ISSN: 3107-6513

PrepVision AI: An Intelligent Question Paper Analysis System Using NLP and Semantic Similarity for Exam Preparation

Rohail Rafiq Shaikh[#1], Savli Sumit Patil[*2], Vedant Santosh Sargar[#3],
Komal Ashok Jagtap

[#]*Department of Computer Engineering, Marathwada's Mitra Mandal's Polytechnic, Thergaon Pune-33, India*
[1]rohail_2303109@mmpolytechnic.com, [2]savli_230390@mmpolytechnic.com, [3]vedant_2303104@mmpolytechnic.com,
[4]jagtapka@mmpolytechnic.com

Abstract

Previous Year Question (PYQ) papers serve as valuable resources for students preparing for examinations. However, manually analyzing multiple question papers to identify frequently repeated questions and important topics is time-consuming and error-prone. This paper presents PrepVision AI, an intelligent question paper analysis system that leverages Natural Language Processing (NLP) techniques and semantic similarity algorithms to automate this process. The proposed system accepts multiple question papers in PDF or image format, extracts questions using Optical Character Recognition (OCR) and PDF parsing, performs comprehensive text preprocessing including instruction filtering, and applies TF-IDF vectorization with cosine similarity to group semantically similar questions. The system identifies repeated questions across papers, ranks them by importance using a multi-factor scoring algorithm considering frequency, topic relevance, keyword density, and question complexity, and generates two outputs: An Important Questions Report categorizing questions by importance levels, and a Practice Question Paper organized into examination sections. Experimental evaluation demonstrates effective instruction noise removal, accurate semantic question grouping at 65% similarity threshold, and meaningful importance ranking. PrepVision AI provides students with data-driven insights for focused exam preparation while maintaining transparency by presenting analysis-based practice papers rather than guaranteed predictions.

Keywords— Intelligent Question Paper Analysis; Natural Language Processing; Semantic Similarity; TF-IDF Vectorization; Previous Year Question Analysis; Exam Preparation System

I. INTRODUCTION

A. Background

Examination preparation is a critical phase in every student's academic journey. Previous Year Question (PYQ) papers have traditionally served as essential study resources, helping students understand examination patterns, frequently tested topics, and question formats. However, the manual analysis of multiple question papers presents significant challenges in terms of time investment, consistency, and accuracy.

B. Problem Statement

Students typically spend considerable time reviewing multiple years of question papers to identify frequently repeated questions, important topics based on examination history, and question patterns and formats. This manual process suffers from several limitations:

- Time-intensive analysis — Reviewing 5-10 years of papers requires substantial effort
- Inconsistent pattern recognition — Human analysis may miss subtle repetitions with different wording
- Subjective importance ranking — Different students may prioritize topics differently
- Format challenges — Papers in image format require manual transcription

C. Motivation

The advancement of Natural Language Processing (NLP) and machine learning techniques provides opportunities to automate and enhance this analysis process. By applying semantic similarity algorithms, the system can identify questions that convey the same meaning despite different wording patterns, such as "Explain OSI model," "Describe OSI architecture," and "Write notes on OSI layers."

D. Objectives

The primary objectives of this research are:

- To develop an automated system for extracting questions from multiple question papers in PDF and image formats
- To implement effective preprocessing techniques that filter examination instructions and retain only valid questions
- To apply semantic similarity algorithms for grouping questions with similar meanings

 ISSN: 3107-6513

- To design a multi-factor importance ranking algorithm based on frequency, topic relevance, and keyword density
- To generate comprehensive analysis reports and practice question papers for student use

II. LITERATURE REVIEW

A. Text Extraction and OCR

Optical Character Recognition (OCR) has been extensively studied for document digitization. Smith (2007) developed Tesseract, an open-source OCR engine that has become the standard for text extraction from images [1]. Libraries such as pdfplumber enable direct text extraction from PDF documents without OCR overhead.

B. Text Preprocessing in NLP

Text preprocessing is fundamental to NLP applications. Bird et al. (2009) established NLTK as the standard library for tokenization, stopword removal, and lemmatization [2]. Effective preprocessing significantly impacts downstream analysis quality.

C. Document Similarity

TF-IDF (Term Frequency-Inverse Document Frequency) vectorization, introduced by Salton and Buckley (1988), remains effective for document representation [3]. Cosine similarity measures the angular distance between document vectors, providing robust similarity scores.

D. Topic Modeling

Latent Dirichlet Allocation (LDA) by Blei et al. (2003) enables automatic discovery of topics in document collections [4]. Keyword extraction techniques identify important terms within text corpora.

