International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

# AI-Based College Lost & Found System Using NLP-Based Text Similarity

Dhananjay Nimbalkar[1], Gaurav Pawar[2], Snehal Pilankar[3]
[1,2,3]*MSc Computer Science, School of Information Technology, Indira University, Pune, India*

***Abstract*—Lost-and-found handling in a college involves matching two independently written descriptions that rarely use the same words. This paper presents an AI-based College Lost & Found web application, built with Streamlit and SQLite, that automatically compares lost and found reports and ranks the most likely matches. The application also serves as a research testbed for comparing two text-similarity approaches: TF-IDF with cosine similarity, which captures shared vocabulary, and Sentence-BERT (all-MiniLM-L6-v2) with cosine similarity, which captures similar meaning expressed in different words. Both scores are shown side by side with a configurable YES / MAYBE / NO status, and a separate evaluation module measures Accuracy, Precision, Recall, F1-score and Top-K accuracy on the same labelled pairs. Exploratory analysis of 276 entries from the college's handwritten lost-item register (2022–2026) shows very short descriptions (88.4% are a single word) and many different wordings for the same kind of object. Because the register contains no found reports, the paper does not claim a final winner; it presents the system design, a literature review of thirty works, the dataset, the experimental procedure and the research gaps that motivate the comparison.**

***Keywords*—Cosine Similarity, Lost and Found System, Natural Language Processing, Sentence-BERT, Streamlit, Text Similarity, TF-IDF**

## I. Introduction

Lost-and-found handling in a college involves matching two independently written descriptions: a report from a student who has lost an item and a report from someone who has found one. Although both reports may refer to the same physical object, they are unlikely to use the same words. One student may report a "black purse" while another submits a found report for a "small black wallet". This project builds a working Lost & Found web application for a college that compares such descriptions automatically and ranks the most likely matches, and it uses the same application as a research testbed for comparing two text-similarity approaches: **TF-IDF with Cosine Similarity** and **Sentence-BERT (SBERT) with Cosine Similarity**.

### *A. Existing System and Need for System*

College lost-and-found services commonly depend on manually written reports and keyword-based searches. While this works for small volumes, it has the following limitations:

*1. Vocabulary mismatch:* A keyword search can fail to connect two reports that use different words for the same object (for example "purse" and "wallet"), and can return misleading results when unrelated objects share common words such as "black" or "bag".

*2. Manual effort:* Staff or students must read through reports to find a possible match, which becomes slow as the number of reports grows.

*3. No ranking:* Results are not ordered by how closely they match, so the most likely candidate is not shown first.

*4. Sensitivity to wording:* Spelling variations, incomplete details and informal language reduce the chance of a successful match.

*5. No evidence for method choice:* There is little evidence on which text-matching approach works best for short, informal campus item reports (see Section II-E).

These limitations show the need for a system that automatically compares lost and found descriptions, ranks the candidates, and shows how each method scored them so that the methods can be compared fairly.

### *B. Need for System*

The proposed system addresses these gaps in the following ways:

*1. Automatic matching:* Every new lost or found report is compared against all active reports of the opposite type, with no manual searching.

*2. Two complementary methods:* TF-IDF captures shared vocabulary [1]–[3], while SBERT captures similar meaning even when different words are used [17].

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

*3. Side-by-side scores:* Both similarity scores are displayed for every candidate, which allows lexical and semantic similarity to be compared directly.

*4. Human confirmation:* Scores are used to rank possible matches for a person to review. They are not treated as proof of ownership.

*5. Evidence-based evaluation:* A separate evaluation module measures Accuracy, Precision, Recall and F1-score for both methods on the same labelled data.

### C. Scope of Work

1. *User Management*:
   - Student registration and login with password hashing (PBKDF2-HMAC-SHA256); no plain-text passwords are stored.
   - Role-based access with student and admin roles.
2. *Reporting and Browsing:*
   - Report Lost Item and Report Found Item forms with input validation.
   - Browse Lost / Found items with keyword search and category filter, and a My Reports page dedicated for browsing and reporting.
3. *Automatic Matching:*
   - Text preprocessing shared by both methods.
   - TF-IDF + Cosine Similarity and SBERT (all-MiniLM-L6-v2) + Cosine Similarity.
   - Configurable Strong Match and Possible Match thresholds; ranked results with YES / MAYBE / NO status.
4. *Administration:*
   - View users and all reports, delete inappropriate reports, mark items as resolved, and view matching statistics.
5. *Research Evaluation:*
   - A standalone script that evaluates both methods on the same labelled pairs using Accuracy, Precision, Recall, F1-score and Top-K accuracy, and saves tables and charts.
6. *Future Scope:*
   - Image-based, location-based and date-based matching, multilingual descriptions, mobile application and email/SMS notifications.

### D. Operating Environment – Hardware and Software

*1) Hardware Requirements:*

- **Processor:** Intel Core i5 or equivalent
- **RAM:** Minimum 8 GB
- **Storage:** Minimum 10 GB of free disk space (Python packages including PyTorch, plus the SBERT model download)

*2) Software Requirements:*

- **Python:** Version Python 3.10–3.12 recommended. Libraries: streamlit, pandas, numpy, scipy, scikit-learn, sentence-transformers, torch, matplotlib.
- **Database:** SQLite: built-in sqlite3 module.
- **NLP Model:** Sentence-BERT model all-MiniLM-L6-v2, downloaded once from Hugging Face on first run (internet required for this step only).
- **Operating System:** Windows 10/11 (64-bit); the application also runs on Linux and macOS.
- **Browser:** Any modern web browser to open the Streamlit interface (default http://localhost:8501).

