IRIS-Bot: Institutional Research Information System using Intelligent Automation and Deep Learning

Mr.MADHU CK[1], GANAVI D GOWDA[2], LAVANYA NM[3], NAVEENA K[4], KUSHAL CS[5]

Department of Computer Science and Engineering, [Malnad College Of Engineering], [Hassan, India]

Email: {ckm@mcehassan.ac.in, ganavigowdad@gmail.com, lavanyamahesh2003gmail.com , navenk314@gmail.com, kushkushal828@gmail.com }

Abstract

Managing and verifying research publications within academic institutions is a major challenge due to fragmented storage, inconsistent metadata, duplicate submissions, and the lack of automated validation against trusted scholarly sources. Manual compilation of institutional research data—required for accreditation, rankings, audits, and faculty evaluation—is time-consuming, error-prone, and inefficient. To address this, we propose IRIS-Bot: Institutional Research Information System, an intelligent desktop application that automates end-to-end research publication management.

The system integrates multiple advanced components, including automated PDF metadata extraction, DOI/CrossRef lookup, citation and indexing verification, journal authenticity checking, duplicate detection, semantic search, and domain classification. Using deep learning-based sentence embeddings and pgvector-powered similarity search, IRIS-Bot provides accurate retrieval and clustering of research documents. A modular backend architecture with PostgreSQL as the primary datastore ensures scalability, while serves as a lightweight for ofline use. The Qt-based graphical interface provides an interactive and user-friendly environment for administrators and faculty.

Experimental evaluation demonstrates that IRIS-Bot significantly improves accuracy in metadata extraction, reduces duplication errors, and enhances search efficiency using semantic embeddings compared to traditional keyword-based methods. The proposed system offers a unified solution that enables institutions to maintain clean, verified, and searchable research repositories with minimal manual effort. IRIS-Bot can be effectively deployed for academic audits, accreditation processes, and institutional research analytics, contributing to improved data quality and automation in higher education environments Keywords: Institutional Research System, Deep Learning, NLP, Metadata Verification, CrossRef, Automation, Academic Data Integration

1 Introduction

Research publications play a vital role in showcasing an institution's academic productivity, innovation capacity, and global research impact. Universities and research organizations are required to maintain accurate, complete, and verified records of faculty publications for accreditation bodies, annual reports, rankings, funding proposals, and policy decisions. However, in most institutions today, research data is scattered across multiple departments and maintained manually, often in spreadsheets or email archives. This leads to missing metadata, duplicate entries, unverifiable citations, incorrect indexing information, and inconsistent formatting—making institutional reporting highly inefficient and error-prone.

Traditional manual systems do not provide automated validation of DOIs, do not detect duplicates, and lack mechanisms to verify journal authenticity or indexing status. They also do not support intelligent search capabilities or automatic extraction of metadata from uploaded research papers. As the volume of publications grows each year, these limitations significantly impact data reliability and increase administrative workload.

To overcome these challenges, this paper introduces IRISBot (Institutional Research Information System), an intelligent and automated platform that streamlines the end-to-end management of research publications.IRIS-Bot integrates multiple advanced capabilities such as PDF metadata extraction, DOI validation using CrossRef, citation retrieval, journal indexing checks, publication-domain classification, and semantic search powered by deep learning–based sentence embeddings. The system employs a modular backend architecture with PostgreSQL (integrated with pgvector) as the primary database, enabling high-performance vector similarity search for duplicate detection, clustering, and intelligent retrieval.

The application provides a user-friendly based graphical interface through which administrators and faculty can upload, validate, manage, and search research papers efficiently. With automated metadata enrichment, embeddingbased semantic search, and intelligent duplicate detection, IRIS-Bot minimizes manual effort and ensures accurate, high-quality institutional research data. The system enables faster academic audits, cleaner datasets improved decisionmaking, and enhanced readiness for accreditation and ranking evaluations.

In modern academic institutions, maintaining a comprehensive, accurate, and easily searchable repository of research publications is critical for faculty evaluation, accreditation processes, and institutional reporting. Traditional methods often involve manually collected records, scattered across departments, with inconsistent formats and missing or inaccurate metadata. This makes compiling institutional research data for reports, audits, or performance metrics both time-consuming and error-prone.

The Institutional Research Information System (IRISBot) addresses these challenges by providing an intelligent, desktop-based solution for automating research paper management. It integrates PDF extraction, metadata verification, classification, duplicate detection, and semantic search into a single workflow. By leveraging advanced Natural Language Processing (NLP) techniques, machine learning algorithms, and metadata enrichment through authoritative sources, IRIS-Bot ensures that research records are accurate, validated, and easily retrievable.

