INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

ScoreSync: Automated Mark Entry App using Image Capture with Computer Vision and Optical Character Recognition

Muhammad Norazlan Abdul Samad
Kulliyyah of Information & Communications Technology
International Islamic University Malaysia
Selangor, Malaysia
Email: alanillaiium@gmail.com

Hafizah Mansor*
Kulliyyah of Information & Communications Technology
International Islamic University Malaysia
Selangor, Malaysia
Email: hafizahmansor@iium.edu.my

Arman Nuri Anuar
Kulliyyah of Information & Communications Technology
International Islamic University Malaysia
Selangor, Malaysia
Email: armananuar7@gmail.com

Submitted: 15/9/2025. Revised edition: 28/11/2025. Accepted: 5/1/2026. Published online: 28/7/2026
DOI: https://doi.org/10.11113/ijic.v16n1-2.703

Abstract*—Traditional score recording, whereby educators manually key in scores in their respective education hubs, is time-consuming and susceptible to error when the volume of papers is high. To address this, ScoreSync, an automated mark entry app, was developed to streamline the grading and recording process by leveraging computer vision and machine learning. The primary objective is to develop an application capable of capturing images of exam papers using a lens or phone camera, ensuring a clear and accurate representation of score sheets. This would not only reduce an educator's administrative burdens but also improve data accuracy and efficiency. Created to be applied by Android devices, ScoreSync captures images of graded papers and extracts scores through Optical Character Recognition (OCR) guided by YOLOv8, a Convolutional Neural Network (CNN) model optimised for bounding box detection. Developed through a Rapid Development Model, the app has been through multiple refines when it comes to the Artificial Intelligence (AI) model, transitioning from TensorFlow to PyTorch for greater flexibility and accuracy. The initial build of the ScoreSync is solely for the purpose of extracting graded papers of any kind, with results ranging from 80% to 90% in terms of detecting the scores in the exam sheets. Future improvements include expanding compatibility across platforms, refining the model adaptability towards different exam paper styles, and a setup that would aid the software when a live-capture solution is implemented, which would minimise more time spent rather than having the educators snap each exam sheet. ScoreSync aims to support educators by transforming the traditional manual process of score recording into a more efficient, technology-aided experience for a more enhanced workflow.

***Keywords*—Automated score transfer, Computer vision, Optical Character Recognition (OCR), YOLOv8, Mobile Application, Machine Learning**

I. INTRODUCTION

Today, rapid technological advancements have been a call for sectors to evolve, including the education sector. The demand for an innovation to streamline academic processes has become increasingly pressing, as educators nowadays hold a lot of responsibility when it comes to their work, including handling school management and student welfare. Therefore, educators will appreciate any help that would ease their burdens. At the realm of assessment and grading, in particular, stands at the forefront the traditional method to transfer scores from the marked papers to their computer by typing one by one, which proves to be both time-consuming and prone to error. Amidst this, ScoreSync aims to give a helping hand to educators by harnessing the power of technology in each of our hands, the smartphone. Coupled with a deep understanding of

challenges faced by educators, our project endeavours to use deep learning in order to capture the scores from the papers that have been marked by the educators and compile them, followed by carrying them directly to their respective educational websites.

II. LITERATURE REVIEW

In the field of educational technology, the automation and efficiency that Optical Character Recognition (OCR)-based systems offer significant benefits, as their ability to convert images of text into editable digital formats has revolutionised data entry across various industries [1]. Teachers are able to save time with these solutions because they eliminate the need for manual data entry, which subsequently reduces the likelihood of errors caused by human intervention [2]. In addition, the integration of deep learning and image processing techniques, such as those utilised in Artificial Intelligence (AI)-based assessment systems, ensures a high level of accuracy in both the recognition and administration of marks. This accuracy helps to guarantee that grading outcomes are consistent and reliable, which is an extremely important aspect in educational settings [3]. As an additional significant advantage, the scalability and adaptability of deep learning models present themselves. These models are intended to be able to adapt to various grading criteria and are also capable of working with a variety of handwriting styles [4]. On account of this, they are suitable for use in a broad variety of educational environments. Furthermore, when EasyOCR is paired with real-time picture capture, it enables quick feedback and grading to be performed on the images. This benefit is especially useful in circumstances that need a rapid evaluation [5]. Due to the fact that it is capable of processing information in real time, the grading system increases its overall efficiency and responsiveness.

In a related study, a records management system was developed that combines the PyTesseract library with both Django and MySQL to automate the digitisation, classification and retrieval of scanned documents [6]. This system demonstrates how open-source OCR technology can effectively extract text and metadata from images. This highlights the practicality and adaptability of PyTesseract for automating text-based documents in institutional environments. Similarly, an analysis showcases EasyOCR in its flexibility across image conditions and support for multiple languages, but shows reduced confidence under severe image degradation such as blur, low contrast and small character sizes [7]. It also displays an uppercase bias when recognising similar letter shapes. PyTesseract remains reliable for clean document scans, and EasyOCR proves to be more adaptive for real-time or varying visual environments, making it suitable for dynamic applications like mobile data capture.

More recently, the effectiveness of Large Language Models (LLMs) such as Gemini in the field of historical document OCR was analysed, with new measures of temporal accuracy being proposed [8]. The findings showed that the Gemini models performed better than standard OCR tools like Tesseract or PyLaia in 18th-century Russian texts, though sometimes ‘over-historicising’ the output by adding historical letters. However, the Gemini models showcased better robustness, flexibility in handling complex page layouts, and enhanced handling of historical letters than their predecessors, thus suggesting the use of multimodal LLMs in improving the effectiveness of historical document OCR.

