International Journal of Science and Research (IJSR)
ISSN: 2319-7064
Impact Factor 2025: 7.089

Automated Web Data Extraction Using OCR and RPA

Abhishek Tiwari[1], Amandeep Ahlawat[2], Mohit Poriya[3], Satyam[4], Himani Chaudhary[5]

[1]Student Department of CS&IT, MIET
Email: *abhisehk.tiwari.csit.2022*

[2]Student Department of CS&IT, MIET
Email: *amandeep.ahlawat.csit.2022*

[3]Student Department of CS&IT, MIET
Email: *mohit.poriya.csit.2022*

[4]Student Department of CS&IT, MIET
Email: *satyam.bablu.csit.2022*

[5]Associate Professor, Department CS&IT, MIET
Email: *himani.chaudhary[at]miet.ac.in*

Abstract: ***This research paper presents an automated system for web data extraction by integrating Optical Character Recognition (OCR) and Robotic Process Automation (RPA). The proposed system addresses the growing need for efficient extraction of unstructured data from web sources, particularly from documents, images, and screenshots embedded within web interfaces. By combining OCR engines for text recognition with RPA bots for workflow automation, the system achieves seamless extraction, validation, and consolidation of data into structured formats. The methodology employs a hybrid approach where RPA automates navigation and data capture while OCR handles text recognition from non-selectable sources. Experimental results demonstrate that the integrated system reduces manual processing time by approximately 65% for batch operations and achieves extraction accuracy exceeding 95% for standardized documents. The system's modular architecture enables deployment across diverse domains including financial document processing, invoice management, and form data extraction, offering significant improvements in operational efficiency and data accuracy.***

Keywords: Optical Character Recognition, Robotic Process Automation, Web Data Extraction, Intelligent Document Processing, Workflow Automation

1. Introduction

The exponential growth of digital data has created an unprecedented challenge for organizations seeking to extract meaningful information from diverse web sources. According to recent industry estimates, businesses spend nearly 40% of their operational time on manual data entry and validation tasks, with a significant portion involving unstructured data embedded in documents, images, and web interfaces. This manual approach not only consumes valuable human resources but also introduces errors that propagate through downstream business processes.

The COVID-19 pandemic accelerated digital transformation initiatives across industries, highlighting the critical need for automated solutions that can handle high-volume data processing without human intervention. Traditional web scraping techniques, while effective for structured HTML data, fail when information is embedded within images, scanned documents, or complex PDF layouts. This limitation has driven the convergence of two complementary technologies: Optical Character Recognition (OCR) for extracting text from visual sources and Robotic Process Automation (RPA) for orchestrating end-to-end workflow automation.

Optical Character Recognition technology has evolved significantly from its early days of simple text digitization. Modern OCR engines leverage deep learning architectures to achieve remarkable accuracy even with challenging inputs such as handwritten text, low-resolution images, and complex document layouts. However, OCR alone cannot address the complete automation challenge—it requires integration with workflow systems that can navigate to data sources, trigger recognition processes, validate extracted information, and route results to destination systems.

Robotic Process Automation fills this gap by providing the orchestration layer that mimics human interactions with digital systems. RPA bots can log into web portals, navigate through multiple pages, trigger OCR processing, handle exceptions, and consolidate results into databases or spreadsheets. The synergy between OCR and RPA creates a powerful automation paradigm capable of handling complex extraction scenarios that neither technology could address independently.

This research focuses on developing an integrated framework for automated web data extraction that combines state-of-the-art OCR engines with RPA workflow automation. The primary objectives include:

1) Designing a modular architecture that supports multiple OCR engines and RPA platforms
2) Developing validation mechanisms to ensure extraction accuracy and handle edge cases
3) Implementing confidence-based escalation workflows for low-certainty extractions

4) Evaluating system performance across diverse document types and web sources
5) Demonstrating practical applications in financial services, invoice processing, and form automation

The remainder of this paper is organized as follows: Section 2 reviews related work in OCR technology, RPA systems, and integrated extraction frameworks. Section 3 presents the proposed methodology and system architecture. Section 4 describes experimental setup and implementation details. Section 5 discusses results and performance evaluation. Section 6 concludes with findings and future research directions.

