

Politecnico di Torino

Master of Science in Data Science and Engineering

Academic Year 2025/2026

Graduation Session, October 2026

An Expert-Finding System for Matching Academic Researchers to Funding Calls

A hybrid sparse–dense retrieval pipeline with neural re-ranking

Supervisors
Prof. Daniele Apiletti
Simone Monaco

Candidate
Alessandro Arneodo

Abstract

Finding the academic researchers that are best suited to answer a research funding call is a frequent and time-consuming task in large technical universities. This thesis presents an expert-finding system that receives the free text of a call and returns a ranked list of researchers of Politecnico di Torino, ordered by how relevant their scientific production is to the topic of the call. The system follows the document-centric formulation of expert finding: it first retrieves the most relevant publications with a hybrid pipeline that combines lexical retrieval (BM25) and dense retrieval (BGE-M3) through reciprocal rank fusion, with an optional cross-encoder re-ranking step; then it aggregates the relevance of the retrieved documents at the author level, using configurable weights for authorship credit and publication recency. The corpus contains more than 60,000 unique publications from the last fifteen years, extracted from the institutional repository and enriched with abstracts from OpenAlex, IRIS, and Scopus. The pipeline is available through an interactive web application, in which the user can write the query manually or generate a structured query from the call text with a large language model. Since no public benchmark exists for this task and corpus, the system is evaluated with a dedicated methodology built on parallel ground truths derived from institutional data. An ablation study with paired significance testing quantifies the contribution of each pipeline component, together with the limitations of the evaluation itself.

Contents

Abstract **3**

List of Figures **9**

List of Tables **11**

List of Acronyms **13**

1 Introduction **15**
1.1 Context and Motivation 15
1.2 Problem Statement 15
1.3 Proposed Approach 16
1.4 Contributions 16
1.5 Structure of the Thesis 17

2 Related Work **19**
2.1 Expert Finding and Expertise Retrieval 19
2.2 Lexical Retrieval 20
2.3 Dense and Learned Representations 20
2.4 Hybrid Retrieval, Rank Fusion, and Re-ranking 21
2.5 Large Language Models for Query Formulation 22

3 Background **23**
3.1 Overview of the Pipeline 23
3.2 BM25 and the Probabilistic Relevance Framework 24
3.2.1 From term specificity to a probabilistic ranking 24
3.2.2 The scoring function 25
3.2.3 Interpretation of the components 25
3.2.4 Strengths and limitations 26
3.3 FAISS and Nearest-Neighbour Search 26
3.3.1 The nearest-neighbour problem in dense retrieval 26
3.3.2 Exact search 26
3.3.3 Design choice 26
3.4 BGE-M3: Multilingual Neural Text Embeddings 27
3.4.1 From transformers to dense retrievers 27
3.4.2 A multilingual, multi-functional encoder 27
3.4.3 Role in this work 27

3.5 Reciprocal Rank Fusion . . . 28
3.5.1 The fusion problem . . . 28
3.5.2 Score-based fusion . . . 28
3.5.3 Rank-based fusion . . . 28
3.5.4 Weighted fusion in this system . . . 29
3.6 Cross-Encoder Re-ranking . . . 29
3.6.1 Bi-encoders versus cross-encoders . . . 29
3.6.2 The monoBERT relevance model . . . 30
3.6.3 Re-ranking in this system . . . 30

4 Methods **31**
4.1 Data and Corpus Construction . . . 31
4.1.1 Sources . . . 31
4.1.2 Abstract Enrichment . . . 31
4.1.3 De-duplication . . . 32
4.1.4 Document Representation . . . 32
4.1.5 Offline Indexing . . . 32
4.2 Query Formulation . . . 32
4.2.1 Structured Query Schema . . . 33
4.2.2 Prompt Design . . . 33
4.2.3 Query Normalisation and Sparse Expansion . . . 34
4.2.4 Model Serving . . . 34
4.3 Hybrid Retrieval and Fusion . . . 34
4.3.1 Lexical Retrieval . . . 34
4.3.2 Dense Retrieval . . . 35
4.3.3 Fusion . . . 35
4.4 Neural Re-ranking . . . 35
4.5 Author-Level Aggregation . . . 36
4.5.1 Authorship Credit . . . 36
4.5.2 Recency Weighting . . . 37
4.5.3 Score Accumulation and Output . . . 37
4.6 Implementation and Deployment . . . 37

5 Evaluation Methodology **39**
5.1 Evaluation Questions and Experimental Design . . . 39
5.2 Metrics and Statistical Testing . . . 40
5.3 Phase One: The Leave-One-Out Benchmark and the Ablation . . . 42
5.4 Phase Two: The Project Registry . . . 44
5.5 Threats to Validity . . . 46

6 Results and Discussion **49**
6.1 Experimental Setup . . . 49
6.2 How the Pipeline Was Designed . . . 51
6.2.1 Error attribution . . . 54
6.3 The System on the Real Task . . . 55
6.4 The Enriched Query . . . 56
6.4.1 Effect of the generated query . . . 57

6.4.2 Sensitivity to the generation backend . . . 57
6.5 Where the System Fails, and Why . . . 59
6.5.1 Corpus representation . . . 60
6.5.2 Language and funding instrument . . . 61
6.6 Discussion . . . 63
6.6.1 The re-ranking discrepancy . . . 64
6.6.2 Implications . . . 64
6.6.3 Limitations . . . 65

7 Conclusions and Future Work **67**
7.1 Summary of the Work . . . 67
7.2 Limitations . . . 68
7.3 Future Work . . . 69

A Query Generation Prompt **71**

B Evaluation Pipelines **73**
B.1 The Leave-One-Out Pipeline . . . 73
B.2 The Project-Registry Pipeline . . . 74

Bibliography **77**

List of Figures

3.1 Overview of the expert-finding pipeline at query time; the corpus and the two indices are built offline. 24

5.1 The two evaluation protocols on the same pipeline. 40

6.1 Outcome by funding instrument; each label reports the number of projects and the mean number of registered members per contract. . . 63

List of Tables

5.1 Design of the two experiments. 40
5.2 Evaluated configuration and default configuration of the application. . 46

6.1 Corpus, models, and evaluation sets. 50
6.2 Ablation of the design choices on 2,000 leave-one-out queries, each row against the evaluated configuration. Paired bootstrap on the per-query average precision, with Holm correction over the sixteen comparisons. 51
6.3 Design choices of Table 6.2 tested on the registry benchmark against their own baselines, with Holm correction over the thirteen decisional contrasts of Table B.1, twelve of which are shown. 54
6.4 Error attribution on 2,000 leave-one-out queries. "Success" is the fraction of queries with at least one reference author, "Recall" the mean fraction of the reference authors. 54
6.5 End-to-end evaluation on the 1,220 funded projects, with the corpus cut at the project start. The metrics are computed on the first ten positions; the conditions with a generated query are computed on 1,219 projects, because one generation failed. 55
6.6 Paired bootstrap against the raw-text condition on the per-query average precision. Holm correction over nineteen conditions (Table B.1), of which the table reports seven. 55
6.7 Effect of the query-generation stage by metric ($n = 1{,}219$), with Holm correction over these seven metrics and average precision. 57
6.8 Effect of the generation backend on the 1,218 projects common to both conditions. Δ is the difference in MAP@10 against the raw text, tested with the Holm correction over nineteen conditions (Table B.1). 58
6.9 Decomposition of the outcome on the registry benchmark ($n = 1{,}219$). 59
6.10 Position of the registered member in the untruncated ranking, on the 581 projects without a member in the top ten. 59
6.11 Documentary profile of each department and effectiveness on the projects its researchers sign. Each project is counted under the department of its first registered member, 1,218 of 1,219. 61
6.12 MAP@10 by proposal language on the registry benchmark. 61
6.13 Effectiveness by funding instrument, with the mean number of registered members per contract ($n = 1{,}219$). 63

B.1 Families over which the Holm correction is computed. 75

List of Acronyms

Acronym	Expansion
BGE-M3	BAAI General Embedding: Multi-Linguality, Multi-Functionality, Multi-Granularity
BM25	Best Matching 25 (probabilistic ranking function)
FAISS	Facebook AI Similarity Search
IR	Information Retrieval
IRIS	Institutional Research Information System (the institutional open-access repository)
LLM	Large Language Model
MAP	Mean Average Precision
MRR	Mean Reciprocal Rank
nDCG	Normalised Discounted Cumulative Gain
RRF	Reciprocal Rank Fusion
TREC	Text REtrieval Conference

Chapter 1

Introduction

1.1 Context and Motivation

Competition for research funding is intense, and a university often has little time to put the right team together. When a new call is published (for example a European programme such as Horizon Europe, a national grant, or an industrial tender), the research-support offices have a few days to identify the academics whose expertise is closest to its topic, who are the natural candidates to lead or join a proposal. In a large technical university such as Politecnico di Torino, with thousands of active researchers in dozens of disciplines, this matching is still done mostly by hand. The officers rely on personal knowledge, on contacts inside the departments, and on keyword searches over internal databases. The process is slow and hard to scale. It also favours the researchers who are already visible, while profiles that are equally relevant but less known are easily missed.

The task of finding the people who have a given competence is known in the information-retrieval literature as *expert finding* or *expertise retrieval* [1]. It is different from classical document retrieval, where the object of interest is a text. Expert finding ranks *people*, and the evidence of a person's expertise is indirect, because it must be derived from the documents associated with that person: publications, patents, projects, and so on. The literature distinguishes two main formulations. The *profile-centric* formulation builds an explicit textual profile for each candidate from the associated documents, and ranks the profiles by their similarity to the query. The *document-centric* formulation, instead, first retrieves the relevant documents and then transfers their relevance to the people who authored them [2]. The system described in this thesis adopts the second view: it retrieves the publications that are most relevant to a call, and then aggregates their relevance at the level of the single researchers.

1.2 Problem Statement

In concrete terms, the system addresses the following problem. Given the free-text description of a funding call, and given a corpus that contains the scientific production of an institution together with the profiles of its researchers, produce a ranked list of

researchers ordered by their topical relevance to the call, where each recommendation is justified by the specific publications that support it.

Three properties make this problem non-trivial. The input is *long, heterogeneous, and noisy*, since a call mixes administrative text with the few technical sentences that really carry the topical signal. The corpus is *multilingual*, because in an Italian university a significant part of the production is written in Italian while the international literature is in English, so a purely lexical matching is not sufficient. The relevant evidence, finally, is *distributed across documents and partially missing*. Many records do not have an abstract, the same publication can appear several times, and authorship must be credited in a fair way before the relevance can be attributed to a person.

1.3 Proposed Approach

This thesis presents an end-to-end expert-finding system that addresses the problem with a multi-stage retrieval pipeline, followed by an aggregation step at the author level. The query is first transformed into a structured representation, which can optionally be generated from the raw text of the call by a large language model. Candidate documents are then retrieved by a *hybrid* retriever, which runs in parallel a lexical model (BM25 [3]) and a dense multilingual encoder (BGE-M3 [4]), and merges the two rankings with reciprocal rank fusion [5]. An optional cross-encoder re-ranker (bge-reranker-v2-m3) refines the top candidates. Finally, the relevance of the documents is propagated to their authors through configurable schemes for authorship credit and recency weighting, producing a ranked list of experts together with their most relevant works.

1.4 Contributions

The main contributions of this work are the following:

- the design of a document-centric expert-finding pipeline for a multilingual institutional corpus, which combines lexical and dense retrieval, neural re-ranking, and author-level aggregation;
- the construction and curation of a corpus of more than 60,000 unique publications covering fifteen years, extracted from the institutional repository and enriched with abstracts from OpenAlex [6], the institutional IRIS archive, and Scopus;
- a configurable author-aggregation model that exposes the authorship-credit scheme (full, fractional, and a square-root compromise) and a temporal-recency weighting, so that the relevance of a researcher can be tuned to different evaluation needs;
- an evaluation methodology for institutional expert finding without a public benchmark, built on two ground truths derived from institutional data (the

authorship record and the register of funded projects); the methodology quantifies the contribution of each pipeline component and its effect on real funding proposals, together with its own limitations;

- the finding that the neural re-ranking stage, which obtains the largest gain of the ablation on publication abstracts, does not transfer to the operational input, also when compared at matched aggregation depth; this is a result about how a retrieval component should be validated, and not only about this pipeline;
- a decomposition of the residual failures, which separates the experts ranked too low from those never produced as candidates, and compares the outcome with the completeness of the institutional record department by department: a quarter of the publications carry no abstract, and the share of output that takes the form of an indexed article varies widely across disciplines;
- an interactive web application that allows research-support staff to query the system either with a manually written query or with a query generated from the call text by a large language model.

1.5 Structure of the Thesis

The rest of this document is organised as follows. Chapter 2 reviews the literature on expert finding, lexical and dense retrieval, hybrid retrieval with rank fusion, neural re-ranking, and the use of large language models for query formulation. Chapter 3 gives an overview of the pipeline and then presents in detail the retrieval models on which the system is built. Chapter 4 describes the data sources, the construction of the corpus, and the design of the retrieval and aggregation pipeline. Chapter 5 presents the evaluation methodology: the metrics, the construction of the ground truths, and the design of the two experiments, the ablation on the authorship record and the end-to-end evaluation on the funded projects. Chapter 6 reports and discusses the experimental results, and Chapter 7 draws the conclusions and outlines future work. The appendices report the prompt used to generate the structured query (Appendix A) and the steps of the two evaluation pipelines (Appendix B).

Chapter 2

Related Work

This chapter places the proposed system in the context of five research areas: expert finding (Section 2.1), lexical retrieval (Section 2.2), dense and learned representations (Section 2.3), hybrid retrieval with rank fusion and neural re-ranking (Section 2.4), and the use of large language models to formulate queries (Section 2.5).

2.1 Expert Finding and Expertise Retrieval

Expert finding emerged as a specific retrieval task inside the TREC Enterprise track [7], and received a principled probabilistic treatment from Balog et al. [2], who introduced the two canonical formulations that still structure the field. In the *candidate* (profile-centric) model, a language model is estimated for each person from the documents associated with that person, and the candidates are ranked by the probability of generating the query. In the *document* (document-centric) model, the documents are ranked first, and their relevance is then transferred to the associated people. The extensive survey by Balog et al. [1] consolidates these models, the association and aggregation strategies behind them, and their evaluation, and it remains the standard reference for this area.

The transfer of relevance from documents to people, which the document model leaves open, was formalised by Macdonald and Ounis [8] as a voting problem: each retrieved document votes for its authors, and data-fusion operators such as CombSUM and CombMAX turn the votes into a candidate score. The aggregation stage of the present system (Section 4.5) belongs to this family, and the combination rules compared in Section 6.2 are its main members. A second line of work adds structure to the association between people and documents, for example by propagating relevance across a graph of documents and candidates, or by profiling researchers through the entities mentioned in their publications. The present system keeps a purely textual association between documents and authors, and Section 7.3 discusses a competence graph of this kind, with its references, as an extension.

Evaluation in this area has relied on a small number of reference collections: the corpora of the TREC Enterprise track [7], built from the documents of a single organisation, and the UvT Expert Collection [9], built from the self-declared expertise of the staff of a Dutch university. None of them matches the present setting, an Italian–English corpus of publications queried with the text of a funding call, which

is why Chapter 5 builds its references from institutional records.

The same task appears under different names in different application domains. In community question answering, it is studied as the problem of routing questions to knowledgeable users. In academia, it is the basis of the closely related *reviewer assignment problem*, where submissions must be matched to reviewers whose past work demonstrates the required competence. Mimno and McCallum [10] treated reviewer matching as expertise modelling over the publications of the authors, and the problem still receives attention: recent work such as exHarmony [11] uses authorship and citation signals to benchmark reviewer–paper matching. These settings share the central feature of the present work, relevance is attributed to people through the documents they produced, and they motivate the document-centric design adopted here.

In the academic domain in particular, de Campos et al. [12] compare information-retrieval and machine-learning approaches to academic expert finding: they build multifaceted textual profiles from the publications of the researchers and compare them with classification-based methods. Their result that IR-based profile methods are in general robust across datasets supports the retrieval-first design followed in this thesis, and their attention to profile construction from scientific output corresponds to the corpus-centric setting of the present system.

2.2 Lexical Retrieval

The lexical component of the pipeline is based on BM25, the probabilistic ranking function formalised inside the probabilistic relevance framework by Robertson and Zaragoza [3]. BM25 scores a document by the frequency of the query terms inside it, reduced by term saturation and normalised by the document length. It is still a very strong and efficient first-stage retriever. Its main limitation is the *vocabulary mismatch* problem: it rewards exact term overlap, and it is therefore blind to synonyms and paraphrases. This is a serious issue in a multilingual corpus, where the query and the relevant documents may be written in different languages.

Rather than changing the lexical representation itself, the present system keeps classical BM25 and pairs it with a separate dense encoder, a design discussed in Section 2.4.

