paper will eventually fade away, while a file kept on a computer can only be lost by deletion. It has become crucial to save any handwritten documents in digital format. This introduces us to the concept of handwriting to text conversion. This project aims to develop a robust and efficient system for converting handwritten text into digital text using advanced machine learning techniques. It is centered on leveraging machine learning techniques to create a system capable of converting written text into digital form so as to enhance accessibility and digitization of handwritten documents, making them readily usable in digital environments. The primary goal is to bridge the gap between traditional written documents and digital text, facilitating easier information retrieval and analysis. The project involves the development of a model based on Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) to accurately recognize and transcribe handwritten characters. The end goal is to provide a user-friendly solution with real-time processing capabilities, ensuring versatility across different handwriting styles and languages.

Systems that read handwritten letters, characters, and numbers enable humans to complete complicated tasks that would otherwise be time consuming and expensive to complete.

PROBLEM FORMULATION

The core challenge addressed by the Handwriting to Text Conversion project lies in the accurate recognition and transcription of diverse handwritten characters into digital text. The primary problem can be defined as developing a machine learning model capable of understanding and deciphering the intricacies of various handwriting styles, ensuring reliable and real-time conversion. Key considerations include handling different languages, adapting to varying writing patterns, and creating an intuitive interface for user interaction. Handwriting is inherently variable, with unique nuances in individual writing styles, making it a complex problem to solve. The key aspects of the problem formulation are:

1. Variability in Handwriting Styles: Handwriting exhibits a wide range of styles, influenced by individual preferences, cultural differences, and educational backgrounds. The model needs to be robust enough to recognize and adapt to the inherent variability, ensuring accurate transcription across diverse writing styles.
2. Multilingual Support: Multilingual support requires the model to generalize well, recognizing characters and patterns across different scripts and linguistic structures.
3. Real-time Processing: Achieving real-time processing is a crucial aspect of the problem formulation, as the system should be capable of providing instant transcription for practical usability.
4. User Interface Design: The formulation includes considerations for designing an intuitive and user-friendly interface to facilitate seamless interaction between the user and the system. The interface should accommodate the input of handwritten text and present the digital transcription in an easily accessible format.
5. Adaptability to Varied Input Formats: Handwritten content may be presented in various formats, including scanned documents, images, or even real-time input from stylus-based devices.

The project aims to strike a balance between computational efficiency and accuracy, ultimately bridging the gap between traditional handwritten documentation and the digital realm.

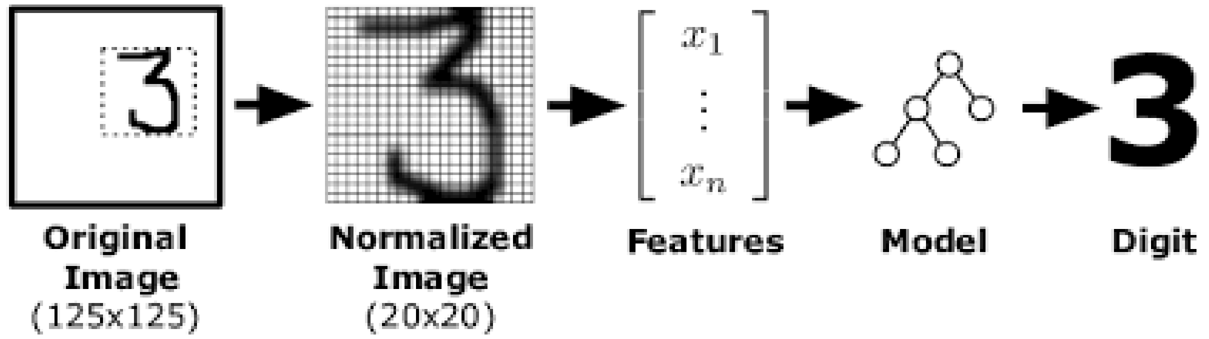
OBJECTIVE

The primary objective of this project is to develop a sophisticated machine learning model capable of accurately converting handwritten text into digital text format. Leveraging algorithms and deep learning techniques, the aim is to create a robust system that enhances accessibility for individuals with visual impairments by providing them with a reliable tool for transcribing handwritten content into easily readable text. Moreover, this model seeks to streamline data digitization processes across various industries, including education, healthcare, finance, and more, by automating the conversion of handwritten documents into machine-readable text, thereby improving efficiency and productivity. Through rigorous training and optimization, the goal is to achieve high precision in transcribing handwritten inputs into digital text, empowering users with enhanced access to information and communication resources.

