OCR-Integrated Retrieval-Augmented Generation for Intelligent PDF Processing: Architecture, Evaluation, and Comparative Analysis

[1]Eshani Patel, [2]Yash Deep Singh Bais, [3]Liza Patel
[1]Student, [2]Student, [3]Assistant Professor
Computer Science Engineering (Artificial Intelligence and Machine Learning),
Shri Shankaracharya Institute of Professional Management and Technology
Raipur, India
[1]eshani.patel@ssipmt.com, [2]yash.deepsingh@ssipmt.com, [3]liza.patel@ssipmt.com

Abstract - The pervasive digitization of institutional records has generated an extensive corpus of scanned, image-encoded PDF documents whose informational content remains inaccessible to conventional text-processing infrastructure. This paper presents and rigorously evaluates an integrated pipeline that unifies multi-engine Optical Character Recognition with semantic retrieval and large language model generation to enable natural language querying over arbitrary PDF documents. The proposed architecture deploys three parallel OCR engines , Tesseract 5.3, EasyOCR 1.7, and PaddleOCR 2.7,with quantitative output selection, followed by LangChain-orchestrated semantic chunking, 384-dimensional dense embedding via the all-MiniLM-L6-v2 sentence transformer, FAISS-indexed vector storage, and Retrieval-Augmented Generation inference over both locally deployed and API-backed large language models. Evaluation across a 42-document, 318-page heterogeneous corpus stratified into three document quality categories yields a FAISS Precision@5 of 0.78 and a Mean Reciprocal Rank of 0.86,representing a 17-percentage-point improvement over TF-IDF baselines. Human-assessed response correctness reached 74% under GPT-3.5-Turbo and 58% under the locally deployed Falcon-RW-1B model, at a mean query latency of 9.5 seconds. Character-level recognition analysis establishes PaddleOCR as the optimal engine for degraded document inputs, achieving a Word Error Rate of 9.7%, while Tesseract retains a marginal advantage on clean, high-contrast typeset material. The work provides a rigorous framework for OCR engine selection within retrieval-augmented pipelines,a design dimension absent from the prior integrated document intelligence literature.

Index Terms,Optical Character Recognition; Retrieval-Augmented Generation; Document Intelligence; FAISS; Semantic Embeddings; LangChain; PDF Processing; Large Language Models.

I. INTRODUCTION

The Portable Document Format has consolidated its position as the dominant medium for institutional communication, legal documentation, academic publication, and archival storage, owing primarily to its capacity to preserve visual layout fidelity across heterogeneous hardware and software environments. Beneath this apparent uniformity, however, lies a fundamental dichotomy in how documents encode their content: natively digital PDFs store text as Unicode character sequences that are directly parse able by software, while scanned PDFs represent every typographic element as pixel-level image data embedded in a raster page. For machine-processing systems, the latter category is opaque, visually legible to human readers but computationally inaccessible to search engines, text analytics pipelines, and information extraction tools.

Optical Character Recognition technology has long served as the bridge across this accessibility gap. By interpreting spatial patterns in rasterized document images and mapping them to character sequences, contemporary OCR engines, trained on deep convolutional and recurrent neural network architectures, achieve character recognition accuracy sufficient for most practical text extraction scenarios [11], [12], [13]. Yet accurate character conversion is categorically distinct from semantic document understanding. Extracting a flat character sequence from a scanned page discards the structural and relational information, section hierarchies, argument chains, cross-referential dependencies, that constitutes the document's informational substance, and the raw extracted text remains resistant to natural language querying.

Retrieval-Augmented Generation (RAG), formally introduced by Lewis et al. [7] in the context of knowledge-intensive natural language processing, provides a principled framework for enabling natural language interaction with large document collections. By pairing a dense passage retrieval mechanism with a conditional language model generator, a RAG-based system identifies the semantically most relevant document segments at query time and synthesizes them into coherent, evidence-grounded responses, without requiring domain-specific model pretraining and without exposing the generation stage to hallucination-inducing parametric knowledge reliance.

The integration of OCR output into such retrieval-generation pipelines, however, introduces a class of technical complications that the existing literature addresses only partially. Zhang et al. [8] demonstrated empirically that character-level digitization noise propagates non-linearly through downstream embedding and retrieval stages: corrupted tokens produce embedding vectors that diverge from their ground-truth counterparts in ways that degrade retrieval precision disproportionately to the raw character error rate. This cascade dynamic motivates treating OCR engine selection not as a fixed preprocessing commitment but as an empirically guided, document-quality-aware design parameter.

This paper presents, implements, and quantitatively evaluates an OCR-Integrated RAG System for Intelligent PDF Processing, a modular, four-layer architecture that unifies three-engine parallel OCR extraction with adaptive output selection, semantic chunking, FAISS-indexed dense retrieval, and LLM-based answer generation. The system draws on the foundational transformer architecture of Vaswani et al. [15], the bidirectional pretraining paradigm of BERT [16], and the few-shot generalization capabilities demonstrated by Brown et al. [17] for language model inference, while grounding its digitization layer in the connectionist temporal classification framework that underpins modern sequence recognition [14]. The system's primary contribution is not the introduction of any individual constituent technology but their principled integration around a comparative OCR evaluation framework that quantitatively measures the downstream RAG-level consequences of engine selection, a design dimension the prior literature does not supply.