## II. Proposed System

### A. Proposed System

The proposed system is a Streamlit web application backed by an SQLite database. When a user submits a LOST report, the system retrieves all active FOUND reports, compares the new description against each one using both algorithms, stores the scores, and shows the candidates ranked from highest to lowest similarity. The same process runs in the opposite direction when a FOUND report is submitted.

**Approach 1 – TF-IDF + Cosine Similarity:** Item description → text preprocessing → TF-IDF vectorization → cosine similarity → similarity score.

**Approach 2 – SBERT + Cosine Similarity:** Item description → text preprocessing → SBERT embedding (all-MiniLM-L6-v2) → cosine similarity → similarity score.

The system is organised into small modules so that each part can be understood and tested separately (Table I).

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

TABLE I
SOFTWARE MODULES OF THE SYSTEM

| Module | Purpose |
|---|---|
| app.py | Streamlit user interface: sidebar navigation, forms, dashboard cards, match cards, admin pages. |
| database.py | SQLite schema (users, items, matches) and all create/read/update/delete functions. |
| auth.py | Registration, login, email/password validation and PBKDF2 password hashing. |
| preprocessing.py | Shared text cleaning pipeline used identically by both algorithms. |
| similarity.py | TF-IDF and SBERT similarity functions; the SBERT model is loaded once and cached. |
| matching.py | Compares an item against all opposite-type items, applies thresholds, ranks and stores matches. |
| research_evaluation.py | Offline experiment comparing TF-IDF and SBERT on labelled pairs; saves CSV results and charts. |
| config.py | Paths, model name, default thresholds, item categories. |

*Match decision.* Each pair receives a YES / MAYBE / NO status. The two thresholds (Strong Match and Possible Match) are adjustable in the application sidebar. The defaults (0.75 and 0.45) are initial values for demonstration only. They are not scientifically validated and will be optimised later using the labelled research dataset. The score that drives the decision (SBERT, TF-IDF or their average) is also selectable, so the application does not assume that one method is better than the other.

*Database design.* The users table stores id, name, email, password_hash, role and created_at. The items table stores id, user_id, report_type (LOST/FOUND), item_name, category, description, color, brand, location, date, status and created_at. The matches table stores lost_item_id, found_item_id, tfidf_score, sbert_score, match_status and created_at.

### B. Feasibility Study

The feasibility study analyses the practicality of the proposed system:

1. *Technical Feasibility:*
   - All components (Python, Streamlit, scikit-learn, sentence-transformers, SQLite) are freely available and run on a normal laptop.
   - A compact model (all-MiniLM-L6-v2) is used so that encoding is practical on modest hardware [18].
2. *Economic Feasibility:*
   - Only open-source tools are used, so there is no licensing cost.
   - No server infrastructure or GPU is needed for a college-scale prototype.
3. *Operational Feasibility:*
   - The interface is a simple web form workflow that students and administrators can use without training.
   - Results are shown as ranked candidates with both scores, so a person can confirm the match.

### C. Objectives

The objectives of the proposed system are as follows:

1. To build a working college Lost & Found web application for reporting, browsing and managing lost and found items.
2. To automatically identify potential matches between lost and found descriptions using TF-IDF + Cosine Similarity and SBERT + Cosine Similarity.
3. To display both similarity scores and a configurable YES / MAYBE / NO result for every candidate match.
4. To experimentally compare the two approaches on the same labelled data using Accuracy, Precision, Recall, F1-score and Top-K accuracy.
5. To answer the research question without assuming in advance that either approach is better.

*Research question:* Which of the two approaches, TF-IDF-based text similarity or SBERT-based semantic similarity, provides better matching accuracy for college Lost & Found item descriptions?

### D. Research Requirements

1. *Dataset:*

Real lost-item entries from the college register (2022–2026), anonymised and transcribed, plus FOUND reports and labelled pairs (Match = 1, Non-match = 0). The FOUND reports and labelled pairs still have to be collected.

2. *Tools:*
   - Python, scikit-learn (TF-IDF, cosine similarity, metrics), sentence-transformers (SBERT), pandas, NumPy, SciPy.
   - Matplotlib for charts; Streamlit and SQLite for the application.
3. *Knowledge Base:*
   - Information retrieval, vector-space models and cosine similarity [2], [3], [6].

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

- Sentence embeddings and Transformer models [14], [15], [17].

*4. Metrics:*

Accuracy, Precision, Recall, F1-score and Top-K accuracy.

### *E. Literature Review*

This section summarises thirty works that form the basis of the project, grouped into five themes: lexical retrieval and TF-IDF; word and sentence embeddings; Transformers and SBERT; matching architectures and evaluation datasets; and surveys, benchmarks and efficiency (Table II). Each summary ends with its citation number, and the full IEEE-style references are given in the References section. This is a narrative literature-based review. It does not report experimental results of its own.