RESEARCH METHODOLOGY

The handwriting recognition and conversion model implements several research methodologies to create a structured framework for the development and evaluation of the same. With the execution of the below given methods and procedures, the model is employed to collect, analyze, interpret, and draw conclusions from data in a rigorous and systematic manner.

Implemented using:
Python 3;
TensorFlow (version 2.10);
mltu==0.1.5



The IAM Dataset comprises handwritten text images, and the target associated with each sample is the corresponding text string within the image. Since the IAM dataset is commonly employed as a benchmark for OCR systems, utilizing this example can provide a valuable foundation for constructing your own OCR system.

Supervised Learning: Utilize a dataset of paired handwritten images and corresponding transcribed text labels to train a neural network model. Supervised learning techniques such as convolutional neural networks (CNNs) or recurrent neural networks (RNNs) can be employed for this purpose.

Unsupervised Learning: Explore unsupervised learning methods such as clustering algorithms or generative models to discover patterns and structures within the handwritten data. This approach may involve techniques like clustering handwritten strokes or generating synthetic handwritten samples.

Semi-Supervised Learning: Combine labeled data with a larger pool of unlabeled data to improve model performance. Techniques such as self-training or co-training can be employed to leverage both labeled and unlabeled data effectively.

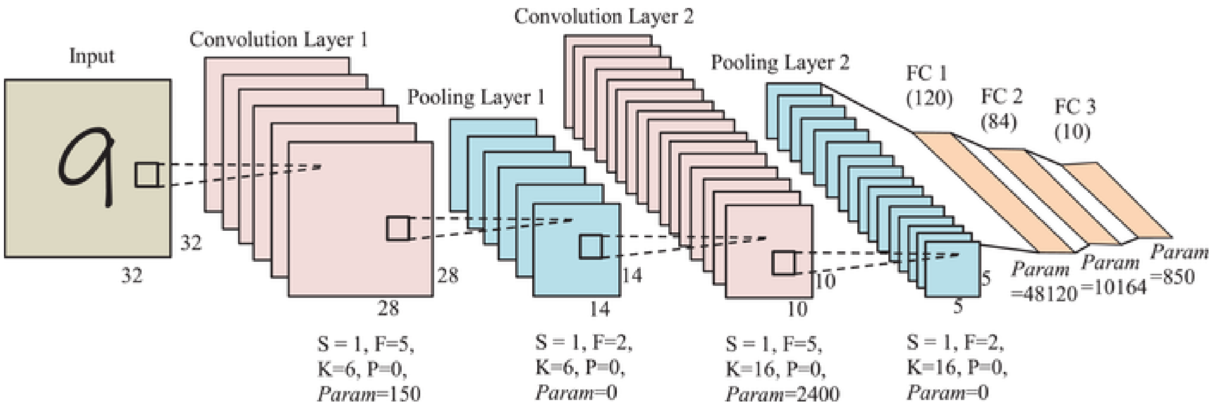
Transfer Learning: Utilize pre-trained models on large-scale handwritten text recognition tasks and fine-tune them on a specific dataset for handwritten-to-digital text conversion. This approach can leverage the knowledge learned from the pre-trained model to accelerate training and improve performance on the target task.

Data Augmentation: Employ techniques such as rotation, scaling, translation, and adding noise to augment the training dataset. This helps improve the model's robustness and generalization ability by exposing it to variations in handwriting styles and conditions.

Feature Engineering: Explore various handcrafted features or representations of handwritten text, such as histogram of oriented gradients (HOG), scale-invariant feature transform (SIFT), or deep learned embeddings, to capture relevant information for the conversion task.

