Automated Plagiarism Detection System

Frequently Asked Questions

Q1: What is the Automated Plagiarism Detection System?

Answer: The Automated Plagiarism Detection System is a comprehensive web-based application designed to identify and analyze textual similarities across student submissions and online sources. It leverages advanced text processing techniques and similarity algorithms to detect plagiarism efficiently in educational settings.

Q2: What are the main features of this system?

Answer: The system includes multi-file PDF document processing (up to 10 submissions), text extraction and preprocessing capabilities, comprehensive pairwise similarity analysis, optional online plagiarism detection, risk level categorization and visualization, and detailed matching segment identification.

Q3: Which institutions can benefit from this system?

Answer: Educational institutions such as universities, colleges, and schools can benefit significantly. The system is particularly valuable for departments managing large numbers of student submissions where manual plagiarism detection would be time-consuming and prone to errors.

Q4: What file formats does the system support?

Answer: The current implementation supports PDF (Portable Document Format) files. Future enhancements are planned to support additional formats including DOCX, TXT, PPTX, and image-based documents with OCR capabilities.

Q5: How many documents can the system process simultaneously?

Answer: The system is designed to process up to 10 student submissions in a single batch. This limit balances practical processing speed (typically 8–18 seconds) with system scalability considerations. Multiple processing runs can be executed for larger batches.

Q6: Is the system accessible to non-technical users?

Answer: Yes, the system is built on the Streamlit framework, providing an intuitive web-based interface that requires minimal technical knowledge. Educators can upload files, configure parameters, and interpret results without programming expertise.

Q7: What technologies are used in the implementation?

Answer: The technology stack includes Streamlit for the web framework, PyPDF2 for PDF text extraction, Python's `re` module for preprocessing, `difflib` for similarity computation, and `requests` and `BeautifulSoup4` for online content fetching.

Q8: How does the system extract text from PDF files?

Answer: The text extraction process uses PyPDF2 library to parse document structure, extract text page-by-page, concatenate page contents, and handle corrupted or password-protected PDFs gracefully. The time complexity is $O(p)$ where p is the number of pages.

Q9: What preprocessing steps are applied to text?

Answer: Preprocessing includes lowercasing text, removing special characters and punctuation using regex patterns, and normalizing whitespace. These steps normalize text variations while preserving semantic content and improve comparison accuracy by approximately 10–15%.

Q10: How is text similarity calculated?

Answer: The system calculates similarity using Python's `difflib.SequenceMatcher` class, which implements the Ratcliff-Obershelp algorithm. Both documents are preprocessed independently, a matching ratio is computed, and converted to percentage scale: $S_{\text{similarity}} = \text{SequenceMatcher}(T_1, T_2) \times 100$.

Q11: What are matching blocks?

Answer: Matching blocks are continuous segments of identical text between two documents. The system identifies blocks by extracting all matching sequences, filtering blocks smaller than 20 characters, and recording position and content of significant matches for educator verification.

Q12: How does the system manage session state?

Answer: The application uses Streamlit's session state mechanism to maintain data across user interactions. Uploaded PDFs and extracted text persist across page reruns, text is cached for efficiency, and multiple analysis runs use consistent data from the same upload session.

Q13: What is the Ratcliff-Obershelp algorithm?

Answer: This is a classic pattern matching algorithm that finds the longest contiguous matching subsequence between two strings, recursively solves matching problems for text segments before and after the match, and computes a ratio as: ratio $= \frac{2 \times M}{T}$ where M is matching characters and T is total characters.

Q14: How does cosine similarity differ from sequence matching?

Answer: Sequence matching finds positional character matches and is best for detecting verbatim plagiarism. Cosine similarity treats documents as vectors in multidimensional space and detects semantic similarity regardless of word order. The system uses sequence matching for computational efficiency.

Q15: What is Levenshtein distance?

Answer: Levenshtein distance quantifies the minimum single-character edits (insertions, deletions, substitutions) required to transform one string into another. It is particularly useful for detecting paraphrased content where wording is slightly changed, though not currently used in the implementation.

Q16: How is the overall plagiarism score calculated?

Answer: The system uses a weighted hybrid model: $S_{\text{overall}} = (S_{\text{peer}} \times 0.6) + (S_{\text{online}} \times 0.4)$ where S_{peer} is maximum peer similarity (60% weight) and S_{online} is online detection score (40% weight).