**TABLE II**
**LITERATURE REVIEW OF THE THIRTY WORKS UNDERLYING THE PROJECT**

| Sr. | Paper | Summary (with citation) |
|---|---|---|
| **A. Lexical Retrieval and TF-IDF** | | |
| 1 | Spärck Jones (1972) [1] | Gave a statistical interpretation of term specificity and inverse document frequency (IDF), reducing the weight of common terms and raising that of rarer ones. This is the theoretical basis of the TF-IDF weighting used here. [1] |
| 2 | Salton, Wong & Yang (1975) [2] | Introduced the vector-space model in which documents are term vectors compared geometrically. It underpins ranking reports by cosine similarity. [2] |
| 3 | Salton & Buckley (1988) [3] | Studied term-weighting approaches and established TF-IDF with length normalisation as a core retrieval technique, which makes reports of different lengths comparable. [3] |
| 4 | Deerwester et al. (1990) [4] | Proposed Latent Semantic Analysis, applying singular value decomposition to term-document matrices to capture some latent relationships beyond exact word overlap. [4] |
| 5 | Robertson & Zaragoza (2009) [5] | Reviewed the probabilistic relevance framework and BM25, an important lexical ranking alternative that remains a strong baseline. [5] |
| 6 | Manning, Raghavan & Schütze (2008) [6] | Textbook covering term weighting, vector-space retrieval and evaluation metrics; used as the foundation for the evaluation design. [6] |
| **B. Word and Sentence Embeddings** | | |
| 7 | Mikolov et al. (2013) [7] | Introduced Word2Vec, which learns word vectors from context so that related words lie close together. Discussed here as background only; it is not implemented in this project. [7] |
| 8 | Pennington, Socher & Manning (2014) [8] | Proposed GloVe, which learns word vectors from global word co-occurrence statistics. [8] |
| 9 | Le & Mikolov (2014) [9] | Proposed Paragraph Vectors, fixed-length representations for sentences and documents, extending embeddings beyond single words. [9] |
| 10 | Kiros et al. (2015) [10] | Introduced Skip-Thought, an unsupervised sentence encoder trained to predict neighbouring sentences. [10] |
| 11 | Arora, Liang & Ma (2017) [11] | Showed that weighted averaging of word vectors is a competitive and inexpensive sentence-embedding baseline. [11] |
| 12 | Conneau et al. (2017) [12] | Presented InferSent, which learns sentence representations from supervised natural-language-inference data. [12] |
| 13 | Cer et al. (2018) [13] | Introduced the Universal Sentence Encoder for semantic similarity and transfer tasks. [13] |
| **C. Transformers and Sentence-BERT** | | |
| 14 | Vaswani et al. (2017) [14] | Introduced the Transformer architecture, which uses self-attention to model relationships among tokens. [14] |
| 15 | Devlin et al. (2019) [15] | Introduced BERT, which learns bidirectional contextual representations through pre-training, so a word is represented in relation to its surrounding words. [15] |
| 16 | Liu et al. (2019) [16] | Proposed RoBERTa, showing that changes in the training procedure improve pretrained language representations. [16] |
| 17 | Reimers & Gurevych (2019) [17] | Introduced Sentence-BERT, a siamese architecture that produces sentence embeddings comparable by cosine similarity. Each report can be encoded once and compared quickly. This is the second method used in this project. [17] |

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

| Sr. | Paper | Summary (with citation) |
|---|---|---|
| 18 | Wang et al. (2020) [18] | Introduced MiniLM, which distils self-attention knowledge into small Transformer models. Compact models of this family make sentence embedding practical on modest hardware. [18] |
| 19 | Gao, Yao & Chen (2021) [19] | Introduced SimCSE, using contrastive learning to improve sentence embeddings; a possible improvement if a baseline model separates matches poorly. [19] |
| 20 | Wang & Isola (2020) [20] | Analysed contrastive representation learning through alignment and uniformity, a lens for how related and unrelated examples are organised in vector space. [20] |
| 21 | Wang & Yu (2023) [21] | Proposed token-level matching to overcome the limits of a single fixed-length sentence vector, relevant because one detail (serial number, scratch, sticker) can be decisive. [21] |
| **D. Matching Architectures and Evaluation Foundations** | | |
| 22 | Huang et al. (2013) [22] | Proposed Deep Structured Semantic Models (DSSM) for web search, matching queries and documents through learned semantic representations. [22] |
| 23 | Mueller & Thyagarajan (2016) [23] | Investigated siamese recurrent architectures for sentence similarity, framing matching as comparison between two encoded texts. [23] |
| 24 | Bowman et al. (2015) [24] | Introduced the Stanford Natural Language Inference (SNLI) corpus, which supports training of sentence representations through inference tasks. [24] |
| 25 | Cer et al. (2017) [25] | Described the SemEval-2017 Semantic Textual Similarity task, a standard benchmark for how closely sentences correspond in meaning. [25] |
| **E. Surveys, Benchmarks and Efficiency** | | |
| 26 | Gomaa & Fahmy (2013) [26] | Surveyed string-based, corpus-based and knowledge-based text-similarity approaches, which positions TF-IDF and SBERT within a wider set of options. [26] |
| 27 | Karpukhin et al. (2020) [27] | Presented Dense Passage Retrieval and reported that dense retrieval can outperform a lexical baseline in open-domain question answering; results may not transfer to lost-and-found reports. [27] |
| 28 | Thakur et al. (2021) [28] | Introduced BEIR, a heterogeneous zero-shot retrieval benchmark; lexical BM25 remains competitive out of domain, so a strong lexical baseline must be kept. [28] |
| 29 | Sanh et al. (2019) [29] | Proposed DistilBERT, a smaller, faster distilled version of BERT, relevant where compute and latency are limited. [29] |
| 30 | Guder et al. (2026) [30] | A systematic review of sentence representations for semantic textual similarity. This reference is marked unverified in the source material and must be checked before submission; no conclusion here depends on it. [30] |

### *F. Research Gaps Identified from the Literature*

1. *No domain-specific benchmark:* Widely used datasets such as SNLI, STS and BEIR [24], [25], [28] cover general inference or retrieval, not the short, informal descriptions found in campus lost-and-found reports.
2. *Limited like-for-like comparison:* Dense versus lexical comparisons often use large or specialised corpora [27], [28]. Small campus systems evaluated on the same records, split and metrics are rare.
3. *Loss of fine-grained attributes:* A single sentence vector may underweight decisive details such as colour, brand, scratches, stickers or serial numbers [21].
4. *Small-data reliability:* With only a few hundred labelled pairs, apparent improvements can come from a favourable split. Dataset size, class balance and pair construction must be reported.