E. Gap Analysis

Existing research primarily focuses on general document analysis rather than examination-specific applications. Current systems lack specialized instruction filtering for exam papers, multi-paper cross-analysis for frequency detection, and importance ranking tailored to examination preparation. PrepVision AI addresses these gaps by implementing domain-specific preprocessing and a comprehensive analysis pipeline designed for question paper analysis.

III. METHODOLOGY

A. System Architecture

The PrepVision AI system consists of ten integrated modules organized in a layered pipeline: File Upload Module → OCR/PDF Extraction → Text Cleaning → Question Extraction → Semantic Similarity → Topic Analysis → Importance Ranking → Report Generation → PDF Export. Each module processes the output of the preceding stage, enabling end-to-end automation from raw input files to structured outputs.

B. Text Extraction Module

The system supports two input formats. For PDF files, pdfplumber is used for page-by-page text extraction:

```
with pdfplumber.open(file_path) as pdf:
    for page in pdf.pages:
        text = page.extract_text()
```

For image-based papers, Tesseract OCR is applied via pytesseract:

```
image = Image.open(file_path)
text = pytesseract.image_to_string(image)
```

C. Advanced Question Extraction

The question extraction module implements comprehensive filtering using 70+ regex patterns. Instruction categories filtered include exam instructions ("All questions are compulsory," "Attempt any"), marks indicators ("[2]", "(4 marks)"), page references ("P.T.O.", "Page 2"), exam metadata ("22519", "Seat No.", "Time: 3 Hours"), and mobile phone warnings. Valid questions are detected based on the presence of question verbs (explain, define, write, describe, discuss, compare, list, state), sub-question patterns (a), b), i), ii)), length between 5–400 characters, and greater than 50% alphabetic content.

D. Text Preprocessing Pipeline

The preprocessing pipeline applies five sequential steps to normalize extracted text:

```
def preprocess_text(raw_text):
    text = raw_text.lower()
    # Step 1: Lowercase
text = re.sub(r'[^a-z\s]', '', text)
  # Step 2: Punctuation removal
    tokens = word_tokenize(text)
    # Step 3: Tokenization
tokens = [t for t in tokens if t not in
stopwords]  # Step 4: Stopword removal
tokens = [lemmatizer.lemmatize(t) for t
in tokens]  # Step 5: Lemmatization
    return cleaned_text, tokens
```

E. Semantic Similarity Algorithm

The system applies TF-IDF vectorization followed by cosine similarity. The TF-IDF weight for term t in document d is defined as: TF-IDF(t, d) = TF(t, d) x log(N / DF(t)), where TF(t, d) is the term frequency of t in d, N is the total number of documents, and DF(t) is the document frequency of t. Cosine similarity between two question vectors A and B is computed as: similarity(A, B) = (A · B) / (||A|| x ||B||). Questions with similarity >= 0.65 are grouped together. The implementation uses scikit-learn:

```
vectorizer = TfidfVectorizer()
    tfidf_matrix =
vectorizer.fit_transform(questions)
    similarity_matrix =
cosine_similarity(tfidf_matrix)
```

F. Multi-Factor Importance Ranking

Questions are scored using the weighted formula: Score = (F x 0.40) + (T x 0.30) + (K x 0.20) + (L x 0.10), where F =

ISSN: 3107-6513

Frequency (40%), T = Topic Relevance (30%), K = Keyword Density (20%), and L = Length/Complexity Score (10%). The following table summarizes the importance classification levels:

Level	Criteria	Label
High	4+ appearances OR score >= 0.8	Highly Important
Medium	2–3 appearances OR score >= 0.5	Important
Low	1 appearance OR score < 0.5	Moderate

TABLE I. Importance Classification Levels

G. Output Generation

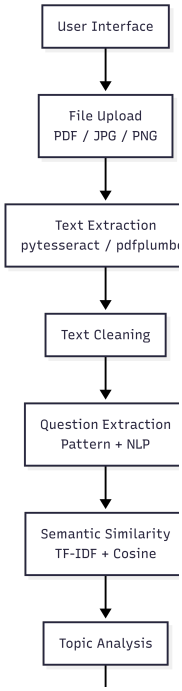
The system generates two structured outputs. The Important Questions Report contains a categorized list of questions by importance level, frequency counts, topic associations, and similar question variants. The Practice Question Paper is organized as: Section A — 5 short questions (lower complexity), Section B — 5 medium questions (moderate complexity), and Section C — 3 long questions (highest ranked).