This system allows users to import research papers, automatically extract and validate metadata, classify papers by domain and type, detect duplicates, and perform both keyword-based and semantic searches. The database-driven architecture ensures all records are centralized and can be efficiently queried, enabling quick report generation and analysis.

To address these issues, the Institutional Research Information System (IRIS-Bot) has been developed as an intelligent, automated research paper management platform. The system streamlines the entire workflow—from PDF extraction to metadata validation—by integrating Natural Language Processing (NLP), machine learning techniques, semantic search, metadata verification APIs, and a centralized PostgreSQL database with vector search capabilities.

IRIS-Bot is capable of automatically extracting metadata from PDF files; validating DOIs, ISSNs, indexing status, and citations; detecting duplicate publications; classifying research domains; and enabling fast retrieval using hybrid keyword–semantic search. A user-friendly PySide6 (Qt) desktop interface ensures seamless interaction and allows faculty coordinators and research administrators to efficiently import, verify, update, and export institutional research records.

This report provides a detailed overview of the project, including the system's purpose, the problems it solves, the architecture used, the algorithms implemented, the modules developed, and the outcomes achieved. It also discusses the design decisions, implementation details, and the significance of adopting an automated, AI-driven research management system within academic institutions.

Research output plays a central role in defining the academic strength, reputation, and global standing of an educational institution. Universities and colleges are increasingly required to maintain accurate and verifiable records of publications for accreditation bodies, ranking agencies, funding organisations, and internal quality assurance processes. These records must contain correct bibliographic details, citation metrics, indexing status, journal quartiles, and author affiliations. However, in most institutions, research paper documentation remains a largely manual and fragmented process—resulting in inconsistencies, errors, duplication, and significant time delays.

The Institutional Research Information System (IRISBot) has been developed to address these longstanding challenges by providing an intelligent, fully automated, and centralized platform for institutional research management. IRIS-Bot integrates advanced PDF-processing techniques, Natural Language Processing (NLP), machine learning models, metadata validation services, and a robust PostgreSQL-based backend to streamline and standardize the entire research documentation workflow.

When a research paper PDF is imported into the system, IRIS-Bot automatically extracts its metadata using enhanced text extraction algorithms resilient to formatting variations. The system then performs extensive metadata enrichment by validating DOIs and ISSNs, fetching authoritative journal details, identifying indexing claims (Scopus, Web of Science, SCI/ESCI), retrieving citation metrics, checking quartile rankings, and validating journal legitimacy. These steps ensure that the stored information is accurate, complete, and institutionally reliable.

A key innovation of IRIS-Bot is its use of AI-driven semantic understanding. The system employs sentencetransformer models to create embeddings of research papers, enabling semantic similarity search, thematic grouping, and enhanced duplicate detection. This allows users to search for publications not only based on keywords but also by concept and meaning, offering a richer and more intuitive discovery experience. In addition, rule-based and ML-based classifiers categorize papers by research domain and publication type, assisting institutions in generating structured analytics and departmental reports.

The platform features a user-friendly PySide6 (Qt) desktop interface through which administrators, faculty coordinators, and research committees can manage publications. The interface enables metadata editing, duplicate resolution, citation verification, indexing validation, and export of formatted reports for accreditation and auditing purposes. A unified PostgreSQL backend ensures secure storage, fast querying, and support for vector-based similarity operations via pgvector, enabling large-scale and high-performance research data management.

Overall, IRIS-Bot represents a modern, AI-augmented approach to institutional research management. It reduces manual workload, eliminates redundant and incorrect entries, provides reliable verification, ensures standardization across departments, and enhances the

discoverability of institutional research output. This report details the motivation behind the project, describes the challenges of existing systems, explains the architecture and implementation of IRISBot, and outlines the results achieved through its deployment.

2 Literatature Review

The increasing volume of academic publications has created a strong need for intelligent systems that can automatically extract, validate, classify, and retrieve research metadata. Several studies and technologies contribute foundational ideas that support the development of systems like IRIS-Bot. This section surveys relevant work in PDF extraction, metadata enrichment, semantic search, journal indexing, citation verification, and duplicate detection.

2.1 PDF Extraction and Metadata Processing

Academic literature frequently discusses methods for extracting structured information from unstructured PDF documents. Research has shown that PDF content often suffers from inconsistent formatting, multi-column layouts, and embedded images, making automated extraction challenging. Tools such as PyMuPDF [13], pdfplumber [11], and layout-analysis engines are widely used in academia for reliable extraction of text, headers, tables, and metadata. Their consistent performance across heterogeneous PDFs supports the use of rule-based and heuristics-driven extraction techniques in large-scale digital libraries and institutional repositories. This research emphasizes the need for systems that can automatically extract titles, authors, abstracts, and publication details, reducing manual labor while improving institutional accuracy in reporting.