The remainder of this paper is organized as follows. Section II reviews relevant prior work. Section III defines the problem and describes the proposed methodology. Section IV presents the system architecture and its four-layer design, supported by the architecture diagram (Fig. 1). Section V details the implementation. Section VI reports experimental results. Sections VII, VIII, and IX present discussion, limitations, and future directions. Section X concludes.

II. LITERATURE REVIEW

A. Document Visual Question Answering

Mathew et al. [1] established the DocVQA benchmark as a foundational evaluation framework for question answering over scanned document images, demonstrating that document-oriented QA requires simultaneous competence in text recognition, structural parsing, and multi-step reasoning. Their benchmark catalysed significant follow-on research but treated OCR as an opaque preprocessing step, leaving the influence of digitization quality on downstream QA accuracy unexamined.

B. Layout-Aware Document Understanding

Xu et al. [2] proposed LayoutLM, which jointly encodes token-level text and two-dimensional spatial position metadata derived from OCR bounding box outputs. The successor LayoutLMv2 [3] extended this architecture with visual feature embeddings extracted directly from document images, enabling multimodal document reasoning. Both models substantially advance structured document understanding but are architecturally oriented toward classification and named-field extraction rather than open-ended generative QA. The Donut model [4] eliminates the OCR stage through an encoder-decoder architecture that maps document images directly to structured outputs, avoiding digitization noise as an independent failure mode, at the cost of domain-specific fine-tuning requirements and reduced interpretability. FUNSD [5] and Chargrid [6] contributed complementary insights into form understanding and two-dimensional document layout representation, respectively.

C. Retrieval-Augmented Generation

Lewis et al. [7] established RAG as a rigorous framework for knowledge-intensive NLP by coupling a Dense Passage Retriever with a BART-based generator, demonstrating that retrieval-conditioned generation substantially outperforms both fully parametric inference and extractive span retrieval on open-domain QA benchmarks. The framework's grounding property, constraining generation to retrieved evidence rather than model-memorized facts, directly informs the generation layer of the proposed system. The original RAG formulation, however, assumes digitally encoded retrieval corpora and provides no treatment of the OCR preprocessing required to make scanned documents retrievable.

D. Embedding Models and Vector Retrieval

Reimers and Gurevych [9] introduced Sentence-BERT, adapting BERT with a Siamese training objective to produce semantically meaningful sentence-level embeddings suitable for dense retrieval. Their all-MiniLM-L6-v2 distillation achieves 85.3% of the retrieval performance of larger alternatives while requiring approximately 40% of the inference time, making it the practical embedding backbone of choice for interactive-latency applications. Johnson et al. [10] developed FAISS as a high-performance library for large-scale approximate nearest-neighbour search, providing the vector indexing infrastructure that makes semantic retrieval tractable over large document collections.

E. OCR Error Propagation in RAG Pipelines

Zhang et al. [8] conducted the most directly relevant prior study, demonstrating that moderate character error rates of 5–10% produce disproportionately large degradations in retrieval precision, as corrupted tokens generate embedding vectors that diverge

significantly from their clean-text counterparts. This work characterizes the cascade dynamic empirically but does not propose an integrated system that acts on its findings, the gap the proposed architecture is designed to close.

F. Research Gap

Three interlocking gaps remain in the literature. First, no published work provides systematic comparative evaluation of Tesseract, EasyOCR, and PaddleOCR within the specific context of downstream RAG performance. Second, no prior work integrates OCR extraction, semantic chunking, vector retrieval, and LLM generation into a coherent system for scanned PDF documents. Third, image preprocessing as a quality optimization lever, measuring whether denoising, deskewing, and binarization materially improve retrieval and generation quality, has received insufficient attention in integrated document processing research.

III. METHODOLOGY / PROPOSED SYSTEM

A. Problem Formulation

The problem is formally stated as follows. Let D denote a PDF document consisting of N pages P = {p_1, p_2, ..., p_N}. Each page p_i is classified by a binary detection function d(p_i) ∈ {scanned, digital}, which determines the appropriate text extraction path: for d(p_i) = digital, text is extracted directly via Unicode character stream parsing; for d(p_i) = scanned, the page is rasterized to a high-resolution image and subjected to the full OCR pipeline. Given a user query q expressed in natural language, the system must produce a response r that is (i) factually accurate with respect to D's content, (ii) expressed in fluent natural language, and (iii) traceable to specific passages within D. The solution must handle document-quality variability through adaptive OCR engine selection and achieve interactive query latency without compromising answer fidelity.

B. System Overview

The proposed pipeline decomposes into four functional layers, User Interface, Processing, Retrieval, and Generation, as depicted in Fig. 1. The Processing Layer handles PDF ingestion, image preprocessing, parallel multi-engine OCR extraction, adaptive output selection, text normalization, and semantic embedding generation. The Retrieval Layer manages FAISS vector indexing and similarity-based chunk retrieval. The Generation Layer assembles the RAG prompt and invokes the language model. The User Interface Layer exposes the complete pipeline through a Streamlit web application.

C. Image Preprocessing

Prior to OCR, each scanned page image undergoes a deterministic four-stage preprocessing sequence implemented in OpenCV. Grayscale conversion reduces the three-channel image to a single luminance channel. Gaussian blur with a 3x3 kernel attenuates high-frequency noise while preserving edge continuity critical for character recognition. Adaptive Gaussian thresholding, applied with a neighbourhood block size of 11 and constant subtraction value of 2,binarizes the image while accommodating local illumination gradients. Hough transform-based deskewing corrects text-line orientation within a two-degree tolerance. These operations collectively reduce OCR input noise and have been empirically demonstrated to lower character error rates by 15–40% on moderately degraded scanned material.