2.3 Dense and Learned Representations

The arrival of pre-trained transformer language models, in particular BERT [13], changed text representation and made retrieval in a learned semantic space effective. Reimers and Gurevych [14] showed with Sentence-BERT that a siamese fine-tuning objective produces sentence embeddings whose cosine similarity reflects semantic relatedness, which makes dense vector search practical. Karpukhin et al. [15] then demonstrated with Dense Passage Retrieval that a dual encoder trained on question–passage pairs can outperform BM25 on open-domain question answering, and dense retrieval became a first-class retrieval paradigm.

For scholarly text, document-level representations that exploit the citation graph have proven particularly effective. SPECTER [16] pre-trains a transformer on a citation-based relatedness signal, in order to embed scientific papers without task-specific fine-tuning, and supports recommendation and classification over academic documents. The encoder adopted in this thesis, BGE-M3 [4], is multilingual by design (it covers more than one hundred languages) and supports dense, multi-vector, and sparse retrieval over inputs of up to 8192 tokens. Its multilinguality is the key property for an Italian–English institutional corpus, because it places the documents written in the two languages in a shared semantic space.

Dense retrieval at the scale of a corpus requires efficient nearest-neighbour search. FAISS [17] is a library for efficient similarity search that scales to billions of vectors; in the present system, FAISS indexes the document embeddings and answers the dense queries with exact search (Section 3.3.2).

2.4 Hybrid Retrieval, Rank Fusion, and Re-ranking

Lexical and dense retrievers fail in complementary ways: BM25 misses semantic matches but guarantees exact-term recall, while dense models capture meaning but can miss rare and discriminative tokens, such as material names or acronyms. Combining the two, *hybrid retrieval*, tends therefore to perform better than each of them in isolation [4].

A robust way to merge two rankings is reciprocal rank fusion (RRF), proposed by Cormack et al. [5], which combines the result lists using only the *rank* of each document and a smoothing constant. In this way, it avoids the difficult problem of normalising score scales that are not comparable. RRF is the fusion mechanism used in this work. Since the optimal balance between the lexical and the dense signals depends on the query, recent work explores adaptive weighting: Hsu and Tzeng [18] propose Dynamic Alpha Tuning, which uses a language model to set the dense–sparse mixing weight for each query, instead of applying a fixed value. The present system exposes this balance as a tunable weight, and leaves the per-query adaptation as a natural extension.

Hybrid retrieval is usually followed by a precision-oriented *re-ranking* stage. Nogueira and Cho [19] showed that a BERT cross-encoder, which encodes the query and a candidate document together and scores their relevance directly, improves the ranking quality of first-stage retrievers in a substantial way. Cross-encoders are too expensive to run over an entire corpus, but they are well suited to re-score a small set of candidates; this thesis adopts the same retrieve-then-re-rank pattern, and applies a multilingual cross-encoder to the fused candidates.

2.5 Large Language Models for Query Formulation

The raw text of a funding call is a poor query: it is long, and most of it is not about the topic. Reformulating the input before retrieval is therefore central to the pipeline, and connects to a wide literature on the expansion of queries and documents and on query generation. HyDE [20] asks an instruction-following language model to generate a *hypothetical* document for a query, and then retrieves the real documents that are similar to it, improving zero-shot dense retrieval without relevance labels. The query generation used in this work follows the same intuition (a large language model transforms the call into a richer, retrieval-oriented representation), but it produces a structured query with separate lexical and semantic fields (Section 4.2.1), instead of a single hypothetical passage.

This use of language models for retrieval is related to the retrieval-augmented generation (RAG) paradigm introduced by Lewis et al. [21], in which a parametric generator is grounded on non-parametric retrieved evidence, and which has since grown into a large family of retrieval-augmented pipelines. The present system inverts the usual emphasis of RAG: instead of generating an answer from retrieved text, it uses the language model only to formulate the query, and it treats high-quality multilingual retrieval and faithful author-level aggregation as the final product.

Chapter 3

Background

After an overview of the pipeline (Section 3.1), this chapter examines in detail the retrieval components on which the system is built: the lexical model BM25 (Section 3.2), the nearest-neighbour search library FAISS (Section 3.3), the multilingual neural encoder BGE-M3 (Section 3.4), reciprocal rank fusion (Section 3.5), and cross-encoder re-ranking (Section 3.6). For each component, the underlying model is presented starting from its original publications and its main equations are stated, so that the design choices made in Chapter 4 can be understood from the foundations.

3.1 Overview of the Pipeline

Before the single models are presented, this section describes the pipeline as a whole, shown in Figure 3.1: what each stage does, what it receives, what it passes on, and where it is explained. The corpus and the two indices used for retrieval are built once, offline (Section 4.1); the stages of the figure run for every query.

Query formulation. The text of a call is long and mostly administrative. A large language model rewrites it into a structured query with two main parts: a *dense field*, a short text written like the abstract of the work of an ideal expert, and a *sparse field*, made of technical terms. The stage is described in Section 4.2.

Hybrid retrieval. The query is used to search the corpus in two ways at the same time. The first way looks for the documents that contain the terms of the sparse field, with BM25 (Section 3.2). The second way looks for the documents whose meaning is close to the dense field, also when they are written in another language: every document is represented as a vector by the BGE-M3 encoder (Section 3.4), and the closest vectors are found with the FAISS library (Section 3.3). The two ranked lists are then merged into a single ranking with reciprocal rank fusion (Section 3.5). The choices made in this system are given in Section 4.3.

Cross-encoder re-ranking. This optional stage takes the first documents of the merged ranking and reads each of them together with the dense field, to give a more precise relevance score. Its model differs from the encoder used in the retrieval: the encoder reads the query and the documents separately, which makes it fast enough

for the whole corpus, whereas the cross-encoder reads each query–document pair jointly, which is more precise but too slow for more than a few dozen documents (Sections 3.6 and 4.4).

Author-level aggregation. The last stage turns the ranked documents into a ranking of researchers: each researcher accumulates the scores of the retrieved documents they co-authored, weighted by the number of authors and by the age of the publication. The output is the list of experts, each with the publications that support the recommendation. The stage is specific to this system and is described in Section 4.5.

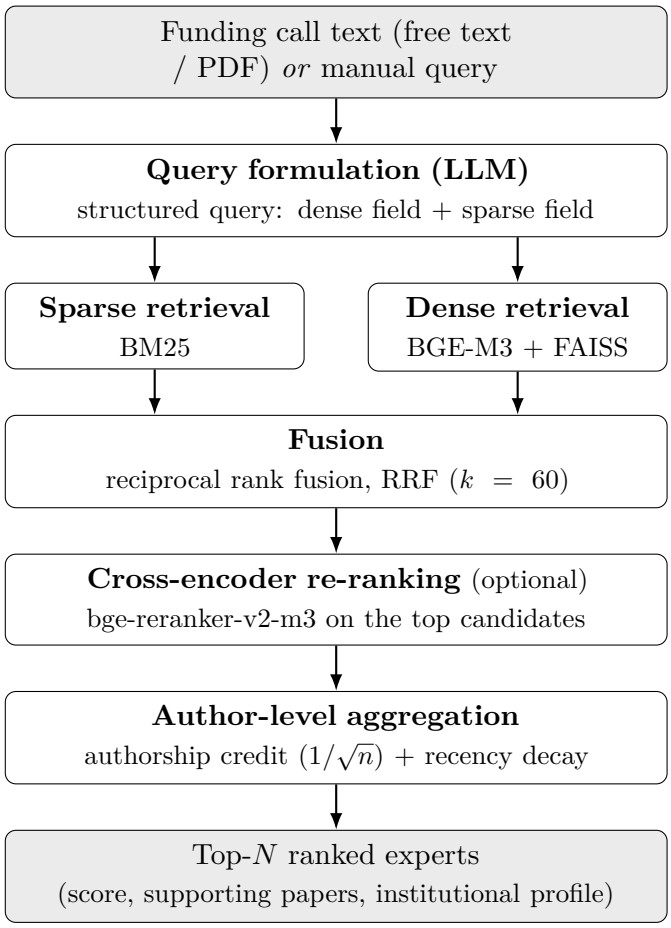


Figure 3.1: Overview of the expert-finding pipeline at query time; the corpus and the two indices are built offline.

3.2 BM25 and the Probabilistic Relevance Framework

3.2.1 From term specificity to a probabilistic ranking

BM25 is the most widely used member of a family of ranking functions derived from the *probabilistic relevance framework*, whose central idea is to rank documents by

their estimated probability of relevance to a query. Two classical results are at its basis. The first is the notion of *term specificity* introduced by Spärck Jones [22]: a term that occurs in few documents is more discriminative than a frequent one, so its weight should grow when its document frequency decreases, the principle that became inverse document frequency (IDF). The second is the Robertson–Spärck Jones relevance weight, which formalises term weighting under a binary independence assumption. The Okapi BM25 scoring function, which combines these ideas with term-frequency saturation and document-length normalisation, was developed in the Okapi experiments at TREC [23] and received its definitive treatment from Robertson and Zaragoza [3].

3.2.2 The scoring function

Given a query $Q = \{q_1, \ldots, q_m\}$ and a document D, BM25 assigns the score

$$\mathrm{BM25}(Q, D) = \sum_{i=1}^{m} \mathrm{IDF}(q_i) \, \frac{f(q_i, D)\,(k_1 + 1)}{f(q_i, D) + k_1 \left(1 - b + b\,\dfrac{|D|}{\mathrm{avgdl}}\right)}, \tag{3.1}$$

where $f(q_i, D)$ is the frequency of the term q_i in D, $|D|$ is the length of the document in tokens, avgdl is the average document length over the collection, and k_1 and b are free parameters. The inverse document frequency is computed as

$$\mathrm{IDF}(q_i) = \ln\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right), \tag{3.2}$$

where N is the number of documents in the collection and $n(q_i)$ is the number of documents that contain q_i. The additive 0.5 smoothing terms and the $+1$ inside the logarithm guarantee a non-negative and well-defined weight, even for very frequent terms. The implementation adopted in this thesis (the Okapi variant of the `rank_bm25` library, Section 4.3.1) computes the logarithm without the $+1$, so that a term contained in more than half of the documents would receive a negative weight; such weights are replaced by a small positive floor, equal to $\varepsilon = 0.25$ times the average IDF of the vocabulary.

3.2.3 Interpretation of the components

Three mechanisms make Equation (3.1) effective.

Term-frequency saturation. The ratio in Equation (3.1) increases with $f(q_i, D)$ but is bounded: when $f(q_i, D) \to \infty$, the contribution of the term approaches $\mathrm{IDF}(q_i)\,(k_1 + 1)$. The parameter k_1 controls how fast the function saturates: a small k_1 makes a single occurrence almost as valuable as many occurrences, a large k_1 keeps the score closer to a linear function of the raw count. This non-linear saturation reflects the intuition that the tenth occurrence of a term gives much less evidence about the topic of the document than the first one.

Document-length normalisation. The factor $\big(1 - b + b\,|D|/\,\mathrm{avgdl}\big)$ penalises long documents, which accumulate term occurrences simply because of their length. The parameter $b \in [0, 1]$ interpolates between no normalisation ($b = 0$) and full normalisation by relative length ($b = 1$). Typical values are $k_1 \in [1.2, 2.0]$ and $b = 0.75$; the implementation adopted in this thesis uses $k_1 = 1.5$ and $b = 0.75$.

Term discrimination. The multiplication by $\mathrm{IDF}(q_i)$ guarantees that matches on rare and specific terms (the material names, acronyms, and technical phrases that characterise a funding call) contribute much more to the score than matches on common words.

3.2.4 Strengths and limitations

BM25 is unsupervised, requires only an inverted index, and runs in a time proportional to the number of postings of the query terms; for this reason it is an excellent and inexpensive first-stage retriever. Its main limitation is that it matches *surface forms*: Equation (3.1) contributes nothing unless a query term appears exactly in the document. It is therefore blind to synonyms, to morphological variation, and (a crucial point for an Italian–English corpus) to cross-lingual paraphrase. This limitation motivates the dense semantic retriever described in Section 3.4.

3.3 FAISS and Nearest-Neighbour Search

3.3.1 The nearest-neighbour problem in dense retrieval

Dense retrieval (Section 3.4) represents every document with a vector $\mathbf{x} \in \mathbb{R}^d$ and answers a query vector $\mathbf{q} \in \mathbb{R}^d$ by returning its nearest neighbours under a similarity measure. With ℓ_2-normalised vectors, ranking by inner product is equivalent to ranking by cosine similarity, so retrieval reduces to the maximum inner-product search

$$\text{top-k}(\mathbf{q}) = \arg\max_{\mathbf{x} \in \mathcal{X}}{}^{(k)}\ \mathbf{q}^\top \mathbf{x}, \tag{3.3}$$

over the corpus $\mathcal{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$. FAISS [17] is a library that solves Equation (3.3) efficiently at scale, also on GPUs.

3.3.2 Exact search

The exact solution evaluates all the N inner products and keeps the k largest ones, with a cost of $\mathcal{O}(Nd)$ per query. FAISS implements this as a flat index (`IndexFlatIP`). For the corpus of this thesis (about 60,000 documents with $d = 1{,}024$), exact search introduces *no* approximation error, and this is why it is the index used here. Approximate methods become necessary only when N reaches the 10^7–10^9 range.

3.3.3 Design choice

Since the corpus used here is small enough for exact search to be both fast and error-free, the system uses the flat inner-product index, and completely avoids the

recall–latency trade-offs of the approximate indexes that FAISS also offers, which partition or compress the vectors. These indexes remain the natural upgrade path in case the corpus grows by orders of magnitude.

3.4 BGE-M3: Multilingual Neural Text Embeddings

3.4.1 From transformers to dense retrievers

Neural retrieval encodes text into a learned vector space in which semantic proximity corresponds to geometric proximity, and in this way it overcomes the surface-form limitation of BM25. The representations are produced by the Transformer architecture [24]. Its self-attention layers let every token of an input build its representation from all the other tokens of the same input, each weighted by a learned similarity, so that the output is a contextual representation of every token. Dense Passage Retrieval [15] made the *bi-encoder* the standard design for retrieval, in which queries and documents are encoded independently, so the document vectors can be precomputed and indexed, and scored with a simple inner product, which is exactly the setting of Equation (3.3).

3.4.2 A multilingual, multi-functional encoder

BGE-M3 [4] is a bi-encoder built on the XLM-RoBERTa [25] backbone, a Transformer pre-trained on one hundred languages, which places Italian and English text in a shared representation space, the property that makes cross-lingual retrieval possible in this work. Its name reflects three forms of versatility: multi-linguality, multi-granularity (it accepts inputs of up to 8192 tokens, enough for a full abstract), and multi-functionality, because a single model produces three different retrieval representations from the same encoder output $\{\mathbf{h}_t\}_{t=1}^{L}$.

Dense retrieval. The hidden state of the special leading token, $\mathbf{e} = \mathrm{norm}(\mathbf{h}_{[\mathrm{CLS}]})$, is taken as the document embedding, and query and document are scored by inner product, $s_{\mathrm{dense}}(q, d) = \mathbf{e}_q^{\top}\mathbf{e}_d$. This is the functionality that the present system indexes in FAISS.

Sparse and multi-vector retrieval. The same encoder output can also be turned into a sparse lexical representation and into a multi-vector representation with one vector per token. The system does not use them, since exact term matching is already provided by BM25 (Section 3.2).

3.4.3 Role in this work

Among the three functionalities, the system uses the dense one: the indexing text of each document (Section 4.1.4) is encoded into a normalised embedding, indexed in FAISS (Section 4.3.2), and matched to the dense field of the query through

Equation (3.3). The multilinguality of the XLM-RoBERTa backbone is the decisive property, because it allows an English funding call to retrieve a relevant Italian publication, and vice versa, through proximity in the shared embedding space, something that BM25 alone cannot do.

3.5 Reciprocal Rank Fusion

3.5.1 The fusion problem

Lexical and dense retrieval each return a ranked list, and the two are complementary: BM25 guarantees exact-term recall, while the dense encoder captures semantic and cross-lingual matches. To exploit both, the two lists must be merged into a single ranking. The obstacle is that the two scoring functions live on scales that are not comparable: a BM25 score is an unbounded sum of IDF-weighted term contributions (Equation (3.1)), while a dense score is a cosine similarity in $[-1, 1]$ (Equation (3.3)), so they cannot simply be added together.

3.5.2 Score-based fusion

The classical approach is represented by the CombSUM operator of Fox and Shaw [26], which combines the scores that a document receives in the different lists; since the scores of different retrievers are not comparable, they are first normalised onto a common range and then summed:

$$s_{\text{CombSUM}}(d) = \sum_{l \in L} \tilde{s}_l(d), \tag{3.4}$$

where $\tilde{s}_l(d)$ is the min–max normalised score of the document d in the list l. This approach requires the choice of a normalisation scheme, and the result is sensitive to score outliers and to the different distributions of the two retrievers, a fragile dependence that is difficult to tune in practice.