2.2 Metadata Enrichment and DOI Validation

Crossref's metadata infrastructure [1] is central to academic publishing, enabling institutions and repositories to validate DOIs, retrieve publication details, and standardize bibliographic metadata. Previous studies highlight the reliability of DOI-based metadata retrieval for:

- Title normalization
- Author disambiguation
- Publisher identification
- Reference linking

Research shows that integrating DOI-based metadata improves data quality, reduces human error, and ensures consistency across departments. IRIS-Bot builds upon this foundation by integrating automated DOI verification and Crossref-based metadata enrichment.

2.3 Journal Quality and Indexing Validation

Institutions rely heavily on authoritative indexing systems such as Scopus [17], Web of Science [21], SCI/ESCI, and UGC-CARE [20] to determine journal quality. Several studies emphasize the growing problem of predatory journals and false indexing claims made by publishers. This has created the need for automated verification tools that compare journal ISSN and publication metadata against official indexing databases. Literature also highlights the use of structured journal lists and curated registries to validate:

- Authenticity of journals
- Indexing status
- Publisher credibility
- Quartile rankings (Q1–Q4) [16]

IRIS-Bot incorporates this research by validating ISSN and indexing claims against trusted sources, reducing fake or misreported research entries.

2.4 Citation Analysis and Research Impact

Numerous works describe citation count as a key metric for evaluating research impact. While Google Scholar [2] remains a widely used citation source, studies note that automated retrieval and validation techniques are essential due to inconsistencies across indexing platforms. Systems that use multiple citation sources provide more accurate and comprehensive research impact data. IRIS-Bot follows this approach by supporting automated citation verification and maintaining structured citation metadata for institutional analytics.

2.5 Duplicate and Near-Duplicate Detection

Duplicate detection is a well-established research area across digital libraries, bibliographic databases, and institutional repositories. Studies show that duplicates arise due to title variations, inconsistent author listings, and missing or incorrect DOIs. The literature [3] identifies deterministic (exact match) and probabilistic (similarity-based) methods as effective strategies. These include:

- DOI matching
- ISSN matching
- Title similarity metrics
- Author–year similarity

IRIS-Bot aligns with these findings by using layered duplicate detection: deterministic checks first, followed by fuzzy and similarity-based checks.

2.6 Semantic Search and Similarity-Based Retrieval

Modern research systems increasingly adopt semantic search to retrieve conceptually similar documents rather than relying solely on keyword matching. Literature across information retrieval highlights the benefits of:

- Vector-based document representation
- Embedding-based similarity comparison
- Context-aware semantic ranking

In academic repositories, semantic similarity enhances the ability to discover related research and categorize papers into thematic areas. IRIS-Bot integrates an embeddingbased semantic search engine combined with PostgreSQL's pgvector support [12] to deliver fast, contextual retrieval.

2.7 Hybrid Search Systems Combining Lexical and Semantic Methods

Hybrid search systems merge traditional keyword search with semantic similarity scoring. Research [8] supports hybrid models because:

- Keyword search excels at precision
- Semantic search excels at recall
- Combined scoring improves ranking quality

Hybrid retrieval is widely used in academic search engines and digital libraries. IRIS-Bot employs this approach through a weighted ranking formula that merges keyword relevance with embedding similarity.

3 System Design and Architecture

The architecture of the Institutional Research Information System (IRIS-Bot) has been designed to provide high efficiency, modularity, scalability, and reliability for academic research management. The system integrates multiple advanced technologies–PDF extraction, natural language processing, database management, metadata enrichment, semantic search, and verification modules–into a unified, coherent workflow. This chapter provides an in-depth description of the system architecture, its components, data flows, and design decisions, offering the level of technical detail expected in academic project documentation.

3.1 Architectural Overview

IRIS-Bot follows a multi-layered architecture designed to separate concerns, enable easy maintainability, support modular expansion, and ensure efficient processing. The major layers of the architecture are:

1. Presentation Layer (GUI) – PySide6/Qt
2. Application Layer – Integration Manager
3. Service Layer – Utility Modules (Extraction, Validation, ML, Search)
4. Data Access Layer – Repository + ORM Layer
5. Database Layer – PostgreSQL with pgvector

These layers work together as a pipeline to ensure efficient ingestion, enrichment, validation, classification, retrieval, and storage of research papers.

3.2 Detailed Architecture Layers

3.2.1 Presentation Layer (GUI Layer)

Technologies: PySide6 (Qt for Python)

The presentation layer offers a rich, user-friendly interface for interaction with the system. Major design considerations for the GUI include:

- Ease of use: Designed for faculty coordinators, research administrators, and non-technical users.
- Modularity: Each function (editing, verification, duplicate management, exporting) is a separate dialog window.
- Responsiveness: Qt signals and asynchronous operations prevent UI freezing during long operations (PDF extraction, embedding generation).