2. Related Work

2.1 Evolution of Optical Character Recognition

Optical Character Recognition has undergone substantial transformation since its inception. Early systems relied on template matching and feature extraction techniques that required carefully formatted inputs and struggled with variations in font, size, and image quality. The introduction of convolutional neural networks (CNNs) revolutionized the field, enabling end-to-end text recognition that could handle real-world variations.

Contemporary OCR engines employ diverse architectural approaches. Tesseract, originally developed by HP and now maintained by Google, combines traditional layout analysis with LSTM-based recognition networks. PaddleOCR, developed by Baidu, utilizes a differentiable binarization approach for text detection followed by attention-based recognition. EasyOCR provides a lightweight alternative supporting multiple languages with pre-trained models. DocTR focuses specifically on document understanding with specialized handling for tables and complex layouts.

Recent advances have integrated Large Language Models (LLMs) into OCR workflows to address challenges in context understanding and ambiguous text interpretation. The LMV-RPA system demonstrated that combining outputs from multiple OCR engines with LLM-based voting mechanisms achieves 99% accuracy, surpassing individual baseline models by significant margins. This multi-engine approach proves particularly effective for documents with complex layouts where no single OCR engine excels across all elements.

2.2 Robotic Process Automation for Document Processing

Robotic Process Automation has emerged as a dominant force in business process automation, with platforms like UiPath, Automation Anywhere, and Blue Prism capturing significant market share. These platforms provide visual designers for creating automation workflows that interact with applications through user interfaces, APIs, and database connections.

The integration of RPA with document processing represents a natural evolution. Traditional RPA excelled at structured data manipulation but struggled with unstructured content. By incorporating OCR capabilities, modern RPA platforms enable end-to-end automation of document-centric processes. Comparative studies have shown that RPA platforms achieve processing times of 18-20 seconds for standardized OCR tasks, though performance varies significantly based on document complexity and platform architecture.

Recent research introduced LMRPA, a large model-driven RPA approach that leverages LLMs to optimize OCR workflows. This system demonstrated 52% reduction in processing time compared to traditional RPA platforms by dynamically selecting optimal OCR strategies and parallelizing extraction tasks. The findings suggest that intelligent orchestration can yield substantial efficiency gains beyond what individual OCR engines provide.

2.3 Integrated OCR-RPA Frameworks

The convergence of OCR and RPA has spawned multiple integrated frameworks targeting specific use cases. ERPA, designed for immigration document processing, combines specialized OCR for identity documents with RPA workflows that validate extracted data against government databases. This system achieved 94% reduction in processing time for ID verification tasks, completing extractions in under 10 seconds compared to manual processing requiring several minutes.

Enterprise solutions like Box Extract demonstrate the maturity of integrated approaches. These systems incorporate multiple layers including document classification, intelligent OCR, validation workflows, and metadata storage. The architecture emphasizes security and compliance, with permission verification at every stage and human-in-the-loop review for low-confidence extractions. Such enterprise-grade systems highlight the importance of operational considerations beyond pure technical performance.

Financial institutions have been early adopters of integrated OCR-RPA solutions. Keiyo Bank's implementation combining WinActor RPA with DX Suite AI-OCR achieved cumulative reduction of over 26,000 work hours across 82 automated processes. The bank's experience demonstrates that while single-document processing may not show dramatic speed improvements, batch processing of multiple documents yields 65% time savings—a pattern consistent with many automation scenarios.

2.4 Hybrid and Multi-Model Approaches

Recent work has explored hybrid approaches combining multiple AI models within unified extraction pipelines. The financial data extraction pipeline developed by zk2k2 integrates PaddleOCR for text extraction with LLM-based semantic parsing to generate structured JSON output. This bimodal approach addresses the limitation that OCR provides raw text without understanding of document semantics, while LLMs can interpret context but require accurate text input.

Community experiments with multi-model workflows reveal practical insights for production deployments. Developers report success with tiered approaches where lightweight OCR models handle initial extraction, with escalation to higher-accuracy models only when confidence thresholds fall below acceptable levels. This strategy balances throughput and

accuracy while managing API costs. Additional techniques include confidence-based validation gates, preprocessing for challenging inputs, and caching intermediate results to avoid redundant processing.

The GitHub community has proposed comprehensive workflows for OCR-based data extraction that incorporate file discovery, multi-format support, human validation, and Excel consolidation. These open-source initiatives demonstrate the growing ecosystem of tools and techniques available for building custom extraction solutions, though they require significant integration effort compared to commercial platforms.

