# COFINDER: MULTIMEDIA CONTEXT LOCATOR WITH LEXICAL SEARCH

**Fabio Vivarelli[1], Ana Paula Abrantes de Castro e Shiguemori[2], Cristiane Oliveira Campos Gonella[2]**

*[1]Federal Institute of Education, Science and Technology of São Paulo, Jacareí, Brazil (fabio.vivarelli@aluno.ifsp.edu.br)*
*[2]Federal Institute of Education, Science and Technology of São Paulo, Jacareí, Brazil*

*Abstract:* This work presents the implemented lexical layer of Cofinder, a multimedia context locator for subtitles, transcripts and text. The prototype ingests and normalizes SubRip Subtitle (SRT), Web Video Text Track (VTT), plain text (TXT) and ASR-generated SRT, then ranks segments with an explainable Term Frequency-Inverse Document Frequency (TF-IDF) and fuzzy score based on Jaccard and Levenshtein similarity.

*Keywords*: information retrieval; TF-IDF; fuzzy search; Jaccard similarity; multimedia context

## INTRODUCTION

Long videos, lectures, interviews, audio recordings and textual corpora create a practical information retrieval challenge: users often remember an approximate idea, expression, scene or fragment, rather than the exact words used in the source. Traditional exact search is fast and transparent, but it fails when a term is misspelled, inflected differently, transcribed with noise or embedded inside a longer segment (Manning et al., 2008).

In information retrieval, TF-IDF remains a transparent baseline for estimating the importance of terms across a corpus (Sparck Jones, 1972; Salton and Buckley, 1988; Manning et al., 2008). However, a multimedia context locator also requires tolerance to partial matches. For this reason, Cofinder combines TF-IDF with fuzzy lexical similarity, using token overlap and edit-based similarity to rank candidate segments in an auditable way.

Cofinder is proposed as a context locator for long media. Given a short query, it aims to retrieve relevant segments with timestamps or line markers from SRT, VTT, TXT and transcribed audio. This expanded abstract focuses on the part already implemented: ingestion and normalization, ASR-supported conversion to SRT, and a lexical ranking module based on TF-IDF, Jaccard similarity and Levenshtein-style fuzzy matching. Sentence embeddings and PMI/PPMI expansion are planned as later modules (Church and Hanks, 1990; Reimers and Gurevych, 2019).

## MATERIALS AND METHODS

The current pipeline accepts subtitle and text files directly and media files through an automatic speech recognition (ASR) stage. SRT and VTT inputs are parsed into temporal segments; TXT inputs are split into textual units; and audio inputs can be transcribed into SRT using a Whisper-style ASR model (Radford et al., 2022). Each segment preserves its original text and metadata, such as source file, timestamp or line identifier. The normalized representation is used only for indexing and scoring, while the original text is preserved for presentation to the user.

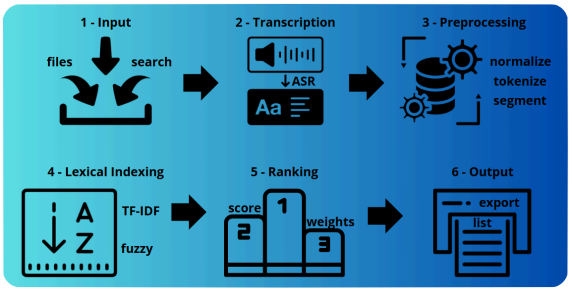



Figure 1. Current lexical retrieval pipeline.

Source: Authors' own work.

Figure 1 summarizes the current lexical retrieval pipeline implemented in Cofinder. After ingestion and optional ASR transcription, the data are normalized, tokenized and segmented, then indexed through a lexical module based on TF-IDF and fuzzy matching. The resulting segments are ranked and returned with their associated timestamps or line metadata. The lexical index uses unigrams and

bigrams. Bigrams are included because many relevant queries are short expressions, such as compound technical terms, and should not be treated only as isolated words.