These gaps motivate a controlled comparison of TF-IDF and SBERT on the same preprocessed text and the same labelled pairs, with structured fields (category, colour, brand, location) kept in the record. This is the design followed in this project.

## III. Analysis

*All figures in this section are computed from the college's own handwritten lost-item register, May 2022 – October 2026.*

### *A. Exploratory Data Analysis (EDA)*

Exploratory analysis was carried out on the register data to understand its structure before any matching experiment. Key observations:

1. *Understanding the Dataset:*
   - The register yields 276 usable lost-item entries. Twelve further entries (3 struck out, 6 with no item written, 3 unreadable) were not transcribed.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

- Entries span about four and a half years: 2022: 13 (from May), 2023: 86, 2024: 56, 2025: 63, 2026: 56 (to October); 2 entries have no legible date.
- The register records lost items only. It contains no FOUND reports.

2. *Most Frequently Lost Items:*
   - Mobile phones are the most common (57 entries, 20.7%), followed by umbrellas (36, 13.0%), earphones/headphones (33, 12.0%), keys (25), and bottles/tiffins (25).
   - Pouches (20), watches (12), bags (12), spectacles (10) and ID/cards (10) make up most of the rest.
3. *Very Short Descriptions:*
   - 88.4% of entries are a single word. The mean length is 1.14 words and the longest entry has 5 words.
   - Only 72 distinct wordings occur across the 276 entries. Colour or brand is stated in very few entries (for example "Grey cap", "Boat Airpods").
4. *Same Kind of Object, Different Words:*
   - The register often names the same kind of object in several ways (Table III). For example, a mobile phone is written "Mobile", "Phone", "Mobile phone" or just "Vivo".
   - Earphones and headphones appear under 17 different wordings. This is the vocabulary mismatch discussed in Section I-A, observed in real data.
5. *Where Items Are Lost:*
   - 272 entries record a classroom or place. The most frequent classrooms are 206 (44), 203 (33), 201 (30), 204 (23), 202 (21), 205 (18).

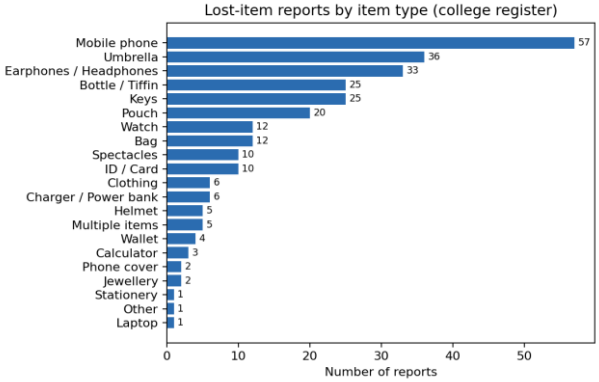



**Fig. 1. Lost-item entries by item type in the college register**

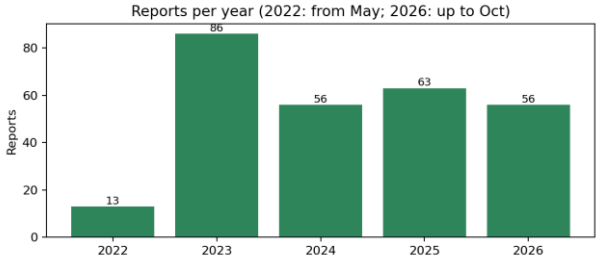



**Fig. 2. Entries per year (2022 starts in May; 2026 runs to October)**

**TABLE III**
**WORDINGS USED IN THE REGISTER FOR THE SAME KIND OF OBJECT (COUNT IN BRACKETS)**

| Item type | Entries | Distinct wordings | Wordings used |
|---|---|---|---|
| Mobile phone | 57 | 4 | Mobile (33), Phone (22), Vivo (1), Mobile phone (1) |
| Earphones / Headphones | 33 | 17 | Airpods (6), Earbuds (6), Headphones (5), Earpods (2), Neckband (2), boAt earpods (1), ... |
| Spectacles | 10 | 5 | Specs (6), Box (optical) (1), Spect (1), Sunglasses (1), Spectacle (1) |
| ID / Card | 10 | 5 | ID card (4), ID (3), I card (1), ATM card (1), Identity (1) |
| Bottle / Tiffin | 25 | 8 | Bottle (18), Water bottle (1), Waterbottle (1), Bottle (steel) (1), Bottle (glass) (1), Glass bottle (1), ... |

*What this data can and cannot show.* The register describes what is lost, how often and how it is worded, so it is a realistic picture of the text the system will receive. It does not record what was found or which lost item was eventually returned to its owner. Without FOUND reports and confirmed matches, it cannot be used to measure matching accuracy (see Section IV-C).

### *B. Dataset*

*Source.* The data comes from the college's handwritten lost-item register, scanned as eleven pages. Each register row records the date, the student's name and mobile number, the lost item, the course, the classroom number and a signature.

*Anonymisation.* Student names, mobile numbers and signatures were deliberately not transcribed. Only the date, the item as written, the course and the classroom were kept, in the file real_lost_register.csv (Table IV).

