ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

Handwriting Recognition Application

[1] Anil V Turukmane, [2] Abdul Rizwan, [3] A Harshavardhan, [4] Nuthakki Deepak, [5] M Karthikeya

[1] [2] Professor, Department of Computer Science, VIT-AP University, Andhra Pradesh, India
[2] [3] [4] [5] Student, Department of Computer Science, VIT-AP University, Andhra Pradesh, India
Corresponding Author Email: [1] anil.turukumane@vitap.ac.in, [2] rizwanahamad.21bce9467@vitapstudent.ac.in, [3] harshavardhan.21bce9478@vitapstudent.ac.in, [4] deepak.21bce9488@vitapstudent.ac.in, [5] karthikeya.21bce9750@vitapstudent.ac.in

***Abstract*— A software program called the Handwriting Recognition Application is used to scan handwritten text from different materials. It digitizes documents, forms, and historical notes for simple administration and searching. A wide range of handwriting styles are supported by the program's accurate recognition of printed and cursive scripts. It addresses issues like illumination and backdrop fluctuations to improve the quality of handwritten material using sophisticated image processing and machine learning. This easy-to-use program offers batch processing and real-time recognition for both personal and commercial usage. It is appropriate for both standalone and cloud-based deployments since it can connect with a variety of input sources, including scanned documents, photos, and digital handwriting. The program places a high priority on security and privacy, protecting sensitive data via encryption and adhering to privacy laws.**

***Index Terms*— Optical character recognition (OCR), LSTM, convolutional neural networks (CNNs), Cloud Vision API.**

I. INTRODUCTION

Handwriting recognition is an interesting and demanding area that combines cutting-edge technology with the subtleties of human handwriting in the modern digital world. The Handwriting Recognition Application bridges the gap between the physical and digital worlds by identifying, interpreting, and converting handwritten text into digital format using cutting-edge machine learning and image processing. With the use of this creative tool, handwritten notes, forms, and old papers may be converted into digital data that is manageable and easily searchable. It guarantees excellent transcription accuracy by accommodating a wide range of printed and cursive handwriting styles. The application's intuitive user interface facilitates document digitisation for both individuals and organisations. The design has a strong emphasis on scalability and security, which enables the program to operate in a variety of deployment scenarios and input formats, including personal devices and cloud-based services. Sensitive data is protected by strong encryption and data privacy protections. Handwritten material may now be managed more effectively, conveniently, and securely thanks to the revolutionary Handwriting Recognition Application.

A. Problem Definition

The goal of handwriting recognition is to create a reliable system that can reliably translate handwritten text into digital form. This is a challenging task. Achieving high accuracy in a variety of handwriting styles and languages, as well as making sure the system is flexible enough to accommodate changes in input tools, picture quality, and writing approaches, are the main goals. Scalability is essential for smoothly adjusting to changing workloads and user demands. It is essential to provide an intuitive interface that makes it simple to submit handwriting samples and retrieve transcription results quickly.

Relentless acquisition of knowledge from user input is essential to gradually increase the accuracy of the system. The project's scope includes training models using a variety of datasets, building APIs for user interaction, supporting several languages, and differentiating between printed and cursive writing styles.

It also emphasises methods for continuous accuracy improvements by integrating user feedback and model upgrades. It is crucial to address issues including optimising speed for near-real-time processing, protecting data privacy and security, making effective use of available resources, and striking a balance between model size and realistic deployment. With this all-encompassing strategy, we want to develop a user-focused, flexible, and efficient handwriting recognition program that can accommodate changing needs and the expectations of a wide range of users.

B. Objectives

To ensure high transcription accuracy, the Handwriting Recognition Application focusses on exact text recognition across various handwriting styles, languages, and variations. Its strong architecture allows it to adjust to various handwriting styles, tools, and image characteristics. The program can distinguish between written and cursive characters and supports a number of languages. Its scalable architecture maintains steady performance even under changing workloads. Adaptability is attained by ongoing learning from fresh data and user input. An easy-to-use interface makes it easier to submit and view results. While data security and privacy protections safeguard user information, resource optimisation guarantees effective

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

functioning. The system may be seamlessly integrated and deployed on a variety of platforms.