Although there are numerous advantages to using these automated systems, there are also some drawbacks involved. When employing OCR technology, one of the most significant challenges is the difficulty in accurately detecting a variety of handwriting styles and unclear text, which can lead to errors [9]. In addition, the implementation of systems that are based on deep learning is a difficult process that requires a significant number of computational resources [3] as well as a significant amount of expertise in computer vision and machine learning. In certain educational settings, the complexity of the situation makes it difficult to adopt and effectively utilise the technology. There is still another disadvantage associated with the fact that accurate mark identification and recognition are highly dependent on the quality of the picture. The performance of these systems is mostly determined by the quality of the photographs that are used as input [5]. However, this can vary based on the method of collection and the configuration settings. The system’s accuracy may be significantly diminished if the image quality is low. The integration of these automated systems with existing software is not only challenging but also resource-intensive [4]. Additionally, maintaining compatibility requires ongoing development and maintenance. It is possible that the integration challenge will limit the system’s capacity to scale and remain viable over the long run.

A. Discussion

Table I offers a complete picture of many automated mark input systems by emphasising their environment, algorithms employed, compatibility, complexity, accuracy, and efficiency. Though they struggle with different handwriting styles, systems using OCR algorithms like those by Koushik *et al*. [2] offer balanced automation and error reduction for scanned papers. Though they are sophisticated and computationally demanding, AI-based examination systems such as those by Logeshvar *et al*. [3] use image processing and Convolutional Neural Networks (CNN) for great accuracy in mark recognition. Appropriate for offline recognition, Lim Lam Ghai [10] uses a Local Binary Pattern (LBP)-based method that effectively handles handwritten digits with modest complexity. Although it may slightly compromise accuracy, their EasyOCR system offers real-time processing for webcam-captured images, therefore balancing ease of use and modest complexity [5]. Faced with difficulties with particular types of images, Mansoor and Olson [9] use contour detection and CNNs for high text recognition accuracy. Although it needs large computational resources, the Learning Pooling (LEAP) operation was used to improve CNN performance with lowered settings [4]. Pooling techniques were also examined with an eye on their significance in CNN performance and regularising capacity [11]. These systems show several techniques that balance complexity, accuracy, and efficiency, thereby guiding the choice of the most appropriate system for particular educational requirements. Jayoma *et al*. [6] use the PyTesseract OCR library to automate document archiving and indexing.

PyTesseract, integrated with Django and MySQL, demonstrated effective text extraction for scanned records, but its accuracy decreased with poor-quality inputs. Similarly, EasyOCR was analysed under image degradation, resulting in strong recognition for Latin characters but suffering from reduced performance under blur and ambiguity for character sizes [7]. Finally, Google Gemini and other LLMs were evaluated for historical document OCR [8]. Google Gemini clearly outperformed traditional OCR systems in both accuracy and stability, though prone to minor "over-historicisation" of archaic characters. More recently, the robustness of YOLOv8 for real-time object detection tasks was demonstrated, achieving high mean average precision (mAP) and fast inference speed, which shows its potential as the backbone for detection in automated grading systems that depend on rapid localisation and recognition of exam components and structures [12].

TABLE I. FEATURE COMPARISON FROM EXISTING WORK

Author	Environment/Situation	Algorithm	Performance
Koushik *et al*. (2019)	Scanned documents (exam sheets)	OCR, scanners	Balanced
Logeshvar *et al*. (2018)	Photographed answer sheets	OpenCV, Convolutional Neural Network (CNN)	High Accuracy, Slower
Lim Lam Ghai (2017)	Photographed answer sheets	Local Binary Pattern (LBP)	Efficient, trade-off in accuracy
Subasri & Meenakumari (2020)	Webcam-captured images	EasyOCR	Real-time, may sacrifice accuracy
Mansoor & Olson (2019)	Images with text	Contour detection, Maximally Stable Extremal Regions (MSER), CNN	High accuracy, challenging with some images
Sun *et al*. (2017)	Benchmark datasets	LEAP, simplified convolutional	Improved accuracy, fewer parameters
Zafar *et al*. (2022)	Classification datasets	Max pooling, S3Pool	Effective regularisation, optimisation
Jayoma *et al*. (2021)	Scanned Administrative Records	PyTesseract, Django, MySQL	Efficient text extraction & indexing, improved accessibility
Salehudin *et al*. (2023)	Latin characters under image degradation	EasyOCR, OpenCV, Pillow	Robust with clear text, reduced accuracy under blur or low contrast
Reis et. Al (2023)	Real-time object detection	YOLOv8 (Ultralytics)	High mAP and fast inference, suitable for grading systems
Levchenko (2025)	Historical 18th-century russian texts	Large Language Model (Gemini, multimodal OCR)	Outperformed traditional OCRs in accuracy and stability

Table II is based on the findings mentioned in Table I, focusing on particular models or algorithms tested for experimentation in this research. The base models describe how conventional detection or OCR techniques develop into multimodal intelligence. YOLOv8 by Reis *et al*. [8] is effective in region of interest detection, while OpenCV enhances pre-processing techniques, enabling accurate localisation **Error! Reference source not found.**. EasyOCR ensures efficient recognition under normal visuals [9], though Google Gemini was chosen [3], which enables this entire functionality by employing sophisticated multimodal logic during scene recognition. YOLOv8 and Gemini were chosen as the final models for region detection and OCR recognition. Individually, all models indicate progress from conventional OCR technology to more adaptive AI-oriented recognition models adequate for current educational use.