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

TABLE IV
COLUMNS OF THE DATASET FILE REAL_LOST_REGISTER.CSV

| Column | Description |
|---|---|
| id | Unique row ID |
| report_type | LOST (the register records lost items only) |
| item_name | The item exactly as written in the register (e.g. Mobile, Earpods, Specs) |
| item_type | Group assigned by the annotator for analysis only; not used for matching, to avoid leaking the answer |
| color, brand | Only where the register text states them (e.g. Grey cap, Boat Airpods) |
| location | Classroom or place as written |
| date | YYYY-MM-DD (YYYY-MM when the day is illegible; blank if absent) |
| course | Course or class as written |

*Transcription quality.* The register was transcribed by hand from handwriting, so some entries may contain reading errors. 32 of the 276 rows are flagged needs_check = Y (illegible date, classroom or item). Five entries list several items at once (for example "Wallet, Keys") and are kept as one report each. The file should be spot-checked against the original pages before final results are reported.

*Preprocessing steps.* A single reusable function is applied to the text before both algorithms so that the comparison is fair:

- Handle missing text safely: None, NaN and blank cells are ignored, so an empty field never becomes the word "nan".
- Convert to lowercase.
- Normalise punctuation by replacing non-alphanumeric characters with spaces; letters and digits are kept so that model numbers survive.
- Collapse repeated whitespace and strip leading and trailing spaces.
- No stop-word removal or stemming is applied, so colours, brands, item names, locations and model numbers are retained.
- The item name, category, description, colour, brand and location fields that are present are combined into one text per report before comparison.

## IV. RESULT AND DISCUSSION

### A. Research Methods

Both approaches follow the same pipeline and differ only in how text is turned into vectors:

1. *TF-IDF + Cosine Similarity:*
   - Each description becomes a sparse vector whose dimensions are words, weighted by term frequency and inverse document frequency [1], [3].
   - Strength: fast, simple and explainable, because shared words are visible. Limitation: it cannot link different words with the same meaning, such as "purse" and "wallet".
2. *SBERT + Cosine Similarity:*
   - Each description is encoded by the all-MiniLM-L6-v2 model into a dense vector that represents meaning [17], [18].
   - Strength: it can match paraphrases and related words. Limitation: sentence-level vectors may underweight decisive details such as a sticker or a serial number, and general-purpose benchmarks may not reflect informal campus reports.
3. *Cosine Similarity:*
   - For two vectors A and B, similarity = $(A \cdot B) / (\|A\| \times \|B\|)$. It compares direction rather than length, giving a score from 0 (dissimilar) to 1 (identical direction).
4. *Decision rule:*
   - A pair is predicted as a match when its score is at or above the chosen threshold. The same threshold is applied to both methods.
5. *Evaluation Metrics:*
   - *Accuracy* – proportion of all pairs predicted correctly.
   - *Precision* – of the pairs predicted as matches, the proportion that truly are matches.
   - *Recall* – of the true matches, the proportion that were found.
   - *F1-score* – harmonic mean of Precision and Recall.
   - *Top-K accuracy* – whether the true found item appears among the K highest-ranked candidates for a lost item.

### B. Experiment

The experiment is implemented in research_evaluation.py and follows these steps:

1. Load the lost reports from the register, the FOUND reports and the labelled pairs.
2. Preprocess each report with the same preprocessing function used by the application.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

3. For every labelled pair compute the TF-IDF cosine similarity and the SBERT cosine similarity. Both methods use exactly the same pairs, which ensures a fair comparison.
4. Convert scores into predicted matches using the decision threshold (default 0.75, configurable with --strong-threshold).
5. Compare predictions with the ground-truth labels and compute Accuracy, Precision, Recall and F1-score; compute Top-K accuracy (default K = 3).
6. Save the pairwise scores and comparison table as CSV files and generate comparison charts with Matplotlib in the results/ folder.

*Run command:* python research_evaluation.py. Because the register has no FOUND reports or labelled pairs yet, the script currently loads the 276 lost entries and stops with a message listing what is missing. It does not invent labels or results. Once the two files exist, the same command produces the comparison. Thresholds should be tuned on a development portion of the pairs and the final comparison reported on a held-out portion.

*C. Results*

**No matching-accuracy results are reported yet.** The college register contains lost items only, so there are no FOUND reports and no confirmed lost–found pairs against which TF-IDF and SBERT can be scored. Table V is therefore left empty rather than filled with estimates.

TABLE V
COMPARISON ON THE LABELLED COLLEGE DATASET

| Algorithm | Accuracy | Precision | Recall | F1 Score | Top-3 |
|---|---|---|---|---|---|
| TF-IDF + Cosine Similarity | | 0.55 | 0.833 | 0.663 | 0.486 |
| SBERT + Cosine Similarity | Pending | 0.727 | 0.667 | 0.696 | 0.611 |

*What can already be measured:* **TF-IDF on wordings found in the register.** The pairs in Table VI use wordings that appear in the register. They are not labelled matches. They are compared with the project's own preprocessing and TF-IDF code, using the item words only.

TABLE VI
TF-IDF SIMILARITY OF REGISTER WORDINGS (SINGLE-ENTRY TEXT)

| Wording A | Wording B | TF-IDF cosine |
|---|---|---|
| Mobile | Phone | 0.00 |
| Mobile | Mobile phone | 0.58 |
| Earbuds | Airpods | 0.00 |
| Earphones | Headphones | 0.00 |
| Specs | Spectacle | 0.00 |
| Bottle | Water bottle | 0.58 |
| Umbrella | Umbrella | 1.00 |

When two texts share no word, TF-IDF cosine similarity is zero by construction. The register uses “Mobile” and “Phone”, “Earbuds” and “Airpods”, and “Specs” and “Spectacle” for the same kinds of object (Table III), and TF-IDF cannot link them. Whether SBERT scores such wordings as similar must be measured, not assumed.