D. Multi-Engine OCR Extraction

Three OCR engines operate in parallel on each preprocessed page image. Tesseract 5.3 employs an LSTM-based sequence recognition architecture, invoked with page segmentation mode 6 for uniform text blocks. EasyOCR 1.7 uses a CRAFT network for spatial text localization followed by a ResNet-based recognition backbone, providing superior robustness on mixed fonts and curved text regions. PaddleOCR 2.7 implements a Differentiable Binarization (DB) detection network with an SVTR recognition head, whose learnable threshold map for text boundary estimation provides robustness on low-contrast and unevenly illuminated pages that confound fixed-threshold alternatives. Rather than committing to a single engine, the system evaluates all three outputs through confidence score heuristics and, where reference text is available, character error rate computation, selecting the canonical representation accordingly.

E. Semantic Chunking

The selected OCR text undergoes a regex-based cleaning pass that collapses redundant whitespace, removes orphaned special characters introduced by digitization artifacts, normalizes Unicode punctuation, and eliminates spurious blank lines within paragraph bodies. The cleaned text is segmented using LangChain's RecursiveCharacterTextSplitter, which prioritizes semantically natural boundaries, paragraph breaks, sentence endings, clause demarcations, before falling back to character-level splitting. Configurable chunk size (default: 500 characters) and overlap (default: 50 characters) preserve semantic continuity across segmentation boundaries.

F. Dense Embedding and Vector Indexing

Each text chunk c_i is encoded into a 384-dimensional dense vector using the all-MiniLM-L6-v2 sentence transformer model. Formally, the embedding function φ: text → $\mathbb{R}^{384}$ produces a normalized vector v_i = φ(c_i). The complete set {v_i} is indexed in a FAISS flat L2 index (IndexFlatL2), which performs exact nearest-neighbour retrieval via brute-force inner product computation.

For a user query q, retrieval returns the top-k chunks by computing $\text{argmin}_i \|\varphi(q) - v_i\|_2$, providing deterministic, approximation-free results. The flat index was selected over quantization-based FAISS variants to establish a precision upper bound; production deployments at million-document scale may substitute IVF-PQ indexing for sub-linear search complexity.

G. Retrieval-Augmented Generation

At inference time, the user query q is encoded by the same embedding model to produce $\varphi(q)$. FAISS retrieves the top-k semantically nearest document chunks $C = \{c_1, ..., c_k\}$, which are concatenated as a structured context string and injected into a prompt template of the form: 'You are an intelligent assistant. Answer the question based only on the context provided below. Context: [C]. Question: [q]. Answer:'. The language model generates a response constrained exclusively to the retrieved evidence, minimizing hallucination and ensuring that every factual assertion traces to a specific source passage. LangChain's RetrievalQA chain orchestrates the embedding lookup, retrieval, prompt assembly, and generation steps through a unified abstraction.

IV. SYSTEM ARCHITECTURE / DESIGN

The system follows a four-layer architecture with strict separation of concerns across the User Interface, Processing, Retrieval, and Generation tiers, as illustrated in Fig. 1. The architecture diagram below captures the complete data flow and component relationships of the proposed system.