C. Challenges

A machine learning system with the right training could be able to read Fig. 1. Challenges in hand writing recognition, and type the text very swiftly. To perform well, however, we need to solve a number of issues with handwritten sentence recognition. Several of these difficulties consist of:

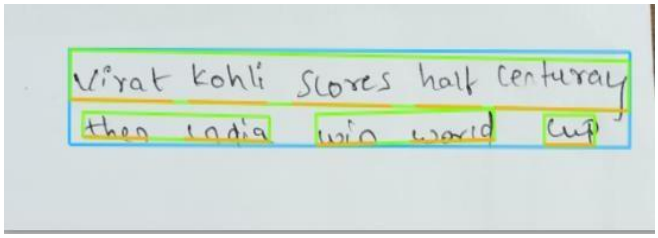

Fig. 1. Challenges in hand writing recognition

The intricacies and fluctuations of penmanship, which vary among individuals, pose obstacles to precise text identification. The handwriting style of a person might vary over time, which increases the amount of inconsistency that models have to deal with. Errors and irregularities like smudges, folds, or inkblots exacerbate the difficulty of recognition. Furthermore, model accuracy may be impacted by the arrangement, context, and placement of handwritten text inside longer text blocks. Differentiating and identifying distinct characters can be particularly challenging when writing in cursive. Unlike printed text, which always has the same orientation, handwritten writing can potentially appear at different angles. Finally, the expense of obtaining a high-quality dataset vs using generated data while training handwriting recognition software can differ.

II. LITERATURE SURVEY

Significant advancements in handwriting recognition technology have resulted from the fusion of artificial intelligence and neural network approaches. With an emphasis on algorithms, systems, and applications, this literature survey examines current developments in handwriting recognition technology.

A. Optical Character Recognition (OCR) Techniques

[1] gives a summary of Tesseract, an open-source OCR system that has seen substantial development since its launch. It uses methods including pattern-matching recognition, character segmentation, and layout analysis. It is a strong option for OCR operations because of its versatility across many fonts and languages. [3] talks on the OCRopus framework, highlighting how its modular nature enables researchers to build different OCR components in a flexible manner. OCRopus is appropriate for a variety of handwriting recognition jobs since it blends conventional methods with machine learning algorithms.

B. Deep Learning in Handwriting Recognition

An end-to-end trainable neural network for sequence recognition with a focus on scene text recognition is proposed by [5]. Their method improves recognition rates for variable-length sequences by using CNNs to extract features from photos. In order to improve handwritten word recognition, [7] presents a gated recurrent convolutional neural network for OCR that incorporates GRNNs to identify long-term relationships in sequential data.

C. Transformer Models

[8] investigates the application of Vision Transformers for text detection in scenes. These models can effectively analyse text in pictures by utilising attention processes, producing quick and precise recognition results.

D. End-to-End Recognition Frameworks

[12] offers Deep TextSpotter, a unified framework for text recognition and localisation. Their end-to-end methodology streamlines the conventional pipeline and offers a more effective way to identify text in pictures. ABCNet, which uses an adaptable Bezier-curve network for real-time scene text detection, is introduced by [14]. Accurately recognising curved text, a frequent problem in handwriting recognition, is the main goal of this model.

E. Challenges in Handwriting Recognition

The difficulties caused by different handwriting styles are addressed by [4] in their competition on scanned receipt OCR. Their research emphasises the necessity for reliable models that are applicable to various layouts and writing styles. [10] combine semantic comprehension with visual cues to develop a Visual-Semantic Transformer for scene text recognition. By improving the model's contextualisation skills, this integration raises the accuracy of recognition.