Ensemble Learning: Combine multiple individual models to form a more robust and accurate ensemble model. Techniques such as bagging, boosting, or stacking can be employed to aggregate predictions from diverse models and enhance overall performance.

Domain Adaptation: Address domain shift between the source domain (e.g., standard fonts) and target domain (e.g., handwritten text) by adapting the model to perform well on the target domain. Techniques such as adversarial training or domain-specific regularization can be employed to mitigate domain discrepancies and improve model generalization.



Proposed Solution:

The proposed solution for this model aims to create an algorithm to serve predictions using a pre-trained Keras model for recognizing handwritten digits from the IAM_Words dataset. The algorithm aims to develop a robust and efficient system for converting handwritten text into digital text using advanced machine learning techniques. It is centred on leveraging machine learning techniques to create a system capable of converting written text into digital form so as to enhance accessibility and digitization of handwritten documents, making them readily usable in digital environments.
Below is a detailed breakdown of the process.

Requirements of the Code:

1. Python environment with required packages (Flask, TensorFlow, NumPy).

2. Pre-trained Keras model for recognizing handwritten digits (e.g., `handwritten_model.h5`).
3. Ability to send POST requests containing image data in JSON format to the Flask server.

Assumptions:

1. The pre-trained Keras model (`handwritten_model.h5`) is trained and saved correctly.

2. The input image data sent via POST request is properly preprocessed and formatted before being passed to the model.

3. The server is running locally, and requests are sent from the same machine or within the local network.

Algorithm :-

1. Data Preprocessing:
   Collect a dataset of handwritten images paired with their corresponding text labels.
   Preprocess the images to a consistent size and format (e.g., grayscale, fixed dimensions).
   Normalize pixel values to a range suitable for the neural network (e.g., 0 to 1).
   Split the dataset into training, validation, and test sets.

2. Model Architecture:
   Design a CNN architecture suitable for image classification tasks.
   Include convolutional layers followed by activation functions (e.g., ReLU) and pooling layers for feature extraction. Add fully connected layers for classification. Use dropout layers to prevent overfitting.

3. Training:
   Initialize the CNN model with random weights.
   Feed the training images into the model and compute the loss between predicted and actual labels using a suitable loss function (e.g., categorical cross-entropy).

Use backpropagation to update the model weights, minimizing the loss function via optimization algorithms such as stochastic gradient descent (SGD) or Adam.
Validate the model performance on the validation set during training to monitor for overfitting.

4. Evaluation:
   After training, evaluate the model's performance on the test set to assess its accuracy and generalization ability.
   Calculate additional metrics like precision, recall, F1-score, and word error rate (WER) if applicable.

5. Inference:
   Use the trained model to predict text labels for new handwritten images.
   Preprocess the new images similarly to the training data before feeding them into the model.
   Obtain the predicted text labels from the model's output layer.

6. Postprocessing:
   Perform any necessary postprocessing steps on the predicted text labels (e.g., correcting spelling errors, formatting).
   Output the final converted text.

7. Optimization and Iteration:
   Experiment with different model architectures, hyperparameters, and optimization techniques to improve performance.
   Iterate on the training process, adjusting as needed based on validation results and insights gained from evaluating model performance.

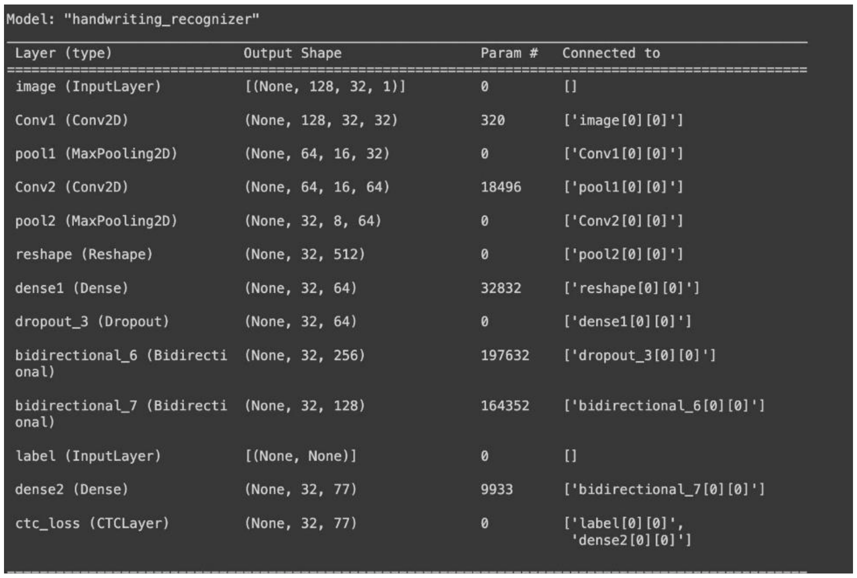
Mode of Operation:

The Flask server listens for incoming POST requests on the `/predict` route.
When a POST request is received, the server extracts the image data from the request.
The image data is preprocessed and passed through the pre-trained model for prediction.
The predicted class is returned as a JSON response.

Performance Evaluation:

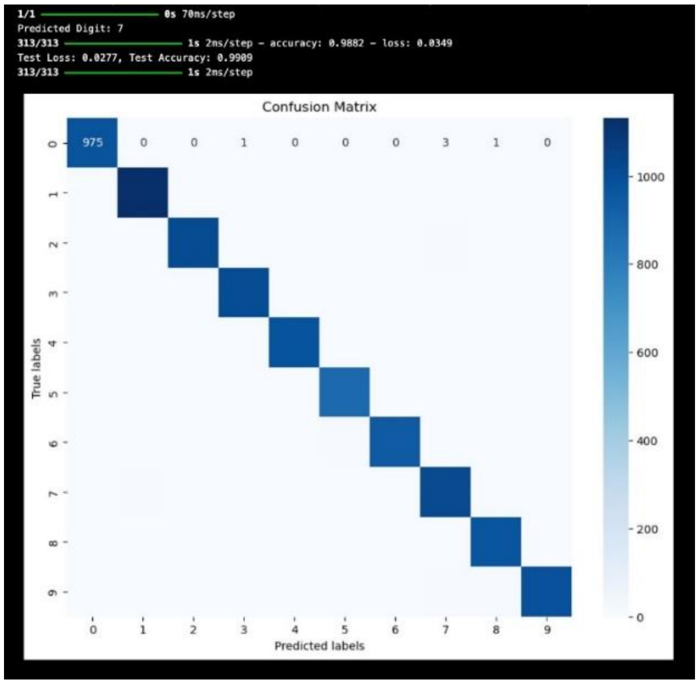
Performance Evaluation Matrix:-

Performance can be evaluated based on the speed and accuracy of predictions. Response time for each prediction request can be measured to assess server responsiveness. Accuracy of predictions can be evaluated using a separate test dataset or real-world data.



```
Model: "handwriting_recognizer"
 ______________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #   Connected to
==============================================================================================
 image (InputLayer)          [(None, 128, 32, 1)]         0         []

 Conv1 (Conv2D)              (None, 128, 32, 32)          320       ['image[0][0]']

 pool1 (MaxPooling2D)        (None, 64, 16, 32)           0         ['Conv1[0][0]']

 Conv2 (Conv2D)              (None, 64, 16, 64)           18496     ['pool1[0][0]']

 pool2 (MaxPooling2D)        (None, 32, 8, 64)            0         ['Conv2[0][0]']

 reshape (Reshape)           (None, 32, 512)              0         ['pool2[0][0]']

 dense1 (Dense)              (None, 32, 64)               32832     ['reshape[0][0]']

 dropout_3 (Dropout)         (None, 32, 64)               0         ['dense1[0][0]']

 bidirectional_6 (Bidirecti  (None, 32, 256)              197632    ['dropout_3[0][0]']
 onal)

 bidirectional_7 (Bidirecti  (None, 32, 128)              164352    ['bidirectional_6[0][0]']
 onal)

 label (InputLayer)          [(None, None)]               0         []

 dense2 (Dense)              (None, 32, 77)               9933      ['bidirectional_7[0][0]']

 ctc_loss (CTCLayer)         (None, 32, 77)               0         ['label[0][0]',
                                                                     'dense2[0][0]']

==============================================================================================
```