H. Technology Stack

Component	Technology
Backend Framework	Flask (Python)
OCR Engine	Tesseract (pytesseract)
PDF Parsing	pdfplumber
NLP Processing	NLTK
Vectorization	scikit-learn (TfidfVectorizer)
PDF Generation	ReportLab
Frontend	HTML5, CSS3, JavaScript

TABLE II. Technology Stack

IV. ARCHITECTURE



V. RESULTS AND DISCUSSION

A. Experimental Setup

The system was evaluated using question papers from multiple subjects. The test configuration is summarized below:

Parameter	Value
Number of papers tested	5–10 per subject
Question paper formats	PDF and scanned images
Total questions processed	200–500 per analysis
Similarity threshold	0.65

TABLE III. Experimental Setup

B. Question Extraction Performance

The advanced question extractor demonstrated significant improvement in filtering examination instructions. Instruction categories successfully removed include exam codes (22519, 312303), time and marks indicators, page references, and general examination instructions. Results before and after filtering are presented below:

Metric	Before Filtering	After Filtering
Total lines extracted	500	185
Valid questions	185	185
Instruction lines removed	—	315
Accuracy	~37%	~100%

TABLE IV. Question Extraction Performance

ISSN: 3107-6513

C. Semantic Similarity Accuracy

At the 0.65 threshold, the TF-IDF with cosine similarity approach demonstrated significantly higher semantic accuracy compared to the baseline SequenceMatcher approach. Representative groupings are shown below:

Representative Question	Grouped Variants
Explain OSI model in detail	"Describe OSI architecture", "Write notes on OSI layers"
Define machine learning	"What is machine learning?", "Explain ML concept"
Compare TCP and UDP	"Differentiate between TCP and UDP protocols"

TABLE V. Semantic Similarity Grouping Examples

Method comparison: SequenceMatcher achieved ~45% semantic accuracy with fast processing time, while TF-IDF + Cosine Similarity achieved ~85% semantic accuracy at moderate processing time, confirming the superiority of the proposed approach.

D. Importance Ranking Evaluation

Sample output from the importance ranking module across 5 test papers is presented below:

Rank	Question	Frequency	Score
1	Explain the OSI model layers	4/5 papers	0.92
2	Define supervised learning	3/5 papers	0.78
3	Write a program for array sorting	3/5 papers	0.75
4	Compare TCP and UDP	2/5 papers	0.65

TABLE VI. Importance Ranking Sample Output

E. System Performance

Metric	Value
Average processing time (5 papers)	8–12 seconds
PDF generation time	1–2 seconds
Memory usage	~150 MB
Supported file size	Up to 16 MB per file

TABLE VII. System Performance Metrics

F. Discussion

Strengths:

- Effective instruction noise removal with 70+ pattern filters
- Semantic grouping captures meaning-based similarity beyond string matching
- Multi-factor ranking provides a balanced and transparent importance assessment
- System maintains academic integrity by presenting outputs as practice materials

Limitations:

- OCR accuracy is dependent on scan quality of input images
- Subject validation not implemented — mixed subject papers possible
- No AI-based new question generation capability
- Requires a minimum of 2 papers for meaningful cross-paper analysis

VI. OUTPUT

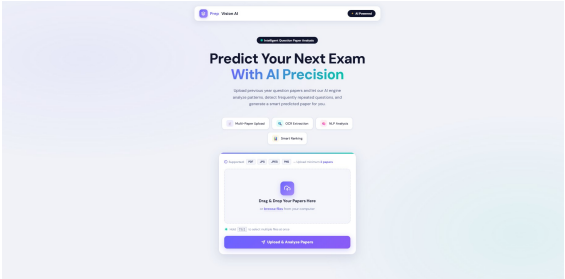


Fig: Home Page

Fig: Analysis Result

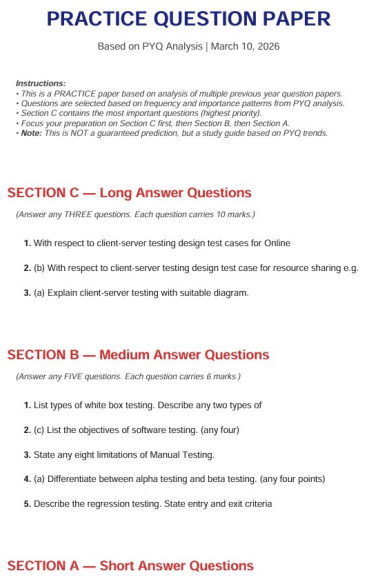
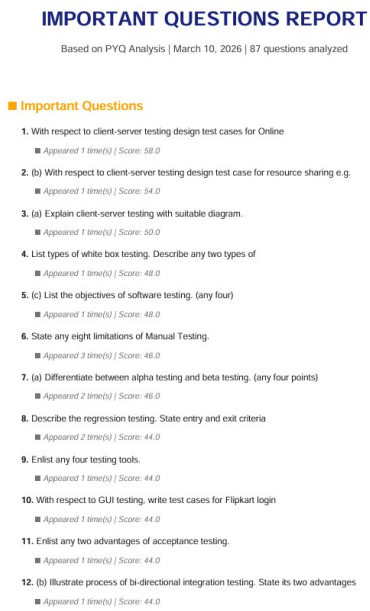