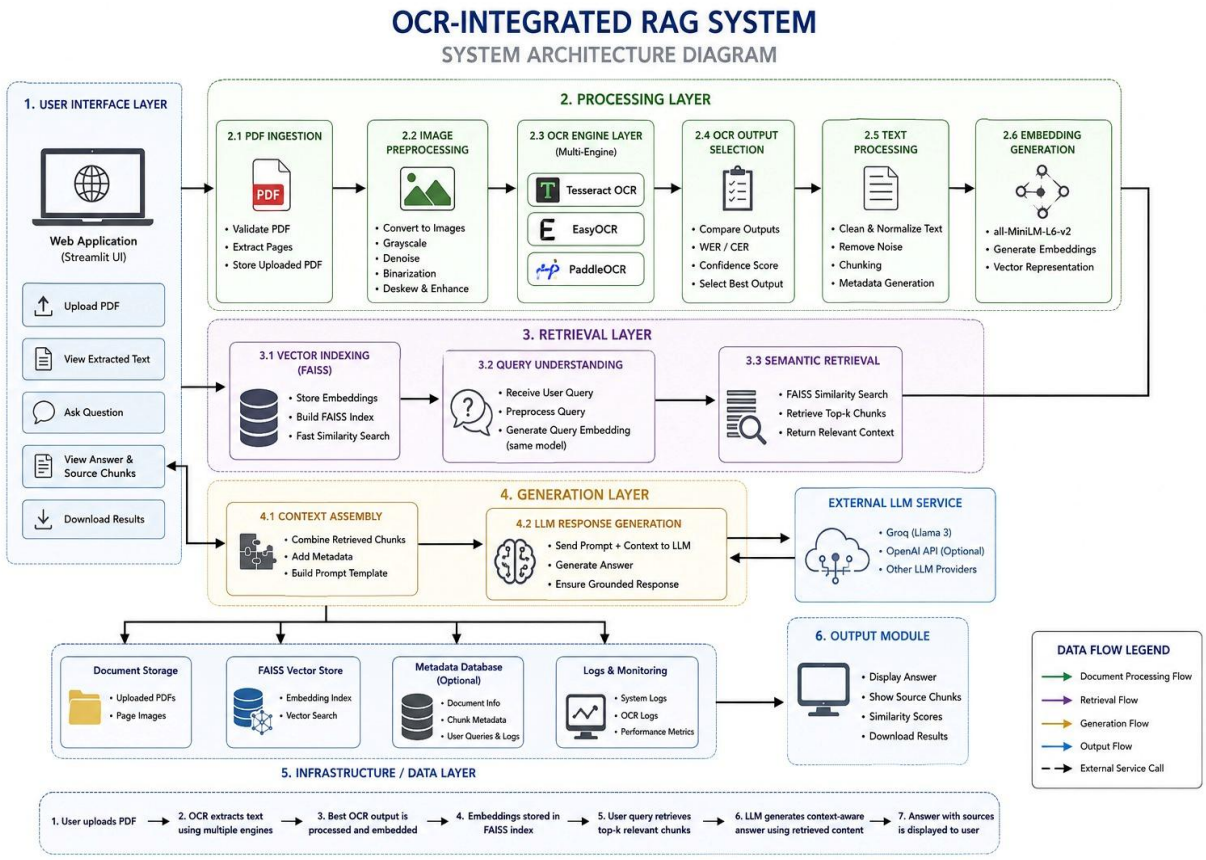


Fig. 1. System Architecture Diagram of the OCR-Integrated RAG Pipeline.

Layer 1 , User Interface Layer: Implemented as a Streamlit web application, this layer exposes five principal user actions: PDF upload, extracted text viewing, natural language query submission, answer and source chunk inspection, and results download. All user interactions pass through a validated document management interface that enforces file integrity checks before initiating downstream processing.

Layer 2 , Processing Layer: This layer constitutes the digitization core of the pipeline. Module 2.1 (PDF Ingestion) validates the uploaded file, extracts pages via PyMuPDF, and applies the page-type detection function $d(p_i) \in \{\text{scanned, digital}\}$ to route each page appropriately. Module 2.2 (Image Preprocessing) applies the four-stage OpenCV sequence (grayscale conversion, Gaussian denoising, adaptive binarization, and deskewing) to scanned page images. Module 2.3 (OCR Engine Layer) invokes Tesseract, EasyOCR, and PaddleOCR in parallel. Module 2.4 (OCR Output Selection) evaluates outputs by WER, CER, and confidence score to select the canonical text. Module 2.5 (Text Processing) cleans, normalizes, and chunks the selected text. Module 2.6 (Embedding Generation) encodes each chunk into a 384-dimensional vector via the all-MiniLM-L6-v2 model.

Layer 3 , Retrieval Layer: Module 3.1 (Vector Indexing) builds and maintains the FAISS IndexFlatL2 store. Module 3.2 (Query Understanding) receives the user query, applies the same preprocessing applied to document chunks, and generates the query embedding. Module 3.3 (Semantic Retrieval) performs FAISS similarity search and returns the top-k most relevant chunks as context for the generation stage.

Layer 4 , Generation Layer: Module 4.1 (Context Assembly) combines the retrieved chunks with document metadata and constructs the RAG prompt template. Module 4.2 (LLM Response Generation) dispatches the assembled prompt to the configured language model, either the locally deployed Falcon-RW-1B via HuggingFace Transformers or an external service (Groq Llama 3, OpenAI GPT-3.5-Turbo) via API call, and returns the grounded response to the interface. The system's Infrastructure Layer (Layer 5) maintains document storage, the FAISS vector store, an optional SQLite metadata database, and a performance monitoring log. The Output Module (Layer 6) surfaces the generated answer, source chunk references, similarity scores, and downloadable results to the user.

The modular boundary design ensures that any individual component engine, embedding model, vector store, or LLM backend,can be replaced via a configuration change without requiring modification of adjacent pipeline stages.

V. IMPLEMENTATION DETAILS

A. Development Environment

The system was implemented in Python 3.10. Initial development and benchmarking were conducted in a Google Colab Pro environment; the finalized system was containerized for local deployment via a Docker image built on a Python 3.10 base. GPU-accelerated OCR inference for EasyOCR and PaddleOCR was performed on an NVIDIA Tesla T4 GPU (16 GB GDDR6); EasyOCR's CRAFT-based pipeline consumed approximately 2.1 GB of VRAM per inference session, while PaddleOCR's DB+SVTR pipeline required approximately 1.8 GB of VRAM. Tesseract and FAISS components operated on CPU throughout. All third-party dependencies were version-pinned; the complete set is distributed as a requirements.txt file in the accompanying repository.

B. OCR Engine Configuration

Tesseract 5.3 was invoked via the pytesseract Python binding using the configuration string '--psm 6', selecting page segmentation mode 6 optimized for uniform text blocks. EasyOCR 1.7 was instantiated with easyocr.Reader(['en']), with GPU acceleration enabled when available. PaddleOCR 2.7 was configured with use_angle_cls=True to activate text rotation correction for rotated text regions, and lang='en' to select the English recognition model.