3.5.3 Rank-based fusion

Reciprocal rank fusion (RRF), proposed by Cormack et al. [5], avoids normalisation completely: it discards the scores and combines only the *ranks*:

$$\text{RRF}(d) = \sum_{l \in L} \frac{1}{k + r_l(d)}, \tag{3.5}$$

where $r_l(d)$ is the rank of d in the list l (documents absent from a list contribute nothing) and k is a smoothing constant, conventionally $k = 60$. The weight $1/(k + r)$ is largest at the top of each list and falls quickly with the rank, so the documents that both lists place near the top receive the highest fused scores. The constant k flattens the first positions. Without it, the very first results of a single list would dominate the fusion, together with whatever idiosyncrasies that list has. Since Equation (3.5) reads only positions, it is invariant to the BM25–cosine scale mismatch, it requires no training, and it has a single robust hyper-parameter k. Cormack et al. showed

that it performs better than Condorcet fusion and than supervised rank-learning methods on standard benchmarks.

3.5.4 Weighted fusion in this system

To let an operator move the fusion towards the lexical or the semantic evidence, the system generalises Equation (3.5) with a mixing weight $\alpha \in [0, 1]$ applied to the two branches, in the form stated in Equation (4.2) in Section 4.3.3; $\alpha = 0.5$ recovers standard RRF.

3.6 Cross-Encoder Re-ranking

The system uses both kinds of encoder, in two different stages of Figure 3.1: the bi-encoder BGE-M3 (Section 3.4) searches the whole corpus in the dense retrieval, and a cross-encoder re-scores only the first documents of the fused ranking. This section explains why the second model is needed and how it scores a query–document pair.

3.6.1 Bi-encoders versus cross-encoders

The first-stage retrievers are *bi-encoders*: the query and the document are encoded independently and compared only at the end with an inner product (Section 3.4.1). This independence is what makes them scalable, the document vectors are pre-computed and indexed, but it is also their weakness: the query never influences the representation of the document, and fine-grained term-level correspondences are lost when the text is pooled into a single vector.

A *cross-encoder* removes this restriction. The query and the candidate document are concatenated into a single input,

$$x = \texttt{[CLS]}\ q\ \texttt{[SEP]}\ d\ \texttt{[SEP]}, \tag{3.6}$$

where:

- q is the query and d is the document, each as a sequence of tokens;
- `[CLS]` is a special token at the start of the input: at the output, its representation summarises the whole pair and gives the relevance score (Equation (3.7));
- `[SEP]` is a special token that marks the end of each text.

The whole input goes through the Transformer as one sequence, so every query token can attend to every document token (Section 3.4.1). This full interaction produces clearly more accurate relevance estimates, but nothing can be pre-computed: a forward pass is required for each query–document pair at query time. Since the query and the document share one input, they also share its maximum length, so a long query leaves less room for the document (Section 4.4).

3.6.2 The monoBERT relevance model

Nogueira and Cho [19] introduced the canonical cross-encoder re-ranker, often called monoBERT. The contextual representation of the leading token, $\mathbf{h}_{[\mathrm{CLS}]}$, is mapped to a scalar relevance score by a learned linear layer,

$$s(q, d) = \mathbf{w}^{\top} \mathbf{h}_{[\mathrm{CLS}]} + b, \tag{3.7}$$

and the model is fine-tuned with a cross-entropy objective that contrasts relevant and non-relevant query–document pairs. Re-ranking with such a model improves the ranking quality of first-stage retrievers in a substantial way, and the BERT-based retrieve-then-re-rank pattern that it introduced is now standard in multi-stage ranking.

3.6.3 Re-ranking in this system

Since the cost of Equation (3.7) grows linearly with the number of scored pairs and cannot be amortised with pre-computation, the cross-encoder is applied only to the top n_{cand} candidates that come out of the fusion stage, and not to the whole corpus. The system uses a multilingual cross-encoder that is built on the same BGE-M3 backbone [4] as the dense encoder, but is trained to output a relevance score for a (query, document) pair directly, and therefore shares the cross-lingual coverage needed for the Italian–English corpus. The re-ranked scores replace the fused scores for the documents that reach this stage, and after that the relevance is propagated to the authors as described in Section 4.5. Since this stage is the main contributor to the query latency on CPU, it can be disabled when a faster retrieval is preferred.

Chapter 4

Methods

This chapter describes the system in detail: the data sources and the construction of the corpus (Section 4.1), and each stage of the pipeline presented in Section 3.1: query formulation (Section 4.2), hybrid retrieval and fusion (Section 4.3), neural re-ranking (Section 4.4), and author-level aggregation (Section 4.5).

4.1 Data and Corpus Construction

4.1.1 Sources

The corpus is built from the scientific production of Politecnico di Torino over a window of fifteen years (2010–2024), exported from the institutional research information system. Each record provides bibliographic metadata (title, authors with internal identifiers, publication type, year, venue, keywords, and external identifiers such as DOI and Scopus ID), but the abstract is often missing in the raw export. Journal articles, reviews and editorials, book chapters, monographs, conference papers, working papers, and patents are retained. Other record types of the repository are left out, among them doctoral theses, posters, conference abstracts, exhibitions, and design works. The profile of each researcher comes from the institutional registry. It contains an internal identifier (the *matricola*), the scientific-disciplinary sector, the home department, and links to external author identifiers.

4.1.2 Abstract Enrichment

Since the abstract is the richest source of topical signal, the records without one are enriched from three external sources, queried in cascade, so that each stage processes only the records that are still missing an abstract:

1. **OpenAlex** [6], queried by DOI; the abstracts are reconstructed from the inverted index returned by the API;
2. **IRIS**, the institutional open-access repository, queried through its public OAI-PMH endpoint by handle;
3. **Scopus**, queried by Scopus ID (used optionally, within the limits of the API quota).

After the enrichment, the abstract coverage increases in a substantial way. The remaining records (mostly journal articles, conference papers, and book chapters without an indexed abstract) are kept in the corpus, but they are represented only by their title, keywords, and venue.

4.1.3 De-duplication

The raw export contains the same publication several times, once for each internal affiliating structure: a paper co-authored across two departments appears twice, with otherwise identical content. The duplicates are collapsed on their repository handle, and when they differ in coverage the copy with an abstract is kept. A handle identifies a record, and a record does not always coincide with a work. A conference paper and its later journal version, for instance, have two handles and remain two separate documents. Without this step, the publications of researchers affiliated with several departments would be counted twice, and their relevance would be inflated. The step also reduces the size of the index. The final corpus contains more than 60,000 unique publications.

4.1.4 Document Representation

For retrieval, each document is represented by two derived text fields. The *indexing text* concatenates the title (written twice, so that it weighs more, a simple form of field weighting that was not tuned), the abstract, the keywords, and the venue; it is used both to build the lexical index and to compute the dense embedding. A separate, shorter *re-ranking text*, composed of the title and the first 2,000 characters of the abstract, is stored for the cross-encoder stage (Section 4.4). The author identifiers are normalised by removing the leading zeros, in order to reconcile the zero-padded form used in the publication records with the integer form used in the registry; this step is essential to link the documents to the researcher profiles correctly.

4.1.5 Offline Indexing

The two indices are built once, offline, from the document representation, and then reused unchanged at every query. The lexical index is a BM25 index over the tokenised indexing text of the whole corpus, serialised to disk so that it does not have to be rebuilt at start-up. The dense index is constructed by encoding the indexing text of each document with BGE-M3, L_2-normalising the resulting vectors, and adding them to a FAISS inner-product index. The embedding of the corpus is by far the most expensive step of the whole system, but it is paid only once: at query time only the single query vector is encoded. This offline construction is what makes the query-time pipeline of Figure 3.1 light enough to run interactively.

4.2 Query Formulation

The raw text of a call is long and dominated by administrative material, so it is a poor query (Section 2.5). For this reason, the system supports two query modes.

In the *manual* mode, the user provides the query fields directly. In the *generated* mode, a large language model transforms the call into a structured query. This follows the same intuition as HyDE [20] (using a generative model to produce a richer, retrieval-oriented representation), but it emits a structured object instead of a single hypothetical passage, so that each retriever can be fed the representation that suits it best.

4.2.1 Structured Query Schema

The generated query is not a single string but a typed object with six fields, each targeting a specific need of the downstream retrievers:

- `main_topic`: the core research domain in at most a few words, used as a compact label and as a fallback when the other fields are empty;
- `dense_query`: a fluent English paragraph of three to five sentences that describes the profile of the ideal expert as if it were the abstract of their work; it is the input of the semantic branch;
- `sparse_query`: a string of technical terms, without stop-words or punctuation, that feeds the lexical branch;
- `keywords`, `phrases`, `acronyms`: three lists of, respectively, single technical terms, multi-word expressions that occur verbatim in titles and abstracts, and domain acronyms (such as *CRM*, *LCA*, or *DFT*).

The query has a *dense field* written in fluent English and *sparse fields* made of discrete terms because the two retrievers read text differently. The dense encoder works best on natural, contextual language, while BM25 rewards precise and rare terms. With the local backend the schema is enforced during generation. The model can only return JSON that validates against the schema, and it decodes at temperature zero, so the same call always yields the same query and the evaluation remains reproducible.

4.2.2 Prompt Design

The quality of the generated query depends on a prompt that knows the retrieval pipeline it feeds, and its full text is reported in Appendix A. Besides the field-by-field instructions, two of its rules are specific to this corpus and task:

- generic words that add noise to a lexical index (*study*, *paper*, *method*, and similar) are not accepted as isolated keywords, and rare, specific terms are requested instead, since BM25 rewards a low document frequency;
- Italian equivalents of the key terms are requested wherever a concept is commonly published in Italian, because the corpus is bilingual and a query written only in English would miss the Italian-language production.

The same concern shapes the dense field. The model is asked to describe the ideal expert instead of summarising the call (Section 4.2.1), and this keeps the field close to the documents to be retrieved (the abstracts of the researchers' publications) and away from the administrative language of the call.

4.2.3 Query Normalisation and Sparse Expansion

Before retrieval, the structured query passes through a normalisation step that serves two purposes. It first cleans every field, collapsing whitespace and lower-casing the discrete terms, so that the same surface form is used at query time and at indexing time. It then builds the effective lexical query by *concatenating* the sparse field with the main topic and with the keyword, phrase, and acronym lists. This is a form of query expansion confined to the lexical branch: the extra terms raise the chance of an exact match against the titles, abstracts, and keywords in the index, while the dense field is left untouched, as a clean English paragraph, so that the semantic branch is not diluted by disconnected tokens. In manual mode, or when a plain string is supplied, the same field is used for both branches. This is also how the raw text of a proposal enters the pipeline in the evaluation of Section 6.4, where it serves as the baseline against which the generated query is tested.

4.2.4 Model Serving

The query generation can be served by a local model run through Ollama or by a hosted model reached through an API, and both produce queries with the same six-field structure. In the evaluation, the local model is Llama 3.1 8B and the hosted model is Claude Haiku 4.5. The local model is the reference of the evaluation: it removes any dependency on an external service and, at temperature zero, makes every experiment reproducible. The deployment relies instead on a hosted model reached through an API, since the only measurement of that path (Section 6.4.2) indicates that the quality of the generator affects the effectiveness of the whole pipeline. In both cases the output is validated against the schema, so a malformed or non-conforming generation is detected and rejected rather than passed silently to the retrievers.

4.3 Hybrid Retrieval and Fusion

4.3.1 Lexical Retrieval

The lexical branch scores the documents with BM25 (Section 3.2) over an index built from the indexing text. The text is tokenised with a deliberately simple procedure: it is put in lower case, every character other than an unaccented Latin letter, a digit, or a hyphen is replaced by a space, and the result is split on white space. The hyphen is kept on purpose, so that a technical term written with a dash, and a code that mixes letters and digits, both stay whole; punctuation, instead, acts as a separator, and so do accented letters and apostrophes (the Italian *dell'energia* yields the two tokens *dell* and *energia*). At indexing time a short bilingual list of

Italian and English function words is also removed. The expanded sparse query (Section 4.2.3) is tokenised with the same procedure at query time, without the list of function words: this changes no score, because a term absent from the index receives no weight, so the query and the documents are compared on identical surface forms. The documents are then ranked by their BM25 score, and the top k_{sparse} with a strictly positive score are kept.

4.3.2 Dense Retrieval

The dense branch encodes both documents and queries with BGE-M3 (Section 3.4), a multilingual encoder that maps Italian and English text into a shared space, so that an Italian publication can be retrieved by an English query and vice versa. The document embeddings are L_2-normalised and indexed with FAISS (Section 3.3) using inner-product search, which on normalised vectors is equivalent to cosine similarity:

$$\text{sim}(q, d) = \frac{\mathbf{e}_q \cdot \mathbf{e}_d}{\|\mathbf{e}_q\| \, \|\mathbf{e}_d\|}, \tag{4.1}$$

where $\mathbf{e}_q$ and $\mathbf{e}_d$ are the embeddings of the dense query field and of the document. The dense query field is encoded with the same normalisation at query time, and the top k_{dense} documents by similarity are kept.

4.3.3 Fusion

The two rankings are merged with reciprocal rank fusion (Section 3.5), which combines the lists using only the ranks and is therefore robust to the incomparable score scales of BM25 and cosine similarity. To control the relative influence of the two branches, a mixing weight $\alpha \in [0, 1]$ is introduced, so that the fused score of a document d is

$$\text{RRF}(d) = (1 - \alpha) \, \frac{\mathbb{I}[d \in R_{\text{sparse}}]}{k + r_{\text{sparse}}(d)} + \alpha \, \frac{\mathbb{I}[d \in R_{\text{dense}}]}{k + r_{\text{dense}}(d)}, \tag{4.2}$$

where $r_{\text{sparse}}(d)$ and $r_{\text{dense}}(d)$ are the ranks of d in the lexical and dense lists, k is the standard RRF smoothing constant (fixed to the customary value of 60), and $\mathbb{I}[\cdot]$ is the indicator function. The constant k damps the influence of the very top ranks, so that a document must be ranked highly by *both* branches, or extremely highly by one, to reach the top of the fused list. Setting $\alpha = 0.5$ recovers standard RRF; $\alpha = 0$ uses only the lexical signal and $\alpha = 1$ only the dense signal. Fixing α globally is a deliberate simplification; making it query-dependent, as in Dynamic Alpha Tuning [18], is a natural extension.

4.4 Neural Re-ranking

Since rank fusion considers only positions, a precision-oriented re-ranking stage is applied to the top of the fused list. A multilingual cross-encoder re-ranker, which follows the design of Nogueira and Cho (Section 3.6), encodes together the dense query field and the re-ranking text of each candidate, and outputs a direct

relevance score. Unlike the dense branch, which embeds the query and the documents *independently* and can therefore pre-compute the document vectors offline, the cross-encoder processes each query–document pair jointly: this joint encoding is what gives it its higher precision, but it also means that the score cannot be pre-computed and must be evaluated at query time. For this reason only the top $n_{\text{cand}} = 50$ fused candidates are re-scored (the cross-encoder is far too expensive to apply to the whole corpus), each read together with the query in a window of 1,024 tokens, and the stage can be disabled altogether when a faster response is preferred. The re-ranked scores replace the fused scores for the documents that reach this stage, while the documents below the n_{cand} cut-off keep their fused scores, so that the two variants produce rankings of the same depth.

4.5 Author-Level Aggregation

The final stage converts the ranked documents into a ranked list of researchers. Each document contributes its relevance to its authors, weighted by three factors: the retrieval score of the document, an authorship-credit weight, and a recency weight. Each of the three is ablated in Section 6.2.

4.5.1 Authorship Credit

How to divide the credit of a multi-authored paper among its authors is a long-standing question in bibliometrics, where the full-counting and fractional-counting conventions are the two opposite extremes [27]. The system exposes this choice as a parameter. For a document with n authors, each author receives a weight $w(n)$ under one of three schemes:

$$w_{\text{full}}(n) = 1, \qquad w_{\text{frac}}(n) = \frac{1}{n}, \qquad w_{\text{sqrt}}(n) = \frac{1}{\sqrt{\min(n, c)}}, \tag{4.3}$$

where *full* counting gives every author the same credit, whatever the size of the team, and *fractional* counting splits one unit of credit among the authors. The *square-root* scheme sits between the two and penalises long author lists more gently than fractional counting. The cap c is set to twenty. It prevents a very large team from pushing the weight towards zero, which would erase the contribution of every member. Because the publication records list only the authors of the institution, n counts internal authors, and the cap binds on very few documents. It works as a safeguard and was not tuned. The records also have a side effect on the credit. A publication with a single internal author and many external co-authors gives that author full credit, because the external co-authors are not listed, so the scheme divides the credit within the internal team only. Unlike position-based schemes, which give extra credit to the first or last author, the system gives all the authors of a paper the same weight. In the Italian academic context, the order of the authors is not a reliable indicator of their contribution. The square-root scheme is the deployed default, as it neither over-rewards members of large teams, as full counting does, nor over-penalises them, as fractional counting does.

