Study on ML Based Automated Paper Checking and Evaluation

Akshat Andhale[1], Rohit Chaudhari[2], Pratik Derle[3], Rohit Chaudhari[4], Kunal Ahire[5]

[1] Department of Information Technology, MET's Bhujbal Knowledge City IoE, Nashik, India
[2] Department of Information Technology, MET's Bhujbal Knowledge City IoE, Nashik, India
[3] Department of Information Technology, MET's Bhujbal Knowledge City IoE, Nashik, India
[4] Department of Information Technology, MET's Bhujbal Knowledge City IoE, Nashik, India
[5] Department of Information Technology, MET's Bhujbal Knowledge City IoE, Nashik, India

Corresponding Author Email: akshat.andhale@gmail.com

How to Cite this Article:

Andhale, A., Chaudhari, R., Derle, P., Chaudhari, R. & Ahire, K. (2026). Study on ML Based Automated Paper Checking and Evaluation. International Journal of Creative and Open Research in Engineering and Management, 02(04). https://doi.org/10.55041/ijcope.v2i4.228

https://doi.org/10.55041/ijcope.v2i4.228

Abstract— Evaluating handwritten answer scripts by hand has long been a resource-intensive process, prone to inconsistency and examiner fatigue. As student enrolments continue to rise across educational institutions, the growing volume of answer sheets presents a serious logistical challenge that traditional grading methods are not equipped to handle efficiently. Recent developments in Artificial Intelligence, Machine Learning (ML), Optical Character Recognition (OCR), and Natural Language Processing (NLP) offer a technically credible path toward automating this workflow.

This paper presents an ML-driven system designed to evaluate handwritten answer sheets without manual intervention. The pipeline begins with OCR-based conversion of scanned scripts into machine-readable text, followed by NLP preprocessing and multi-strategy scoring. Evaluation draws on keyword matching, semantic similarity, and syntactic analysis, with transformer-based models—BERT and Sentence-BERT—paired with cosine similarity and Jaccard index metrics to achieve greater precision.

The resulting system produces consistent, bias-free scores, generates detailed performance reports for individual students, and can operate at scale across diverse exam formats, file types, and languages. By uniting document recognition with semantic understanding, the proposed solution addresses key shortcomings in fairness, turnaround time, and transparency that persist in conventional grading practice.

Keywords— Automated Grading, Optical Character Recognition, Natural Language Processing, Semantic Similarity, Machine Learning, Transformer Models

I. INTRODUCTION

Manual grading of examination scripts has served as the primary mechanism for academic assessment for decades. Yet its fundamental limitations have become harder to overlook as institutions scale. Evaluating answer sheets by hand demands considerable time, and the quality of feedback is inevitably shaped by an examiner's workload, fatigue, and individual interpretation of marking criteria. When hundreds or thousands of scripts require grading within a short window, both speed and consistency suffer—and feedback, when it eventually reaches students, often arrives too late to inform any meaningful course of correction.

Against this backdrop, advances in deep learning, OCR, and NLP have opened the possibility of delegating evaluation tasks to automated systems. Machines can now process descriptive and handwritten responses with a degree of linguistic understanding that was simply not achievable a decade ago.

The system described in this paper takes this further by combining text recognition with NLP preprocessing, deep neural network-based feature extraction, and transformer-driven semantic analysis. The goal is a grading pipeline that does not simply match keywords but understands the conceptual content of a student's response—producing evaluations that are fairer, faster, and more transparent than those achieved by prior OCR-centric approaches.

Figure I: Entity Relationship Diagram

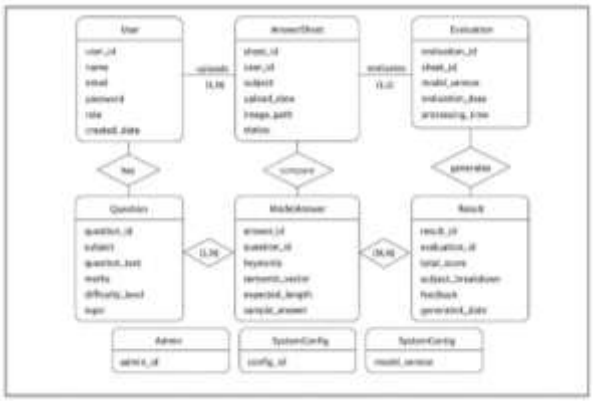


Figure 1 presents the entity relationship diagram underlying the system design. A user submits a handwritten answer sheet through the front-end interface, which persisted as an AnswerSheet entity. This entity maintains associations with Question, ModelAnswer, Evaluation, and Result entities, capturing both one-to-many and many-to-many relationships that reflect the full lifecycle of a grading transaction.