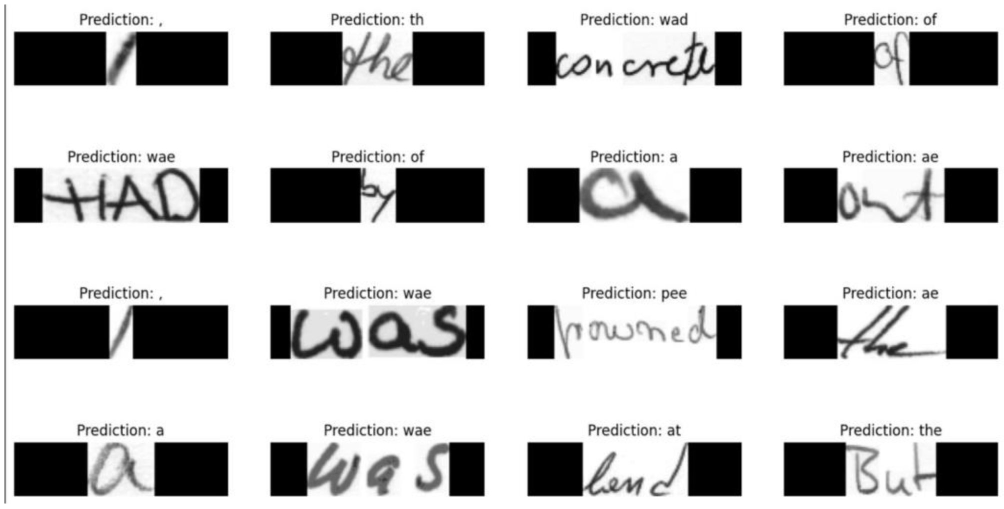
Keras Layers



```
1/1 ━━━━━━━━━━━━━━━━━━━━ 0s 70ms/step
Predicted Digit: 7
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step - accuracy: 0.9882 - loss: 0.0349
Test Loss: 0.0277, Test Accuracy: 0.9909
313/313 ━━━━━━━━━━━━━━━━━━━━ 1s 2ms/step
```

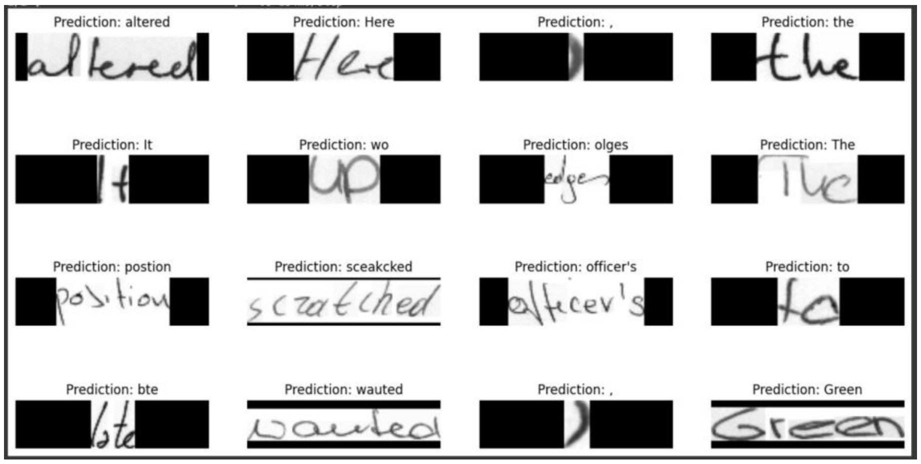
Performance matrix

Test Case:-



Predictions on 10 epoch iterations and training set of 8000 words

Here, it is observed that when presented with fewer iterations and a smaller dataset, the model lacks the depth of learning required to adequately capture the nuances of handwriting across different styles. Limited exposure to diverse examples during training impedes its ability to generalize effectively, resulting in weaker performance when confronted with handwriting styles. Consequently, this model exhibits lower accuracy and reliability showcasing a CTC loss value of up to 12.0014