TABLE II. INVESTIGATION OF MODELS AND ALGORITHMS UTILISED IN THE PROPOSED SYSTEM

Model / Author	Algorithm & Function	Performance	Remarks
YOLOv8 by Reis *et al*., 2023)	Object detection for identifying exam regions	High precision, real-time inference	Requires large annotated datasets for fine-tuning
OpenCV + CNN by Logeshvar *et al*. (2023)	Image preprocessing and mark recognition	Strong baseline in structural detection	Computationally demanding on high-resolution inputs
EasyOCR by Salehudin *et al*. (2023)	Text recognition from localised regions	Fast and lightweight implementation	Accuracy drops with blur or low-contrast images
Google Gemini by Levchenko *et al*. (2025)	Multimodal OCR with contextual understanding	Exceptional accuracy on varied document layouts	Needs stable internet and incurs API cost

III. METHODOLOGY

Through the digitization and evaluation of scored academic papers, we want to utilize two AI models in one system, also known as a hybrid model. The first model concentrates on improving feature recognition, which is essential for precise mark detection and table of marks detection on the front of the captured exam paper. The second model concentrates on identifying the marks themselves. It must accurately interpret the educators' annotations on the table or on the paper. Therefore, the first model is a recognition model, whilst the second model is an OCR. This hybrid model technique guarantees high reliability in mark recognition, which is essential not only because the papers that are used across educational institutes are various but also to uphold academic integrity and transparency. Our goal is to incorporate this hybrid model into a mobile application, which would then process scores in real-time, giving the educators a digital format to be used either for reporting or to upload to institutional databases of their respective institutes.

A. Requirements Specification

1) Functional Requirements:

Functional requirements include image capture and processing, mark detection and analysis and scores compilation and upload.

- **Image Capture and Processing:** The application must allow users to capture images of marked papers using any Android device camera. The images must be preprocessed for clarity and noise reduction to ensure high accuracy in mark detection.
- **Mark Detection and Analysis:** Utilising CNN, the application must accurately detect and analyse handwritten marks. This includes recognizing different marking schemes and quantifying them appropriately.
- **Scores Compilation and Upload:** Post mark detection, the scores should be automatically compiled and formatted according to institutional requirements and securely uploaded to the designated educational platform.

2) Non-Functional Requirements:

The non-functional requirements include usability, performance, and scalability. Usability is supported by an intuitive and simple-to-use interface with minimal technical expertise, primarily focusing on educators who may not be familiar with advanced technology. Performance is ensured where the application should process images and upload scores with minimal delay. There should be no more than a few seconds of latency. Scalability is the ability to handle simultaneous uploads without significant loss of performance.

B. Artificial Intelligence Model

In this section, we outline the methodologies employed to develop our AI model for precise mark detection. The process involved several stages, including data collection and preparation, model selection and design, as well as training and validation of the final chosen model. Each stage was crucial in enhancing the model's accuracy and efficiency, ensuring that it meets the project's objectives effectively.

- **Data Collection**: Collected data through various means, such as community outreach, whereby we reached out to friends and family members to gather a diverse set of 100 exam paper sets. This approach yields 19 distinct table and paper formats that are properly marked by educators via a red pen. This approach helped in acquiring a wide range of handwriting styles and marking schemes.
- **Data Preprocessing:** Each image was carefully cropped to isolate individual questions, as most exam papers have multiple questions per page, including the marking table on the front page. All images are labelled using the LabelImg tool and were annotated into XML and YOLO format, which includes the coordinates of the bounding boxes and the corresponding class labels. The data is then split into two categories:70% for training, 20% for validation, and 10% for testing.
- **Model Design:** The hybrid model integrates YOLOv8 and OCR to achieve automated detection and recognition. YOLOv8 is responsible for detecting and localising objects within images based on the data that has been fed to it, specifically identifying tables and related elements. OCR is responsible for recognising and distinguishing numerical values from bounding boxes labelled as marks in the table. Additionally, OCR processes both numeric and characters from bounding boxes labelled as names or IDs, using a whitelist of characters to ensure accurate recognition.

C. Application Development

1) Application Design:

The phase is the detailing stage in the software development lifecycle, where the conceptual framework of the application is visualized into a detailed blueprint. This phase involves designing the structure of the application, user interactions and the data flow process. It ensures that all the functional requirements are clearly reviewed and visualized before their implementation begins. Examples of a key diagram include use case diagrams that provide a high-level overview of the interactions between the users and the system. On the other hand, system architecture diagrams highlight the technical components to offer a clear depiction of how the application's various parts function together. These diagrams together serve as vital tools for communicating design choices and system behaviour.

The Use Case Diagram, as shown in Fig. 1, highlights the main tasks that the user can complete and how they connect to the ScoreSync system, illuminating the interactions between the user and the system. The User and the ScoreSync system are the main characters in this diagram. The user starts a number of operations, such as activating across cameras, taking pictures, confirming uploads, reviewing the scores they have processed, and closing the application. Every one of these activities is associated with a certain function in the system. Receiving, processing, and storing data are all handled by the ScoreSync system. Details about the connections between these use cases are also provided. To illustrate that the upload confirmation is an optimal extension of capturing an image, consider how the "Confirm Upload" use case expands upon the "Capture Image" use case. Comparably, the "Store Data" use case is included in the "Process Image" use case, demonstrating that storing data is required in order to process the image. Moreover, the "Confirm Upload" use case is extended by the "Review Processed Scores" use case, suggesting that scoring reviews is an optional step following upload confirmation.