F. Applications of Handwriting Recognition

[6] uses deep convolutional and recurrent neural networks to recognise text in document photos captured by smartphones. Their research is on the usefulness of handwriting recognition in mobile environments. [9] describes end-to-end text recognition with hybrid HMM Maxout models, focussing on its use in extracting structured information from unstructured handwritten materials.

G. Conclusion

Deep learning and neural network design advancements have made substantial strides in the field of handwriting detection. Modern methods are rapidly being added to traditional OCR systems to increase the precision and effectiveness of handwritten text recognition. Ongoing research keeps improving the capabilities and uses of handwriting recognition technology, despite obstacles relating to the integration of visual cues and the diversity of handwriting styles.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

III. CLOUD VISION API ESSENTIALS

A. Cloud Vision API

Using computer vision and machine learning, the Cloud Vision API from Google Cloud Platform analyses and interprets picture content. It gives programmers access to a collection of pre-trained models for extracting useful data from photos, opening up a variety of industrial applications. Businesses may now take advantage of new opportunities in image processing and analysis thanks to this technology. The API's image categorisation capability allows it to identify objects, locations, brands, and even various animal species in photos by classifying them into distinct labels. This is further enhanced by the object detection capability, which can identify and highlight certain objects inside an image while providing details about their locations and attributes. Optical Character Recognition (OCR) is another advanced text recognition feature of the API. It correctly extracts handwritten and printed text from photos, enabling editing and searching of the data for a variety of applications, including translation and document analysis. Among the many effective tools available in the Cloud Vision API's toolbox is facial detection and analysis. It can identify features such as age, gender, emotions, and facial landmarks in photos by recognising faces, which is very helpful for applications pertaining to social media, user profiling, and security. Furthermore, photographs are screened for possibly unsuitable content via the Safe Search and content moderation tool, which looks for sexual material, violent content, or sensitive imagery. Enforcing content restrictions on digital platforms and ensuring user safety depend on this skill. With simple REST API calls that work with a variety of platforms and programming languages, developers can include the Cloud Vision API into their apps with ease. Additionally, the API integrates easily with other Google Cloud services, offering a strong way to improve image processing and analysis in a larger cloud-based ecosystem. With the help of the Cloud Vision API, developers can now leverage AI and machine learning to analyse visual content more effectively, giving them the tools they need to create innovative applications.

B. Types of API

Application Programming Interfaces, or APIs, serve as bridges that enable interaction and communication between various software systems. They specify sets of rules and guidelines that let programs to access information, features, or services from one another. APIs facilitate interaction between apps, databases, and devices by providing specified methods for retrieving or altering data, which speeds up development. They come in many varieties, such as RESTful, SOAP, and GraphQL, as shown in Fig. 2. Four major types of API protocols, and they can be used for tasks like information retrieval and real-time communication. APIs drive innovation by enabling developers to expand on current features and use external resources, enabling the linked digital world we live in every day.

Various online offers can safely communicate using REST (Representational State Transfer) API using common HTTP protocols like GET, POST, PUT, and DELETE. Its unified interface, which makes information accessible with lightweight codecs like JSON or XML, is what makes it so simple. REST would not require a strict agreement, in contrast to SOAP, which makes it more adaptable for a variety of applications. On the other hand, GraphQL provides a more customised approach to information retrieval. An example of a REST API is the Cloud Vision API, which uses machine learning to analyse photos and perform tasks like object detection, textual content popularity, and content moderation. This API's smooth integration and HTTP accessibility make it possible for a wide range of programs to analyse and process photographs.