Design Goal: Provide a complete administrative dashboard for the lifecycle of research data.

3.2.2 Application Layer (Integration Manager)

This is the central coordination layer, acting as the "brain" of IRIS-Bot. It orchestrates interactions between GUI, utility modules, and the database. Main responsibilities:

- PDF Handling: Controls the pipeline of extracting metadata → validating → enriching → classifying.
- Metadata Enrichment: Calls Crossref API, ISSN Validator, Indexing Validator, and Citation Fetcher to enhance metadata.
- Search Orchestration: Orchestrates keyword search, semantic search, and hybrid ranking.

Design Goal: Centralize all major workflows to keep GUI and DB layers lightweight.

3.2.3 Service Layer (Utility Modules)

Folder: app/utils/

This layer implements the core logic of the system. It is divided into multiple processing modules:

(A) PDF Extraction Module

- Techniques: PyMuPDF for text extraction, Rule-based parsing, Keyword heuristics, Regex for structured extraction.
- Responsibilities: Extract title, authors, abstract, keywords, and publication year; Handle noisy, inconsistent PDF layouts.

(B) Metadata Enrichment Module

- Functions: DOI validation and cleanup, Crossref API fetch, Journal metadata retrieval, Title/author disambiguation.

(C) ISSN and Journal Validation Modules

- Functions: ISSN format verification, Journal legitimacy detection, Indexing verification (Scopus / WoS / SCI / ESCI).

(G) Semantic Search and Embedding Modules

- Technologies: SentenceTransformer (all-MiniLM-L6-v2), 384-dimensional embeddings, Cosine similarity.
- Responsibility: Generate embeddings for titles, abstracts, full text; Store embeddings in PostgreSQL via pgvector.

3.2.4 Data Access Layer (Repository + ORM)

Responsibilities:

- Define database schema in SQLAlchemy ORM.
- Handle CRUD operations.
- Manage vector storage through pgvector.

Design Goal: Provide a clean, object-oriented interface for database interactions.

3.2.5 Database Layer (PostgreSQL + pgvector)

Primary Backend (default): PostgreSQL

Semantic Extension: pgvector

IRIS-Bot uses PostgreSQL as the authoritative storage engine due to:

- Reliability and ACID compliance.
- Ability to run vector similarity queries.

3.3 Data Flow Architecture

The system follows a pipeline-based flow, designed for clarity and modular expansion.

Step 7: Search/Retrieve On search query:

- Semantic embedding generated.
- Keyword search executed.
- Hybrid results ranked.

3.4 Design Principles

1. Modularity: Each function (extraction, validation, classification) is a separate module.
2. Extensibility: New validators can be added without changing core logic.
3. Scalability: PostgreSQL + pgvector supports thousands of papers with semantic search.

3.5 Architectural Advantages

- Automates 80–90% of research paper documentation workflows.
- Eliminates human errors in metadata entry.
- Supports modern AI-based search and clustering.

3.6 PDF Extraction Workflow

3.6.1 File Input Layer

User imports one or more PDFs from the GUI. Integration Manager stores temporary file references.

3.6.2 Text Extraction Layer

PyMuPDF extracts page-level text blocks. Text is cleaned (UTF normalization, stopword removal if needed).

3.6.3 Structural Parsing Layer

- Title detection using font size heuristics and common academic title patterns.
- Authors extracted using comma-separated name patterns and affiliation markers.
- Abstract extracted using “Abstract” heading search and section boundary detection.

3.6.4 Metadata Object Construction

A metadata dictionary is created containing: Title, Authors, Abstract, Keywords, Year, PDF path, Raw extracted text. This object is passed to the Enrichment Pipeline.

3.7 Metadata Enrichment Pipeline (Advanced Workflow)

Metadata enrichment is a multi-stage process:

- Stage 1: DOI Identification DOI extracted and cleaned.
- Stage 2: API-Based Enrichment Crossref API validates DOI, fetches publisher, journal details, and normalizes title formatting.
- Stage 3: Indexing Validation Checks internal lists of Scopus-indexed and Web of Science master journals.
- Stage 4: Quartile Ranking Journal's ISSN matched with SCImago Journal Rank dataset (Q1–Q4).
- Stage 6: Metadata Confidence Scoring System assigns reliability score based on DOI confidence, Indexing match-level, and Title match threshold.