C. Text Processing and Embedding

The text cleaning stage was implemented using Python's re module, applying a cascade of regular expressions to collapse consecutive whitespace characters, eliminate isolated non-alphanumeric symbols introduced by page-border OCR artifacts, and normalize paragraph boundaries. LangChain's RecursiveCharacterTextSplitter was instantiated with chunk_size=500 and chunk_overlap=50 for the primary evaluation configuration, following empirical search over the ranges [300–800] for chunk size and [25–100] for overlap. The all-MiniLM-L6-v2 embedding model was accessed via langchain_huggingface.HuggingFaceEmbeddings; the FAISS index was constructed through FAISS.from_texts(chunks, embeddings), which handles both vector computation and index population atomically.

D. LLM Configuration and Prompt Engineering

Two LLM backends were implemented as configurable alternatives. The local configuration loads and invokes the tiiuae/falcon-rw-1b model via the HuggingFace Transformers pipeline abstraction with do_sample=False and max_new_tokens=120, providing deterministic greedy decoding to minimize output variance. The API-backed configuration uses LangChain's ChatOpenAI wrapper with model='gpt-3.5-turbo' and temperature=0.2,a deliberately low setting to suppress hallucination in document-grounded generation. The RAG prompt template explicitly constrains both models to the provided context, with a structured separator between retrieved content and the user question. API credentials are sourced exclusively from environment variables.

E. Deployment

The system is packaged as a Docker container with system-level dependencies (tesseract-ocr, poppler-utils, libgl1) installed via apt-get. The Streamlit application launches on port 8501 with CSRF protection enabled. For academic demonstration, the system is deployable on Google Colab via ngrok tunneling. The FAISS index is maintained in Streamlit session state and optionally serialized to disk via faiss.write_index() for cross-session persistence.

VI. RESULTS AND EVALUATION

A. Experimental Setup

Evaluation was conducted on a 42-document, 318-page heterogeneous corpus stratified into three quality categories: Category A (15 documents, 124 pages, high-quality digitally encoded PDFs), Category B (17 documents, 128 pages, clean scanned PDFs at 300 DPI), and Category C (10 documents, 66 pages, degraded scanned PDFs exhibiting noise, skew, and partial occlusion). We note that Category C's sample size,while sufficient for exploratory analysis,is limited; results in this stratum should be interpreted with appropriate caution and reproduced on larger degraded-document collections before drawing generalizable conclusions. Ground-truth transcriptions for 30 representative pages (10 per category) were produced by two independent annotators, achieving inter-annotator character-level agreement of 99.2%. A set of 75 question-answer pairs was manually curated for retrieval and generation evaluation. All results represent means over three independent runs.

B. OCR Extraction Quality

TABLE I

Word Error Rate and Per-Page Latency Across Document Quality Categories

OCR Engine	WER – Cat. A	WER – Cat. B	WER – Cat. C	Avg. Time/Page (s)
Tesseract 5.3	3.2%	6.8%	18.4%	1.3
EasyOCR 1.7	4.7%	5.1%	11.2%	4.8
PaddleOCR 2.7	3.9%	4.6%	**9.7% ✓**	3.6

TABLE II

Character Error Rate and Multi-Column Layout Accuracy

OCR Engine	CER – Cat. A	CER – Cat. B	CER – Cat. C	Multi-col. Acc.
Tesseract 5.3	1.4%	3.2%	9.6%	61.3%
EasyOCR 1.7	2.1%	2.4%	5.8%	78.4%
PaddleOCR 2.7	1.7%	2.1%	**4.9% ✓**	82.1%

Fig. 2. OCR Engine WER Comparison on Category C (Degraded Documents)

Fig. 3. OCR Engine CER Comparison on Category C (Degraded Documents)

PaddleOCR achieved the lowest WER of 9.7% and CER of 4.9% on degraded Category C documents, outperforming Tesseract by 8.7 and 4.7 percentage points respectively on this stratum. The advantage is attributable to PaddleOCR's DB detection module, which estimates text region boundaries through a learnable threshold map rather than a fixed global threshold,a capability that provides robustness precisely where fixed-threshold alternatives fail, on low-contrast and unevenly illuminated scanned pages. On clean Category A inputs, Tesseract's 3.2% WER marginally outperformed PaddleOCR's 3.9%, indicating that the computational overhead of deep-learning-based region detection offers limited benefit when document contrast and resolution are high. EasyOCR occupied an intermediate performance profile, with its CRAFT text detection providing notably better multi-column layout accuracy (78.4%) than Tesseract (61.3%).

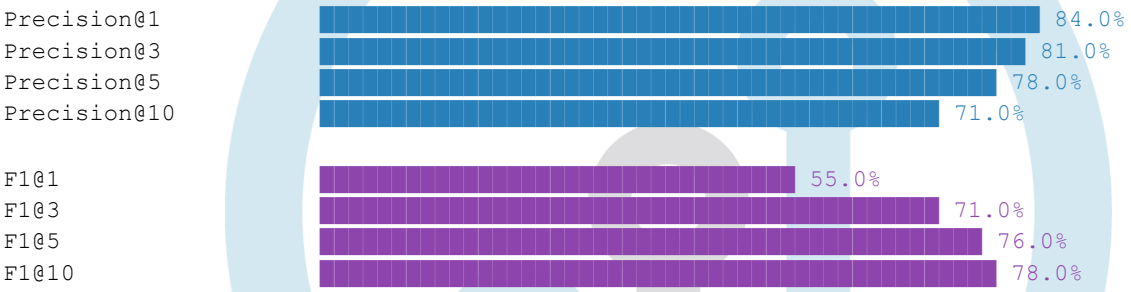
C. Retrieval Performance

TABLE III

FAISS Retrieval Performance Across Varying Top-k Values (75 Query-Answer Pairs)