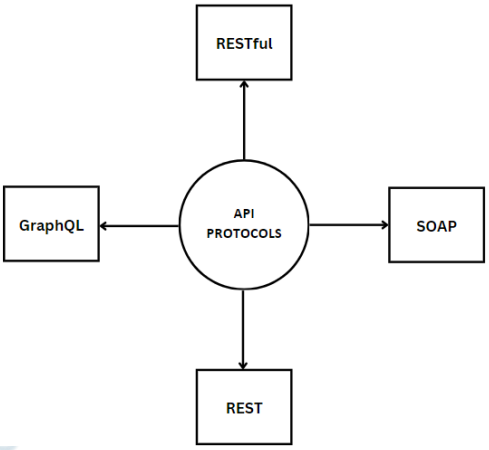

Fig. 2. Four major types of API protocols

C. Authentication Tools

To protect sensitive information and capabilities, a variety of authentication approaches, including OAuth, token-based authentication, and HMAC, are used in API access security. The authentication mechanism we have chosen for our application's connection with the Cloud Vision API is the use of API keys. With each request, these special IDs are sent, giving our program permission to use the robust picture analysis features offered by the Cloud Vision service. The ease of use and effectiveness of API keys meet our unique needs, guaranteeing safe and regulated access to the extensive image processing features provided by the Cloud Vision API.

D. Processing Behind the Cloud Vision API

Now we will go into detail about the step-by-step process of the working shown in Fig. 3., The initial step of the procedure involves uploading the image to a cloud storage provider, which offers a safe location for additional processing. After being saved, the picture stays intact and readable in the cloud. Upon upload, a cloud function might be activated to automate processes or get the picture ready for

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

processing. After that, the stored image is downloaded and transferred to the Cloud Vision API, which scans it for objects, labels, text, and facial features using machine learning models. The processed information is kept for future use or analysis by storing the outcomes or analysed data from the Vision API back in the cloud. For additional optimisation or targeted insights, this data can also be incorporated into bespoke AI models or machine learning algorithms.

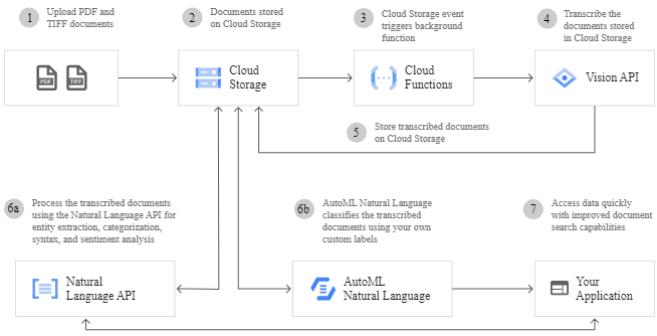

Fig. 3. Cloud Vision API

IV. METHODOLOGY

Let us talk about the various machine learning models we will be using in this cloud:

A. CNN-RESNET

There are several important steps in the process of using a Residual Network (ResNet) in a handwriting recognition program. It starts with gathering a variety of handwritten text samples into a dataset, then preprocessing the images to adjust their size and normalise pixel values. The annotation of the dataset which involves identifying areas with handwritten text comes next. After that, the ResNet architecture is created, configuring it to accept handwritten text pictures as input and producing binary classification output that indicates text or non-text regions by using its residual blocks for efficient feature extraction. The residual blocks of a common ResNet architecture, like ResNet-18, are used for effective feature extraction. Because it strikes a mix between performance and efficiency, ResNet-18 is frequently chosen over more complex ResNet architectures, making it perfect for applications involving handwriting recognition. Using a suitable optimiser and backpropagation for ResNet training, the dataset is divided into training and validation sets during the training phase. Hyperparameter fine-tuning is made possible by constant monitoring of training and validation metrics. To improve the model's predictions, post-processing methods like thresholding and morphological procedures are used. Finally, the model's performance is evaluated on an independent test set using evaluation measures like precision, recall, and F1 score.