The TF-IDF formulation follows classical information retrieval models (Sparck Jones, 1972; Salton and Buckley, 1988; Manning et al., 2008). The term frequency component follows a sublinear formulation, reducing the dominance of repeated terms. For a term t in a segment d, with frequency f(t,d), the term frequency is defined as:

$$\mathrm{TF}(t,d) = \begin{cases} 1 + \ln(f_{t,d}), & \text{if } f_{t,d} > 0 \\ 0, & \text{if } f_{t,d} = 0 \end{cases} \quad (1)$$

Before indexing, the system keeps two parallel representations of each segment. The display representation preserves the original subtitle or transcript text, which is important for user inspection. The normalized representation is used for retrieval and applies lowercasing, punctuation handling and token extraction. This separation avoids modifying the material shown to the user while still reducing noise during ranking.

The inverse document frequency gives greater weight to terms that occur in fewer segments. Using smoothed IDF, where N is the total number of segments and DF(t) is the number of segments containing term t, the formulation is:

$$\mathrm{IDF}(t) = \ln\left(\frac{1+N}{1+\mathrm{DF}(t)}\right) + 1 \quad (2)$$

Following the standard vector-space model for information retrieval, TF-IDF weighting, L2 normalization and cosine similarity are used to represent and compare queries and segments (Salton and Buckley, 1988; Manning et al., 2008).

After computing term frequency and inverse document frequency, each term receives a TF-IDF weight. This weight represents how relevant a term is to a specific segment while also considering how rare it is across the full corpus:

$$w_{t,d} = \mathrm{TF}(t,d) \cdot \mathrm{IDF}(t) \quad (3)$$

In Equation (3), w_t,d is the TF-IDF weight of term t in segment d. A term receives a higher weight when it occurs in the segment but is not frequent across all other segments. Since longer segments may naturally contain more terms and therefore larger raw vectors, L2 normalization is applied to reduce the advantage of segment length and make comparison depend more on vector direction than magnitude:

$$\vec{v}_d = \frac{\vec{w}_d}{\|\vec{w}_d\|_2} = \frac{\vec{w}_d}{\sqrt{\sum_{t \in V} w_{t,d}^2}} \quad (4)$$

In Equation (4), w_d is the original TF-IDF vector of segment d, V is the vocabulary, and v_d is the normalized vector. After this normalization, the vector has unit length. The lexical relevance between the query and each segment is then computed through the dot product between their normalized vectors, which is equivalent to cosine similarity:

$$\mathrm{TFIDF}(q,d) = \vec{v}_q \cdot \vec{v}_d \quad (5)$$

Therefore, TFIDF(q,d) measures how close the query q and segment d are in the TF-IDF vector space. Fuzzy matching complements TF-IDF with two approximate lexical signals. First, Jaccard similarity measures token overlap between the query token set Q and the segment token set D (Jaccard, 1901):

$$J(Q,D) = \frac{|Q \cap D|}{|Q \cup D|} \quad (6)$$

Second, Levenshtein-style similarity estimates closeness by the minimum number of insertions, removals and substitutions required to transform one string into another (Levenshtein, 1966). In normalized form:

$$\mathrm{LevSim}(q,d) = 1 - \frac{d_{\mathrm{lev}}(q,d)}{\max(|q|,|d|)} \quad (7)$$

In the prototype, the edit-based component can also be implemented through a token-based fuzzy ratio when available, because subtitle segments and short queries frequently contain extra words, punctuation differences or changes in word order. Therefore, the fuzzy module combines token overlap and edit-based similarity before the final lexical ranking step:

$$\mathrm{Fuzzy}(q,d) = 0.55 \cdot J + 0.45 \cdot \mathrm{LevSim} \quad (8)$$

In Equation (8), J abbreviates J(Q,D), and LevSim abbreviates LevSim(q,d). The fuzzy score gives slightly more weight to Jaccard similarity, favoring segments that share actual query tokens, while still preserving the contribution of Levenshtein-style similarity for spelling variation, plural forms, punctuation changes and ASR transcription artifacts.

After computing the TF-IDF and fuzzy components, the final lexical ranking score is obtained by combining both signals:

$$\mathrm{Score}(q,d) = 0.70 \cdot \mathrm{TFIDF} + 0.30 \cdot \mathrm{Fuzzy} \quad (9)$$

In Equation (9), TFIDF abbreviates TFIDF(q,d), and Fuzzy abbreviates Fuzzy(q,d). The weights used in the current prototype define TF-IDF as the main ranking signal and fuzzy similarity as a corrective signal. These weights are initial design choices and

will be adjusted through validation experiments in later stages.