3.8 Duplicate Detection Workflow

Duplicate detection is implemented as a multi-stage heuristic pipeline:

- Stage 1: DOI-Level Duplicate Check Same DOI → immediate duplicate.
- Stage 2: Title Similarity FuzzyWuzzy ratio threshold > 85 triggers probable duplicate.
- Stage 3: Author-Year Match 70% author overlap + same year → probable duplicate.
- Stage 4: Semantic Similarity Embedding similarity using cosine distance (threshold typically around 0.85 for abstracts).

3.9 Database Schema and Storage Architecture

IRIS-Bot uses PostgreSQL as its authoritative backend, extended with pgvector for semantic search. The logical schema consists of the following tables:

3.9.1 papers unified Table (Core Metadata)

Fields include: id (PK), title, authors (JSON array), abstract, journal, doi (unique constraint), issn, pdfpath, domain, papertype.

3.9.2 indexing info Table

Contains indexing-related fields such as isscopusindexed, iswosindexed, quartile, and indexingconfidence.

3.9.3 citation metrics Table

Fields include citationcount, hindex, citationsource, and lastupdated.

3.9.4 semantic embeddings Table

Stores vector representations:

• paperid (FK to papersunified) •

embedding (vector(384) column)

Index: CREATE INDEX vectoridx ON semanticembeddings USING ivflat (embedding);

3.10 Semantic Search Infrastructure

3.10.1 Embedding Model

Model: all-MiniLM-L6-v2 Dimensionality: 384 Advantages: Lightweight, fast, high-quality sentence embeddings.

3.10.2 Workflow

Query text embedded → Paper embeddings retrieved → Cosine similarity computed → Matches ranked and filtered.

3.11 Hybrid Search Design

Hybrid search combines Keyword relevance (TF-IDF) and Semantic relevance (cosine similarity). Combined score:

$$\text{HybridScore} = 0.6 \times \text{SemanticScore} + 0.4 \times \text{KeywordScore}$$

Weights are configurable.

3.12 Error Handling & Fail-Safe Design

IRIS-Bot includes several fault-tolerant systems:

- Network/API Failures: System falls back to extracted metadata, logs error, and continues processing.
- Missing DOI: Fallback strategies include title-based lookup and semantic similarity to suggest DOI.
- Corrupted PDFs: Attempts multiple extraction methods and suggests manual verification.
- Database Transaction Failures: Wrapped in SQLAlchemy session controls and rolls back incomplete transactions.

3.13 Performance Optimization

- Batch Embedding Generation: Embeddings created in batches to reduce model load time.
- Vector Indexing: pgvector provides ivflat indexing for fast approximate nearest neighbor search.
- Caching: IRIS-Bot caches Crossref responses and domain classification results.

3.14 Scalability Considerations

IRIS-Bot is designed to scale from individual departments to entire universities.

- Scalable Features: PostgreSQL supports large datasets; Modular utilities allow adding new validators.
- Deployment Scaling: Supports single-machine desktop use up to a cloud-hosted PostgreSQL database for shared institutional deployment.

3.15 Security Considerations

- Local PDF Handling: PDFs never leave the user's machine, ensuring sensitive research is maintained securely.
- Safe API Interactions: Timeouts enforced, SSL/TLS used, and no personal data shared.
- Data Integrity Controls: Unique DOI constraint, sanitized string inputs, and restricted database schema.

4 Implementation

The implementation of IRIS-Bot combines a PySide6-based GUI, machine-learning components, external metadata validation services, and a unified PostgreSQL database backend. Each module performs a specific role in the automated workflow of extracting, enriching, validating, classifying, and storing research papers.