B. Long Short-Term Memory (LSTM)

Because they can recognise sequential patterns and contextual dependencies in handwriting samples, Long Short-Term Memory (LSTM) networks are essential in the field of handwriting recognition applications. LSTMs, a subclass of recurrent neural networks (RNNs), are well-suited to handle the sequential nature of handwritten text because they are especially good at retaining and applying information over long sequences. LSTMs are used in both handwriting recognition and other tasks. First of all, they efficiently handle sequential input data, viewing each writing stroke or pixel sequence as a sequence of information stored in time. This capacity helps individuals to identify the temporal relationships that exist between the strokes, which is crucial for comprehending the structure and flow of handwriting. Because they can analyze input sequences both forward and backward and capture contextual information from both past and future strokes, bidirectional LSTMs (BiLSTMs) are very useful for handwriting recognition. Additionally, managing both spatial information and sequential dependencies in handwriting images can be accomplished with Convolutional LSTMs (ConvLSTMs), which blend convolutional layers with LSTM layers. By enhancing feature extraction and context comprehension, both variants increase recognition accuracy. Because they can better handle both spatial and sequential dependencies and collect greater contextual information, BiLSTMs and ConvLSTMs are recommended over regular LSTMs for handwriting recognition.

C. You Only Look Once (YOLO)

In order to use YOLO (You Only Look Once) for a handwriting recognition application, this robust object detection framework must be modified in order to locate and identify handwritten text within photos. YOLO's strength is its real-time object detection with grid-based picture segmentation and simultaneous class probability and bounding box prediction. YOLOv4 was used to train the model on annotated datasets with a range of handwritten text samples, YOLO must be customised for handwriting recognition. During the training phase, the network's parameters are adjusted and hyperparameters are optimised in order to precisely locate and identify text instances against various handwriting styles and backdrops. The anchor boxes and feature pyramid of the YOLO architecture facilitate the effective identification of text sections in images with varying scales and aspect ratios. Non-maximum suppression (NMS) is one post-processing technique that improves detection results by keeping the most likely text regions and eliminating unnecessary bounding boxes. Metrics including precision, recall, and Intersection over Union (IoU) are used to evaluate the performance of the modified YOLO model for handwriting recognition. These metrics offer important insights into the model's accuracy and robustness in handwritten text recognition.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

D. Android Studio

The central location for developers creating handwriting recognition application is Android Studio. It provides an extensive toolkit that is especially made to simplify the difficult process of developing apps, and it is the official IDE endorsed by Google. Fundamentally, Android Studio offers programmers a sophisticated code editor that supports languages including C, Java, and Kotlin. This editor serves as the cornerstone around which handwriting recognition programs are constructed, allowing programmers to turn creative ideas into real-world digital solutions. Beyond code editing, Android Studio's user-friendly layout editor which combines XML with visual design elements enables developers to create user interfaces. This capability makes it easy to create aesthetically pleasing interfaces that combine functionality and style to improve user experience. We leverage Android Studio's Java and XML features in our handwriting recognition application development to provide a solid and user-focused experience. The main programming language used in the application is Java, which provides effective data processing, recognition algorithms, and interactive features. It also forms the core of the application's logic. Its object-oriented design principles give developers the adaptability they need to create scalable apps that work well on a variety of Android handsets. As the markup language for creating the application's user interface, XML is a useful addition to Java. It makes it easier to create responsive and well-organised layouts, guaranteeing that the placement of UI elements is understandable and intuitive. We can construct intuitive and functional handwriting recognition programs by combining XML for interface design with Java for logic. Android Studio's combination of Java and XML is a testament to our dedication to producing handwriting recognition software of the highest calibre. This method makes it possible to create complex user interfaces and strong recognition capabilities, giving users an effective and entertaining way to interact with handwritten text.

E. User Interface

For our handwriting recognition application, we used Android Studio to carefully craft an engaging user interface (UI) that strikes a balance between utility and aesthetics by combining XML and Java as shown in Fig. 4. Home page of mobile application. The foundation for designing user interface elements is XML, which allows components to be efficiently organised across various screen sizes and orientations using layout types like Linear Layout and Relative Layout. Java gives these interfaces life by integrating the logic and features required to guarantee smooth execution. Following Google's Material Design guidelines, our UI design prioritises responsiveness, dependable typography, and eye-catching aesthetics to produce a visually appealing and intuitive user experience. Real-time previewing is made possible by Android Studio's visual toolkit, which greatly expedites the process of design iteration and validation. A fundamental component of our design is inclusivity, which includes accessibility features like high contrast settings and screen readers to accommodate a range of user needs. Our handwriting recognition software prioritises usability and visual coherence while integrating technology to create a compelling user experience. Our devotion to developing an engaging and user-friendly application for our users is shown in our commitment to providing a well-designed, user-centric user interface (UI).