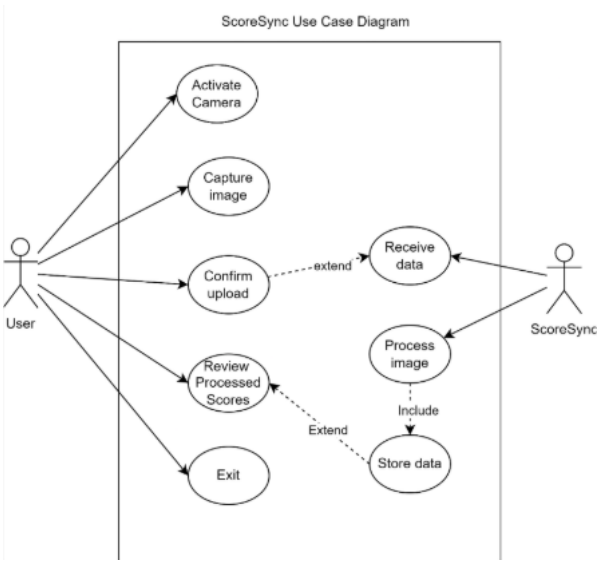

Fig. 1. Use Case Diagram

The ScoreSync application's process flow, from the point of program launch to the conclusion of the processing cycle, is shown step-by-step in the flowchart in Fig. 2. The process begins when the user launches the application and concludes once all documents have been processed. The system handles key decisions, such as whether to retake a picture or confirm that the captured image is clear. The system turns on the camera if another capture is required. After the user takes a picture, the system evaluates its quality. The user is prompted to snap another picture if it is not clear. The system turns on the camera if another capture is required. After the user takes a picture, the system evaluates its quality. The user is prompted to snap another picture if it is not clear. When the image is clear, the system processes and detects marks using the CNN model. It then computes the user's overall score and allows the user to review the computed scores. If there are additional papers to process, the procedure is repeated.

2) Application Implementation:

The application is developed in such a manner that it will accept the latest changes in AI with cases. This, in addition to being compatible with all devices, is built with Flutter, a cross-platform mobile application framework, making it run very smoothly and responsively.

- **Frontend Development with Flutter:** This application is made with Flutter, a very flexible cross-platform application framework. It created a uniform user interface besides creating a responsible user interface besides creating a responsive user interface across devices. The application delivered a smooth and high-performance user experience with versatility.
- **Backend Integrated to Firebase:** Because it provides a real-time database, the team adopted Firebase as the back-end service for the real-time database capabilities. In fact, it provided effective data management, secure storage, and instant synchronization between the mobile application and the AI model.
- **Development Methodology – Rapid Development Model:** The Rapid Development Model is implemented to guide the process of development. This iterative model allows testing and feedback at every stage, along with modifications and ensures that end products match real-world requirements.
- **System Design and Architecture**: Use case diagrams and architectural blueprints were used to chart out user interactions, functionalities available in the system, and data flows. Clarity in design, streamlining the development process, and ensuring effort-target alignment with the overall project objectives are the targeted outcomes.

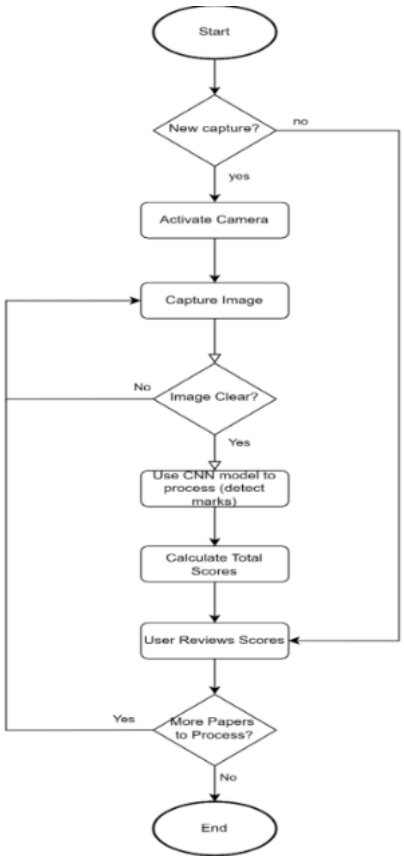

Fig. 2. Flowchart of Application

The application captures images of marked exam papers, preprocesses them, and transmits them to the AI model for analysis. The results are then displayed in an intuitive interface that allows educators to review and finalise scores efficiently. This structured approach to application development ensures reliability, scalability, and alignment with the project's overarching goal of automating the grading and score transfer process.

IV. FINDINGS

A. Decision on choosing an AI model

A hybrid model consists of two different AI models, which complement each other by having each model perform two different tasks that are crucial to achieve the output according to the objective. The input would be an image of an exam paper, whereby the output is a list of scores that were graded by educators in digital format. One of the first requirements of the AI model is that it can identify the table of marks. Thus, in this section, the results and findings of different AI models are discussed.