II. SYSTEM ARCHITECTURE

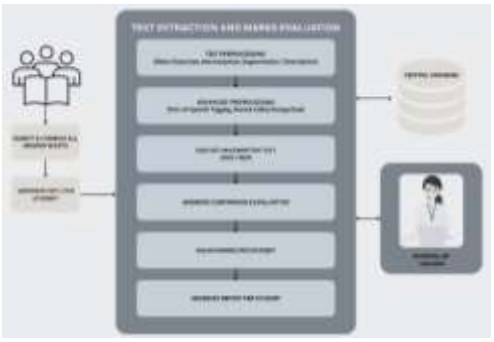
The system is organized as a layered, modular pipeline that takes a scanned answer sheet as input and produces scored results with accompanying feedback. Each component operates through well-defined interfaces, which permits parallel processing and simplifies maintenance. Eight functional modules constitute the full architecture:

The modular design ensures that each component can be tested, updated, or replaced independently without disrupting the rest of the pipeline. A brief account of each module follows:

1) Input and User Interface Module: Teachers and administrators interact with the system through this module. It accepts scanned answer sheets in PDF, JPG, or PNG format and captures associated metadata—exam details, subject identifiers, and student records. The interface is deliberately kept simple and responsive, so that users without technical backgrounds can operate it without difficulty.

2) Image Preprocessing Module: Recognition quality depends heavily on the clarity of the input image. This module applies grayscale conversion, adaptive binarization, noise removal, skew correction, and morphological operations to produce clean, well-structured images. A text region detector further ensures that only answer-bearing areas are forwarded to the recognition stage.

3) OCR and Handwriting Recognition Module: Handwritten text is converted to machine-readable form through a hybrid architecture combining classical OCR with deep learning-based recognition. Convolutional Neural Networks



(CNNs) extract spatial features at the character level, while Bidirectional LSTM (BiLSTM) networks capture sequential context across character sequences. This combination handles the considerable variability found across individual handwriting styles.

4) Semantic Analysis Module: Conceptual similarity between student and model answers is measured by converting text into numerical vector representations. The module employs TF-IDF, Word2Vec, and transformer-based embeddings (BERT and Sentence-BERT). Similarity is then quantified using cosine similarity and the Jaccard index, which assesses how closely ideas align rather than simply checking for shared vocabulary.

5) Machine Learning Scoring Engine: This engine aggregates the similarity scores produced by the semantic module alongside additional features—grammatical correctness, response length, keyword coverage, and factual accuracy. Supervised ML models, including both regression and classification algorithms, produce a final mark for each answer. A weighted scheme balances semantic relevance against structural quality.

6) Result Generation and Feedback Module: Once individual answers have been scored, the module computes total marks and assembles detailed evaluation reports. Reports include topic-level performance breakdowns and are exportable as PDFs or viewable directly through the interface. Feedback is structured to highlight both strengths and areas where a student's understanding appears incomplete.

7) Database and Storage Module: Scanned scripts, extracted text, model answers, scores, and system logs are stored securely through this module. A combination of relational and NoSQL storage is used depending on data structure requirements. Access controls and audit trails are maintained to ensure compliance with institutional data governance policies.

Figure II: System Architecture of the Automated Paper Checking System

III. OPTICAL CHARACTER RECOGNITION TEHNIQUES

Optical Character Recognition converts scanned or photographed text—whether printed or handwritten—into a machine-readable digital form that can be analyzed by NLP and ML components. In the grading context, OCR accuracy is particularly consequential: any downstream semantic analysis is only as good as the text it receives. A systematic character misrecognition at the recognition stage can cascade into incorrect similarity scores and unfair marks.