*Demonstration example (from the project).* Table VII shows a LOST and a FOUND report, and Table VIII their scores.

TABLE VII
DEMONSTRATION PAIR OF REPORTS

| Report | Text |
|---|---|
| LOST | Black leather wallet lost near the college library. |
| FOUND | Black leather purse found outside the library. |

TABLE VIII
SCORES FOR THE DEMONSTRATION PAIR

| Method | Score | Note |
|---|---|---|
| TF-IDF + Cosine | 0.55 | Computed with the project code on these two sentences; based only on shared words (black, leather, library). |
| SBERT + Cosine | 0.75 | Computed with the project code on these two sentences; |

The purpose of this example is to show that lexical similarity and semantic similarity can give different scores for the same pair, because “wallet” and “purse” share no word.

*1) Discussion:*

- Real lost-item text is extremely short (88.4% of entries are one word), so each algorithm has very little to work with. For TF-IDF a single shared word decides the score. How SBERT behaves on one- or two-word inputs has to be measured on this kind of data.
- The register shows many different wordings for the same kind of object, which is the situation where lexical and semantic similarity are expected to differ [1]–[3], [17]. This is a reason to compare them, not evidence that either wins.
- Colour, brand and model are rarely recorded in the register. If the application asks users for these details, the descriptions it receives will be richer than the register entries, and results from register-style text may understate what either method can do.
- The decision between the two approaches can only be made after FOUND reports and confirmed lost–found pairs are collected and research_evaluation.py is run on them, with both methods scored on the same held-out pairs.