4.5.2 Recency Weighting

To favour the researchers that are currently active on a topic, the contribution of a document can optionally be decayed with its age, using an exponential half-life model:

$$\rho(d) = 0.5^{\,(y_{\text{now}} - y_d)/h}, \tag{4.4}$$

where y_d is the publication year, y_{now} is the reference year, and h is the half-life, that is, the number of years after which a paper keeps half of its weight. A larger h flattens the decay. The deployed default sets h to ten years, which halves the weight of a decade-old paper but still counts it, and the weighting can also be switched off. Section 6.2 measures its effect on the ranking. The weighting is enabled by default and stays under the user's control, because it encodes a preference that no measurement can settle, namely whether a researcher who is active now should come before an established authority whose most relevant work is older.

4.5.3 Score Accumulation and Output

The score of a researcher a is the sum, over the retrieved documents that they co-authored, of the document score scaled by the two weights:

$$S(a) = \sum_{d\,:\,a \in \mathcal{A}(d)} s(d)\; w(|\mathcal{A}(d)|)\; \rho(d), \tag{4.5}$$

where $\mathcal{A}(d)$ is the author set of d and $s(d)$ is its retrieval score (the re-ranked score when re-ranking is enabled, otherwise the fused score). A researcher therefore accumulates score by co-authoring *several* relevant documents and by being highly ranked on each of them, which rewards a sustained body of work on the topic over a single well-placed paper. The researchers are ranked by $S(a)$. In the terms of the expert-finding literature, this is the document model of Balog et al. [2] with the retrieval score as the document–candidate association, and the sum is the CombSUM rule of the voting model of Macdonald and Ounis [8]; the alternative rules compared in Section 6.2 are other members of the same family.

For each of the top-ranked researchers, the system returns the supporting evidence (their most relevant retrieved publications, ordered by their contribution to the score) together with profile attributes: the home department, the scientific-disciplinary sector and its broader area, external identifiers such as ORCID and Scopus, and the number of projects and awards. With this evidence a user can check each recommendation against the publications behind it, which matters when the list is used to justify the composition of a proposal team. Authors are keyed by their normalised internal identifier. Almost every author occurrence in the corpus carries one (only 5 of 138,111 do not). The profile attributes come instead from the author registry, which covers 2,980 researchers, so an author whose identifier is missing from the registry is ranked like any other but shown without a profile.

4.6 Implementation and Deployment

The pipeline is implemented in Python. The lexical index, the FAISS dense index, and the pre-processed corpus are built offline and saved to disk; at start-up, the

application loads them once and keeps the embedding and re-ranker models in memory. The system is exposed through an interactive web application, in which the user selects the query mode, tunes the retrieval and aggregation parameters (k_{sparse}, k_{dense}, α, the re-ranking toggle, the authorship-credit scheme, the recency half-life, and the number of experts and supporting papers to display), launches the search, and exports the ranked results. Access is protected by authentication, so that the application can be hosted behind an institutional endpoint.

Chapter 5

Evaluation Methodology

No public benchmark exists for this corpus and this task: no dataset states, for a given funding call, which researchers of Politecnico di Torino are the relevant experts. This chapter describes the methodology built to work around that constraint. Two experiments carry the conclusions of this work, and neither requires a human relevance judgment to exist: a leave-one-out benchmark derived automatically from the authorship record of the corpus (Section 5.3), and an end-to-end benchmark built from the institutional register of funded projects, whose reference experts are established by an administrative act rather than by any retrieval run (Section 5.4). The two benchmarks run in parallel on the same pipeline: the first gives the statistical power to compare the design choices, the second measures the same choices and the whole system on the input the system was built for. The experimental results obtained with this methodology are reported in Chapter 6.

5.1 Evaluation Questions and Experimental Design

Two questions, one for each experiment, guide the rest of this chapter and the results of Chapter 6; they are stated here as an aid to reading rather than as a formal hypothesis framework.

1. How much does each design choice of the pipeline (the fusion weight, the aggregation depth, the authorship credit, the recency weighting, the combination rule, the cross-encoder) contribute to the expert ranking, on a large sample where the correct answer is known by construction? (Section 5.3)

2. Given the real text of a funded proposal, which stands in for the call, does the pipeline recover the researchers that an administrative record already lists for that project? (Section 5.4)

Table 5.1 summarises how the two experiments are built: what a query is and where it comes from, how large the sample is, what counts as ground truth, what is removed or cut from the input so that the evaluation does not leak, which conditions are compared, and which metrics and test apply. Figure 5.1 shows the two protocols

that share the same pipeline but differ in one respect: where in the pipeline the evaluation intervenes.

	Leave-one-out (Section 5.3)	**Project registry** (Section 5.4)
Query unit and input	Publication; title and first 2,000 characters of the abstract	Funded project; proposal title and abstract
Sample	2,000 queries, fixed seed, 2010–2024; journal articles with an abstract of at least 200 characters and at most 15 reference authors	1,220 projects
Ground truth	PoliTo authors with at least two publications in the corpus, including authors outside the registry; mean 2.28 per query	Registry members assigned to the project with at least two publications before its start (2,980 profiled researchers); mean 1.58 per project
Removed or cut	Query publication removed *after* fusion; no temporal cut	Corpus cut to publications before the project start, *per branch, before* fusion
Factors or conditions	Fusion weight, aggregation depth, authorship credit, recency, combination rule, cross-encoder (16 comparisons)	Raw text against generated query (local reference and hosted); cross-encoder, also at aggregation depth fifty; aggregation depth, combination rule, recency; single branches; random and popularity floors
Metrics and test	AP and RR on the 50-expert list; S@k and R@k at 1, 5, 10, 20, 50; paired bootstrap, Holm	First ten positions and R@50; paired bootstrap, Holm

Table 5.1: Design of the two experiments.

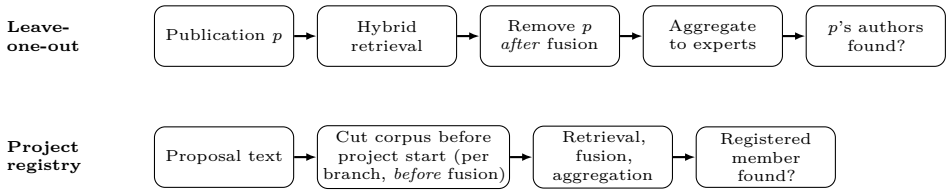


Figure 5.1: The two evaluation protocols on the same pipeline.

5.2 Metrics and Statistical Testing

All metrics are standard information-retrieval measures over binary relevance judgments [28]. In the following, $L = (d_1, d_2, \dots)$ is the ranked list returned for a query, R is the set of items relevant to that query, and $\mathrm{rel}(i) \in \{0, 1\}$ indicates whether the item at rank i belongs to R.

Success and recall at cut-off.

$$\mathrm{S@}k = \mathbb{I}\Big[\, L_{1:k} \cap R \neq \emptyset \,\Big], \qquad \mathrm{R@}k = \frac{|\, L_{1:k} \cap R \,|}{|R|}. \tag{5.1}$$

S@k, the indicator that at least one relevant item appears in the first k positions, is the reading figure on both operational benchmarks: it states the outcome a user experiences, a shortlist worth reading. R@k states how much of the reference the shortlist covers, and at $k = 50$ it separates a retrieval failure from an aggregation failure.

Average precision.

$$\mathrm{AP} = \frac{1}{|R|} \sum_{i\,:\,d_i \in R} \frac{|L_{1:i} \cap R|}{i}, \tag{5.2}$$

and its mean over the query set is MAP. AP is rank-aware and uses the whole truncated ranking, which is why MAP is the primary metric on which significance is tested throughout this work; S@k remains the figure used to report effectiveness, because it is the one a reader interprets directly.

Reciprocal rank. RR is $1/r$, where r is the position of the first relevant item, and 0 if none appears within the cut-off; its mean is MRR. It is reported as a diagnostic alongside AP on the project-registry benchmark, since it is not defined on an interval scale and is not used for model selection [29].

nDCG. The normalised discounted cumulative gain at cut-off k [30],

$$\mathrm{nDCG@}k = \frac{\mathrm{DCG@}k}{\mathrm{IDCG@}k}, \qquad \mathrm{DCG@}k = \sum_{i=1}^{k} \frac{\mathrm{rel}(i)}{\log_2(i+1)}, \tag{5.3}$$

where IDCG@k is the DCG of an ideal ranking, discounts each relevant item by the logarithm of its rank, so it is sensitive to the order inside the top k in a way S@k and R@k are not.

Cut-offs. On the leave-one-out benchmark, AP and RR are computed on the ranked list of fifty experts that the aggregation produces, and S@k and R@k are reported at $k \in \{1, 5, 10, 20, 50\}$. On the project-registry benchmark the shortlist a user reads is the first ten positions, so AP, RR, and S@k are computed there. R@50 is kept as a depth diagnostic. It separates the case where the system never places an expert in the candidate set from the case where it places one but ranks them beyond the tenth position.

Statistical testing. Every comparison between two configurations, on either benchmark, is tested with a paired bootstrap on the per-query average precision [31]. From 10,000 resamples of the query set the test obtains the mean difference in MAP, its confidence interval, and a two-sided p-value. The pairing is possible because both configurations are evaluated on the same queries, and it removes the between-query

variance from the estimate. Where a table compares more than one configuration against the same reference, the p-values are corrected with the Holm step-down procedure [32], which keeps the adjusted p-values monotone non-decreasing with the sorted raw values so that the correction cannot reverse the ranking of the comparisons. The correction is computed separately over four families: the sixteen comparisons of the leave-one-out ablation (Table 6.2); the nineteen registry conditions tested against the raw proposal text, whose main rows are reported in Tables 6.6 and 6.8; the thirteen *decisional* contrasts of the registry, which test each design choice against its own baseline and give the corrected values quoted in Section 6.2; and the eight metrics of the query-generation comparison (Table 6.7), average precision included; Table B.1 lists the comparisons of each family. MAP is the primary test metric throughout this work. A difference counts as significant only when its corrected value stays below 0.05. When the test does not resolve a difference, the two configurations are reported as unresolved and are not ranked. Comparisons between groups of projects, rather than between configurations on the same projects, use unpaired two-sample tests, and associations across departments, funding instruments, or projects are measured with a correlation coefficient and its test.

5.3 Phase One: The Leave-One-Out Benchmark and the Ablation

The document that the pipeline ranks is a publication, but the output a user reads is a ranking of researchers, and building a reference for that ranking without manual labelling requires a signal that is already a matter of record. Authorship meets this requirement, because the authors of a publication are, by construction, experts on its topic. Reviewer-assignment research uses authorship and citations as a relevance signal in the same way [10], [11].

Sampling. A publication p is drawn from the corpus and used as a query, its title concatenated with the first 2,000 characters of its abstract. The sample comprises 2,000 such queries, drawn with a fixed random seed from the publications dated 2010–2024, restricted to journal articles with an abstract of at least 200 characters and at most fifteen reference authors, so that every query carries a substantial and unambiguous topical signal.

Ground truth. The reference for a query p is the set of matricole of the PoliTo authors of p who have at least two publications in the corpus. The condition applies whether or not the author is profiled in the registry, so the leave-one-out reference is not confined to the registry. On average a query has 2.28 reference authors. Authors with no further publication in the corpus are excluded, because once p is removed nothing in the corpus could lead to them, and missing them would measure a gap in the data instead of a retrieval failure.

Removal and retrieval. The query publication is removed from the ranking only *after* the two retrieval branches are fused. Retrieval and fusion therefore run exactly

as they would for any other query, and p is stripped from the fused list just before the aggregation stage. No temporal cut is applied. Two hundred documents per branch feed the fusion, a hundred fused documents are passed to the author aggregation, and the aggregation returns fifty experts.

Factors and levels. The ablation starts from the reference configuration and changes one design choice at a time, while every other choice keeps its reference value. Six choices are varied, each against the value it takes in the reference configuration:

- the fusion weight α of Equation (4.2): 0 (lexical only), 0.3, 0.7, or 1 (dense only), instead of 0.5;
- the aggregation depth, that is the number of fused documents passed to the aggregation: 25, 50, or 200, instead of 100;
- the authorship credit of Section 4.5.1: full or fractional, instead of the square root;
- the recency weighting of Section 4.5.2: a half-life of five, ten, or twenty years, instead of no weighting;
- the combination rule, which turns the contributions of several documents into the score of one researcher (Section 4.5.3): the maximum, the mean, or the sum of the three largest contributions, instead of the sum of all of them;
- the cross-encoder re-ranking of Section 4.4: enabled, instead of disabled.

The alternatives add up to sixteen settings, each compared with the reference configuration on the same 2,000 queries, and these sixteen comparisons form the family on which the Holm correction of Section 5.2 is computed. No setting changes the lists that the two branches retrieve. The fusion weight and the choices of the aggregation work on the same two lists, and the cross-encoder only re-scores the first fifty fused documents, so each difference in Table 6.2 can be traced to a single choice. The one interaction that bears on the conclusions, between the cross-encoder and the aggregation depth, is measured separately, at matched depth (Section 6.6.1).

The query-generation stage is deliberately left out of this sweep and ablated instead on the project-registry benchmark (Section 5.4), where the input is the text of a funded project rather than a publication abstract: a stage that rewrites administrative text cannot be attributed cleanly on a benchmark whose queries already are abstracts.

From the ablation to the evaluated configuration. The two benchmarks are used in parallel. The ablation measures every design choice with the power of a large sample, and the registry benchmark of Section 5.4 measures on the real input the choices whose effect could depend on the query being a publication abstract: a shallower aggregation depth, a different combination rule, and the cross-encoder. A level that the ablation favours is adopted only if the registry does not contradict it. None of these alternatives is adopted, so the configuration evaluated on the registry is the reference of the sweep, unchanged, and the effectiveness reported there does not result from a choice made on the registry itself.

Error attribution. The metrics of Section 5.2 say how often a reference author reaches the top of the ranking, but they do not say which stage loses the others. On the same cached runs, the authors of the retrieved documents are the only people the aggregation can rank. The share of reference authors among them is therefore the ceiling set by retrieval, and the gap between this ceiling and the share actually returned in the top ten is the loss due to the aggregation. Section 6.2.1 reports the measurement.

5.4 Phase Two: The Project Registry

The leave-one-out reference is objective and large, but its queries are publication abstracts and its reference experts are co-authors, topically close to the query by construction. A second benchmark uses an administrative rather than bibliometric reference: the institutional register of funded research projects, harvested through the public personnel web service of the institution.

Construction. A project is often registered as several contracts, one per participant or per funding year, with the same abstract. These copies are merged into a single project with the union of their member lists, so that the same query does not enter the benchmark twice and its reference is not split across the copies. A project enters the benchmark if it passes three filters:

- its proposal abstract is at least 300 characters long, so that every query is a full text (this is the most selective filter, because many industrial contracts are registered with a title only);
- its start date falls between 2015 and 2027, so that at least five years of corpus precede every query;
- at least one of its researchers has two or more publications indexed before the project start.

After the filters, the benchmark contains 1,220 funded projects, spanning all eleven departments and eight funding families (European collaborative programmes, ERC and MSCA actions, national, regional, and industrial funding, and the PNRR), with 366 proposals written in Italian.

Ground truth. The reference of a project is the set of its assigned researchers who have at least two publications indexed before the project start. The principle is the same as in Section 5.3, since every reference expert has a record in the corpus that is visible when the project starts. On average a project has 1.58 reference researchers.

Temporal cut. For every project, the corpus visible to the system is restricted to publications dated before the project start. The cut is applied to *each branch separately, before fusion.* Both the lexical and the dense candidate lists lose the publications dated from the project start onwards, and only the remaining candidates are fused, aggregated, and ranked. The leave-one-out removal of Section 5.3 works

differently, since it strips a single item *after* fusion, whereas here entire spans of the corpus are unavailable to both branches before they are combined. Without the cut the evaluation would leak, because the publications produced by the project itself share its vocabulary and would reveal the team trivially. With the cut, the benchmark asks an operational question, namely whether the system would have identified the researchers from what they had published before the project started. A residual anachronism remains in the BM25 collection statistics, which are computed on the full corpus, a second-order effect that is declared rather than corrected. Part of the benchmark starts after the last corpus year, where the cut has nothing to remove; on that subset the runs with and without the cut must coincide, which checks the implementation of the cut rather than its effect, and the check is reported in Section 6.3.

Evaluated configuration. All the conditions on this benchmark share the retrieval and aggregation parameters of the reference configuration of the ablation (Section 5.3): 200 documents per branch, 100 passed to the aggregation, and 50 experts returned, with the cross-encoder and the recency weighting disabled, so that the two benchmarks speak about the same pipeline setting. With these parameters, the query generated by the local model is the *reference configuration* of this benchmark, and the raw proposal text is the *baseline* against which every other condition is tested.