2.5 Research Gaps and Contributions

Despite substantial progress, several research gaps remain in OCR-RPA integration. First, systematic comparison of different integration architectures is lacking—most studies focus on point solutions rather than generalizable frameworks. Second, confidence estimation and uncertainty handling in extraction pipelines require deeper investigation, particularly for edge cases where all models produce low-confidence outputs. Third, the trade-offs between cloud-based and on-premise deployment for sensitive document processing deserve more attention.

This research contributes to addressing these gaps by proposing a modular architecture that supports multiple OCR engines and RPA platforms, implementing confidence-based validation with escalation workflows, and evaluating performance across diverse document types. The findings provide practical guidance for organizations seeking to implement automated extraction systems while contributing theoretical insights into integrated AI-RPA system design.

3. Proposed Methodology

3.1 System Architecture

The proposed Automated Web Data Extraction system follows a modular, layered architecture designed for flexibility, scalability, and maintainability. Figure 1 illustrates the high-level system components and their interactions.

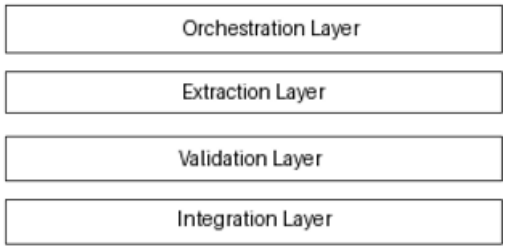

Figure 1: System Architecture Diagram

The architecture comprises four primary layers:

1) **Orchestration Layer**: Implemented using RPA bots that manage workflow execution, including navigation to data sources, triggering extraction processes, handling exceptions, and routing results. This layer maintains workflow state, manages retry logic, and provides audit logging for compliance purposes.
2) **Extraction Layer**: Houses multiple OCR engines configured for different document types and quality levels. The extraction manager dynamically selects appropriate engines based on document characteristics and confidence requirements. Supported engines include Tesseract for general-purpose extraction, PaddleOCR for complex layouts, and specialized engines for specific document formats.
3) **Validation Layer**: Applies multi-stage validation to extracted data, including format checking, cross-field consistency verification, and confidence scoring. Extractions falling below configurable thresholds trigger escalation workflows that may involve alternative OCR engines or human review.
4) **Integration Layer**: Handles data transformation and routing to destination systems including databases, spreadsheets, APIs, and document management platforms. This layer maintains mappings between extracted fields and target schemas, handling data type conversions and format normalization.

3.2 Workflow Design

Figure 2: Extraction Workflow Diagram

The extraction workflow follows a systematic sequence designed to maximize automation while maintaining data quality. Figure 2 presents the workflow as a Unified Modeling Language (UML) activity diagram.

Phase 1: Source Discovery and Navigation
The RPA bot begins by accessing configured web sources through one of several methods: direct URL navigation with login credentials, API-based data retrieval, or scheduled file downloads from monitored folders. For web-based sources, the bot handles authentication challenges, session management, and navigation through multi-page interfaces. The bot maintains a queue of pending extraction tasks with metadata including source type, expected document format, and priority level.

Phase 2: Document Acquisition
Once the source is accessed, the bot acquires documents requiring extraction. This may involve downloading files, capturing screenshots of web content, or extracting embedded images from HTML pages. For each document, the system records source metadata including acquisition timestamp, file size, format, and any available context information that may aid extraction.

Phase 3: Preprocessing and OCR Selection
Acquired documents undergo preprocessing appropriate to their format and quality. Image-based documents receive enhancement including deskewing, contrast adjustment, and noise reduction. PDF documents may require page extraction and format conversion. Based on document characteristics,

the extraction manager selects optimal OCR engines- for example, using PaddleOCR for documents with tables and complex layouts, or Tesseract for straightforward text documents.

Phase 4: Text Extraction
Selected OCR engines process the prepared documents, generating raw text output with confidence scores for each recognized element. For critical documents or those with initially low confidence, the system may employ multi-engine extraction where multiple OCR engines process the same document and results are combined through voting mechanisms. This approach, inspired by LMV-RPA, significantly improves accuracy for challenging documents.