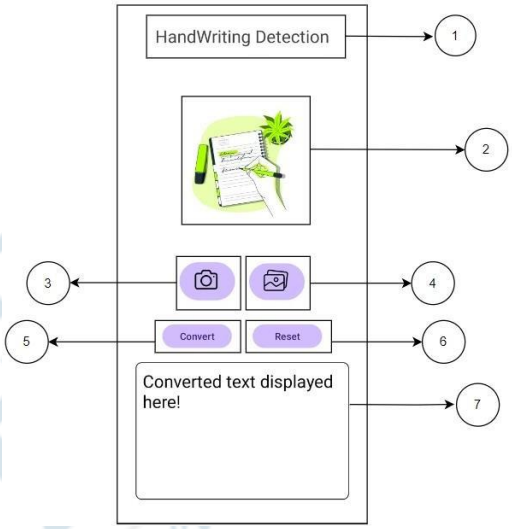

Fig. 4. Home page of mobile application

The UI has a number of crucial components. At the top of the screen, the title section clearly presents the application's name or logo, so creating its identity. The chosen image that is subjected to recognition processing is displayed in the image display section, providing viewers with a visual representation of the content under analysis. The camera button allows users to take new pictures within the app for instant analysis by turning on the device's camera. In the meanwhile, users can choose and use pre-existing photographs for recognition by clicking on the gallery icon, which gives access to stored images. By pressing the convert button, you can start the process of handwriting recognition, which sets off an algorithm that examines the presented image and extracts text from it. A reset button lets you start over with new interactions by offering the option to remove selected items or go back to the original state. When the recognition analysis is finished, a text view shows the user the extracted text in a straightforward and understandable manner. This well-considered layout guarantees that every app interaction is simple to understand and efficient for the best possible user experience.

V. RESULTS AND DISCUSSION

A. Word Error Rate (WER)

A statistic called Word Error Rate (WER) is used to assess how accurate systems such as machine translation, speech recognition, and handwriting recognition are. Three

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

categories of errors—substitutions, in which a word in the prediction deviates from the reference; insertions, in which additional words are included in the prediction; and deletions, in which words from the reference text are absent from the prediction—are used to quantify the difference between the predicted and reference texts. These mistakes are added together and divided by the total word count of the reference text to determine WER. Better system accuracy is indicated by a lower WER, underscoring its significance in enhancing translation and text recognition models.

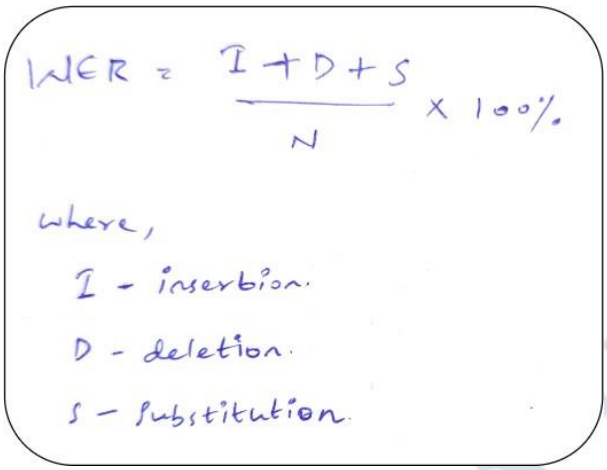

Fig. 5. Formula for word error rate (WER)