Generation backends. The query fed to the pipeline is either the raw proposal text or a query generated by a language model, and three generation conditions are compared. The local model, Llama 3.1 8B at temperature zero, is run once over the whole benchmark and is the *reference configuration*: deterministic and reproducible, and the only one analysed in full. The same local model is also run three times at temperature 0.7 on a sample of 400 proposals, to measure the variance that sampling introduces across repeated runs, and once more at temperature zero on the same sample, to check that the reference is reproducible. A hosted model, Claude Haiku 4.5 (Section 4.2.4), is run once over the whole benchmark, and this single run is the only measurement of that path. Its provenance is declared in full: the queries were produced outside the client the application uses to call its own generator, with a separate instructions file, in Italian, of content equivalent to but not identical with the prompt of Appendix A, so the hosted condition changes the model, the prompt, and the channel together.

Conditions and floors. Absolute values on this benchmark are read against two floors computed under the same protocol: a random ranking of registry authors, and a popularity ranking of the authors with the most publications before the temporal cut, the more demanding of the two, since document-centric aggregation is known to favour prolific authors.

Application defaults. The application exposes the parameters of the evaluated configuration to the user, and the values it proposes when it opens differ from the evaluated ones on three points (Table 5.2). The figures of Chapter 6 describe the evaluated configuration, not what a user sees unless the interface is reconfigured.

- The shorter lists (100 documents retrieved, 50 aggregated, 20 experts) make the interface responsive and match what a user reads in one sitting. Their cost is small: fifty aggregated documents score higher than one hundred on the leave-one-out benchmark and leave MAP@10 unchanged on the registry, although they reach slightly fewer reference experts (Section 6.2), and twenty experts exceed the ten positions on which the registry metrics are computed. The evaluation uses the deeper setting so that the measurement is not limited by an interface choice.

- The recency weighting is enabled by default because it encodes a user preference that no measurement settles (Section 4.5.2), and it is disabled in the evaluation so that its cost does not confound the other comparisons.

- The cross-encoder is the one difference that the measurements do not support: on the input of this benchmark the stage does not help (Section 6.2), so the recommendation of this work is that the default be changed.

Candidate set. The pipeline ranks every author identifier present in the corpus, 10,175 matricole, while the registry ground truth is drawn from the 2,980 profiled researchers. Consequently, some of the names that could occupy the first ten positions belong to authors the registry does not cover and can never count as a success, regardless of how the system ranks them, so every precision-like reading on this benchmark is a lower bound. A sensitivity check repeats seven conditions with the ranking restricted to the profiled researchers (Section 6.3).

Parameter	Evaluated here	Application default
Documents retrieved per branch	200	100
Documents passed to aggregation	100	50
Experts returned	50	20
Fusion weight α	0.5	0.5
Authorship credit	$1/\sqrt{n}$	$1/\sqrt{n}$
Cross-encoder re-ranking	disabled	enabled
Recency weighting	disabled	half-life of ten years

Table 5.2: Evaluated configuration and default configuration of the application.

5.5 Threats to Validity

Where a language model can, and cannot, influence the conclusions. Neither reference used in this work is produced by a language model: the leave-one-out reference is the authorship record of the corpus, and the registry reference is an administrative act, so neither can be shaped by a model's output. A model appears only at the input side, where it reformulates the text of a call: this is a stage of the system under test rather than a part of any reference, and it is evaluated as such (Section 5.4).

Statistical power. The two benchmarks that carry the conclusions differ in size, with 2,000 leave-one-out queries against 1,220 registry projects. On the smaller one, the paired bootstrap of Section 5.2 gives 95% confidence intervals of about ± 0.005 to ± 0.015 MAP between two configurations, depending on the contrast. A true difference narrower than the interval cannot be told apart from noise, and for a factor tested within a larger family the Holm correction raises the threshold further. For this reason absolute values are read with more caution than the relative ordering of configurations, and a comparison that the test does not resolve is reported as unresolved.

The registry as a construct. The register is authoritative about who is assigned to a project, but it is not exhaustive. With a mean of 1.58 listed researchers per project, it plausibly records the principal investigators and leaves out the rest of the team, so every precision-like reading against it is a lower bound of the true value. Being listed on a contract does not imply having written its proposal. On collaborative instruments, the recorded signatory may be the contact for an institutional work package, and not necessarily the person who wrote the text from which the query is built, so the pairing between project and researcher is less exact there than on instruments with a single holder. Finally, as Section 5.4 states, the system ranks every author identifier in the corpus while the register covers only a subset of it, so some of the candidates a ranking could legitimately surface can never be credited by this reference, which is a property of the reference rather than a ranking failure.

The proposal as a stand-in for the call. The register stores the funded proposal and does not keep the call it answered, so the benchmark uses the proposal as the query. The two genres differ. A call is written by the funding body and names nobody, whereas a proposal is written by the researchers the system is asked to find, in their own vocabulary and on their own topic. The temporal cut removes the publications that follow the project. It cannot remove this proximity between the query and the earlier work of its own authors, which a call would not carry. The proposal is used because it is the only call-like text for which an administrative answer exists; the readings on this benchmark are therefore taken on an input somewhat closer to the reference than a call would be, a threat to external validity that is declared here and whose size is not established.

Leakage and contamination. The temporal cut of Section 5.4 removes the dominant leakage channel, the project's own publications, by withholding them from both retrieval branches before fusion; its implementation is checked on the projects where it has nothing to remove. A residual channel is addressed with a textual check that flags a project whose acronym already appears in a pre-start publication of its own team, which bounds rather than eliminates the remaining contamination; its result is reported in Section 6.6.3. A further channel concerns the leave-one-out benchmark rather than the registry: de-duplication collapses records on their repository handle (Section 4.1.3), so two records that describe the same work, such as a conference paper and its journal version, can both remain in the index, and when one of them is the query the other reveals its authors as a near-duplicate. This

channel is not quantified; it inflates the proxy, not the registry, and it acts in the same direction as the conclusion on the cross-encoder (Section 6.6.2). A different channel concerns the generation stage: a language model may know a funded project from its training data. This cannot be measured directly, and the only available check, on the projects that start after the last corpus year, is reported in Section 6.4.2.

Chapter 6

Results and Discussion

This chapter reports the results of the evaluation methodology described in Chapter 5, organised around the decisions that the system embodies rather than around the benchmarks that produced them. The two benchmarks, both built from data fixed before any retrieval, play complementary roles:

- Section 6.2 compares the design choices of the pipeline on 2,000 leave-one-out queries, enough to separate configurations that differ by a few points of MAP, and sets each result next to the same comparison on the real input. It answers the question *why is the system built this way*.
- Section 6.3 evaluates the resulting configuration end to end on 1,220 funded projects, against an administrative reference and with the corpus cut at the project start. It answers the question *how well does it work*.
- Section 6.4 measures the one stage that the ablation cannot exercise on call-like input, the generation of the query.
- Section 6.5 looks at what the residual failures are made of, and traces part of the answer to the corpus itself.
- Section 6.6 draws the two evaluations together, examines where they disagree, and states the limitations of both.

Before the results, Section 6.1 summarises the corpus, the models, and the evaluation sets shared by all the experiments, together with the conventions used throughout the chapter.

6.1 Experimental Setup

Table 6.1 summarises the corpus, the models, and the evaluation sets. The corpus covers fifteen years of institutional scientific production. After de-duplication (Section 4.1.3) and abstract enrichment, about three quarters of the publications carry an abstract, and the others are represented by title, keywords, and venue only.

Component	Value
Corpus	
Unique publications	60,272
With abstract	44,869 (74.4%)
Publication years	2010–2024
Author profiles in the registry	2,980
Models	
Dense encoder	BGE-M3
Cross-encoder re-ranker	bge-reranker-v2-m3
Lexical model	BM25 (Okapi)
Query generator (local)	Llama 3.1 8B
Author-level benchmark (leave-one-out)	
Query publications	2,000
Publication years sampled	2010–2024
Ground-truth authors per query (mean)	2.28
End-to-end benchmark (project registry)	
Funded projects	1,220
Project start years	2015–2027
Reference researchers per project (mean)	1.58
Funding families / departments covered	8 / 11
Proposals in Italian	366

Table 6.1: Corpus, models, and evaluation sets.

Conventions. The following conventions hold throughout the chapter.

- On the registry, the *evaluated configuration* is the reference configuration of Section 5.4, that is, the hybrid retriever on the query generated by the local model, with the parameters of Table 5.2. The same parameters define the reference configuration of the ablation. The raw proposal text, with the same parameters, is the baseline against which every other registry condition is tested.

- On the registry, the metrics are computed on the first ten positions of the expert ranking, the shortlist a user actually reads, with R@50 as a depth diagnostic (Section 5.2). On that benchmark MAP is therefore MAP@10.

- A difference is called significant only when its p-value, corrected with the Holm procedure over the family to which the comparison belongs (Section 5.2), stays below 0.05. Where both values are quoted, p_{Holm} is the corrected one.

- The queries of the language-model stage are generated once and stored, so that every evaluation run is deterministic given the stored queries (Appendix B).

Factor	Setting	MAP	S@1	S@10	ΔMAP	p
Evaluated configuration		**0.574**	0.559	0.884		
Fusion weight	$\alpha = 0$ (lexical)	0.535	0.523	0.854	−0.039	< 0.001
	$\alpha = 0.3$	0.561	0.548	0.870	−0.013	< 0.001
	$\alpha = 0.7$	0.557	0.541	0.874	−0.017	< 0.001
	$\alpha = 1$ (dense)	0.524	0.507	0.851	−0.050	< 0.001
Aggregation depth	25 documents	0.628	0.623	0.895	+0.054	< 0.001
	50 documents	0.603	0.595	0.892	+0.030	< 0.001
	200 documents	0.551	0.540	0.879	−0.023	< 0.001
Authorship credit	Fractional	0.566	0.547	0.889	−0.008	0.002
	Full	0.558	0.546	0.871	−0.016	< 0.001
Recency weighting	Half-life 5	0.549	0.543	0.869	−0.024	< 0.001
	Half-life 10	0.568	0.555	0.880	−0.006	0.034
	Half-life 20	0.572	0.558	0.883	−0.002	n.s.
Combination rule	Maximum	0.556	0.501	0.874	−0.018	0.034
	Mean	0.305	0.175	0.696	−0.269	< 0.001
	Sum of top three	0.614	0.581	**0.896**	+0.040	< 0.001
Cross-encoder	Enabled	**0.639**	**0.636**	0.880	+0.066	< 0.001

Table 6.2: Ablation of the design choices on 2,000 leave-one-out queries, each row against the evaluated configuration. Paired bootstrap on the per-query average precision, with Holm correction over the sixteen comparisons.

6.2 How the Pipeline Was Designed

This section reports the ablation of Section 5.3. Each alternative in Table 6.2 changes one choice of the reference configuration, on the same 2,000 queries and the same retrieved lists, so its difference from the reference is attributable to that choice alone. Each of the paragraphs below closes with the choice the system makes and, where the same comparison was run on the registry benchmark of Section 6.3, with its outcome there: a level that the ablation favours is adopted only if the registry does not contradict it (Section 5.3). No alternative level is adopted, so the reference configuration of Table 6.2 is also the configuration evaluated on the registry. Table 6.3 collects the comparisons repeated on the registry.

Fusion weight. Hybrid retrieval gives the clearest result of the ablation. In Table 6.2 the balanced fusion is the best of the five settings on every metric. Lexical-only retrieval loses 0.039 MAP and dense-only retrieval 0.050, and even the two intermediate weights lose a small but significant amount. On the raw proposal text the registry points the same way, with hybrid retrieval ahead of the lexical branch by 0.042 MAP and of the dense branch by 0.030, both significant (Table 6.3). Two ground truths of different origin, one taken from authorship and one from an administrative register, thus agree on the balanced fusion. No other result of this work supports the hybrid design as strongly, and it comes directly from confronting references built from different records (Chapter 5). The benchmarks disagree only

on the weaker branch, which is the dense one on the leave-one-out benchmark and the lexical one on the registry. The system keeps the balanced fusion.

Aggregation depth. On this benchmark a shorter document list helps, and the shorter the list, the larger the gain. Fifty documents gain 0.030 MAP over one hundred and twenty-five gain 0.054, while two hundred documents lose 0.023. Twenty-five documents are nevertheless excluded, for a structural reason. At about two internal authors each, they can hardly supply the fifty experts that the aggregation returns, and most recommendations would then rest on a single publication, which would also be the only evidence shown to the user. Fifty documents avoid this problem and were tested on the registry. There they leave the top of the ranking unchanged on both inputs (Table 6.3) but reach fewer reference experts, with S@10 down by 0.018 on both inputs and R@50 by 0.032 and 0.033, all significant. The gain of a shorter list is specific to abstract queries. The evaluation keeps one hundred documents, while the application aggregates fifty to keep the interface responsive, at a small cost in coverage. At the other end the ablation leaves no doubt, since two hundred documents let in enough weakly relevant material to damage the ranking.

Authorship credit. On MAP the square-root scheme is the best of the three. Full counting comes last, as expected, because it gives the same credit to every member of a large collaboration and so inflates the score of researchers who co-author many papers with large teams. Fractional counting loses less, 0.008 MAP ($p = 0.002$), a difference that only a sample of this size can resolve. The deployed default thus rests on a measurement as well as on the argument of Section 4.5.1.

Recency weighting. On this benchmark the recency weighting harms the ranking. It loses 0.024 MAP at a half-life of five years and 0.006 at ten, and only the flattest decay, twenty years, cannot be told apart from no weighting at all. The leave-one-out index keeps the publications that follow the query, so for a query drawn from an earlier year the weighting rewards documents published *after* it. The cost indeed comes from the early queries (at a half-life of ten years, -0.051 MAP on those of 2010–2012 against $+0.023$ on those of 2022–2024), and when the later publications are removed from the ranking the same weighting gains 0.019 MAP. Measuring the age of each publication from the year of the query instead of the last year of the corpus would leave the ranking unchanged, because with an exponential decay a change of reference year multiplies every weight by the same constant. On the registry, where the temporal cut removes the publications that follow the start of each project (Section 5.4), the weighting is neutral on both inputs (Table 6.3). It remains the default of the application, a ten-year half-life, as a declared preference for currently active researchers, and every evaluation in this chapter is run without it, so that its proxy cost does not confound the other comparisons.

Combination rule. How the contributions of several documents turn into the score of a researcher matters more than most parameters. Averaging them is by far the worst choice (-0.269 MAP). Dividing by the number of documents throws away exactly the signal carried by a sustained body of work on a topic. Taking the

maximum also does worse than the sum (-0.018), since one well-placed paper is weaker evidence than several. Summing only the three best contributions gains 0.040 MAP and gives the best S@10 of the table. Both results have the same explanation, which is also behind the gain of a shallower aggregation depth. Accumulating evidence helps, up to the point where a tail of weakly relevant documents starts to dilute it. On the registry, however, the sum of the top three loses 0.027 MAP on the raw proposal text and 0.028 on the generated query, both significant (Table 6.3). A rule that helps on paper abstracts hurts on the operational input, so the system keeps the plain sum.

Cross-encoder. Re-scoring the first fifty candidates with the cross-encoder and re-aggregating raises MAP by 0.066, the largest single gain of the table, and raises the success rate at the first position by almost eight points. The gain comes through two channels. The re-ordered documents enter the aggregation carrying the scores of the cross-encoder, which span the unit interval, whereas the reciprocal-rank scores of the fused list are almost flat (the score of the first document is about twice that of the fiftieth) and remain on the documents below the cut-off. Enabling the stage therefore also concentrates the author scores on the first fifty documents, an effect of the same kind as a shallower aggregation depth. The stage, however, is not reduced to a shorter list. On a separate sample of 500 leave-one-out queries, drawn with the same criteria from the publications of 2018–2024, an aggregation over the first fifty documents gains $+0.025$ MAP over one hundred, and the same fifty documents scored by the cross-encoder add a further $+0.023$ (95% CI $[+0.002, +0.044]$, $p = 0.032$). On paper-abstract queries, the scores of the cross-encoder weight the evidence better than the nearly flat fused scores, although this comparison does not separate which documents the stage places on top from how much weight their scores carry.

Cross-encoder on the registry. On the registry the same stage, with the same windows, does not help on either input (Table 6.3). The effect is -0.007 MAP on the raw proposal text and -0.019 on the generated query, measured against the same query without re-ranking, and neither is significant. On the generated query the stage also loses the depth gained by the generation stage, with S@10 falling from 0.523 to 0.489 and R@10 from 0.455 to 0.421. At matched depth, where the stage keeps its gain on abstracts, it still does not help on proposals. Compared with an aggregation over the same fifty documents, it changes MAP by -0.005 on the raw text and by -0.027 on the generated query, and the second loss is significant. The window does not explain the result either. It was controlled on both benchmarks, and on the registry the smaller window only makes things worse. At 512 tokens the raw-text effect becomes -0.030 ($p < 0.001$), because the long proposal queries leave the document side of the pair truncated to half. The stage therefore stays disabled for call-like input, and Section 6.6.1 examines why the two benchmarks disagree.