Handwritten text presents substantially greater challenges than printed input. The variability inherent in individual writing styles—differences in letter forms, character spacing, line inclination, ink density, and the artifacts introduced by scanning hardware—means that recognition systems must be both flexible and robust. Contemporary approaches address this by coupling image preprocessing, region segmentation, feature extraction, and sequence-based deep learning models.

A complete OCR pipeline proceeds through the following stages, each of which contributes to the overall quality of the extracted text:

- Image Preprocessing
- Segmentation
- Feature Extraction
- Character Recognition
- Post-processing and Error Correction

Digitization of handwritten answer sheets is carried out using flatbed scanners or high-resolution cameras. Image quality at this stage directly conditions everything that follows. Resolution, lighting uniformity, page alignment, and background noise each affect how accurate characters can be recognized. For academic applications, scanned PDFs or JPEG images at a minimum resolution of 300 DPI are generally recommended to preserve fine character detail.

Preprocessing refines the raw scanned image before recognition begins. The following operations are applied in sequence:

1) Grayscale Conversion: Color images are reduced to grayscale to lower processing overhead while preserving the structural features that distinguish character strokes.

2) Binarization: Grayscale images are converted to binary black-and-white form using adaptive thresholding. Otsu's method is particularly suited to handwritten documents, as it copes well with uneven illumination across a page.

3) Noise Removal: Scanning artifacts—speckles, smudges, and background texture—are suppressed using median filtering, which smooths noise without substantially blurring character edges.

4) Skew Detection and Correction: Pages that have been placed unevenly in the scanner are realigned using Hough Transform or projection-based methods, ensuring that text lines run horizontally before segmentation begins.

5) Morphological Operations: Dilation and erosion strengthen character strokes and close small gaps caused by thin ink or imperfect scanning, producing cleaner shapes for the recognition stage.

Segmentation partitions the pre-processed image into lines, words, and where necessary individual characters.

1) Line Segmentation: Horizontal projection profiles identify text lines by locating rows of elevated pixel density. This is generally straightforward for neatly written text but requires more adaptive approaches when lines run close together or overlap.

2) Word Segmentation: Word boundaries are inferred from the spacing between connected components. Because interword spacing varies considerably across individuals, adaptive thresholds are applied rather than fixed values.

3) Character Segmentation: Separating characters from cursive handwriting is among the most technically demanding steps in the pipeline. Connected Component Analysis and contour-based methods are used where explicit segmentation is needed; however, many modern deep learning approaches bypass this stage entirely by treating character recognition as a sequence prediction problem.

IV. DEEP LEARNING BASED OCR MODELS

Deep learning has become the dominant paradigm in handwritten text recognition. Four architectural families are most relevant to this domain:

1) Convolutional Neural Networks (CNNs): CNNs learn spatial features—edges, curves, stroke patterns—directly from image data without the need for hand-crafted feature descriptors. They are commonly used for offline character and word recognition.

2) Recurrent Neural Networks (RNNs): LSTM and BiLSTM networks model sequential dependencies across character positions. They are effective for recognizing character runs without requiring prior segmentation into individual glyphs.

3) Convolutional Recurrent Neural Networks (CRNNs): The CRNN architecture combines a CNN front-end for visual feature extraction with an RNN back-end for sequence decoding. Processing an entire word or line image, the model predicts the full character sequence in a single pass—a significant practical advantage for handwriting recognition.

4) Transformer-Based OCR Models: Self-attention mechanisms give transformer models the capacity to capture long-range dependencies across

an image. When paired with vision encoders, they achieve strong robustness against the stylistic diversity and scanning noise encountered in real examination scripts.

Three categories of OCR tools are commonly employed in automated evaluation pipelines:

1. Tesseract OCR: An open-source engine with LSTM-based recognition and multilingual support. It can be fine-tuned on domain-specific handwriting datasets to improve performance for particular script styles.

2. Cloud-Based OCR APIs: Commercially managed services offer high recognition accuracy but introduce recurring costs, external service dependencies, and potential data privacy obligations that may be incompatible with institutional requirements.