OpenCV for the first AI model: OpenCV is evaluated for suitability for the task of extracting tables from the exam paper. We used a single table with marks as our test input, as shown in Fig. 3. OpenCV preprocesses the input by applying thresholding to emphasise the structure of the table and suppress any noise or random handwritten markings. We also converted the paper into grayscale to accommodate the AI model. Despite these preprocessing efforts, OpenCV successfully detected table cells with consistent dimensions, as shown in Fig. 4. However, it struggled with flexibility when faced with irregular table sizes or varying cell dimensions, especially in cases of merged cells or handwritten text overlapping the table lines. The weakness of OpenCV is that it does not have any flexibility towards different cell sizes in a table.

YOLOv8 for first AI model: YOLOv8, a live detection model, is evaluated for its ability to extract tables and marks from exam papers. Training on a custom dataset annotated with bounding boxes for seven classes (Name, ID, Table, Question Numbers, Full Marks, Marks Given, and Total Marks) were conducted based on the front page exam paper as shown in Fig. 5. Fig. 6 demonstrated that the model correctly identified and annotated its classes during the validation process.

YOLOv8 for first AI model: YOLOv8, a live detection model, is evaluated for its ability to extract tables and marks from exam papers. Training on a custom dataset annotated with bounding boxes for seven classes (Name, ID, Table, Question Numbers, Full Marks, Marks Given, and Total Marks). Fig. 6 demonstrated that the model correctly identified and annotated its classes during the validation process.

Untuk Kegunaan Pemeriksa			
Nama Pemeriksa:			
Bahagian	Soalan	Markah Penuh	Markah Diperoleh
A	1	05	5
	2	05	4
B	3	10	6
	4	10	8
	5	05	4
	6	10	8
C	7	05	5
Jumlah		50	40

Fig 3: Table with marks on an exam paper

Untuk Kegunaan Pemeriksa			
Nama Pemeriksa:			
Bahagian	Soalan	Markah Penuh	Markah Diperoleh
A	1	05	5
	2	05	4
B	3	10	6
	4	10	8
	5	05	4
	6	10	8
C	7	05	5
Jumlah		50	40

Fig. 4: Output of the OpenCV model

However, during testing, the model struggled to detect objects accurately, mainly due to insufficient training data, as ScoreSync detects 7 classes, a dataset of 500 – 1000 data would be sufficient. Training used pre-trained weights to leverage knowledge from large datasets. Ultralytics' documentation states that pre-trained weights provide the model with a solid understanding of basic features. Given the limited dataset of 100 images, leveraging pre-trained weights is crucial for performance improvement despite the small dataset. The model's F1 score of 0.81 during validation demonstrates its capability to detect structure information with a moderate performance range from 0.7 to 0.8, justifying its selection as the first AI model in the hybrid pipeline.

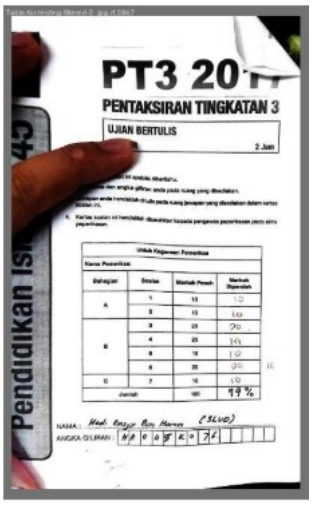

Fig. 5. Full page exam paper

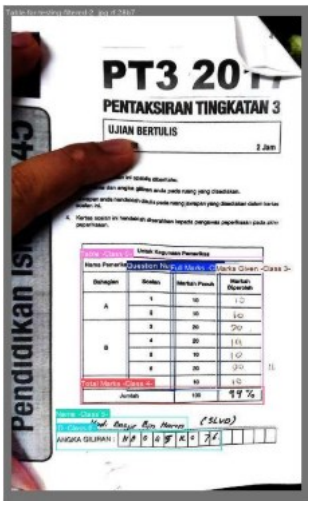

Fig. 6. Output of the YOLOv8 model

PyTesseract for the second AI model: For the first model in OCR, our initial experiment uses the PyTesseract library. PyTesseract is a Python wrapper for the Tesseract OCR engine. However, in practice, the engine often produced inaccurate character recognition, misclassifying numbers, letters, and symbols even after preprocessing. While Tesseract remains actively maintained by the open-source community, its rule-based architecture and older design make it less reliable for the dataset. Because of these limitations, an alternative OCR solution needs to be identified to suit the project requirements.

EasyOCR for second AI model: EasyOCR, a widely used Optical Character Recognition (OCR) library, served as the second model in the hybrid pipeline to extract textual data from exam papers that have been detected and localised specific regions by the first model. EasyOCR allows text extraction, flexibility, and adaptability, which will help in recognising handwritten and printed text that is localised by the first model. EasyOCR also supports multi-language text recognition and is robust against minor variations in handwriting, making it well-suited for diverse exam paper formats. It also demonstrated noticeably better recognition accuracy compared to PyTesseract and effectively handled a variety of text styles. However, the library required a large model footprint and additional dependencies, which raised concerns regarding deployment efficiency and integration with the lightweight

requirements of a mobile app. For this reason, another OCR solution alternative needs to be identified.

Google Gemini for second AI model: Google Gemini is Google's latest multimodal large language model (LLM), and is capable of processing both text and images, making it suitable for interpreting cropped image inputs generated by the first AI model. Unlike traditional OCR engines, Gemini leverages deep learning and natural language processing to improve recognition accuracy and contextual interpretation. Moreover, since Gemini is accessed via an API, it eliminates the need to host large models locally or at the back end of the application, making the deployment lightweight and easier.