In order to compare the real text (ground truth) taken from a collection of 500 photos with the projected text retrieved from the Cloud Vision API, Python programming was used. In order to analyse and digitise the handwritten text, the Python script most likely iterated over the dataset, sending each image to the Cloud Vision API. The actual text labelled inside the photos was then compared to the anticipated text that was retrieved from the API. The error rate was computed as a numerical number by adding up the variances or discrepancies between the real and anticipated text for each of the 500 photos. The percentage of discrepancies or mistakes between the predicted and ground truth text across the dataset is around 0.242121, or 24.21% in this instance, as shown in Fig. 6. Calculated WER. The accuracy and performance of the Cloud Vision API for handwriting detection were examined in this Python-based study, which helped pinpoint areas in need of development and improve the system's text recognition capabilities.

Fig. 6. Calculated WER

B. Basic Analysis

The durability of our tool in reading and understanding handwritten information was demonstrated when we tested our system on a dataset of 500 photos and obtained an amazing accuracy rate of 87.2% in predictions. After a thorough analysis of each image, the findings revealed that 87.2% of the predictions accurately represented the content. To illustrate this accuracy and assist pinpoint areas that might require work, we created a pie chart titled as Fig. 7. Visual representation of detection analysis, that breaks down accurate forecasts into various categories. This analysis demonstrates our dedication to managing handwritten content with accuracy and dependability.

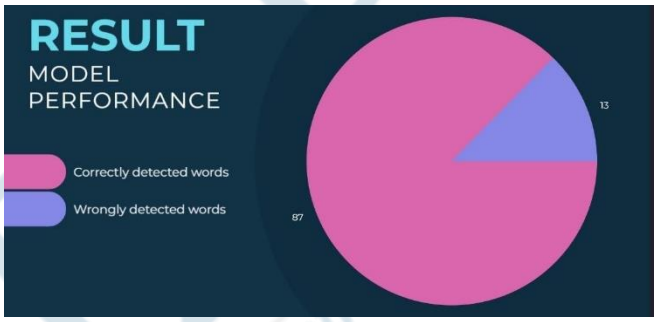

Fig. 7. Visual representation of detection analysis

C. Android Application Results

Now we will be showing the first result of our application titled as Fig. 8. Result 1.

Fig. 8. Result 1

Expected Output: handwriting
Predicted Output: handwriting

The second outcome will now be presented, which is titled as Fig. 9. Result 2.

ISSN (Online) 2394-2320

International Journal of Engineering Research in Computer Science and Engineering (IJERCSE)

Vol 12, Issue 1, January 2025

Fig. 9. Result 2

Expected Output: Hello, how are you?
Predicted Output: Hello, how are you?

VI. CONCLUSION

With its wide range of applications across several domain names, the Handwriting Recognition program showcases the potential of using generation to transform handwritten text into digital layout. This well-received development highlights the significance of precise popularity structures, improving both accessibility and value.

VII. FUTURE SCOPE

Considering its present progress, the application for handwriting recognition holds great potential for further enhancements in the future. Improving identification accuracy is still a top priority, and to better support a variety of handwriting styles and languages, deep learning models are being integrated and algorithms are continuously being refined. Incorporating intuitive features like as intelligent editing tools, real-time feedback during text capture, and language specific optimisations to increase user satisfaction may greatly improve the user experience.Increasing the application's multi-platform accessibility by incorporating web and iOS compatibility will also be essential to increasing its usability and accessibility. The software's adaptability will increase with the addition of offline features, which will allow users to execute handwriting recognition without continual internet access. The application's usefulness will be further enhanced by integration with productivity tools such as note-taking applications, task organisers, or document management systems, which will enable the smooth integration of recognised text into routine operations. Essentially, the application for handwriting recognition establishes a strong basis for the advancement of this technology. Its future development might bring forth more accuracy, better functionality, and easier use, making the transition from analogue to digital media more fluid than in the past.