Phase 5: Semantic Parsing and Structuring
Raw OCR output undergoes semantic parsing to extract structured fields according to predefined schemas. This phase may employ regular expressions for predictable patterns (dates, amounts, identifiers), layout-based extraction using template matching, or LLM-based parsing for complex, variable-format documents. The parser generates structured JSON output with field-level confidence scores.

Phase 6: Validation and Quality Assurance
Extracted structured data passes through validation gates that verify:
- Field presence (all required fields extracted)
- Format compliance (dates valid, amounts numeric)
- Cross-field consistency (total matches line items)
- Confidence thresholds (field-level scores exceed minimum)

Documents failing validation trigger escalation workflows. Low-confidence extractions may be reprocessed with alternative OCR engines or enhanced preprocessing. Persistent failures route to human review queues with complete context including original document, extraction history, and validation failures.

Phase 7: Output Integration
Validated data is transformed according to destination requirements and routed to target systems. Common destinations include Excel spreadsheets for business analysis, databases for application integration, and cloud storage platforms for archival. The integration layer maintains audit trails linking extracted data to source documents, supporting traceability and compliance requirements.

Phase 8: File Management and Cleanup
Processed documents are moved to appropriate storage locations with metadata recording processing date, extraction results, and quality metrics. The system maintains folder structures mirroring original organization while separating processed from unprocessed files. Detailed logs support performance analysis and troubleshooting.

3.3 Multi-Engine OCR Strategy

A key innovation in the proposed system is the adaptive multi-engine OCR strategy that dynamically selects and combines recognition engines based on document characteristics. Table 1 summarizes the OCR engines integrated into the system and their respective strengths.

Table 1: OCR Engine Comparison

Engine	Strengths	Optimal Use Cases	Average Confidence
Tesseract	Lightweight, open-source, good with printed text	Simple forms, standard fonts	92%
PaddleOCR	Excellent with complex layouts, table handling	Invoices, multi-column documents	96%
EasyOCR	Multi-language support, easy integration	International documents	91%
DocTR	Document-specific optimization	Structured forms, templates	94%

The selection algorithm considers document type, layout complexity, image quality, and language requirements. For high-value documents, the system may employ parallel extraction where multiple engines process simultaneously and results are merged through a voting mechanism. This approach, inspired by recent research, achieves accuracy improvements of 5-8% compared to single-engine baselines.

3.4 Confidence Scoring and Uncertainty Handling

Reliable confidence estimation is critical for determining when automated extraction can be trusted versus when human review is required. The proposed system implements multi-level confidence scoring:

Character-level confidence: OCR engines provide per-character confidence scores based on recognition probability. These scores aggregate to word and field levels.

Field-level confidence: Combined score incorporating character confidences, format compliance, and contextual consistency. For example, a date field may receive additional confidence if it matches expected patterns and falls within reasonable ranges.

Document-level confidence: Overall confidence score derived from field-level scores weighted by field importance. Critical fields (invoice total, customer ID) receive higher weight in overall assessment.

Configurable thresholds determine automation decisions. Documents with overall confidence above 95% proceed directly to output without review. Documents between 80-95% confidence route to validation queues where automated checks may be supplemented by quick human verification. Documents below 80% confidence require full human review, with system-provided context to accelerate manual processing.

3.5 Human-in-the-Loop Integration

While full automation remains the goal, practical extraction systems must accommodate edge cases where automated processing fails. The proposed architecture includes carefully

designed human-in-the-loop workflows that balance efficiency and quality.

When documents trigger human review, the system presents reviewers with a unified interface showing:
- Original document image or PDF
- Raw OCR output from all attempted engines
- Proposed structured data with confidence indicators
- Validation failures requiring resolution

Reviewers can correct extracted fields, with corrections feeding back to improve future extraction through continuous learning mechanisms. The system tracks review time and correction rates, providing metrics for ongoing optimization.