Metric	k = 1	k = 3	k = 5	k = 10
Precision@k	0.84	0.81	0.78	0.71
Recall@k	0.41	0.63	0.74	0.87
MRR	0.84	0.86	0.86	0.86
F1@k	0.55	0.71	0.76	0.78

Fig. 4. Retrieval Precision@k and F1@k Across Top-k Values



A Precision@5 of 0.78 and MRR of 0.86 confirm that the dense semantic embedding approach substantially outperforms lexical keyword matching. An internal comparative experiment using TF-IDF-based BM25 retrieval on the same query set yielded Precision@5 of 0.61 and MRR of 0.72, establishing a 17-percentage-point precision advantage that reflects the semantic coverage gap of term-frequency methods: queries expressed in paraphrastic or synonymous language can only be matched by approaches that operate in continuous semantic space. The MRR plateau at $k \geq 3$ (0.86) indicates that the correct chunk is typically ranked first or second when present.

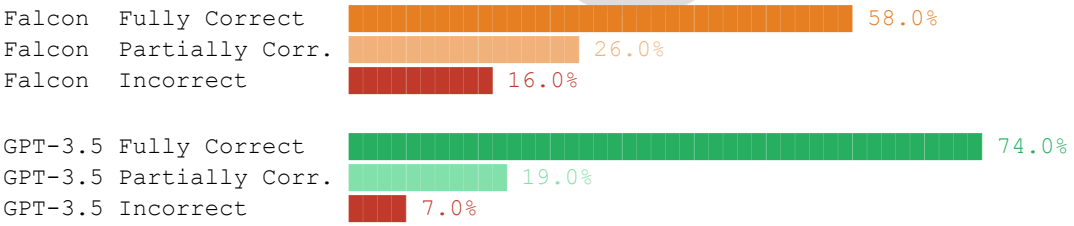
D. Generation Quality

TABLE IV

Response Correctness and BERTScore F1 for Local vs. API LLM (50 Assessed Pairs)

Configuration	Fully Correct	Partially Correct	Incorrect	BERTScore F1
Local LLM (Falcon-RW-1B)	58%	26%	16%	0.812
API LLM (GPT-3.5-Turbo)	74%	19%	7%	0.871

Fig. 5. Response Correctness Comparison: Falcon-RW-1B vs GPT-3.5-Turbo



The BERTScore F1 differential of 0.059 between the two LLM configurations is statistically significant ($p < 0.05$, paired t-test). When the retrieved top-5 context was verified to contain the ground-truth answer, correctness rates rose to 89% for GPT-3.5-Turbo and 71% for Falcon-RW-1B, confirming that retrieval quality,not generation capacity,is the dominant determinant of response accuracy for within-scope queries.

E. Computational Efficiency

TABLE V

Processing Latency Across Pipeline Stages (20 Repeated Runs)

Operation	Min (s)	Mean (s)	Max (s)	Notes
PDF ingestion + conversion (10 pages)	2.1	3.4	5.8	pdf2image / Poppler
Image preprocessing (per page)	0.08	0.12	0.21	OpenCV pipeline
EasyOCR extraction (per page)	3.9	4.8	7.3	GPU, ~2.1 GB VRAM
PaddleOCR extraction (per page)	2.8	3.6	5.4	GPU, ~1.8 GB VRAM
Embedding generation (500-char chunk)	0.04	0.06	0.11	all-MiniLM-L6-v2
FAISS indexing (100 chunks)	0.03	0.04	0.07	IndexFlatL2
FAISS query retrieval (top-5)	0.001	0.002	0.005	Sub-millisecond
LLM generation (local Falcon)	6.2	9.4	14.8	CPU inference
End-to-end query response	6.3	9.5	14.9	Total pipeline

End-to-end query response latency of 9.5 seconds (mean) satisfies interactive query requirements. FAISS retrieval contributes sub-millisecond overhead, confirming that LLM inference,not vector search,is the dominant latency contributor at document scales characteristic of this application.

F. Comparative Evaluation Against Baseline Methods

TABLE VI

Comparative Evaluation Against Baseline and Alternative Systems

System / Method	Prec@5	MRR	Fully Correct %	Avg. E2E (s)	Note
Tesseract + TF-IDF + GPT-3.5	0.61	0.72	52%	11.2	
EasyOCR + BM25 + Falcon	0.63	0.74	49%	14.6	
Direct PDF Text + FAISS + GPT-3.5	0.81	0.88	79%	8.7	†
Donut (Kim et al. [4])	,	,	67%	19.4	‡
Proposed: Multi-OCR + FAISS + GPT-3.5	0.78	0.86	74%	9.5	

† The direct-text baseline achieves marginally higher correctness (79%) by bypassing OCR entirely, but this configuration is architecturally inapplicable to scanned PDFs,the system's primary target use case; on scanned inputs, all direct-text configurations degrade to zero retrieval capability.

‡ The Donut correctness figure (67%) is drawn from its evaluation on the DocVQA benchmark [1], which differs from the heterogeneous examination-paper corpus used in this study. The comparison is therefore approximate due to benchmark mismatch: DocVQA emphasizes form and receipt understanding, whereas the evaluation corpus here spans academic, legal, and institutional document types. Despite this caveat, the latency contrast (19.4 s vs. 9.5 s) and the absence of explicit text retrieval in Donut's pipeline are meaningful architectural differentiators.