REFERENCES

[1] R. Smith, “An Overview of the Tesseract OCR Engine,” in Ninth International Conference on Document Analysis and Recognition (ICDAR2007), vol. 2, Sep. 2007, pp. 629–633, iSSN: 2379- 2140. [2] “EasyOCR,” Sep. 2022, original-date: 2020-03-14T11:46:39Z. [Online].

[3] T. M. Breuel, “The OCRopus Open-Source OCR System.”

[4] Z. Huang, K. Chen, J. He, X. Bai, D. Karatzas, S. Lu, and C. V. Jawahar, “ICDAR2019 Competition on Scanned Receipt OCR and Information Extraction,” in 2019 International Conference on Document Analysis and Recognition (ICDAR), Sep. 2019, pp. 1516–1520, arXiv:2103.10213 [cs]. [Online].

[5] B. Shi, X. Bai, and C. Yao, “An End-to-End Trainable Neural Network for Image-based Sequence Recognition and Its Application to Scene Text Recognition,” Jul. 2015, arXiv:1507.05717 [cs]. [Online].

[6] H. El Bahi and A. Zatni, “Text recognition in document images obtained by a smartphone based on deep convolutional and recurrent neural network,” Multimedia Tools and Applications, vol. 78, no. 18, pp. 26 453–26 481, Sep. 2019. [Online].

[7] J. Wang and X. Hu, “Gated Recurrent Convolution Neural Network for OCR,” in Advances in Neural Information Processing Systems, vol. 30. Curran Associates, Inc., 2017. [Online].

[8] R. Atienza, “Vision Transformer for Fast and Efficient Scene Text Recognition,” May 2021, arXiv:2105.08582 [cs]. [Online].

[9] O. Alsharif and J. Pineau, “End-to-End Text Recognition with Hybrid HMM Maxout Models,” Oct. 2013, arXiv:1310.1811 [cs]. [Online]

[10] X. Tang, Y. Lai, Y. Liu, Y. Fu, and R. Fang, “Visual-Semantic Transformer for Scene Text Recognition,” Dec. 2021, arXiv:2112.00948 [cs]. [Online].

[11] K. Wang, B. Babenko, and S. Belongie, “End-to-end scene text recognition,” in 2011 International Conference on Computer Vision, Nov. 2011, pp. 1457–1464, iSSN: 2380-7504.

[12] M. Busta, L. Neumann, and J. Matas, “Deep TextSpotter: An End-to-End Trainable Scene Text Localization and Recognition Framework,” in 2017 IEEE International Conference on Computer Vision (ICCV). Venice: IEEE, Oct. 2017, pp. 2223–2231. [Online]. 45

[13] T. He, Z. Tian, W. Huang, C. Shen, Y. Qiao, and C. Sun, “An Endto-End TextSpotter with Explicit Alignment and Attention,” in 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. Salt Lake City, UT, USA: IEEE, Jun. 2018, pp. 5020–5029. [Online].

[14] Y. Liu, H. Chen, C. Shen, T. He, L. Jin, and L. Wang, “ABCNet: Real-Time Scene Text Spotting With Adaptive Bezier-Curve Network,” in 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, Jun. 2020, pp. 9806–9815. [Online].

[15] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: A Database and Web-Based Tool for Image Annotation," International Journal of Computer Vision, vol. 77, no. 1- 3, pp.157–173, May 2008. [Online].

[16] P. Iakubovskii, "qubvel/segmentation models," Sep. 2022, originaldate: 2018-06- 05T13:27:56Z. [Online].

[17] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," May 2015, arXiv:1505.04597 [cs]. [Online].

[18] A. Chaurasia and E. Culurciello, "LinkNet: Exploiting Encoder Representations for Efficient Semantic Segmentation," in 2017 IEEE Visual Communications and Image Processing (VCIP), Dec. 2017, pp. 1–4, arXiv:1707.03718 [cs]. [Online].

[19] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," Apr. 2015, arXiv:1409.1556 [cs]. [Online].

[20] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," Dec. 2015, arXiv:1512.03385 [cs]. [Online]