B. Choosing the first model

The initial model in the hybrid pipeline is crucial for identifying the structural components of the exam paper. Its function is to identify and categorise essential elements such as the Table, Question Numbers, Full Marks, Marks Given, Total Marks, Name, and ID. The second phase of the pipeline (OCR) can concentrate solely on extracting the essential text segments by isolating these areas from the remainder of the page. The precision of the OCR is highly dependent on the efficacy of the initial model's detection capabilities, underscoring its significance within the entire system.

However, we quickly discovered that making a unique detection pipeline from scratch took a lot of effort and wasn't strong enough to handle the many layouts of exam papers. This made us choose YOLOv8, a cutting-edge object detection architecture that is based on PyTorch. YOLOv8 is better than raw PyTorch implementations since it comes with a ready-to-use, highly optimised detection pipeline that can learn from other datasets. This makes it perfect for our small custom dataset. The best pick for the initial model in our pipeline is PyTorch because it was fast, accurate, and easy to train.

C. Model Performance & Figures

Fig. 7 illustrates that the steady decline of the loss functions (box loss, classification loss, and DFL loss) continues to decline during training, which shows that convergence is stable. The total mAP@0.5 hit its highest point at 0.945, and recall hit its highest point at 0.88. The model still got a score of about 0.8 even when the mAP@0.5–0.95 was tightened. This shows that it is strong across all IoU thresholds. These results show that the model generalises successfully instead of fitting too closely to the training data.

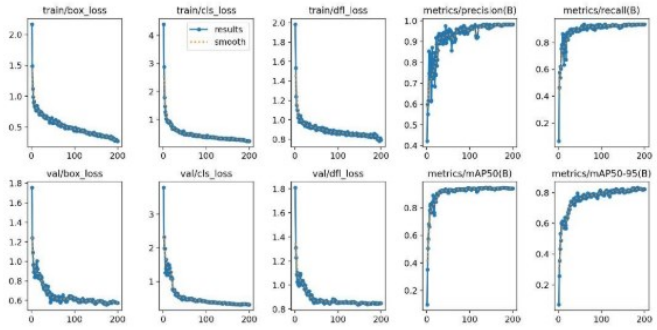

Fig. 7: Training and Validation Curves

As shown in Fig. 8, most of the categories did exceptionally well. Classes such as Full Marks, Marks Given, Question Numbers, Table, and Total Marks got almost perfect precision and recall (around 0.995). The Name class did a little worse at 0.907, and the ID class did the worst at 0.732. This is probably because there weren't many samples with annotations for that group. The weighted average across all classes stayed quite high (mAP@0.5 = 0.945) even if certain classes were weaker.

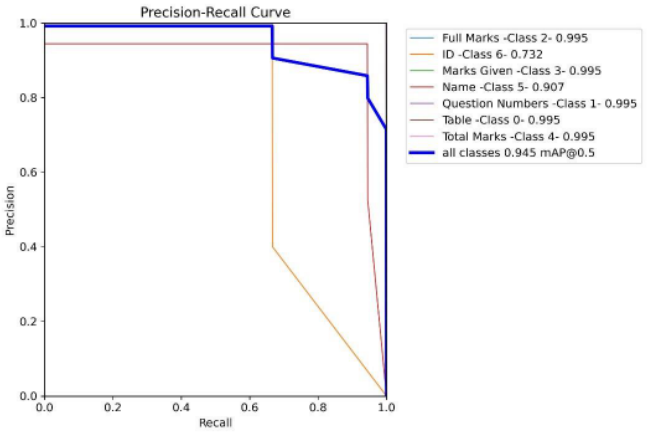

Fig. 8. Precision-Recall Curves by Class

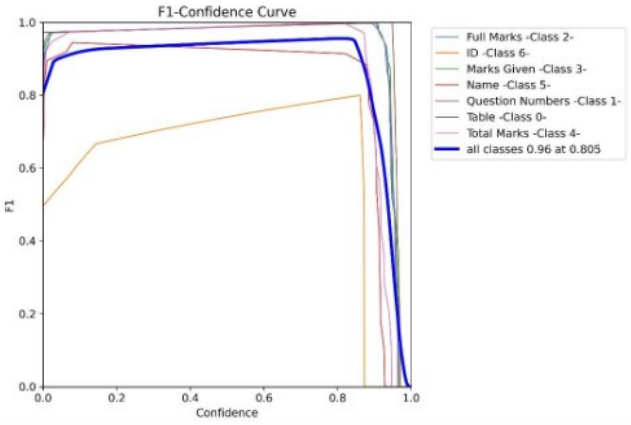

Fig. 9. F1-Confidence Curve

Figure 9 demonstrates how the F1 score changes when the confidence level changes. Most classes preserve their F1 scores near 1.0 throughout a wide range of confidence levels, which further proves that detection is reliable. The Name class drops somewhat to 0.89, while the ID class stands out again with inferior performance, staying below 0.8. The overall (blue) curve shows that the model works best when the F1 score is 0.96 and the threshold is approximately 0.8. This is a solid and useful position to work on.

The normalised confusion matrix in Fig. 10 renders it easy to see how accurate the classification is. The diagonal of most classes (Full Marks, Marks Given, Question Numbers, Table, Total Marks) showed perfect scores of 1.0, which implies near flawless recognition. The Name class got a score of 0.89, which means that some samples were sometimes misclassified as background. The ID class achieved a poorer score of 0.67, with some samples being misclassified or confused with

background regions. This shows that we need more examples with annotations to make ID detection stronger.