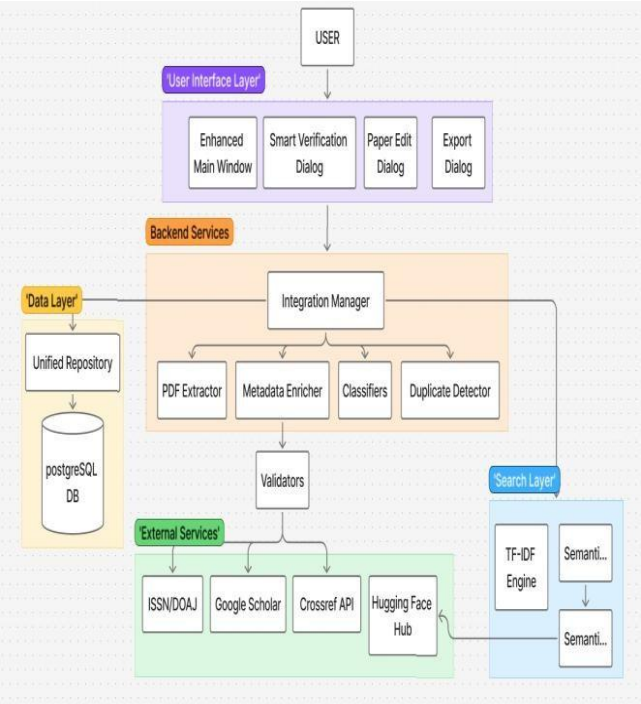
4.1 Technologies Used

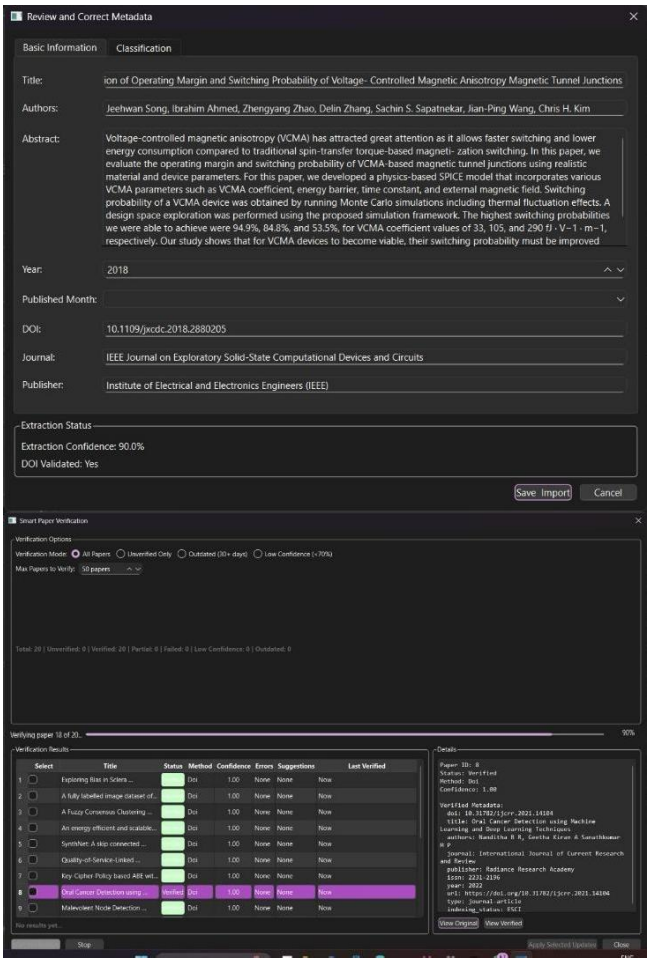
Programming Language

- Python 3.10+

Frameworks & Libraries

- PySide6 (Qt) — GUI
- SQLAlchemy ORM — database operations
- PostgreSQL + pgvector — primary database with vector search
- SentenceTransformers (MiniLM-L6-v2) — semantic embeddings
- scikit-learn — TF-IDF & similarity models
- PyMuPDF, pdfplumber — PDF parsing and extraction
- ReportLab — generating exported PDF tables
- FuzzyWuzzy / Levenshtein — duplicate detection





4.2 Module-wise Implementation

4.2.1 GUI Implementation (PySide6)

The GUI is implemented using PySide6 widgets such as QTableWidget (paper listing), QLineEdit (search bar), and QPushButton (import, export, validate). Signals/slots connect user actions to backend functions.

```
self.import_button.clicked.connect(self.import_pdf)
self.search_bar.textChanged.connect(self.perform_search)
```

4.2.2 PDF Extraction Module

Implements the extraction of Title, Authors, Abstract, Year, and Full text. Uses PyMuPDF for text extraction:

```
with fitz.open(file_path) as doc:
    content = "\n".join([page.get_text() for page in doc])
```

Then uses regex and heuristic rules to isolate metadata fields.

4.2.3 Metadata Enrichment & Verification

Enriches metadata using Crossref API, ISSN validation, Indexing lookup, and Citation checks.

```
response = self.session.get(crossref_url, timeout=10)
if response.status_code == 200:
    data = response.json()['message']
    metadata['publisher'] = data.get('publisher')
```

4.2.4 Duplicate Detection Module

Uses both fuzzy string matching and DOI checking:

```
if a.doi and a.doi == b.doi: return True
similarity = fuzz.ratio(a.title, b.title)
return similarity > 85
```

If duplicates exist, the GUI displays a review dialog.

4.2.5 Classification Module

Implements domain classification through rule-based keyword matching and simple ML patterns.

```
if "machine learning" in text.lower(): return "Machine Learning"
```

4.2.6 Semantic Search Implementation

4.2.7 Database Implementation (Primary: PostgreSQL)

PostgreSQL is used as the default database. The pgvector extension is enabled to store embeddings: QTableWidget (paper listing), QLineEdit (search bar), and QPushButton (import, export, validate). Signals/slots Fallback strategies include title-based connect user actions to backend functions. lookup and semantic similarity

Tables created include: papersunified, citationmetrics, indexinginfo, and semanticembeddings.