The methodology deliberately avoids treating fuzzy matching as a black-box correction. Jaccard and Levenshtein-style similarity capture different aspects of approximation: Jaccard is discrete and token-oriented, whereas edit similarity is character- or token-sequence-oriented. Keeping them separate before fusion allows future ablation tests to measure which type of approximation is more useful for subtitle files, long transcripts and ASR-generated segments.

## RESULTS AND DISCUSSION

The current prototype implements the initial technical goals of the project as a reproducible lexical baseline. It can ingest SRT, VTT and TXT files, generate searchable SRT output from audio, normalize segments and rank them using the lexical-fuzzy score. Returned results preserve source information, timestamps or line metadata, original segment text and partial score values, supporting auditability of the retrieved passages.

The implemented components address complementary retrieval conditions. TF-IDF favors terms that are relevant in a segment but not overly frequent across the corpus. Jaccard similarity contributes to cases in which query terms occur inside longer segments or in different positions, while Levenshtein-style or token-based fuzzy similarity increases robustness to spelling variation, inflectional differences, punctuation changes and minor ASR transcription artifacts. This combination is especially relevant for subtitle and transcript retrieval, where exact keyword matching may be too restrictive.

The main contribution at this stage is the establishment of an explainable lexical baseline. Instead of returning ranked segments as opaque matches, the prototype allows the retrieval process to be interpreted through its components: corpus-based term relevance, token overlap and approximate textual similarity. This structure is useful both for user-facing explanation and for later experimental analysis, since each component can be isolated in ablation tests.

The current limitation is semantic mismatch. When a query and a relevant segment express the same idea using substantially different vocabulary, lexical-fuzzy retrieval may still fail. This limitation motivates the planned integration of sentence embeddings and PMI/PPMI co-occurrence expansion, which are expected to improve conceptual and corpus-specific retrieval. No final effectiveness claims are made at this stage. The next evaluation step will use manually annotated queries and metrics such as Recall@K, MRR and nDCG to compare exact search, TF-IDF alone, fuzzy components and the combined lexical score.

## CONCLUSION

This expanded abstract presents the implemented lexical foundation of Cofinder: ingestion and normalization of subtitle/text sources, ASR-supported conversion to SRT, and explainable ranking based on TF-IDF, Jaccard and Levenshtein-style fuzzy similarity. By presenting the scoring equations, the work makes the current retrieval process explicit and reproducible. The implemented layer provides a transparent basis for locating relevant segments in long multimedia and textual sources. Future work will integrate semantic embeddings, PMI/PPMI expansion, a web interface and formal retrieval evaluation.

## ACKNOWLEDGMENTS

The authors thank the Federal Institute of São Paulo (IFSP) and the project advisor for providing institutional, methodological, and financial support.

## REFERENCES

CHURCH, K.; HANKS, P. Word association norms, mutual information, and lexicography. Computational Linguistics, v. 16, n. 1, p. 22-29, 1990.

JACCARD, P. Etude comparative de la distribution florale dans une portion des Alpes et des Jura. Bulletin de la Société Vaudoise des Sciences Naturelles, v. 37, p. 547-579, 1901.

LEVENSHTEIN, V. I. Binary codes capable of correcting deletions, insertions and reversals. Soviet Physics Doklady, v. 10, n. 8, p. 707-710, 1966.

MANNING, C. D.; RAGHAVAN, P.; SCHUTZE, H. Introduction to Information Retrieval. Cambridge: Cambridge University Press, 2008.

RADFORD, A. et al. Robust Speech Recognition via Large-Scale Weak Supervision. arXiv preprint, 2022.

REIMERS, N.; GUREVYCH, I. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. Proceedings of EMNLP 2019, 2019.

SALTON, G.; BUCKLEY, C. Term-weighting approaches in automatic text retrieval. Information Processing & Management, v. 24, n. 5, p. 513-523, 1988.

SPARCK JONES, K. A statistical interpretation of term specificity and its application in retrieval. Journal of Documentation, v. 28, n. 1, p. 11-21, 1972.