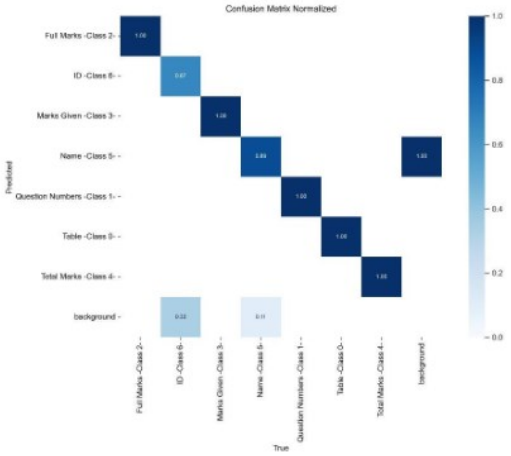

Fig. 10. Confusion Matrix

D. Choosing the second model

Gemini, as the second model, is used to interpret handwritten or printed text within the detected regions of the exam paper. After the YOLOv8 model identifies key components, which are the classes, Google Gemini processes these cropped sections to extract their textual content. The selection of Gemini is due to its multimodal architecture, which integrates both vision and language understanding. Unlike conventional OCR frameworks such as PyTesseract or EasyOCR, which rely solely on pixel pattern matching, Gemini leverages contextual reasoning to interpret numeric and textual information even when handwriting is inconsistent, incomplete, or partially obstructed.

This contextual ability makes Gemini suited for interpreting real-world examination papers, which often contain handwritten annotations, variable pen colours, and table structures that differ between institutions. Gemini API configures the model by using the Gemini-1.5-flash version, chosen for its balance between inference speed and text comprehension accuracy. Gemini will receive an image input encoded in base64 along with a concise prompt that instructs it to identify the relevant marks and return the extracted values in a structured JSON format, enabling it to perform beyond traditional rule-based recognition, providing more flexible and generalizable mark extraction.

However, one might argue that the first model is not necessary since Google Gemini can process the entire page directly without the need for YOLOv8 to identify and extract specific regions; however, providing the entire page often leads to errors that errors such as the model attempting to interpret unnecessary regions beyond the score boxes or being influenced by them. These limitations are evident from the example in the Fig. 11 and Fig. 12.

Testing used the student's name twice in the paper example and resulted in two differing outputs for the class "Name". The name appeared once on a sticker, and once more on the designated space provided on the paper. Despite using the same prompt in both cases, Gemini produced two different interpretations of the student's name. This inconsistency highlights how contextual redundancy within the image can influence the model's understanding. These observations suggest that integrating Gemini with precise detection outputs, being the cropped regions from YOLOv8, is the most effective configuration for achieving mark extraction in future iterations of the ScoreSync system.

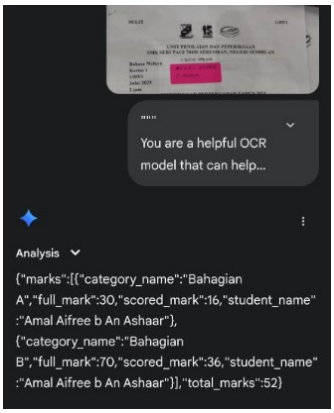

Fig 11. Output from Gemini without YOLOv8

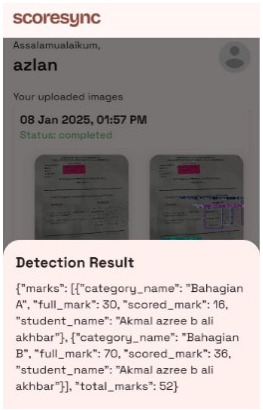

Fig. 12. Output of Gemini with YOLOv8

E. Result of both models in the application

In preliminary private testing conducted on a small dataset of twenty (20) exam papers with varying layouts, handwriting styles, and score table formats, the hybrid ScoreSync system (YOLOv8 + Gemini) demonstrated encouraging qualitative performance. Out of the twenty papers tested, eight (8) were processed perfectly with all score regions and associated classes detected and recognised. Nine (9) papers contained minor recognition errors, some are from YOLOv8, and some are from Gemini. Primarily, typographical inconsistencies in non-numeric classes such as Name and ID. The remaining three (3) papers exhibit incorrect score interpretations.

V. FUTURE WORKS

ScoreSync aims to help educators transform marks from their students' papers into a digital format, reducing the workload involved in manual data entry. Currently, ScoreSync focuses on detecting marks in tables located on the front page of exam papers. In the future, we envision expanding the functionality that could further ease educators' workloads. Some future works are discussed in the next subsections:

A. Enhanced Table Detection and Mark Extraction

Data constraints were a big problem in developing YOLOv8 from scratch. ScoreSync uses a pre-trained weight that would cause a few mishaps in terms of detection. A large dataset is needed to train a weight from scratch that would refine the bounding box detection accuracy and would be able to handle a more complex table structure across even more various exam paper formats.

B. Rigorous Quantitative Evaluation using Metrics

The hybrid model's results in the application were obtained from limited internal testing without standardised benchmarks. A quantitative evaluation using established object detection and

OCR metrics is proposed as part of the future work, allowing the system's performance to be validated objectively and compared across different detection configurations.

C. Real-Time Capture

YOLOv8 has the capability for real-time capture. By introducing this method, educators can hover their mobile device's camera and the app automatically detects, extracts, and transfers marks in real time.