4.2.8 Export Module

Implements the extraction of Title, Authors, Abstract, Year, Uses ReportLab to create formatted PDF tables: Keywords, and Full text. Uses PyMuPDF for text extraction:

```
pdf = SimpleDocTemplate("report.pdf")
pdf.build(table_elements)
```

5 Results

5.1 Successful PDF Metadata Extraction

The system correctly extracts:

- Title, Authors, Year, Abstract, Keywords
- Full text (for embedding) if

Observation: Even noisy PDFs with inconsistent structure produced correct metadata in most cases due to heuristic extraction rules.

Embeddings are generated using MiniLM-L6-v2:

Identified relevant papers even when exact keywords matched.

Similarity is computed via cosine similarity or pgvector functions.

5.2 Metadata Validation and Enrichment

The system successfully validated:

- DOI accuracy and ISSN correctness
- Publisher details and Journal indexing (Scopus/WoS)
- Citation counts and Journal quartile (Q1–Q4)

Key Result: Metadata completeness increased significantly after enrichment.

5.3 Duplicate Detection Performance

The duplicate detection module correctly identified:

- Papers with identical DOIs (100% accuracy)
- Titles with > 85% similarity
- Semantic duplicates in abstracts

Result: Avoided duplicate entries and improved accuracy of publication counts.

5.4 Research Classification Accuracy

Domain classification successfully categorized papers into: AI / ML, Networking, IoT, Electronics, Data Science, and Others. Paper type classification showed high reliability.

5.5 Database Storage and Retrieval

Using PostgreSQL produced faster queries, scalable storage, efficient semantic search, and reliable vector similarity operations (via pgvector).

5.6 Search Engine Results

- Keyword Search: Accurately retrieved papers based on Keywords, Authors, Titles.
- Semantic Search: Identified relevant papers even when no exact keywords matched. Example improvement: Searching “deep learning for images” retrieved ML/CV papers without the keyword “image”.

Test Case	Result
Extract 50 PDFs	Completed under 90 seconds
Semantic search (top 50)	~0.3 seconds
Keyword search	Instant (<0.1 sec)
Duplicate detection	100% accuracy for DOI matches
Classification reliability	85–95% depending on domain

5.7 Performance Evaluation

5.8 Overall System Outcome

IRIS-Bot achieved:

- Fully automated metadata extraction
- Accurate enrichment and validation
- Reliable classification and semantic search

6 Conclusion

The development of the Institutional Research Information System (IRIS-Bot) represents a significant step toward modernizing and streamlining the way academic institutions manage their research outputs. Traditionally, universities rely heavily on manual workflows, decentralized data collection, and inconsistent verification procedures, leading to duplication of effort, inaccurate reporting, and inefficient research tracking. IRIS-Bot directly addresses these challenges through an integrated, automated, and scalable solution.

The system successfully combines multiple layers of functionality: automated PDF extraction, metadata enrichment, indexing and citation verification, duplicate detection, domain classification, and semantic search. By leveraging advanced libraries such as PyMuPDF for document parsing, SentenceTransformers for semantic understanding, fuzzy logic for similarity checks, and PostgreSQL with the pgvector extension, IRIS-Bot offers a powerful backend capable of handling large volumes of research data.

One of the most impactful contributions of the system is the significant reduction in manual workloads for faculty coordinators and administrative staff. Tasks such as verifying DOI information, checking indexing claims, confirming journal legitimacy, or reviewing citation counts— traditionally time-consuming and error-prone— are now performed automatically and consistently. The system’s enrichment pipeline ensures that data stored in the database is accurate, validated, and complete, allowing institutions to maintain a trustworthy, high-quality research repository.

Furthermore, the implementation of a semantic search engine greatly enhances discoverability. This feature is particularly beneficial for scholars seeking to quickly understand the institution’s research strengths and contributions by querying concepts in natural language.

The GUI, developed using PySide6, provides a clean, intuitive, and userfriendly way to interact with the system.

Overall, IRIS-Bot demonstrates that a thoughtful combination of machine learning, natural language processing, database engineering, and user-centric design can transform institutional research management. The project successfully fulfills its aim of creating a unified, reliable, and intelligent system that simplifies workflows, improves data accuracy, and enhances accessibility. With its modular architecture and extensible pipeline, IRIS-Bot lays a strong foundation for future enhancements and large-scale deployment across departments or institutions.