4. Implementation

4.1 Technology Stack

The proposed system was implemented using a modern technology stack selected for flexibility, performance, and ease of integration:
- **Programming Language**: Python 3.10+ for core extraction logic, providing access to rich OCR and machine learning libraries. JavaScript for web automation components where browser-based interaction is required.
- **OCR Engines**: Tesseract 5.x with LSTM models for baseline extraction; PaddleOCR 2.7 for complex document handling; EasyOCR 1.7 for multi-language support. All engines integrated through unified Python interfaces enabling dynamic selection.
- **RPA Framework**: Custom RPA implementation using Python with Selenium for web automation and PyAutoGUI for desktop interaction. This approach provides greater flexibility than commercial platforms while maintaining necessary automation capabilities.
- **Web Automation**: Selenium WebDriver 4.15 for browser automation, supporting Chrome, Firefox, and Edge. Headless execution for server deployment with screenshot capture for debugging.
- **Data Processing**: Pandas 2.1 for data manipulation and Excel integration; NumPy 1.24 for numerical operations; OpenCV 4.8 for image preprocessing.
- **Validation and Parsing**: Regular expressions for pattern-based extraction; Custom parsers for layout-based extraction; Integration with local LLM models via Ollama for semantic parsing of complex documents.
- **Storage and Logging**: SQLite for local metadata storage; JSON for configuration and extraction results; Python logging module for comprehensive audit trails.
- **API Layer**: FastAPI 0.104 for REST endpoints enabling programmatic access to extraction services; Swagger UI for API documentation and testing.

4.2 OCR Engine Integration

Each OCR engine was integrated through a standardized interface enabling seamless switching and parallel execution. The integration abstracts engine-specific details while exposing common functionality:

```
class OCRAdapter(ABC):
    @abstractmethod
    def extract_text(self, image_path: str, language: str = 'eng') -> Dict:
        """Extract text from image and return with confidence scores"""
        pass

    @abstractmethod
    def get_supported_languages(self) -> List[str]:
        """Return languages supported by this engine"""
        pass

    @abstractmethod
    def get_confidence(self, extraction_result: Dict) -> float:
        """Calculate overall confidence from engine-specific scores"""
        pass
```

Engine-specific adapters implement this interface, handling the unique requirements of each OCR system. For example, the Tesseract adapter manages configuration parameters and output parsing, while the PaddleOCR adapter handles model loading and GPU acceleration options.

4.3 Web Automation Implementation

The web automation component uses Selenium WebDriver with a custom wrapper providing high-level operations common to extraction workflows:
- **Navigation**: URL access with configurable timeouts and retry logic
- **Authentication**: Credential management with secure storage and session handling
- **Element interaction**: Clicking, form filling, and dropdown selection
- **Screenshot capture**: Full-page and element-specific screenshots for OCR processing
- **DOM extraction**: Direct text extraction when available (fallback to OCR when needed)

The automation layer includes robust error handling for common issues including element not found, timeout, and unexpected page changes. Retry policies with exponential backoff handle transient failures while logging provides diagnostic information for persistent problems.

4.4 Validation Framework

The validation framework implements configurable rules that verify extracted data quality:
- **Format validation**: Regular expression patterns for common fields including dates, email addresses, phone numbers, currency amounts, and identifiers. Format validation ensures extracted values match expected patterns before acceptance.
- **Range validation**: Numeric bounds checking for quantities, amounts, and dates. For example, invoice dates must fall within reasonable business periods, and amounts must be positive.
- **Cross-field validation**: Consistency checks across related fields. For invoice processing, line-item totals must sum to invoice total; for forms, postal codes must match country format.

- **Reference validation**: Where applicable, extracted data may be validated against external reference data. For example, customer IDs can be checked against CRM systems, or product codes against catalog databases.

Validation results generate detailed reports enabling root cause analysis of extraction failures. Common failure patterns inform system improvements and training data collection.

4.5 Configuration Management

The system employs YAML-based configuration enabling non-technical users to customize extraction workflows:

```
workflow:
 name: "invoice_extraction"
 source:
   type: "folder"
   path: "./incoming/invoices"
   file_pattern: "*.pdf|*.jpg|*.png"

 extraction:
   primary_engine: "paddleocr"
   fallback_engines: ["tesseract", "easyocr"]
   confidence_threshold: 0.85

 fields:
   - name: "invoice_number"
     pattern: "INV-\d{6}"
     required: true
   - name: "invoice_date"
     pattern: "\d{4}-\d{2}-\d{2}"
     required: true
   - name: "total_amount"
     pattern: "\$\d+\.\d{2}"
     required: true
     validation:
       min: 0
       max: 1000000

 output:
   type: "excel"
   path: "./output/invoices.xlsx"
   sheet: "Extracted Data"
   append: true

 logging:
   level: "INFO"
   audit_trail: true
   retention_days: 90
```

This configuration-driven approach enables rapid deployment for new document types without code changes, significantly reducing implementation time for new use cases.