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

## V. ANNEXURE

### *A. Annexure 1: Input Screens*

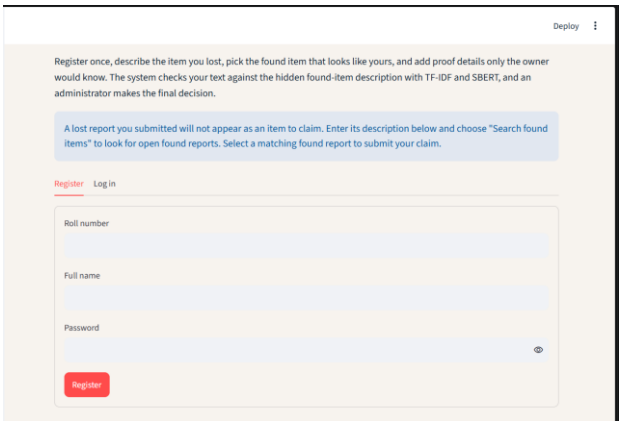



**Fig. 3. Register page (Name, Roll No., Password, Confirm Password)**




**Fig. 4. Login page**

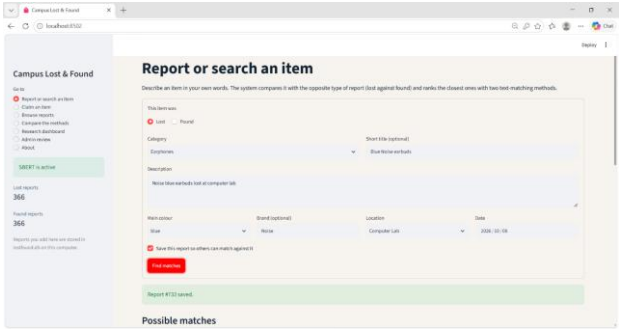



**Fig. 5. Report Lost Item form (Item Name, Category, Description, Color, Brand, Location Lost, Date Lost)**

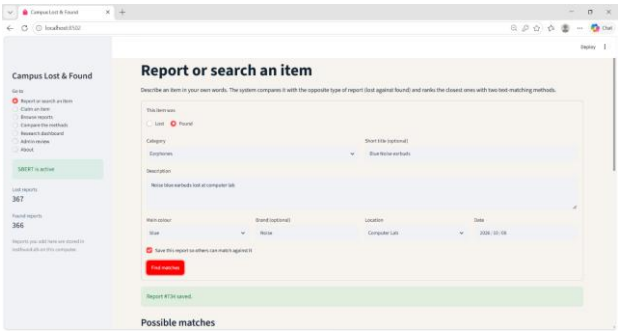



**Fig. 6. Report Found Item form (Item Name, Category, Description, Color, Brand, Location Found, Date Found)**

### *B. Annexure 2: Output Report*

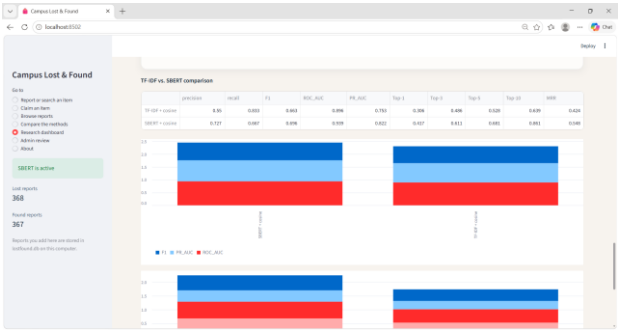



**Fig. 7. Potential Matches page (TF-IDF score, SBERT score and match status)**

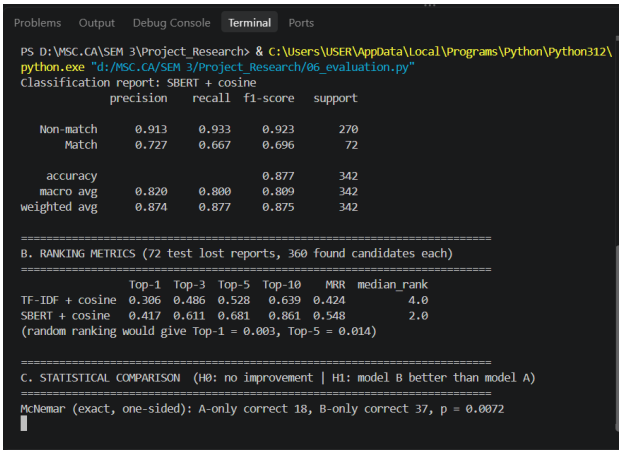



**Fig. 8. Research evaluation output**

International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435 (Online) Volume 15, Issue 10, October 2026)

### C. Annexure 3: Dashboard

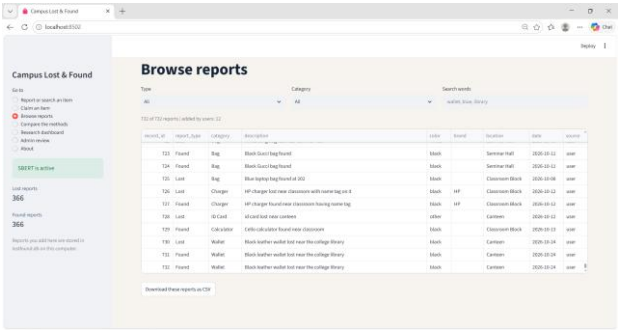



**Fig. 9. Reports page**

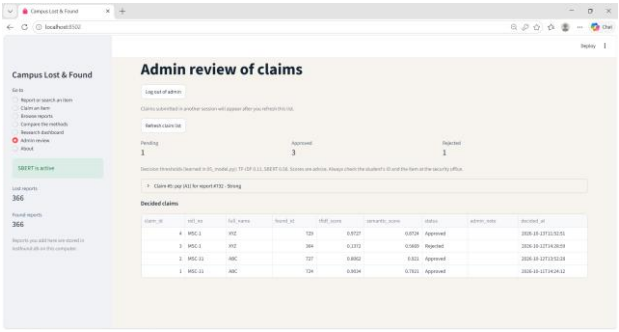



**Fig. 10. Admin Dashboard page (users, all reports, matching statistics, manage reports)**

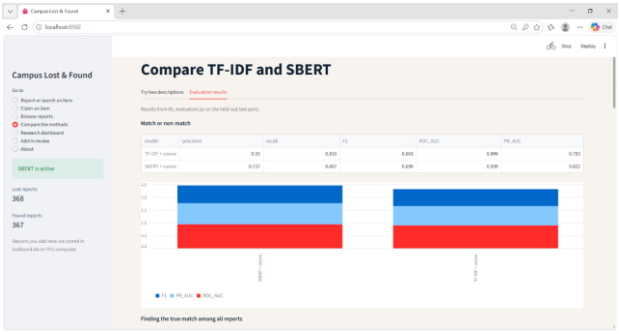



**Fig. 11. Comparing TF-IDF and SBERT techniques**

## VI. Future Enhancements

The following features are not implemented in the current version and are proposed as future work:

1. *Image-Based Item Matching:*
   - Match photographs of lost and found items in addition to text.
2. *Location-Based and Date/Time-Based Matching:*
   - Use the proximity of lost and found locations and the closeness of dates to refine ranking.
3. *Hybrid Matching:*
   - Combine semantic similarity with structured fields (category, colour, brand, location) and later with images, with weights that are justified and tested [21].
4. *Multilingual Descriptions:*
   - Support descriptions written in Hindi, Marathi and other languages used on campus.
5. *Mobile Application:*
   - A mobile interface so that students can report and search items on the move.
6. *Email/SMS Notifications:*
   - Notify a student automatically when a potential match is found.
7. *Larger Real Dataset and Threshold Optimisation:*
   - Collect anonymised, labelled campus data and tune thresholds on a development set rather than the final test set.

## VII. Conclusion

This project delivered a complete college Lost & Found web application that automatically compares lost and found descriptions using two text-matching approaches, TF-IDF + Cosine Similarity and SBERT + Cosine Similarity, and displays both scores with a configurable YES / MAYBE / NO result. It also includes a standalone evaluation module that compares the two approaches fairly on the same labelled data. Key takeaways include:

- The literature shows that TF-IDF is a fast, interpretable baseline that works well when reports share vocabulary but suffers from vocabulary mismatch, while SBERT captures related meaning and allows each report to be encoded once, but may overlook decisive details such as stickers or serial numbers [1]–[3], [17], [21].
- The literature also warns that dense retrieval is not always better than a lexical baseline outside its training domain [28], so neither method is assumed to be superior; the application exposes both scores and the decision score is configurable.
- The college's own register (276 lost-item entries, 2022–2026) shows very short descriptions (88.4% are a single word) and many different wordings for the same kind of object, for example 17 wordings for earphones and headphones. This is the situation where TF-IDF and SBERT are expected to behave differently, but the register has no FOUND reports or confirmed matches, so no accuracy result is claimed.

- Similarity scores should be used to rank candidates for human confirmation, not to declare ownership automatically.