Q17: How is the online plagiarism score determined?

Answer: The online score is calculated as: $S_{\text{online}} = \min(\text{matching queries} \times 25\%, 75\%)$. The process extracts up to 3 sentences, converts them to search queries, implements 1-second rate limiting, and calculates the score with a 75% maximum cap.

Q18: What preprocessing techniques are most important?

Answer: Lowercasing eliminates false negatives from case variations. Punctuation removal normalizes formatting differences. Whitespace normalization handles spacing inconsistencies. These techniques improve system performance by 10–15%. Stop word removal is not employed as these words provide structural information useful for plagiarism detection.

Q19: What is the time complexity of the system?

Answer: Time complexity for key operations: Text extraction $O(p)$, preprocessing $O(n)$, similarity calculation $O(n \times m)$, pairwise comparison $O(s^2 \times n \times m)$. Overall complexity for s students: $T_{\text{total}} = O(s^2 \times n^2)$, indicating quadratic growth with document quantity and length.

Q20: What is the space complexity?

Answer: Storage requirements are: $S = O(s \times n)$ where s is the number of students and n is average document length. For 10 students with 5000-character documents, total memory footprint is approximately 650 KB, including extracted text storage (500 KB), session state overhead (50 KB), and comparison matrices (100 KB).

Q21: How accurate is the plagiarism detection?

Answer: Accuracy varies by plagiarism type: verbatim plagiarism achieves 95–99% detection accuracy, paraphrased content achieves 40–60% accuracy due to semantic variations, and preprocessing improves overall accuracy by 10–15%.

Q22: How long does it take to process documents?

Answer: Processing times for documents averaging 5000 characters: 2 students (0.3s without online, 3.2s with online), 5 students (1.8s without online, 8.5s with online), 10 students (8.2s without online, 18.3s with online). Formula: $T = T_{\text{extract}} + T_{\text{preprocess}} + T_{\text{compare}} + T_{\text{online}}$.

Q23: What is the memory footprint for typical use cases?

Answer: Memory usage for 10-student batch with 5000-character documents: extracted text storage (500 KB), session state overhead (50 KB), comparison matrices (100 KB), total footprint (650 KB). This modest requirement enables deployment on standard web servers and local machines.

Q24: How does the system compare to manual plagiarism detection?

Answer: Automated system processes 10 documents in 8–18 seconds versus 2–4 hours for manual review. System provides very high consistency compared to moderate consistency

in manual review. Automated approach has low cost and excellent scalability versus high cost and poor scalability of manual methods with minimal human error versus possible errors.

Q25: What are the risk level categories?

Answer: High Risk (70% or above) indicates substantial plagiarism requiring immediate investigation. Medium Risk (40–70%) indicates moderate plagiarism concerns warranting further review. Low Risk (below 40%) indicates minimal plagiarism generally acceptable for submission. Thresholds can be adjusted based on institutional policies.

Q26: How are risk levels visually represented?

Answer: Color coding provides intuitive visualization: Red for High Risk (score 70% or above), Orange for Medium Risk (40–70% range), and Green for Low Risk (below 40%). Color-coded indicators provide immediate visual feedback enabling educators to quickly identify submissions requiring attention.

Q27: What information is provided in the results?

Answer: Results include overall plagiarism score (percentage 0–100%), risk level classification, maximum peer similarity score, online plagiarism detection results, list of matching segments with context, and detailed comparison with each peer document. Results are presented in expandable cards with detailed metrics.

Q28: What are the current system limitations?

Answer: Current limitations include paraphrasing detection limited to 40–60% accuracy, optimization for English only, mock online search implementation without real API integration, limitation to PDF format, quadratic processing time growth requiring optimization for 50+ submissions, and limited semantic analysis capabilities.

Q29: What machine learning enhancements are planned?

Answer: Future plans include BERT integration for advanced semantic similarity detection, neural network models for paraphrasing identification, GPT models for semantic plagiarism detection, and custom training on institutional plagiarism datasets. These enhancements are expected to improve paraphrasing detection to 70–80% accuracy.

Q30: How should educators use the system in their workflow?

Answer: Educators should access the Streamlit web interface, upload student PDF submissions via sidebar file uploader, configure analysis parameters, click the Analyze button, review results in the main content area, examine matching blocks for plagiarism evidence, consult with students about medium/high-risk submissions, and document decisions. The system should support but not replace established academic integrity policies.