		Raw text		Generated query	
Factor	Setting	ΔMAP@10	p_{Holm}	ΔMAP@10	p_{Holm}
Fusion weight	BM25 only	−0.042	< 0.001		
	Dense only	−0.030	< 0.001		
Aggregation depth	50 documents	−0.004	0.900	+0.000	0.971
Recency weighting	Half-life 10	+0.006	0.077	+0.005	0.550
Combination rule	Sum of top three	−0.027	< 0.001	−0.028	< 0.001
Cross-encoder	Enabled	−0.007	0.900	−0.019	0.058
	Enabled, depth 50	−0.005	0.900	−0.027	< 0.001

Table 6.3: Design choices of Table 6.2 tested on the registry benchmark against their own baselines, with Holm correction over the thirteen decisional contrasts of Table B.1, twelve of which are shown.

6.2.1 Error attribution

The decisions examined so far all act on the stage that turns retrieved documents into a ranking of authors. This subsection shows why that stage matters more than retrieval. A reference author can miss the shortlist in two ways. Either no retrieved document lists them among its authors, and then no aggregation could repair the loss, or they are among the candidates and the aggregation ranks them too low. Table 6.4 separates the two cases with the measurement described in Section 5.3.

Stage	Success	Recall
Reachable from the retrieved documents	**0.963**	**0.932**
Ranked within the first ten	0.884	0.764
Ranked within the first fifty	0.948	0.904

Table 6.4: Error attribution on 2,000 leave-one-out queries. “Success” is the fraction of queries with at least one reference author, “Recall” the mean fraction of the reference authors.

Retrieval is close to saturation. For 96.3% of the queries the retrieved documents include at least one reference author, and on average they contain 93.2% of the reference authors, drawn from a median of 115 candidate authors per query. The shortlist of ten recovers 76.4% of the reference authors. The other 16.8 points of recall are lost while ordering candidates that were already there, and at fifty positions the gap is down to 2.8 points. Most missing authors are thus in the candidate set, just below the tenth position. On this benchmark a larger retrieval has little left to give. The margin is in the document-to-author aggregation, the same stage to which the registry attributes its ordering loss (Section 6.5).

6.3 The System on the Real Task

This section evaluates the complete pipeline on the 1,220 funded projects of Section 5.4, with the proposal text as input and the corpus cut at the project start. It is the only experiment of this work in which the input belongs to the real genre, the reference was produced with no retrieval in the loop, and the sample is large enough for corrected significance testing. Table 6.5 reports the evaluated conditions, and Table 6.6 the paired tests against the raw-text baseline. The rows of the single branches and of the cross-encoder are discussed in Section 6.2, and the generated query in Section 6.4.

Condition	MAP@10	MRR	S@1	S@5	S@10	R@10
Hybrid, generated query	**0.242**	**0.279**	**0.173**	**0.421**	**0.523**	**0.455**
Hybrid, raw text	0.233	0.271	0.170	0.403	0.489	0.420
Hybrid + cross-enc., raw text	0.226	0.258	0.168	0.384	0.472	0.413
Hybrid + cross-enc., generated	0.223	0.256	0.172	0.367	0.489	0.421
Dense only, raw text	0.203	0.235	0.139	0.366	0.451	0.388
BM25 only, raw text	0.191	0.220	0.139	0.330	0.399	0.350
Popularity floor	0.011	0.015	0.009	0.023	0.048	0.034
Random floor	0.001	0.001	0.000	0.004	0.005	0.002

Table 6.5: End-to-end evaluation on the 1,220 funded projects, with the corpus cut at the project start. The metrics are computed on the first ten positions; the conditions with a generated query are computed on 1,219 projects, because one generation failed.

Condition vs. raw text	ΔMAP@10	95% CI	p	Holm
Generated query	+0.009	$[-0.002, +0.021]$	0.115	n.s.
+ cross-enc., raw text	−0.007	$[-0.018, +0.004]$	0.225	n.s.
+ cross-enc., generated	−0.009	$[-0.024, +0.005]$	0.207	n.s.
Dense only	−0.030	$[-0.039, -0.021]$	< 0.001	sig.
BM25 only	−0.042	$[-0.054, -0.030]$	< 0.001	sig.
Popularity floor	−0.222	$[-0.242, -0.203]$	< 0.001	sig.
Random floor	−0.232	$[-0.252, -0.213]$	< 0.001	sig.

Table 6.6: Paired bootstrap against the raw-text condition on the per-query average precision. Holm correction over nineteen conditions (Table B.1), of which the table reports seven.

Absolute values. The evaluated configuration places a correct researcher first in 17.3% of the projects, and at least one correct researcher in the top ten for 52.3% of them. With 1.58 listed researchers per project, a name ranked first and absent from the register may well be a legitimate expert, so these precision-like readings are lower bounds (Section 5.5). The system also finds correct researchers far more often than it ranks them in the shortlist, with R@50 at 0.654 against 0.455 for R@10.

For about a fifth of the projects it finds a reference expert but places them between positions ten and fifty, and this loss belongs to the aggregation stage (Section 6.5).

Floors. The ranking reflects topical pertinence, and productivity alone does not explain it. The evaluated configuration exceeds the popularity floor by a factor of about twenty on MAP and eleven on S@10, and the random floor by more than two orders of magnitude. Document-centric aggregation is known to favour prolific authors, but here it cannot be reduced to that bias, since ranking the most productive researchers of the period answers almost none of the queries.

Registered researchers only. About a third of the names in the first ten positions (34.9% on the raw text) belong to researchers the register does not profile, who can never count as a success (Section 5.4). Seven conditions were therefore repeated with the ranking restricted to the 2,980 profiled researchers, with the Holm correction over the seven contrasts among them.

- S@10 rises from 0.489 to 0.551 on the raw text, from 0.523 to 0.598 with the local generator, and from 0.550 to 0.605 with the hosted one.
- The conclusions hold. Hybrid retrieval still beats both single branches, and the hosted generator still beats the raw text, all with $p_{\text{Holm}} \leq 0.002$. The cross-encoder still brings no significant change (+0.001 MAP on the raw text, −0.015 on the generated query), and the gain of the local generator stays unresolved (+0.013, $p_{\text{Holm}} = 0.107$).
- Only the comparison between the two generators changes status, since the hosted one no longer separates from the local one (+0.013, $p_{\text{Holm}} = 0.107$).

Temporal leakage. Running the same pipeline *without* the temporal cut raises MAP from 0.233 to 0.284 (+0.051, $p < 0.001$), and the figure understates the effect. On the 287 projects that start after the last corpus year the cut has nothing to remove, and the runs with and without it coincide exactly (MAP 0.193 in both), which also checks the implementation. On the 933 projects where the cut actually bites, the difference is +0.067 ($p < 0.001$), an inflation of about 27% over the leak-free baseline. Most of the inflation comes from publications that did not exist when the project started, many of them produced by the project itself. An effect of this size makes the cut necessary. Without it every condition would be overstated, and not by the same amount, since projects with many follow-up publications would gain more.

6.4 The Enriched Query

The query-generation stage rewrites the free-text proposal into the structured, enriched query described in Section 4.2.1. This section measures what the rewriting is worth on the real input, first with the reproducible local generator and then with a hosted backend, and what the difference implies for deployment.

6.4.1 Effect of the generated query

Primary metric. With the local 8B generator the effect on MAP is positive but not significant, at +0.009 (95% CI $[-0.002, +0.021]$, $p = 0.115$ before correction). In a head-to-head count, 277 projects improve, 213 get worse, and 729 do not change, so the effect is real but small and diffuse. On the 366 Italian proposals MAP rises from 0.143 to 0.159 (+0.016, $p = 0.139$). The direction fits the repair mechanism described in Section 6.5.2, although this sample is too small to establish its size.

Top and depth of the shortlist. Although the average effect is small, it follows a consistent pattern across the metrics (Table 6.7). At the top of the shortlist the generated query has no measurable effect, since neither S@1 nor MRR changes significantly. Further down, the gain is consistent and survives the Holm correction over the eight metrics tested, with +0.035 on S@10 and on R@10, +0.039 on R@50, and +0.015 on nDCG@10. The generated query leaves the first position as it is and widens the set of correct experts that enter the shortlist at all. The cross-encoder behaved in the opposite way on the leave-one-out benchmark, improving only the first positions (Section 6.2), so the two stages act on opposite ends of the ranking.

Metric	Raw text	Generated	Δ	p	Holm
S@1	0.170	0.173	+0.003	0.755	n.s.
MRR	0.271	0.279	+0.008	0.245	n.s.
S@5	0.403	0.421	+0.018	0.073	n.s.
S@10	0.488	0.523	+0.035	0.001	sig.
R@10	0.420	0.455	+0.035	< 0.001	sig.
nDCG@10	0.289	0.304	+0.015	0.008	sig.
R@50	0.615	0.654	+0.039	< 0.001	sig.

Table 6.7: Effect of the query-generation stage by metric ($n = 1{,}219$), with Holm correction over these seven metrics and average precision.

6.4.2 Sensitivity to the generation backend

The same structured query is also generated by a hosted large language model, Claude Haiku 4.5, and scored with the same protocol on the 1,218 projects for which both backends produced a valid query (Table 6.8). The two runs are not a controlled comparison of the models alone. The hosted backend was served through a different path, which truncates the proposal text at three thousand characters (a limit that binds on 12.3% of the proposals) and constrains the output format by instruction, whereas the local generator enforces a decoding-time schema. Its queries were also produced with separate instructions (Section 5.4). The run therefore bounds what a stronger generation stage is worth on this benchmark, without attributing the difference to the model alone.

Condition	MAP@10	MRR	S@1	S@10	R@10	R@50	Δ	Holm
Raw text	0.233	0.271	0.170	0.488	0.419	0.615		
Generated query								
Local (Llama 3.1 8B)	0.242	0.279	0.173	0.524	0.455	0.654	+0.009	n.s.
Hosted (Claude Haiku 4.5)	**0.260**	**0.295**	**0.187**	**0.549**	**0.479**	**0.669**	+0.027	sig.

Table 6.8: Effect of the generation backend on the 1,218 projects common to both conditions. Δ is the difference in MAP@10 against the raw text, tested with the Holm correction over nineteen conditions (Table B.1).

Against the raw text. A stronger backend also improves the top of the shortlist. With the hosted backend the effect of query generation becomes significant on the primary metric too, at +0.027 MAP against the raw proposal text ($p_{\text{Holm}} < 0.001$), whereas the +0.009 of the local backend does not survive the same correction. The stage widens the depth of the shortlist (Section 6.4.1), and under this configuration it widens the top as well.

Against the local backend. The direct comparison between the two backends supports this reading. The hosted backend gains 0.018 average precision (95% CI $[+0.006, +0.030]$, $p = 0.004$), 0.025 on S@10 ($p = 0.017$), and 0.016 on MRR ($p = 0.018$), while R@50 stays level (+0.015, $p = 0.116$, not significant). At depth fifty the two runs retrieve the same correct experts, and the hosted one simply ranks more of them near the top. Its advantage lies in the ordering, and it exceeds anything the local backend gains over the raw text. The truncation of the input does not account for it. On the 1,067 proposals shorter than three thousand characters, which both backends read in full, the hosted backend still gains 0.019 average precision over the local one ($p = 0.003$).

Robustness. The margin over the local backend is less robust than the gain over the raw text.

- With the ranking restricted to the registered researchers, the margin narrows to +0.013 and does not survive the Holm correction (Section 6.3), while the gain over the raw text remains significant.
- The run cannot exclude contamination, since a hosted model may know the earlier funded projects from its training data. On the 287 projects that start after the last corpus year its gain over the raw text is +0.005 MAP ($p = 0.61$), against +0.034 on the earlier ones, a difference between the two groups of +0.028 ($p = 0.027$). The local generator shows the same direction, with a smaller and non-significant difference (+0.000 against +0.012, $p = 0.35$), and the later projects score lower overall (Section 6.5.2), so the check cannot separate contamination from the composition of the two groups.

Role of the two backends. The local backend is the reproducible reference of this evaluation (Section 4.2.4), because its output can be regenerated offline under the same settings. A model billed per call, which may change behind the same API,

offers no such guarantee. The deployment calls a hosted model through an API, and the run with Claude Haiku 4.5 is the only measurement available of that path. Its queries were stored, and every figure of this section is computed from them, so the reported results do not depend on the backend remaining unchanged.

Operational reliability. The same campaign also measured the reliability of the local generator:

- it failed on exactly one of the 1,227 pre-merge proposals, a failure rate of 0.08%;
- at temperature 0.7, its run-to-run variance over three independent generations on 400 proposals spans 0.005 MAP and 0.023 S@10, about half and about two thirds, respectively, of the measured effect of the stage;
- at temperature zero, the setting of the reference configuration, a second generation on the same 400 proposals reproduces the same query for 397 of the 399 with a valid output in both runs.

6.5 Where the System Fails, and Why

The aggregate figure of Table 6.5 hides two stages that behave differently, and the distinction matters because they call for different remedies. Table 6.9 decomposes the outcome.

Outcome	Projects	Share
Registered member ranked in the top ten	638	52.3%
Retrieved in the top fifty, but not top ten	249	20.4%
Outside the top fifty	332	27.2%

Table 6.9: Decomposition of the outcome on the registry benchmark ($n = 1{,}219$).

Position of the registered member	Projects	Share of failures
11–25	160	27.5%
26–50	89	15.3%
51–100	86	14.8%
101–200	43	7.4%
Never produced as a candidate	203	**34.9%**

Table 6.10: Position of the registered member in the untruncated ranking, on the 581 projects without a member in the top ten.

The success rate is the product of two shares, the projects with a registered member among the first fifty experts and, among these, the projects where that member reaches the first ten ($0.728 \times 0.719 = 0.523$). To see where the other projects are lost, the aggregation was re-run without truncation, ranking every candidate that the retrieved documents produce, and the position of the registered member

was recorded. Table 6.10 reports it for the 581 projects without a member in the top ten.

Most failures fall into two distinct kinds, in comparable proportion, with a smaller group between them:

- In 42.9% of the failures the person is produced as a candidate between positions eleven and fifty. The evidence linking them to the topic is in the corpus, but the aggregation ranks it below other evidence. This ordering loss is the margin that Section 6.6.2 attributes to the aggregation stage.
- In a further 22.2% of the failures the person falls between positions fifty-one and two hundred, where the evidence exists but ranks lower still.
- In 34.9% of the failures (203 projects, 16.7% of the benchmark) the registered member is *never produced as a candidate*. The aggregation yields a median of 139 candidates per project, drawn from the authors of the hundred retrieved documents, and the person is not among them. Every reference member has at least two prior publications, so the person is in the corpus. Their record, however, contains nothing that the topic of their own project selects.

The following subsections relate this to the way the corpus represents different kinds of scientific output, and to the language and the funding instrument of the project.

6.5.1 Corpus representation

The corpus is not a uniform record of the institution's activity. As reported in Table 6.1, 25.6% of the publications carry no abstract after the enrichment cascade and are represented by title, keywords, and venue alone. The share of output that takes the form of an indexed article also varies widely across disciplines. Book chapters and monographs, which carry an abstract far less often than journal articles, weigh more in some departments, while exhibitions, catalogues, and design works fall outside the publication types that the corpus indexes (Section 4.1.1). Both properties were fixed before any retrieval, and both vary systematically by department. Table 6.11 pairs, for each department, the documentary profile of its researchers with the effectiveness on the projects they sign.

Association with effectiveness. The association is weaker than the sample can certify. Across the eleven departments, the correlation between abstract coverage and S@10 is $r = 0.52$ ($p = 0.10$), and the share of indexed journal articles shows no association at all ($r = -0.07$, $p = 0.84$). DIST and DAD have the least complete corpus record (0.63 coverage, against 0.86 for DET and DENERG), and DIST is also the department on which the system performs worst (S@10 = 0.350, against 0.632 for DENERG, the best). DAD, however, stays within five points of the benchmark average (S@10 = 0.482 against 0.523), so a thin record does not by itself condemn a department to poor results. These are the departments of territorial planning and of architecture and design, whose output is in substantial part books, competition entries, exhibitions, and built work. The books often enter the corpus without an abstract, and the other artefacts are not among the publication types it indexes, so a system reading titles and abstracts cannot represent them.

The record. Part of the failure is plausibly a property of the record itself. A researcher whose publications are stored without an abstract is represented in the index by a title of a dozen words. No weighting of the lexical and semantic branches can recover from a representation that thin, and the deficit is concentrated in identifiable disciplines instead of being spread at random. Its size is not established, however. Across eleven departments the association with the success rate is not significant, so the share of the 27.2% that the pipeline could not have avoided remains unmeasured.