3. Custom CNN-Based Models: Task-specific models trained on targeted handwriting datasets can outperform general-purpose engines on narrow domains, at the cost of additional training data and infrastructure investment.

V. NATURAL LANGUAGE PROCESSING (NLP) PREPROCESSING

NLP preprocessing occupies a critical position in the evaluation pipeline—bridging the gap between raw OCR output and the scoring algorithms that follow. The text produced by OCR is rarely clean: it may contain misrecognized characters, structural inconsistencies, redundant terms, and grammatical irregularities introduced by the recognition process. Unless these artefacts are addressed, even sophisticated semantic models will struggle to produce reliable scores.

1) Text Normalization*:* All characters are converted to lowercase, contractions are expanded, and punctuation is standardized to establish a uniform text representation before any further processing.

2) Tokenization: Text is divided into individual units—words or sentences—depending on the analytical requirement. Word tokenization enables term-level comparison; sentence tokenization supports structural analysis.

3) Stop Word Removal: High-frequency function words that contribute little semantic content (such as is, the, a, and of) are removed. This reduces dimensionality and sharpens the focus of similarity computations on content-bearing terms.

4) Stemming: Words are reduced to their root forms by stripping grammatical suffixes. While computing is fast, stemming can produce non-standard word fragments. It is most appropriate in contexts where processing speed takes priority over linguistic precision.

5) Lemmatization: Words are mapped to their canonical dictionary forms using grammatical rules and vocabulary knowledge. Lemmatization preserves semantic coherence more reliably than stemming, making it the preferred option in evaluation systems where answer quality is the primary concern.

6) Spelling Correction and OCR Noise Handling: OCR-generated text frequently contains spelling errors from character misrecognition. Dictionary-based lookup and probabilistic correction methods are applied to recover the intended words and improve text quality before semantic analysis.

7) Part-of-Speech (POS) Tagging: Grammatical roles—noun, verb, adjective, and so on—are assigned to each token. POS tags help the system identify key content words, action-oriented phrases, and structurally informative elements within a student's response.

8) Vectorization and Feature Representation: Preprocessed text is converted into numerical vectors for use in ML models and semantic comparison. Approaches include Bag of Words (BoW), TF-IDF, word embeddings (Word2Vec, GloVe), and contextual embeddings (BERT, Sentence-BERT), each offering a different trade-off between computational cost and representational richness.

VI. SEMANTIC SIMILARITY ALGORITHMS

Scoring descriptive answers requires the system to determine not merely whether a student used the right words but whether their response conveys the right meaning. Traditional approaches use TF-IDF vectors paired with cosine similarity to measure the angular distance between term-weighted representations of the student's answer and the model answer. This approach is computationally economical and performs reasonably well when answers closely follow the model wording, but it fails to recognize conceptually equivalent responses that employ different vocabulary.

Transformer-based models—primarily BERT and Sentence-BERT—address this limitation by producing contextual embeddings that encode the semantic content of a text rather than its surface form. Applied to the answer evaluation problem, cosine similarity computed over these embeddings captures meaning-level alignment far more accurately than vector-space models built on raw term frequencies. The practical consequence is that paraphrased answers, which correctly express the required concept in the student's own words, receive scores commensurate with their actual merit.

Hybrid architecture that combines vector-space representations with classification models trained on graded examples achieve further gains by incorporating grader-level judgments alongside algorithmic similarity signals. The resulting ensemble is more robust across different question types and subject domains than any single method applied in isolation.

VII. PERFORMANCE METRICS

System performance is assessed through the following quantitative measures:

1. Character Error Rate (CER): Captures OCR quality as the proportion of incorrectly recognized characters relative to the total character count. Lower values indicate more reliable text extraction.

2. Semantic Similarity Score: Quantifies the contextual closeness between student and model answers via cosine similarity applied to embedding vectors. Scores range from 0 to 1, with values approaching 1 indicating strong conceptual alignment.

3. Scoring Accuracy: Measures of agreement between system-generated marks and human-assigned marks, typically expressed as mean absolute error or as the proportion of predictions falling within an acceptable tolerance.