Predictions on 30 epoch iterations and training set of 80,000 words

On the other hand, it is observed that with consistent training and a robust data set, it is observed that the model has adapted to learn intricate patterns and variations present in the diverse samples of the robust dataset thus giving it a higher accuracy in its predictions particularly with longer words of various handwriting styles, shapes, and sizes. This comprehensive learning process enables the model to generalize well to unseen data, thereby enhancing its overall performance. It showcases a loss value as low as 0.9843

Result:

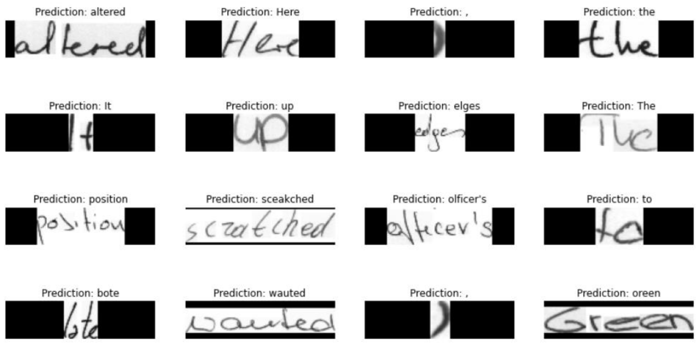
The Flask server successfully handles incoming requests and returns predicted classes. Performance metrics such as response time
and prediction accuracy can be monitored
and analyzed.

Proposed Outcome:-

Model Dataset-

sure
ne
auring
or
booty
gastronomy
boy
The
and
in
his
flying
presented
No
as
few

Prediction Outcome –



CONCLUSION & FUTURE RECOMMENDATIONS

Handwritten character recognition has remained a complex problem, and the solution to the same will not only lead to efficient transcribing of data but also in a streamlined manner. This model is built to analyze text written and convert it to digital text and voice formats. When compared to the prior system, the new system achieves relatively good recognition rates for both the evaluation of text line elements and the collection of horizontal, vertical, and skewed lines.
The ability to automatically convert handwritten text into digital format using ML and CNN can save significant time and resources. This detection and implementation using Tensor flow provides suggestions for improved accuracy. It can help preserve significant handwritten texts/scriptures so they can withstand the test of time. Perseverance of historical data allows us to preserve cultural heritage, trace lineages and genealogy and support academic research.
As we improve this model over the years, we can expect such machine learning models to be able to adapt to mediaeval languages and transcribe ancient text that is unreadable to the modern man. Implementation of such models could help academic fields such as Paleography and Philology reach new heights.
Digital translation from one language to another has also led to FASTag deployment in India.
The Flask application provides a simple and efficient way to serve predictions using a pre-trained Keras model. It helps to demonstrate the integration of machine learning models with web applications for real-time inference.
Overall, this model has a wide range of applications across different fields, including education, historical studies, healthcare, finance, and legal industries, and can bring significant benefits to these industries. The future research in this field can further improve the accuracy and efficiency of handwritten to digital text conversion using AI and Machine Learning.

Future Directions:

1. Implementing error handling and input validation to handle edge cases and improve robustness.
2. Scaling the application to handle larger loads by deploying it on a cloud platform or using containerization.

3. Adding support for additional features such as batch prediction, model versioning, and model retraining.
4. Enhancing the user interface and adding visualisation capabilities for better interaction and feedback.

REFERENCES

1. https://pylessons.com/handwriting-recognition
2. www.jetir.org
3. https://www.ijert.org/machine-learning-approach-for-translating-handwritten-document-to-digital-form
4. https://github.com/pythonlessons/mltu/tree/main/Tutorials/03_handwriting_recognition
5. https://digitallibrary.aau.ac.ae/bitstream/handle/123456789/751/Hand-Written%20Text%20Recognition%20Methods%20Review%20Study.pdf?sequence=1#:~:text=Nowadays%20there%20are%20many%20number,ML)%20machine%2D%20learning%20algorithms
6. https://towardsdatascience.com/build-a-handwritten-text-recognition-system-using-tensorflow-2326a3487cd5

PROJECT BY:-

A058 – Aditya Kurup
A070 – Kaavya Nair
A078 – Haripriya Saraf