The pairing. DISMA has high abstract coverage yet an above-average failure rate, and DAUIN sits at the departmental average despite the highest coverage of all, so representation is not the whole story. For those cases a plausible explanation lies on the other side of the pairing. The registry attaches to a project the person who signs the contract, and on the collaborative instruments this need not be the person whose publications describe the work. Section 5.5 treats this as a construct-validity limit of the ground truth.

Department	Projects	Abstract cov.	Journal share	Never retrieved	S@10
DIST	60	0.633	0.412	0.167	0.350
DIATI	84	0.834	0.601	0.143	0.452
DISMA	30	0.856	0.753	0.233	0.467
DAD	56	0.626	0.373	0.286	0.482
DAUIN	99	0.904	0.370	0.182	0.495
DIGEP	88	0.742	0.566	0.216	0.500
DISAT	283	0.791	0.761	0.159	0.516
DIMEAS	199	0.797	0.489	0.161	0.518
DISEG	59	0.755	0.364	0.203	0.559
DET	124	0.863	0.428	0.129	0.621
DENERG	136	0.855	0.527	0.118	**0.632**

Table 6.11: Documentary profile of each department and effectiveness on the projects its researchers sign. Each project is counted under the department of its first registered member, 1,218 of 1,219.

6.5.2 Language and funding instrument

The effectiveness of the system also varies with the language of the proposal and with the funding instrument of the project.

Condition	English ($n = 854$)	Italian ($n = 366$)
BM25 only, raw text	0.260	0.030
Dense only, raw text	0.227	0.147
Hybrid, raw text	0.271	0.143
Hybrid, generated	**0.277**	**0.159**

Table 6.12: MAP@10 by proposal language on the registry benchmark.

Language. On genuinely Italian proposals, hybrid retrieval on the raw proposal text loses a factor of 1.9 with respect to English (0.143 against 0.271). On the generated query, the reference configuration, the factor narrows to 1.74 (0.159 against 0.277; Table 6.12). The decomposition shows where the loss comes from. BM25 alone collapses to 0.030 on Italian input, a factor of 8.7 below its English value, while the dense branch holds at 0.147, essentially the level of the full hybrid. On Italian queries the lexical branch therefore contributes nothing, which points to a mismatch of surface forms between Italian queries and a corpus whose indexing text (Section 4.1.4) is predominantly English. The gap is not only lexical, however. The dense branch also falls, from 0.227 on English proposals to 0.147 on Italian ones, so part of the loss affects the multilingual encoder as well, or depends on what distinguishes the Italian proposals from the English ones.

Language and funding instrument are intertwined, since 131 of the 140 regional contracts are in Italian, but the gap persists within two of the three largest instruments with proposals in both languages. On the raw text:

- EU-collaborative projects score 0.239 in English against 0.060 in Italian (375 and 27 proposals);
- national projects score 0.333 against 0.155 (214 and 91 proposals);
- on industrial contracts the gap nearly closes, 0.248 against 0.213 (68 and 65 proposals).

Query generation, which rewrites the dense field in English and adds bilingual terminology, recovers a small and non-significant part of the loss on real proposals (Section 6.4.1).

Funding instrument. Table 6.13 reports the breakdown by funding instrument, and Figure 6.1 shows the same decomposition as three outcomes per project. ERC grants stand out at S@10 = 0.714 and 88.6% retrieval, against 0.518 on the remainder (+0.197, 95% CI $[+0.040, +0.340]$, $p = 0.021$ on a two-sample bootstrap). Once again the explanation is in the data. An ERC grant has one holder (the registry lists 1.03 members per contract, against 2.19 for regional contracts), so the pairing between the project and the person is unambiguous, and the recorded person is by construction the scientific author of the proposal. It is plausibly the subset of the benchmark with the least noisy ground truth. With 35 projects, its figure describes the effectiveness on an unambiguous pairing and should not be read as an estimate for the whole benchmark.

Team size. Team size acts differently on the two metrics. MAP falls from 0.269 for a single listed researcher to 0.124 for four, since recovering every member of a larger team in ten positions is harder than recovering one; S@10, which only asks for one correct name, does not fall (per-project correlation $\rho = +0.015$, $p = 0.599$, $n = 1{,}219$). Across funding instruments the mean number of registered members does correlate strongly with S@10 (Pearson $r = -0.816$, $p = 0.013$ over the eight families of Table 6.13), but this is an ecological correlation, which vanishes within

the instruments. The number of members is an indicator of the kind of instrument, not a cause of the failure, and concluding from it that the system degrades on larger teams would be an error of inference that the project-level data does not support.

Start year. No monotone trend is detectable across individual project start years once the composition of the later cohort is accounted for (per-project correlation $\rho = -0.021$, $p = 0.455$, $n = 1{,}219$). The 287 projects starting after the corpus ends do score lower (0.193 MAP against 0.257). Standardising the earlier group on the composition of the later one, however, reproduces the same gap, so the gap depends on what those projects are and not on when they start.

Funding instrument	n	Members	Retrieved @50	S@10
EU-ERC	35	1.03	**0.886**	**0.714**
EU-MSCA	47	1.49	0.830	0.617
National	305	1.33	0.754	0.567
Industrial	132	1.54	0.705	0.553
PNRR	122	1.93	0.738	0.541
Other	36	2.06	0.694	0.528
EU-collaborative	402	1.49	0.744	0.505
Regional	140	2.19	0.571	0.357

Table 6.13: Effectiveness by funding instrument, with the mean number of registered members per contract ($n = 1{,}219$).

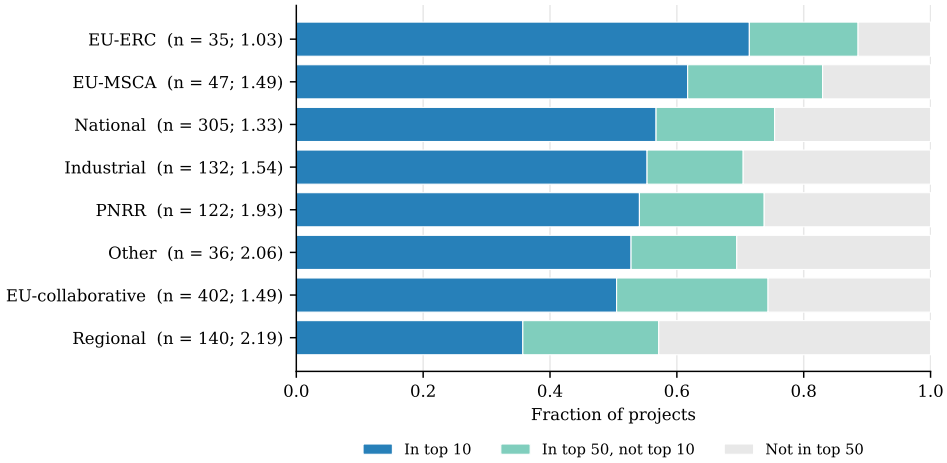


Figure 6.1: Outcome by funding instrument; each label reports the number of projects and the mean number of registered members per contract.

6.6 Discussion

The two evaluations of this chapter rest on ground truths of different origin, one derived from authorship and the other from an administrative register, both fixed

before any retrieval. Where they can be compared, on hybrid retrieval, they agree (Section 6.2). On the cross-encoder they disagree, and the disagreement is itself informative.

6.6.1 The re-ranking discrepancy

The most instructive result of this evaluation is an apparent contradiction between the two benchmarks. The cross-encoder gains 0.066 MAP on the leave-one-out benchmark (Table 6.2) and does not help on the registry, on either input (Section 6.2). The disagreement is not an effect of the length of the aggregated list, since at matched aggregation depth the stage still gains on abstracts (+0.023) and still does not help on proposals (−0.005 on the raw text, −0.027 on the generated query). It concerns the top of the ranking, MAP and S@1, while on S@10 neither benchmark shows a gain.

Query genre. The two benchmarks differ above all in the genre of their queries. A leave-one-out query is a paper abstract, the same genre as the corpus, and it looks much like the query–passage pairs on which a cross-encoder is trained. A proposal is administrative prose, and the generated dense field (Section 4.2.1) is a synthetic profile of an ideal expert. Both are far from that training distribution. This is the most direct reading of the result, even though the benchmarks also differ in their reference and in the temporal cut. The stage carries over the assumptions of its training data, and on proposals those assumptions no longer hold.

Near-duplicate records. A further reading, which does not exclude the first, is the near-duplicate channel declared in Section 5.5. A record that describes the same work as the query and remains in the index is exactly what a cross-encoder places on top, and it would inflate the proxy at the first positions, where the gain is measured. Its size is not established, and it points to the same deployment choice (Section 6.6.2).

6.6.2 Implications

Where the margin lies. The margin lies mostly in the aggregation. On the leave-one-out benchmark retrieval is close to saturation, and the loss lies in the ordering of candidates already present (Section 6.2.1). On the registry the same holds for the 42.9% of the failures in which the registered member is a candidate between positions eleven and fifty. It does not hold for the 34.9% in which the member is never produced as a candidate (Section 6.5). There the limit is the retrieved evidence itself, so the saturation measured on the proxy does not extend to the registry. Part of that loss is plausibly a property of the corpus (Section 6.5.1). Improving the aggregation therefore addresses the ordering loss, and cannot address a person absent from the retrieved evidence altogether.

Deployment. Three of the choices examined in this chapter carry a direct deployment implication:

- the cross-encoder should stay disabled for call-like input (Section 6.6.1);
- the generation stage should use the strongest generator the deployment can afford, since the effect grows from a diffuse and non-significant gain with the local generator to a significant one with Claude Haiku 4.5 (Section 6.4.2), and a frontier model is the natural next step (Section 7.3);
- the recency weighting stays in the application as a declared user preference, with no claim that it improves the ranking, since it costs on the proxy and brings no compensating benefit on the registry (Section 6.2).

6.6.3 Limitations

The limitations fall into two groups. Those of the evaluation bound the strength of the conclusions, and those of the system bound its current usefulness in practice.

Limitations of the evaluation.

- The registry reference is incomplete (1.58 listed researchers per project, plausibly the principal investigator), so precision-like figures are lower bounds, and success and recall are depressed as well whenever an unlisted expert occupies a position of the shortlist. The check with the ranking restricted to the profiled researchers (Section 6.3) bounds this effect at about six points of S@10.
- Some contamination remains through projects whose acronym already appears in pre-start publications of their team, typically continuations of an earlier grant. They are 33, or 2.7% of the benchmark, and excluding them moves the headline MAP from 0.233 to 0.228 without changing any conclusion.
- The BM25 collection statistics come from the full corpus and are not recomputed for each temporal cut. This is a declared, second-order anachronism.
- The two benchmarks are not on the same scale. At the same cut-off the pipeline reaches a higher success rate on the leave-one-out benchmark (Table 6.4) than on the registry (Table 6.5), so the leave-one-out figures say which design choice is better, while the reading closest to deployment is the one on the registry. The near-duplicate records of Section 5.5 may account for part of the distance.
- Sampling the generator at temperature 0.7 adds a stochastic component whose run-to-run variance is about half of the measured effect of the stage. The reference configuration decodes at temperature zero, where a second generation almost always reproduces the same query (Section 6.4.2).
- No human relevance judgment was available for this work, so both references are records and not assessments. The feedback loop of Section 7.3 is the intended way to obtain one.

Limitations of the system.

- *Corpus coverage and missing text.* The retrieval can only rank what the corpus contains, and a quarter of the publications have no abstract, so their relevance signal rests on the title alone. A researcher whose most relevant work is under-represented in the institutional repository is penalised for reasons unrelated to the retrieval model (Section 6.5.1).

- *Author disambiguation and cold start.* Relevance reaches a person through the identifiers with which the institutional repository links each publication to its authors. The pipeline does not resolve names to identifiers itself, so any error made upstream in that link propagates to the ranking. Both author-level protocols also keep in the reference only researchers with a bibliographic record (at least one further publication on the leave-one-out benchmark, two publications before the project start on the registry). The reported figures therefore say nothing about researchers with almost no record.

- *A single configuration for heterogeneous input.* The balance between the lexical and the dense branch is fixed once for every query, and on Italian input the lexical branch contributes nothing (Table 6.12), so the reported figures average over inputs on which the two branches contribute very unevenly. Structural remedies exist, such as indexing translated surrogates or a query-dependent fusion weight (Section 4.3.3); neither is part of the present system.

- *Recency weighting.* The optional weighting, meant to surface currently active researchers, structurally disadvantages established experts whose most relevant work is older, and on the proxy its cost exceeds its benefit (Section 6.2). It remains the application default because the preference it encodes is legitimate, while no benchmark of this work shows a significant benefit on the proposal input, where the weighting is neutral.

Chapter 7

Conclusions and Future Work

7.1 Summary of the Work

The system built in this thesis takes the free text of a funding call and returns the researchers of Politecnico di Torino whose publications are most relevant to it, each with the papers that justify the recommendation. It follows the document-centric approach introduced in Chapter 1. BM25 and BGE-M3 retrieve publications from a corpus of more than 60,000 records covering fifteen years, reciprocal rank fusion merges the two lists, and an optional cross-encoder can re-rank the top of the fused list. The relevance of the retrieved documents is then aggregated at the author level, with configurable weights for authorship credit and recency. The corpus comes from the institutional repository, enriched with abstracts from OpenAlex, IRIS, and Scopus, and users reach the system through an interactive web application (Chapter 4).

Most of the work went into the evaluation, since no public benchmark exists for this corpus. Chapter 5 built two ground truths from different institutional records, and neither needs a relevance judgment. The first is a large leave-one-out benchmark derived from the authorship record of the corpus. The second contains 1,220 funded projects from the institutional register, where the reference experts are fixed administratively and the input is the real proposal text. There the corpus is cut at the project start, so the system has to find the team from what it had published before. Every comparison between configurations is tested with a paired bootstrap, with a correction for multiplicity whenever it belongs to a family of conditions.

The experiments confirm the retrieval design. Hybrid retrieval makes the clearest contribution, and the two author-level benchmarks agree on it. On 2,000 leave-one-out queries the balanced fusion improves over lexical-only retrieval by 0.039 MAP and over dense-only retrieval by 0.050. On the raw text of the 1,220 funded projects the same comparisons give +0.042 and +0.030, all with $p < 0.001$. With the generated query, the evaluated configuration places a listed researcher first in 17.3% of the projects and in the top ten for 52.3%. That is eleven times the popularity baseline on the same measure and two orders of magnitude above random, so the aggregation measures topical pertinence and cannot be reduced to productivity. The ablation also settles the authorship credit. It excludes full counting and separates the square-root scheme from the fractional one, which loses 0.008 MAP ($p = 0.002$),

so the deployed default is backed by a measurement as well as by the argument of Section 4.5.1.

Two earlier beliefs had to be corrected. The cross-encoder produces the largest single gain of the ablation on paper-abstract queries (+0.066 MAP), and the gain survives at matched aggregation depth. It does not transfer to the operational input, however, where it helps neither on the raw proposal text nor on the generated query, at either depth. Query generation with the local backend, in turn, has a small and non-significant effect on the primary metric (+0.009 MAP, $p = 0.115$) and leaves the top of the shortlist unchanged. Its contribution is in the depth of the shortlist, with +0.035 on S@10 and R@10 over 1,219 funded projects, significant after the Holm correction.

A hosted backend, Claude Haiku 4.5, also recovers part of the top of the shortlist, with +0.027 MAP on the raw text and +0.018 over the local backend. It was measured once, through a different channel and with separate instructions, and its margin over the local backend disappears when the ranking is restricted to the registered researchers. This single run is the only measurement of the hosted path (Section 6.4.2). It suggests that a stronger generator ranks the experts better, although it does not isolate the model as the only factor that varies, nor exclude that the model already knew the earlier projects. A frontier model is the natural next step.

The third correction concerns a default of the application. The recency weighting, applied by default with a ten-year half-life, costs 0.006 MAP on the leave-one-out proxy at that half-life and up to 0.024 at five years, because the later publications remain in the index of that benchmark. On the registry it is neutral (Section 6.2). It stays enabled by default as a declared user preference, without the support of a measured gain.

The correction on the cross-encoder and the reassessment of the recency default both came from confronting benchmarks built from different records, which is the central methodological argument of this work. The correction on query generation came from the registry alone, the one benchmark that can exercise that stage on the real input. In short, the leave-one-out benchmark compares the design choices with the power of a large sample, and the registry, used in parallel, measures on the real input the choices that depend on it.

7.2 Limitations

The relevance judgments of both benchmarks are binary, which makes rank-sensitive metrics such as nDCG less informative. On the registry, the ground truth lists on average 1.58 researchers per project, plausibly the principal investigator, so every precision-like figure on that benchmark is a lower bound. Attributing a publication to a person depends on the identifiers with which the institutional repository links each publication to its authors, and any error in that link carries over to the ranking. On Italian proposals the effectiveness drops by a factor of 1.74 on MAP@10, and the lexical branch adds nothing to the dense one, so the hybrid rests on the multilingual encoder alone (Section 6.5.2). Finally, the system ranks researchers only by the topical relevance of their publications. It does not model availability, seniority, or

collaboration networks, and it does not address fairness, for example the balance across departments, career stages, and gender.