5. Results and Evaluation

5.1 Experimental Setup

The system was evaluated using a diverse test dataset comprising 1,000 documents across multiple categories:

Document Type	Count	Sources	Characteristics
Invoices	400	Real business documents	Varied layouts, tables, logos
Forms	300	Scanned applications	Handwriting, checkboxes
Reports	200	PDF exports	Multi-column, graphics
Receipts	100	Mobile photos	Low quality, varied angles

Evaluation metrics included extraction accuracy (field-level correct extraction rate), processing time (seconds per document), automation rate (percentage processed without human review), and error rate (incorrect extractions passing validation).

5.2 Accuracy Results

Table 2 presents field-level extraction accuracy across document types for different OCR strategies.

Table 2: Extraction Accuracy by Document Type and Strategy

Document Type	Tesseract Only	PaddleOCR Only	Multi-Engine Voting	Multi-Engine + LLM
Invoices	89.20%	94.70%	96.30%	98.10%
Forms	84.50%	91.20%	93.80%	95.40%
Reports	91.30%	93.60%	95.20%	96.80%
Receipts	76.80%	84.30%	87.90%	91.20%
Overall	86.70%	91.90%	94.10%	96.20%

The results demonstrate consistent accuracy improvements through multi-engine and LLM-enhanced approaches. Multi-engine voting provides 2-4% improvement over best single engine, while LLM-based semantic parsing adds another 1-3% by resolving ambiguities and correcting context errors. Receipts remain the most challenging category due to quality variations and non-standard formats, though even here the combined approach achieves 91% accuracy.

5.3 Processing Time Analysis

Processing time measurements reveal important trade-offs between accuracy and speed:

Table 3: Processing Time by Strategy (seconds per document)

Strategy	Average Time	90th Percentile	Accuracy
Tesseract Only	2.4	3.8	86.70%
Paddle OCR Only	4.7	6.9	91.90%
Sequential Multi- Engine	8.2	12.4	94.10%
Parallel Multi- Engine	5.1	7.8	94.10%
Parallel + LLM	6.3	9.5	96.20%

Parallel execution of multiple OCR engines significantly reduces the time penalty of multi-engine approaches, with parallel processing completing in only 8% more time than PaddleOCR alone while achieving 2.2% accuracy improvement. Adding LLM processing increases time by approximately 24% but delivers the highest overall accuracy.

Batch processing tests with 50 documents demonstrated the system's scalability. While single-document processing times averaged 6.3 seconds, batch throughput reached 12.5

documents per minute (4.8 seconds per document effective) through parallelization and pipeline optimization.

5.4 Automation Rate and Human Review

The confidence-based escalation strategy achieved the following automation rates:

Confidence Threshold	Documents Fully Automated	Documents Requiring Review	Review Accuracy
0.9	73%	27%	99.20%
0.85	81%	19%	98.70%
0.8	86%	14%	97.80%
0.75	91%	9%	96.10%

Selecting an 85% confidence threshold provides optimal balance, automating 81% of documents while maintaining 98.7% accuracy after human review. Higher thresholds increase review burden with diminishing quality returns; lower thresholds risk automation of questionable extractions.

Average human review time was 45 seconds per document, significantly faster than full manual processing requiring 3-5 minutes per document. The combined automated-plus-review approach achieved 87% time savings compared to fully manual processing.

5.5 Real-World Case Study: Invoice Processing

A real-world deployment processed 2,400 invoices over six months for a mid-sized organization. Key results included:

- **Time savings**: 340 hours of manual data entry eliminated (85% reduction)
- **Accuracy improvement**: Error rate decreased from 4.2% (manual) to 1.1% (automated + review)
- **Processing speed**: Average 45 seconds per invoice versus 4 minutes manual
- **Cost reduction**: 72% decrease in processing cost per invoice
- **Scalability**: Successfully handled peak volumes of 200 invoices/day without backlog

These results align with findings from financial institutions reporting 65-80% efficiency gains from integrated OCR-RPA implementations.