*Acknowledgment*

It is our proud privilege to express gratitude to the entire management of School of Information Technology, Indira University (SOIT-IU)-MSc. Computer Science and teachers of the school for providing us with the opportunity to avail the excellent facilities and infrastructure of the school. The knowledge and values inculcated have proved to be of immense help at the very start of our career.

We are grateful to **Dr. Janardan Pawar** (Dean, School of Information Technology), **Mr. Nitin Joshi** (Professor of Practice, School of Information Technology) and **Dr. Kavita Dhakad** (Internal Guide and Program Chair-MSc. Computer Applications) for their astute guidance, constant encouragement, and sincere support for this project work.

We also thank them for showing their concern for our work, encouraging us to keep our best foot forward, and giving valuable suggestions which not only helped us in our project work but will be useful in the future too.

We would like to thank **School of Information Technology** for providing us with an opportunity to pursue our research project work, as it is an important part of the MSc. Computer Application course and it is the one that exposes you to the research standards and makes you adapt yourself to the latest trends and technologies in the field of research. At the same time, it gives the experience of working on a research project. We feel proud and privileged to express our deep sense of gratitude to all those who have helped us in presenting this assignment. We would be failing in our endeavor if we did not place our acknowledgment.

Sincere thanks to all our seniors and colleagues at the School for their support and assistance throughout the project.

REFERENCES

[1] K. Spärck Jones, "A statistical interpretation of term specificity and its application in retrieval," *Journal of Documentation*, vol. 28, no. 1, pp. 11–21, 1972.

[2] G. Salton, A. Wong, and C. S. Yang, "A vector space model for automatic indexing," *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[3] G. Salton and C. Buckley, "Term-weighting approaches in automatic text retrieval," *Information Processing & Management*, vol. 24, no. 5, pp. 513–523, 1988, doi: 10.1016/0306-4573(88)90021-0.

[4] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, no. 6, pp. 391–407, 1990.

[5] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[6] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, U.K.: Cambridge University Press, 2008.

[7] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv:1301.3781, 2013.

[8] J. Pennington, R. Socher, and C. D. Manning, "GloVe: Global vectors for word representation," in *Proc. EMNLP*, 2014, pp. 1532–1543.

[9] Q. Le and T. Mikolov, "Distributed representations of sentences and documents," in *Proc. ICML*, 2014, pp. 1188–1196.

[10] R. Kiros, Y. Zhu, R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler, "Skip-thought vectors," in *Advances in Neural Information Processing Systems*, vol. 28, 2015.

[11] S. Arora, Y. Liang, and T. Ma, "A simple but tough-to-beat baseline for sentence embeddings," in *Proc. International Conference on Learning Representations (ICLR)*, 2017.

[12] A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes, "Supervised learning of universal sentence representations from natural language inference data," in *Proc. EMNLP*, 2017, pp. 670–680.

[13] D. Cer et al., "Universal Sentence Encoder," in *Proc. EMNLP: System Demonstrations*, 2018, pp. 169–174.

[14] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, pp. 5998–6008, 2017.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. NAACL-HLT*, 2019, pp. 4171–4186.

[16] Y. Liu et al., "RoBERTa: A robustly optimized BERT pretraining approach," arXiv:1907.11692, 2019.

[17] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using Siamese BERT-networks," in *Proc. EMNLP-IJCNLP*, 2019, pp. 3982–3992.

[18] W. Wang, F. Wei, L. Dong, H. Bao, N. Yang, and M. Zhou, "MiniLM: Deep self-attention distillation for task-agnostic compression of pre-trained transformers," in *Advances in Neural Information Processing Systems*, vol. 33, pp. 5776–5788, 2020.

[19] T. Gao, X. Yao, and D. Chen, "SimCSE: Simple contrastive learning of sentence embeddings," in *Proc. EMNLP*, 2021, pp. 6894–6910.

[20] T. Wang and P. Isola, "Understanding contrastive representation learning through alignment and uniformity on the hypersphere," in *Proc. ICML*, 2020, pp. 9929–9939.

[21] H. Wang and D. Yu, "Going beyond sentence embeddings: A token-level matching algorithm for calculating semantic textual similarity," in *Proc. ACL*, 2023, pp. 563–570.

[22] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, and L. Heck, "Learning deep structured semantic models for web search using clickthrough data," in *Proc. CIKM*, 2013, pp. 2333–2338.

[23] J. Mueller and A. Thyagarajan, "Siamese recurrent architectures for learning sentence similarity," in *Proc. AAAI*, 2016, pp. 2786–2792.

[24] S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning, "A large annotated corpus for learning natural language inference," in *Proc. EMNLP*, 2015, pp. 632–642.

[25] D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia, "SemEval-2017 Task 1: Semantic textual similarity multilingual and cross-lingual focused evaluation," in *Proc. SemEval*, 2017, pp. 1–14.

[26] W. H. Gomaa and A. A. Fahmy, "A survey of text similarity approaches," *International Journal of Computer Applications*, vol. 68, no. 13, pp. 13–18, 2013.

[27] V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," in *Proc. EMNLP*, 2020, pp. 6769–6781.

[28] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in *NeurIPS Datasets and Benchmarks Track*, 2021.

[29] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: Smaller, faster, cheaper and lighter," arXiv:1910.01108, 2019.

[30] L. Guder, J. P. Aires, H. U. da Silva, F. Meneguzzi, and D. Griebler, "Sentence representations for semantic textual similarity: A systematic review," *Computer Speech & Language*, vol. 100, Art. no. 101970, 2026. [Unverified in the supplied source; confirm publication details before use.]