7.3 Future Work

Several extensions follow naturally from this work:

- **A competence graph.** The document-centric pipeline could be complemented by a graph that connects the researchers to one another, so that expertise can be followed through people as well as through documents. Relevance can be propagated across documents and candidate experts over several steps [33], or each researcher can be profiled by the entities mentioned in their publications, as in WISER [34], a system built for the researchers of an Italian university. At present, however, most of the information needed to connect the researchers is missing, including the composition of the research groups, comprehensive information on department memberships, and the collaborations that go beyond the co-authorships recorded in the publications. The field of self-declared keywords of the institutional personnel service, which would add self-reported expertise to such a graph, is also filled for only 1,062 of the 2,980 profiles in the registry (35.6%).

- **Data from sources not accessible at present.** About a quarter of the publication records (25.6%, Table 6.1) still lack an abstract, and the corpus could be enriched by sources that are not currently integrated. More generally, extending the author registry used as ground truth (Section 5.4) is a problem of data access more than of modelling.

- **Effectiveness as a primary direction.** The lever this work has actually measured is the quality of the query-generation backend (Section 6.4). A hosted model, Claude Haiku 4.5, measured in a single run, improves over the local reference configuration. With the reference configuration at 52.3% on S@10, considerable room remains, and a frontier model is the most promising extension of the generation stage. Two variants are worth testing before a deployment is fixed. A larger local model would keep the pipeline self-contained, and a frontier model called through the application's own client would make the model the only thing that changes.

- **Feedback loop in production.** The web application could collect lightweight relevance feedback from the research-support staff, such as accepted and rejected recommendations, building over time an in-house benchmark whose labels are produced by the people who actually use the shortlists. Such feedback would also supply graded relevance judgments, which the current binary ground truths cannot provide, and could be used to learn the aggregation weights directly from usage instead of setting them by design.

Appendix A

Query Generation Prompt

This appendix reports, for reproducibility, the full prompt used by the generated query mode of Section 4.2. The prompt is filled with the call text and with the maximum number of keywords, and is sent to the language model under a schema constraint that forces the output to conform to the six-field structure of Section 4.2.1. The text below is transcribed verbatim, with the only change that a few non-ASCII characters (such as the "less than or equal" sign and the en-dash) are rendered in their ASCII form.

```
You are an expert query builder for an academic expert-finding system at
Politecnico di Torino (PoliTo).

RETRIEVAL PIPELINE CONTEXT:
- BM25 sparse search over titles, abstracts, keywords (keyword matching --
  language-sensitive)
- BGE-M3 dense semantic search (cross-lingual: handles both Italian and English
  documents)
- Goal: identify the most relevant faculty experts for a given research call or
  topic

FIELD-BY-FIELD INSTRUCTIONS:

main_topic (<= 8 words):
  Core research domain, concise and specific. No verbs, no generic words.

dense_query (3-5 sentences):
  A rich semantic paragraph for embedding-based retrieval.
  - Describe the ideal expert's research profile as if writing an abstract
  - Cover: main domain, key methodologies, application areas, materials/systems,
    and goals
  - Include both high-level concepts and specific technical details
  - Avoid generic phrases ("this paper studies...", "we propose...")

sparse_query (space-separated keyword string):
  Technical terms optimized for BM25 exact matching.
  - No stopwords, no filler words, no punctuation
  - Prioritize rare/specific terms (BM25 rewards low-frequency terms)
  - Include materials, compounds, algorithms, standards, instrument names
  - Add Italian equivalents of key terms where relevant (corpus is partially
    Italian)

keywords (up to {max_keywords} items):
  Individual technical terms. Include:
  - Core domain terms and their direct synonyms
  - Specific materials, compounds, components, organisms, or datasets
  - Methods, algorithms, models, and experimental techniques
  - Italian equivalents of key terms when the corpus may contain Italian papers

phrases (multi-word expressions, 2-4 words each):
```

```
  Compound technical expressions that appear verbatim in academic
  titles/abstracts. Include Italian variants if relevant.

acronyms:
  Relevant acronyms as they appear in literature (e.g. "CRM", "LCA", "DFT").
  List only the acronym, not the expansion.

STRICT RULES:
- Write dense_query in English only.
- For keywords and sparse_query, include Italian synonyms where the concept is
  commonly published in Italian at Italian universities.
- Never use: "study", "paper", "research", "approach", "method", "work",
  "analysis" as standalone keywords -- they add noise to BM25.
- Be specific: prefer "lithium iron phosphate" over "battery chemistry".

Input text:
{text}
```

Appendix B

Evaluation Pipelines

This appendix documents, at the operational level, the two pipelines that produce the results of Chapter 6, so that each result can be traced to the procedure that produced it. The reasons behind each design choice are given in Chapter 5. Here the emphasis is on the sequence of steps, on what each step produces, and on the parameters that control it. Both pipelines start from the offline indices of Section 4.1.5 and are independent of each other, so that neither evaluation depends on the results of the other.

B.1 The Leave-One-Out Pipeline

This pipeline produces the author-level, known-item recall results of Section 5.3 and Section 6.2. It uses no language model and no external service: every step runs offline on the corpus and the indices. It proceeds as follows.

1. **Index loading.** The indices, the corpus, the embedding model, and the author registry are loaded once; the cross-encoder is loaded only if the re-ranked variant is evaluated.

2. **Sampling.** The query publications are drawn from the journal articles of the corpus that have a substantial abstract, fall within the chosen years (2010–2024 for the reported results), and have at least one PoliTo author with a further publication in the corpus. These authors, identified by their matricole, form the ground truth of the query.

3. **Retrieval.** Title and abstract of each query publication form the query, and the hybrid retriever runs exactly as it does at deployment time. The query publication is then removed from the retrieved list, so its authors can only be found through their other work.

4. **Aggregation and scoring.** The aggregation of Section 4.5 turns the retrieved documents into a ranked list of experts, which is scored against the ground truth with average precision, reciprocal rank, success rate, and recall at n.

5. **Aggregation over queries.** The per-query metrics are averaged over the whole sample. The paired bootstrap of Section 5.2 then tests the contribution

of the cross-encoder and of the retrieval parameters, and the large query set gives it the power it needs.

The ablation runs on one cached retrieval. Every row of Table 6.2 comes from a single pass over the sampled queries. The pipeline runs the retrieval once and stores the ranked candidates of each branch, and every grid then reads these candidates instead of repeating the retrieval. The cross-encoder row belongs to the same pass. It re-scores the cached candidates and re-aggregates them, so it shares the sample and the retrieval with the rest of the table. The error attribution of Table 6.4 uses the same cached candidates, comparing the authors reachable from the retrieved documents with those the aggregation ranks.

Answerable ground truth. An author whose only publication is the query cannot be recovered once that publication is removed. Keeping such authors in the ground truth would measure a gap in the data instead of the retrieval. The sampling therefore keeps only the authors with at least one further publication, so every query is answerable and the metrics describe the retrieval alone.

B.2 The Project-Registry Pipeline

This pipeline produces the end-to-end benchmark and results of Section 5.4 and Section 6.3. It proceeds as follows.

1. **Harvesting.** The public personnel web service of the institution is queried for every researcher of the author registry, returning the funded projects each one is assigned to, with title, abstract, funding type, and start and end dates.

2. **Benchmark construction.** The researcher–project links are grouped by project, and contracts sharing the same proposal abstract are merged into one project with the union of their member lists. The admissibility filters of Section 5.4 (abstract length, start-year window, at least one answerable member) then select the final projects. Each project stores as ground truth the matricole of its answerable members, together with the year at which its corpus is cut.

3. **Query generation.** The local model generates the structured query of each proposal once, and the query is cached with its latency and failure status. Every later evaluation is therefore deterministic, and the operational properties of the stage come as a by-product. Independent repetitions at zero and at non-zero temperature measure the run-to-run variance.

4. **Evaluation.** Every condition (query mode, retrieval branch, cross-encoder setting, reference floors) runs the evaluated pipeline with the corpus cut at the project start. The author-level metrics are computed on the first ten positions, and each condition is tested against the raw-text baseline with the paired bootstrap and the Holm correction.

5. **Stratified reporting.** The per-query results are broken down by language, funding family, start year, team size, department, and leakage-free status, which gives the tables of Sections 6.3 and 6.5.

Leakage controls. The temporal cut of Section 5.4 is applied to the retrieved lists before rank fusion, so the fused ranks are those of an index built on the past. The no-cut condition quantifies the leakage that the cut removes, and the projects starting after the last corpus year serve as a control on which the two runs must coincide. A textual scan for project acronyms in pre-start publications of the ground-truth members bounds the residual contamination.

Table provenance. To place a registered member beyond position fifty, the aggregation is re-run without a truncation limit, for a single declared condition. The four aggregate tables of Section 6.5 all come from one declared condition, so each table can be traced to the condition it summarises. Throughout Chapter 6 the Holm correction is applied step-down, and the adjusted p-values are kept monotonically non-decreasing along the ordered family of comparisons. Table B.1 lists the families over which it is computed.

Complementarity. Two of these pipelines carry the conclusions of this work, and they rest on ground truths of different origin: the authorship record of the corpus and the administrative project register. Agreement between the configuration rankings they produce is evidence that neither is an artefact of its own construction. Disagreement, as in the case of the cross-encoder, localises a property of the benchmarks themselves and is informative in its own right. Confronting references built in these different ways is the core of the evaluation strategy adopted in this work (Section 5.5).

Family	Size	Comparisons
Leave-one-out ablation	16	Each alternative level against the reference configuration: four fusion weights, three aggregation depths, two credit schemes, three recency half-lives, three combination rules, and the cross-encoder (Table 6.2).
Registry, raw text	19	All against the raw text: the local and the hosted generated query; on the raw text and on the local generated query, the cross-encoder with the 1,024-token and with the 512-token window, the cross-encoder at aggregation depth fifty, recency, aggregation depth fifty, and the top-three combination rule; dense only; BM25 only; no temporal cut; the popularity and random floors (Tables 6.6 and 6.8).
Registry, decisional	13	Each design choice against its own baseline, on the raw text and on the local generated query: the cross-encoder, the top-three combination rule, recency, aggregation depth fifty, and the cross-encoder at depth fifty against depth fifty alone (ten contrasts); query generation against the raw text; dense only and BM25 only against hybrid retrieval (Section 6.2).
Registry, metrics	8	The generated query against the raw text on AP, MRR, S@1, S@5, S@10, R@10, nDCG@10, and R@50 (Table 6.7).

Table B.1: Families over which the Holm correction is computed.

Bibliography

[1] K. Balog, Y. Fang, M. de Rijke, P. Serdyukov, and L. Si, "Expertise retrieval," *Foundations and Trends in Information Retrieval*, vol. 6, no. 2–3, pp. 127–256, 2012.

[2] K. Balog, L. Azzopardi, and M. de Rijke, "Formal models for expert finding in enterprise corpora," in *Proc. 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2006, pp. 43–50.

[3] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: BM25 and beyond," *Foundations and Trends in Information Retrieval*, vol. 4, no. 1–2, pp. 1–174, 2009. DOI: 10.1561/1500000019

[4] J. Chen, S. Xiao, P. Zhang, K. Luo, D. Lian, and Z. Liu, "M3-Embedding: Multi-linguality, multi-functionality, multi-granularity text embeddings through self-knowledge distillation," in *Findings of the Association for Computational Linguistics: ACL 2024*, 2024, pp. 2318–2335.

[5] G. V. Cormack, C. L. A. Clarke, and S. Büttcher, "Reciprocal rank fusion outperforms Condorcet and individual rank learning methods," in *Proc. 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009, pp. 758–759.

[6] J. Priem, H. Piwowar, and R. Orr, "OpenAlex: A fully-open index of scholarly works, authors, venues, institutions, and concepts," *arXiv preprint arXiv:2205.01833*, 2022.

[7] N. Craswell, A. P. de Vries, and I. Soboroff, "Overview of the TREC 2005 enterprise track," in *Proc. Fourteenth Text REtrieval Conference (TREC 2005)*, 2005.

[8] C. Macdonald and I. Ounis, "Voting for candidates: Adapting data fusion techniques for an expert search task," in *Proc. 15th ACM International Conference on Information and Knowledge Management (CIKM)*, 2006, pp. 387–396. DOI: 10.1145/1183614.1183671

[9] K. Balog, T. Bogers, L. Azzopardi, M. de Rijke, and A. van den Bosch, "Broad expertise retrieval in sparse data environments," in *Proc. 30th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2007, pp. 551–558. DOI: 10.1145/1277741.1277836

[10] D. Mimno and A. McCallum, "Expertise modeling for matching papers with reviewers," in *Proc. 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2007, pp. 500–509.

[11] S. Ebrahimi, S. Salamat, N. Arabzadeh, M. Bashari, and E. Bagheri, "ExHarmony: Authorship and citations for benchmarking the reviewer assignment problem," *arXiv preprint arXiv:2502.07683*, 2025.

[12] L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, F. J. Ribadas-Peña, and N. Bolaños, "Information retrieval and machine learning methods for academic expert finding," *Algorithms*, vol. 17, no. 2, p. 51, 2024.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," in *Proc. 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2019, pp. 4171–4186.

[14] N. Reimers and I. Gurevych, "Sentence-BERT: Sentence embeddings using siamese BERT-networks," in *Proc. 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP-IJCNLP)*, 2019, pp. 3982–3992.

[15] V. Karpukhin et al., "Dense passage retrieval for open-domain question answering," in *Proc. 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6769–6781.

[16] A. Cohan, S. Feldman, I. Beltagy, D. Downey, and D. S. Weld, "SPECTER: Document-level representation learning using citation-informed transformers," in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 2270–2282.

[17] M. Douze et al., "The faiss library," *arXiv preprint arXiv:2401.08281*, 2024.

[18] H.-L. Hsu and J. Tzeng, "DAT: Dynamic alpha tuning for hybrid retrieval in retrieval-augmented generation," *arXiv preprint arXiv:2503.23013*, 2025.

[19] R. Nogueira and K. Cho, "Passage re-ranking with BERT," *arXiv preprint arXiv:1901.04085*, 2019.

[20] L. Gao, X. Ma, J. Lin, and J. Callan, "Precise zero-shot dense retrieval without relevance labels," in *Proc. 61st Annual Meeting of the Association for Computational Linguistics (ACL)*, 2023, pp. 1762–1777.

[21] P. Lewis et al., "Retrieval-augmented generation for knowledge-intensive NLP tasks," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 33, 2020, pp. 9459–9474.

[22] 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.

[23] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, and M. Gatford, "Okapi at TREC-3," in *Proc. Third Text REtrieval Conference (TREC-3)*, 1994, pp. 109–126.

[24] A. Vaswani et al., "Attention is all you need," in *Advances in Neural Information Processing Systems (NeurIPS)*, vol. 30, 2017.

[25] A. Conneau et al., "Unsupervised cross-lingual representation learning at scale," in *Proc. 58th Annual Meeting of the Association for Computational Linguistics (ACL)*, 2020, pp. 8440–8451.

[26] E. A. Fox and J. A. Shaw, "Combination of multiple searches," in *Proc. Second Text REtrieval Conference (TREC-2)*, 1994, pp. 243–252.

[27] A. Perianes-Rodriguez, L. Waltman, and N. J. van Eck, "Constructing bibliometric networks: A comparison between full and fractional counting," *Journal of Informetrics*, vol. 10, no. 4, pp. 1178–1195, 2016.

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

[29] N. Fuhr, "Some common mistakes in IR evaluation, and how they can be avoided," *ACM SIGIR Forum*, vol. 51, no. 3, pp. 32–41, 2018.

[30] K. Järvelin and J. Kekäläinen, "Cumulated gain-based evaluation of IR techniques," *ACM Transactions on Information Systems*, vol. 20, no. 4, pp. 422–446, 2002.

[31] M. D. Smucker, J. Allan, and B. Carterette, "A comparison of statistical significance tests for information retrieval evaluation," in *Proceedings of the 16th ACM Conference on Information and Knowledge Management (CIKM)*, 2007, pp. 623–632.

[32] S. Holm, "A simple sequentially rejective multiple test procedure," *Scandinavian Journal of Statistics*, vol. 6, no. 2, pp. 65–70, 1979.

[33] P. Serdyukov, H. Rode, and D. Hiemstra, "Modeling multi-step relevance propagation for expert finding," in *Proc. 17th ACM Conference on Information and Knowledge Management (CIKM)*, 2008, pp. 1133–1142. DOI: 10.1145/1458082.1458232

[34] P. Cifariello, P. Ferragina, and M. Ponza, "WISER: A semantic approach for expert finding in academia based on entity linking," *Information Systems*, vol. 82, pp. 1–16, 2019. DOI: 10.1016/j.is.2018.12.003