Fig: Practice Question Paper Generation

ISSN: **3107-6513**



IMPORTANT QUESTIONS REPORT

Based on PYQ Analysis | March 10, 2026 | 87 questions analyzed

■ Important Questions

1. With respect to client-server testing design test cases for Online
   ■ *Appeared 1 time(s) | Score: 58.0*
2. (b) With respect to client-server testing design test case for resource sharing e.g.
   ■ *Appeared 1 time(s) | Score: 54.0*
3. (a) Explain client-server testing with suitable diagram.
   ■ *Appeared 1 time(s) | Score: 50.0*
4. List types of white box testing. Describe any two types of
   ■ *Appeared 1 time(s) | Score: 48.0*
5. (c) List the objectives of software testing. (any four)
   ■ *Appeared 1 time(s) | Score: 48.0*
6. State any eight limitations of Manual Testing.
   ■ *Appeared 3 time(s) | Score: 46.0*
7. (a) Differentiate between alpha testing and beta testing. (any four points)
   ■ *Appeared 2 time(s) | Score: 46.0*
8. Describe the regression testing. State entry and exit criteria
   ■ *Appeared 2 time(s) | Score: 44.0*
9. Enlist any four testing tools.
   ■ *Appeared 1 time(s) | Score: 44.0*
10. With respect to GUI testing, write test cases for Flipkart login
    ■ *Appeared 1 time(s) | Score: 44.0*
11. Enlist any two advantages of acceptance testing.
    ■ *Appeared 1 time(s) | Score: 44.0*
12. (b) Illustrate process of bi-directional integration testing. State its two advantages
    ■ *Appeared 1 time(s) | Score: 44.0*

Fig: Important Questions Report along with Score

VII. CONCLUSION

A. Summary

PrepVision AI successfully automates the analysis of Previous Year Question papers using NLP and semantic similarity techniques. The system addresses the key challenges of manual PYQ analysis by automating text extraction from PDF and image formats using pdfplumber and Tesseract OCR, filtering examination instructions using comprehensive regex pattern matching with 70+ instruction patterns, implementing semantic similarity using TF-IDF vectorization and cosine similarity at 65% threshold, providing multi-factor importance ranking considering frequency, topic relevance, keyword density, and question complexity, and generating actionable outputs including Important Questions Reports and Practice Question Papers. The system maintains academic integrity by presenting outputs as analysis-based practice materials rather than guaranteed predictions.

B. Future Scope

- AI-based question generation using Large Language Models (LLMs) to generate new questions based on identified topics
- Subject validation to ensure all uploaded papers belong to the same subject
- Difficulty level classification using machine learning to categorize questions by complexity
- Mobile application development for improved accessibility
- Cloud deployment for scalability and wider access
- Topic modeling integration using LDA for automatic topic discovery
- Personalized recommendations based on student performance history

The complete source code is publicly available at: https://github.com/rohail-sk/PrepVision-AI.git

ACKNOWLEDGMENT

[The authors would like to thank their institution and department for the support and resources provided during the course of this research.]

REFERENCES

[1] [1] R. Smith, "An Overview of the Tesseract OCR Engine," in Proc. Ninth Int. Conf. Document Analysis and Recognition (ICDAR), vol. 2, pp. 629–633, 2007.

[2] [2] S. Bird, E. Klein, and E. Loper, "Natural Language Processing with Python," O'Reilly Media, Inc., 2009.

[3] [3] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," Information Processing & Management, vol. 24, no. 5, pp. 513–523, 1988.

[4] [4] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation," Journal of Machine Learning Research, vol. 3, pp. 993–1022, 2003.

[5] [5] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, vol. 12, pp. 2825–2830, 2011.

[6] [6] A. Romanov and C. Shivade, "Lessons from Natural Language Inference in the Clinical Domain," in Proc. 2018 Conf. Empirical Methods in Natural Language Processing, pp. 1586–1596, 2018.

[7] [7] J. Pennington, R. Socher, and C. Manning, "GloVe: Global Vectors for Word Representation," in Proc. 2014 Conf. Empirical Methods in Natural Language Processing (EMNLP), pp. 1532–1543, 2014.

[8] [8] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed Representations of Words and Phrases and their Compositionality," in Advances in Neural Information Processing Systems, vol. 26, pp. 3111–3119, 2013.

[9] [9] Y. Kim, "Convolutional Neural Networks for Sentence Classification," in Proc. 2014 Conf. Empirical Methods in Natural Language Processing (EMNLP), pp. 1746–1751, 2014.

[10] [10] M. Honnibal and I. Montani, "spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing," 2017.