5.6 Error Analysis

Analysis of extraction failures revealed common patterns:

Error Type	Frequency	Primary Causes
OCR character errors	42%	Poor image quality, small fonts, unusual typefaces
Field misidentification	28%	Ambiguous layouts, missing labels, complex tables
Format parsing errors	18%	Non-standard date formats, currency variations
Missing fields	12%	Fields outside capture area, document variations

These findings guide ongoing improvements including enhanced preprocessing for poor-quality images, expanded training data for layout variations, and more flexible format parsers.

6. Conclusion and Future Work

6.1 Summary of Contributions

This research presented an integrated framework for automated web data extraction combining Optical Character Recognition and Robotic Process Automation. The key contributions include:

1) **Modular architecture** enabling flexible integration of multiple OCR engines and RPA components, supporting both cloud-based and on-premise deployment scenarios.
2) **Adaptive multi-engine strategy** that dynamically selects and combines OCR engines based on document characteristics, achieving 96.2% overall accuracy—a 4.3% improvement over single-engine baselines.
3) **Confidence-based escalation workflow** that automatically routes uncertain extractions to appropriate fallback mechanisms including alternative engines and human review, maintaining high quality while maximizing automation.
4) **Comprehensive validation framework** implementing multi-level quality checks that detect extraction errors before they propagate to downstream systems.
5) **Practical deployment insights** from real-world implementations demonstrating 85% time savings and 72% cost reduction for document processing workflows.

The system successfully addresses the challenge of extracting structured data from unstructured web sources, offering organizations a practical path to automation for document-intensive processes.

6.2 Limitations

Several limitations warrant acknowledgement:

- **Document dependence**: Performance varies significantly with document quality and format, with receipts and handwritten forms remaining challenging
- **Initial setup effort**: Configuration for new document types requires domain expertise and testing
- **Infrastructure requirements**: Multi-engine parallel processing demands adequate computational resources
- **Language coverage**: Current implementation focuses on English documents, with limited multi-language support

6.3 Future Research Directions

Several avenues for future work emerge from this research:

- **Enhanced LLM Integration**: Recent advances in vision-language models suggest potential for end-to-end document understanding that bypasses traditional OCR pipelines. Exploring integration of models like GPT-4V and Claude 3 for direct document-to-structured-data extraction could simplify architectures while potentially improving accuracy.
- **Continuous Learning Mechanisms**: Implementing feedback loops where human corrections improve future extractions would accelerate accuracy improvements and reduce review burden over time. Techniques from active learning and few-shot adaptation show promise for this application.
- **Real-Time Extraction Optimization**: For time-sensitive applications such as automated form submission or real-

time data validation, optimizing extraction pipelines for minimal latency while maintaining acceptable accuracy deserves investigation.

- **Expanded Document Types**: Extending the framework to handle additional document types including handwritten medical forms, technical drawings with embedded text, and multilingual documents would broaden applicability.
- **Blockchain-Based Audit Trails**: For regulated industries, implementing immutable audit trails of extraction processes using blockchain technology could enhance compliance and trust in automated systems.

6.4 Practical Implications

For organizations considering OCR-RPA automation, this research offers practical guidance:

- Start with high-volume, standardized documents where automation delivers quick wins
- Implement confidence-based escalation rather than pursuing perfect automation
- Plan for human review capacity even with highly accurate systems
- Invest in document quality improvements where possible
- Measure and monitor extraction quality continuously

The convergence of OCR and RPA technologies, enhanced by recent advances in large language models, creates unprecedented opportunities for automating document-intensive workflows. As these technologies continue to mature, the vision of fully automated document processing- where documents flow seamlessly from receipt to structured data without human intervention- moves closer to reality.

References

[1] A.K., I. Sut. and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," Advances in Neural Information Processing Systems 25, pp. 1097-1105.
[2] LMV-RPA: Large Model Voting-based Robotic Process Automation, arXiv:2412.17965, December 2024.
[3] Box Extract Architecture: Beyond "Just Wrappers," Box Blog, January 2026.
[4] Keiyo Bank Journey: How WinActor & DX Suite Transformed Paperwork into Digital Success, WinActor Support, August 2025.
[5] Financial Data Extraction Pipeline, GitHub Repository, April 2025.
[6] Mixing multiple AI models for OCR and element detection in headless workflows, Latenode Community, October 2025.
[7] LMRPA: Large Language Model-Driven Efficient Robotic Process Automation for OCR, arXiv:2412.18063, December 2024.
[8] Extract Data Activity, Infor Documentation, March 2025.