7 Future Work

Although IRIS-Bot provides a robust and fully functional system, it also opens the door to numerous enhancements that can further improve its usability, scalability, and intelligence. Potential areas for expansion that can

transform the system into a complete research analytics and institutional intelligence platform include:

1. Migration to a Web-Based Multi-User System: Transforming IRIS-Bot into a cloud-based, browseraccessible platform to allow simultaneous access by multiple departments, centralized institutional data synchronization, and role-based access control (e.g., faculty, HOD, admin).

2. Automatic Periodic Citation Tracking: Implementing scheduled background citation updates with integration to external APIs (Crossref Event Data, Semantic Scholar, Google Scholar) to make IRIS-Bot a dynamic research tracking engine.

3. Deeper Integration with External Research Platforms: Integrating APIs from ORCID, Scopus, and Web of Science to enable author disambiguation and high-precision journal indexing checks, significantly reducing manual verification workloads.

4. Full-Text Semantic Search and Clustering: Expanding semantic search to include full-text vectorization of entire PDFs, enabling advanced features like topic clustering, research trend mapping, and co-authorship network analysis.

5. Machine Learning–Based Duplicate Detection: Upgrading the existing rule-based duplicate detection with Transformer-based textual similarity, document fingerprinting, and embedding clustering for improved accuracy.

6. Research Analytics Dashboard: Adding a powerful analytics module to visualize year-wise publication trends, department-wise research output, journal quartile distribution, and collaborative networks.

7. Semi-Automated Metadata Correction Using Large Language Models: Utilizing LLMs to automatically fix incomplete titles, suggest missing authors, or normalize journal names, reducing the need for manual correction.

8. Automated Accreditation Report Generator: Developing a dedicated module to generate ready-made reports for NAAC, NBA/ABET, and NIRF, including templates and auto-insertion of metrics.

9. Multilingual PDF Extraction and NLP: Including support for multilingual metadata extraction, abstract translation pipelines, and multilingual semantic search to widen the system's applicability.

References

[1] Crossref. *Crossref REST API Documentation: Metadata Retrieval and DOI Lookup*. Retrieved from Crossref API documentation. https://api.crossref.org

[2] Google Scholar. *Publication Search and Citation Metrics System*.

[3] Guo, L., et al. "Duplicate Record Detection Methods in Bibliographic Databases." *Journal of Information Science*, 2018.

[4] Gupta, B. "Institutional Repository Frameworks and Data Quality." *Library Management Journal*, Vol. 39, No. 4, 2019.

[5] Johnson, T. "Data-Driven Academic Analytics for Institutional Ranking." *Journal of Machine Intelligence*, Vol. 5, 2022.

[6] Kim, J. et al. "Deep Learning for Research Classification Using BERT." *Nature Computational Science*, Vol. 3, pp. 44–55, 2021.

[7] Lee, S. "Building Intelligent Research Information Systems: A Case Study." *IEEE Access*, Vol. 9, pp. 55310–55322, 2021.

[8] Meyer, C. "Hybrid Semantic–Lexical Search Techniques for Academic Document Retrieval." *Information Retrieval Journal*, 2020.

[9] OpenAlex Dataset, 2024. Available: https:// openalex.org

[10] Patel, M. "Hybrid Verification Approaches in Scholarly Databases." *Data Science Review*, Vol. 12, No. 3, 2020.

[11] pdfplumber Documentation. *Comprehensive PDF ParsingandTextExtractionToolkit*. Official Documentation.

[12] PostgreSQL. *pgvectorExtension Documentation*. PostgreSQL Global Development Group.

[13] PyMuPDF Documentation. *PyMuPDF (Fitz) PDF Text Extraction and Layout Processing*. Official PyMuPDF Reference.

[14] Roy, A. and Thomas, P. "Hybrid Metadata Verification Framework for Scholarly Records." *Information Systems Research*, 2023.

[15] Google Scholar Data Parsing API (ScholarPy), 2023. Available: https://scholarpy. readthedocs.io/

[16] Scimago Journal & Country Rank. *Quartile (Q1–Q4) Computation and Journal Ranking Methodology*.

[17] Scopus Database. *Journal Indexing and Evaluation Guidelines*. Elsevier.

[18] Scopus API Developer Guide, Elsevier, 2024. Available: https://dev.elsevier.com/

[19] Smith, A. "Automated Extraction of Scientific Metadata Using NLP Techniques." *IEEE Transactions on Information Systems*, 2021.

[20] UGC CARE. *Reference List of Quality Journals for Indian Academia*.

[21] Web of Science. *JournalCitationReportsandIndexing Framework*. Clarivate Analytics.

[22] Academic Research on Semantic Similarity and Document Embeddings. *Various IR and NLP Publications*.