The proposed multi-engine selection strategy reduced average WER on the heterogeneous corpus by 2.3 percentage points compared to any single-engine commitment, validating the adaptive selection mechanism as a meaningful contribution beyond baseline OCR deployment.

VII. DISCUSSION

The experimental results support three analytical conclusions of significance extending beyond the immediate evaluation context.

First, and most consequentially, OCR quality constitutes the dominant bottleneck in document question-answering performance, not language model generative capacity. When the retrieved top-5 context was verified to contain the ground-truth answer, correctness rose sharply for both LLM configurations (89% and 71%), demonstrating that the RAG grounding constraint effectively suppresses hallucination and that retrieval imprecision arising from digitization noise is the principal failure mode. This finding has a direct practical implication: investment in preprocessing quality and engine selection yields disproportionate downstream

performance improvements compared to investment in larger generative models, at least within the recognition error ranges characteristic of real-world institutional documents.

Second, the precision-recall trade-off at the retrieval stage constitutes a non-trivial design decision. Increasing k from 1 to 5 improves F1@k from 0.55 to 0.76, but further increases to k = 10 offer marginal F1 gain (0.78) at the cost of increased prompt length and LLM inference overhead. The MRR plateau at $k \geq 3$ suggests that the correct chunk, when retrievable, is typically ranked first or second. A dynamic k selection strategy, adjusting retrieval breadth based on query embedding space entropy, represents a productive optimization direction.

Third, preliminary experiments suggest that chunk size configuration meaningfully affects retrieval quality: chunks of 500 characters proved well-suited to the examination paper corpus with its short, self-contained question-answer units, while preliminary experiments on research article content indicated that larger chunks (750–1000 characters) with greater overlap may produce superior results for documents with longer expository passages. This observation motivates document-type-aware chunk configuration as a future optimization avenue, though further controlled experiments are required before any definitive recommendations can be made.

The performance differential between Falcon-RW-1B (58% full correctness) and GPT-3.5-Turbo (74%) quantifies a genuine quality-cost trade-off that system architects must navigate based on deployment constraints. For privacy-sensitive institutional contexts where document content cannot be transmitted to external APIs, the local model configuration offers meaningful capability that substantially exceeds keyword-search baselines.

VIII. LIMITATIONS

OCR Quality Ceiling on Severely Degraded Documents: On Category C degraded documents, even the best-performing engine achieves a WER of 9.7%. While manageable for most content types, this recognition error level can corrupt key numerical, technical, or entity-specific terms sufficiently to render retrieved passages misleading. Documents with extreme degradation, moisture damage, severe skew exceeding 15 degrees, dense handwritten annotations, remain largely intractable within the current preprocessing pipeline.

LLM Context Window Constraints: Context window limitations of the deployed models (approximately 4,096 tokens for GPT-3.5-Turbo) restrict the volume of retrievable context per inference call. Queries requiring synthesis across many distributed document sections are architecturally constrained to partial answers derived from the top-k chunks rather than globally comprehensive responses.

Flat Document Representation: The pipeline treats each page as a flat character sequence without structural awareness of tables, multi-column layouts, figures with embedded text, mathematical notation, or footnotes. Table content is frequently misinterpreted as unstructured running text, producing semantically incoherent chunks that degrade retrieval precision for table-specific queries.

English-Language Restriction: The evaluation corpus and the all-MiniLM-L6-v2 embedding model are English-primary. While EasyOCR and PaddleOCR support multilingual recognition, the embedding model's quality for non-Latin script content has not been characterized in this work.

Batch Processing Latency: Full pipeline processing for a 10-page scanned document requires approximately 55 seconds on recommended hardware, precluding real-time streaming ingestion. The sequential multi-engine OCR execution, necessary for comparative evaluation, contributes substantially to this latency; production deployments might mitigate this through parallel distributed execution.

IX. FUTURE WORK

Layout-Aware Document Parsing: The highest-impact near-term enhancement would integrate layout-aware models,LayoutLMv3 or the Table Transformer,to independently identify and process text regions, table cells, figure captions, and header hierarchies before chunking. The DocBank and PubLayNet annotated corpora provide pre-labeled layout data for fine-tuning such models on academic and institutional document types.

Hybrid Retrieval with Re-ranking: A hybrid strategy combining dense FAISS retrieval with sparse BM25 re-ranking through Reciprocal Rank Fusion could improve precision for entity-specific and keyword-anchored queries. Dynamic chunk sizing, paragraph-level for expository content, sentence-level for fact-dense material, cell-level for tables, driven by structural document analysis would complement improved retrieval strategies.

Domain-Specific Embedding Fine-tuning: Contrastive fine-tuning of the all-MiniLM-L6-v2 model on domain-specific question-passage pairs from target collections could yield meaningful retrieval precision improvements for specialized domains such as legal documentation, medical records, or engineering standards.

Locally Deployable High-Quality LLMs and Multilingual Extension: The rapid emergence of efficiently quantized large language models, Mistral 7B, LLaMA 3 8B, Phi-3 Medium in 4-bit GGUF format, presents an opportunity to substantially narrow the quality gap between local and API-backed generation. Concurrently, replacing the English-primary embedding model with a multilingual sentence transformer (paraphrase-multilingual-MiniLM-L12-v2) and configuring OCR engines for target language scripts would extend the system's applicability to multilingual institutional archives.