4. Precision, Recall, and F1-Score: Applied to classification-based grading tasks to assess how well the system identifies correctly and incorrectly answered questions, and to balance the two types of error.

5. Pearson Correlation Coefficient: Measures the linear relationship between automated scores and human scores across a test set, providing a holistic indication of system fairness.

6. Average Processing Time: The meantime required to evaluate a complete answer sheet, which determines whether the system can operate at the throughput levels demanded by large examination cohorts.

VIII. CHALLENGES AND LIMITATIONS

Significant engineering challenges remain. Highly stylized or degraded handwriting continues to exceed the tolerance of current recognition models. Poor scan quality, particularly from student-submitted photographs taken under variable lighting, introduces noise that preprocessing cannot fully compensate for. Mathematical notation and inline diagrams lie outside the scope of text-based OCR altogether, requiring separate specialized components. Finally, the computational resources needed to run transformer-based models at scale are non-trivial, and cost-efficient deployment on institutional infrastructure remains an open engineering problem. Addressing these constraints is a prerequisite for broad real-world adoption.

IX. CONCLUSION

The work described in this paper demonstrates that ML-driven automated evaluation of handwritten answer scripts is both technically feasible and practically beneficial. By linking OCR-based text extraction with NLP preprocessing, transformer-based semantic analysis, and supervised scoring models, the proposed pipeline reduces evaluation time substantially while maintaining consistent, examiner-independent scoring. The modular architecture supports deployment across a range of institutional settings and examination formats, and the system's performance on standard metrics compares favorably with prior approaches. As semantic understanding techniques continue to mature, the scope for accurate, fair, and transparent automated grading across diverse educational domains will only broaden.

REFERENCES

[1] S. Haller, "Survey on Automated Short Answer Grading with Deep Learning," arXiv preprint arXiv:2204.03503, Apr. 2022.

[2] S. Bonthu et al., "Automated Short Answer Grading Using Deep Learning," in Lecture Notes in Computer Science (LNCS), 2021.

[3] U. V. Marti and H. Bunke, "The IAM-database: An English sentence database for offline handwriting recognition," International Journal on Document Analysis and Recognition, vol. 5, no. 1, pp. 39–46, 2002.

[4] 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," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 11, pp. 2298–2304, Nov. 2017.

[5] W. AlKendi et al., "Advancements and Challenges in Handwritten Text Recognition," Vision, vol. 10, no. 1, 2024.

[6] H. Mouchère, R. Zanibbi, U. Garain, and C. Viard-Gaudin, "Advancing the state-of-the-art for handwritten math recognition: The CROHME competitions, 2011–2014," International Journal on Document Analysis and Recognition, vol. 19, no. 2, pp. 173–189, 2016.

[7] J. Zhang, J. Du, and L. Dai, "Multi-scale attention with dense encoder for handwritten mathematical expression recognition," arXiv preprint arXiv:1801.03530, 2018.

[8] S. Bansal et al., "Evaluating Handwritten Answers Using Deep Learning," in Proceedings of an International Conference on Artificial Intelligence and Education, 2025.

[9] P. A. H. Pawar, "Automating Handwritten Answer Evaluation: A Deep Learning and OCR Integrated Approach," Technical Report, 2025.

[10] Mathpix, "Mathpix: Image to LaTeX and Document Conversion API," [Online]. Available: https://mathpix.com, 2024.

[11] S. Truong, "A survey on handwritten mathematical expression recognition," Pattern Recognition Letters, 2024.

[12] V. Ruiz-Parrado et al., "A bibliometric analysis of off-line handwritten document analysis," Pattern Recognition, vol. 123, 2022.

[13] S. H. Louloudis et al., "Text line and word segmentation of handwritten documents," in Proceedings of the International Conference on Document Analysis and Recognition (ICDAR), 2009.

[14] "AI-Powered Exam Assessment System for Handwritten Answer Sheets," International Journal of Innovative Science and Research Technology (IJISRT), Apr. 2025.

[15] "Automated Handwritten Subjective Answer Evaluation System," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), 2025.