D. Data Analytics for Class Performance

ScoreSync transforms marks from student papers into a digital format. Using this obtained data allows for a detailed analytical report to be generated, including trends in student performance, insights and comparative analyses. These reports can be visualised using graphs and dashboards, assisting educators in identifying areas for improvement.

E. Optimised Setup with Dedicated Phone Stand and Document Tray

During the development phase, we considered an optimised physical setup of the phone stand with an attached document tray. This ensures consistent image capture at fixed distances and angles relative to a mobile camera phone and exam papers. Such consistency will greatly improve image quality, reduce variability of input data, and increase accuracy in mark detection and extraction. For the setup, the process begins by taking pictures of each page of the exam papers and summing the marks automatically within the application. This will greatly improve the efficiency of ScoreSync.

VI. CONCLUSION

ScoreSync represents a significant step toward automating the process of transferring marks from exam papers to a digital format. By integrating YOLOv8 for table and mark detection and Gemini for text recognition, the project demonstrates the use of computer vision and OCR technologies to streamline a traditional, manual and labour-intensive task. While the current implementation successfully creates an app that transfers marks from paper to digital format, several challenges were encountered, particularly in terms of data constraints, causing the app to not achieve 100% accuracy, which is important as the main objective aims to lessen educators' burden without compromising the integrity of the marks and grades. These limitations have provided valuable insights into the importance of dataset quality, annotation consistency and model optimisation for such applications. Despite these challenges, this project lays a solid foundation for further development of the ScoreSync app. To conclude, ScoreSync has demonstrated strong potential to become an application that helps educators in mark entry tasks, demonstrating reliability and accuracy in detecting and extracting information from exam sheets. With further refinement and expanded features, ScoreSync can evolve into a robust and indispensable tool, enhancing the efficiency of educational systems.

ACKNOWLEDGMENT

The authors would like to acknowledge the support of Islamic International University Malaysia (IIUM) and Kulliyah of Information and Communication Technology, Gombak, Selangor, Malaysia, for providing the facilities and support for this project.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest regarding the publication of this paper.

REFERENCES

[1] Raj, A., Sharma, S., Singh, J., & Singh, A. (2023). Revolutionizing data entry: An in-depth study of optical character recognition technology and its future potential. *International Journal for Research in Applied Science and Engineering Technology, 11*(2), 645–653. https://doi.org/10.22214/ijraset.2023.49108

[2] Koushik, K. S., Sourav, C., & Chendan, R. P. (2022). Automated marks entry processing in handwritten answer scripts using character recognition techniques. In *Proceedings of the International Conference* (pp. 728–733). Retrieved from https://ieeexplore.ieee.org/document/9885493

[3] Logeshvar, L., Premnath, A. B., Geethan, R., & Suganya, R. (2021). AI-based examination assessment mark management system. In *International Conference on Secure Cyber Computing and Communications* (pp. 144–149). Retrieved from https://ieeexplore.ieee.org/document/9478091

[4] Sun, M., Song, Z., Jiang, X., Pan, J., & Pang, Y. (2017). Learning pooling for convolutional neural network. *Neurocomputing, 224*, 96–104. Retrieved from https://www.sciencedirect.com/science/article/abs/pii/S0925231216312905

[5] Subasri, R., & Meenakumari, R. (2023). AI-based automatic mark entry system. Retrieved from https://openreview.net/forum?id=-eCgVcWbnzE

[6] Jayoma, J. M., Moyon, E. S., & Morales, E. M. O. (2021). OCR-based document archiving and indexing using PyTesseract: A record management system for DSWD Caraga, Philippines. In *2020 IEEE 12th International Conference on Humanoid, Nanotechnology, Information Technology, Communication and Control, Environment, and Management (HNICEM)*. https://doi.org/10.1109/HNICEM51456.2020.9400000

[7] Salehudin, M. A. M., Basah, S. N., Yazid, H., Basaruddin, K. S., Safar, M. J. A., Mat Som, M. H., & Sidek, K. A. (2023). Analysis of optical character recognition using EasyOCR under image degradation. In *8th International Conference on Man Machine Systems 2023*. *Journal of Physics: Conference Series, 2641*(1), 012001. https://doi.org/10.1088/1742-6596/2641/1/012001

[8] Levchenko, M. (2025). *Evaluating LLMs for historical document OCR: A methodological framework for digital humanities* (arXiv Preprint). Retrieved from https://arxiv.org/abs/2510.06743

[9] Mansoor, K., & Olson, C. F. (2019). Recognizing text with a CNN. In *International Conference on Image and Vision Computing New Zealand* (pp. 1–6). Retrieved from https://ieeexplore.ieee.org/document/8961031

[10] Lim, L. G., Suhaila, B. H., & Norashikin, Y. (2014). Automatic assessment mark entry system using local binary pattern (LBP). In *IEEE International Conference on Control System, Computing and Engineering* (pp. 372–377). Retrieved from https://ieeexplore.ieee.org/document/7072747

[11] Zafar, A., Aamir, M., Mohd Nawi, N., Arshad, A., Riaz, S., Alruban, A., Dutta, A. K., & Almotairi, S. (2022). A comparison of pooling methods for convolutional neural networks. *Applied Sciences, 12*(17), 8643. https://doi.org/10.3390/app12178643

[12] Reis, D., Kupec, J., Hong, J., & Daoudi, A. (2023). *Real-time flying object detection with YOLOv8* (arXiv Preprint). https://doi.org/10.48550/arXiv.2305.09972