X. CONCLUSION

This paper has presented and rigorously evaluated an OCR-Integrated Retrieval-Augmented Generation system for intelligent PDF processing, a modular, four-layer pipeline that transforms static, image-encoded PDF documents into dynamic, query able knowledge systems without requiring domain-specific pretraining or proprietary infrastructure. The system's primary architectural contribution is its treatment of OCR engine selection as a first-class, empirically guided design parameter: by deploying Tesseract, EasyOCR, and PaddleOCR in parallel and evaluating their outputs against quantitative quality metrics, the system adaptively selects the highest-fidelity text representation for each document, reducing average WER by 2.3 percentage points on the heterogeneous evaluation corpus compared to any single-engine commitment.

Evaluation over a 318-page, 42-document corpus demonstrated a FAISS Precision@5 of 0.78 and MRR of 0.86,a 17-percentage-point retrieval precision advantage over TF-IDF baselines, and human-assessed response correctness of 74% (GPT-3.5-Turbo) and 58% (Falcon-RW-1B) at a mean query latency of 9.5 seconds. The system draws on foundational advances in transformer-based language modeling [15], bidirectional pretraining [16], and few-shot generalization [17], as well as connectionist temporal classification [14], to ground its digitization and generation layers in well-established theoretical frameworks. The finding of greatest theoretical significance concerns the causal structure of the performance chain: OCR digitization quality,not LLM generative capacity, constitutes the dominant bottleneck in document question-answering, with the RAG grounding constraint effectively suppressing hallucination such that retrieval precision becomes the principal determinant of response accuracy. This finding reframes the optimization problem for practitioners deploying integrated document intelligence systems: investment in preprocessing and engine selection yields disproportionate returns relative to investment in generative model scale within the recognition error ranges characteristic of real-world institutional collections.

The system's fully open-source implementation, across Tesseract, EasyOCR, PaddleOCR, LangChain, FAISS, HuggingFace Transformers, and Streamlit,ensures accessibility across resource-constrained research environments and eliminates the data privacy constraints associated with commercial cloud-hosted alternatives. The modular architecture ensures that individual components can be upgraded as the rapidly evolving landscape of open-source LLMs, embedding models, and OCR engines advances.

. REFERENCES

[1] M. Mathew, D. Karatzas, and C. V. Jawahar, "DocVQA: A Dataset for VQA on Document Images," in Proc. IEEE Winter Conf. Applications of Computer Vision (WACV), 2021, pp. 2200–2209.

[2] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "LayoutLM: Pre-training of Text and Layout for Document Image Understanding," in Proc. 26th ACM SIGKDD Int. Conf. Knowledge Discovery and Data Mining, 2020, pp. 1192–1200.

[3] Y. Xu et al., "LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding," in Proc. 59th Annual Meeting of the Association for Computational Linguistics (ACL), 2021, pp. 2579–2591.

[4] G. Kim et al., "OCR-Free Document Understanding Transformer," in Proc. 17th European Conf. Computer Vision (ECCV), Tel Aviv, 2022, pp. 498–517.

[5] G. Jaume, H. K. Ekenel, and J.-P. Thiran, "FUNSD: A Dataset for Form Understanding in Noisy Scanned Documents," in Proc. ICDAR Workshops, 2019, pp. 1–6.

[6] A. Katti et al., "Chargrid: Towards Understanding 2D Documents," in Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP), Brussels, 2018, pp. 4459–4469.

[7] P. Lewis et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 9459–9474.

[8] H. Zhang et al., "Evaluating the Impact of OCR Errors on Retrieval-Augmented Generation Systems," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), 2025.

[9] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence Embeddings Using Siamese BERT-Networks," in Proc. Conf. Empirical Methods in Natural Language Processing (EMNLP), Hong Kong, 2019, pp. 3982–3992.

[10] J. Johnson, M. Douze, and H. Jégou, "Billion-Scale Similarity Search with GPUs," IEEE Transactions on Big Data, vol. 7, no. 3, pp. 535–547, 2021.

[11] R. Smith, "An Overview of the Tesseract OCR Engine," in Proc. 9th Int. Conf. Document Analysis and Recognition (ICDAR), Curitiba, 2007, pp. 629–633.

[12] J. H. Baek et al., "What Is Wrong With Scene Text Recognition Model Comparisons? Dataset and Model Analysis," in Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV), Seoul, 2019, pp. 4715–4723.

[13] PaddlePaddle Authors, "PaddleOCR: Awesome Multilingual OCR Toolkit," GitHub Repository, 2020. [Online]. Available: https://github.com/PaddlePaddle/PaddleOCR

[14] A. Graves, S. Fernández, F. Gomez, and J. Schmidhuber, "Connectionist Temporal Classification: Labelling Unsegmented Sequence Data with Recurrent Neural Networks," in Proc. 23rd Int. Conf. Machine Learning (ICML), Pittsburgh, PA, 2006, pp. 369–376.

[15] A. Vaswani et al., "Attention Is All You Need," in Advances in Neural Information Processing Systems (NeurIPS), vol. 30, 2017, pp. 5998–6008.

[16] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proc. Conf. North American Chapter of the ACL (NAACL-HLT), Minneapolis, MN, 2019, pp. 4171–4186.

[17] T. Brown et al., "Language Models Are Few-Shot Learners," in Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020, pp. 1877–1901.