SCUOLA DI DOTTORATO
UNIVERSITÀ DEGLI STUDI DI MILANO-BICOCCA

Department of
Informatics, Systems and Communication
Ph.D. program in Computer Science, Cycle XXXV

A CONTENT-BASED RECOMMENDATION MODEL FOR LIVING EVIDENCE IN THE HEALTH CARE DOMAIN

Author: Paolo Tenti
Registration number: 744757

Tutor: Prof. Fabio Stella
Supervisor: Prof. Gabriella Pasi
Co-Supervisor: Prof. Rafael Peñaloza

Coordinator: Prof. Leonardo Mariani

Academic Year 2021/22

A CONTENT-BASED RECOMMENDATION MODEL FOR LIVING EVIDENCE IN THE HEALTH CARE DOMAIN

Paolo Tenti

January 2024

Acknowledgements

I acknowledge the Evidence for Policy and Practice Information (EPPI) Center, based at the Social Science Research Unit within the UCL Social Research Institute, for their constant collaboration in this research and for providing the computational resources.

I would like to extend my appreciation to the Cochrane Collaboration for generously providing access to the extensive data on studies included in thousands of their systematic reviews.

I would like to thank my supervisor, Prof. Gabriella Pasi, for inspiring me to embark on this incredible journey, for assisting me with her always effective guidance, and for leading me to the finish line.

I would like to thank my supervisor, Prof. Rafael Peñaloza, for his continuous help, genuine support, and the many opportunities for discussion and sharing of ideas.

I would like to express my gratitude to Prof. James Thomas, who has played a crucial role in guiding this research from the very beginning. His extensive domain expertise, profound curiosity, and genuine support have been invaluable.

A heartfelt thank you goes to my family for their loving support, understanding... and limitless patience!

Pursuing a PhD has been a lifelong dream for me. I was fortunate enough to have the opportunity to pursue this path. I am deeply grateful to everyone who accompanied me on this journey.

To my family

Abstract

Systematic reviews (SR) summarise the knowledge available in the literature on a specific topic. Keeping SRs up to date with new publications as soon as they become available—a problem known as *living SR*—is fundamental to avoid their early obsolescence. Recently, automated workflows have been proposed to deal with one or a few living SRs. However, there is a need to scale these workflows across science to maintain large numbers of living SRs across entire domains of research in a task known as *living evidence*.

The typical workflow to update an existing SR—or to create it from scratch—is usually the following: (i) identify the bibliographic databases which are relevant for updating the SR; (ii) search useful new citations, which were published after the last SR update; (iii) perform citation screening to discard the citations which are clearly irrelevant to the SR; (iii) perform abstract screening to assess in more details the remaining citations after the previous step; (iv) manually review all the publications selected in the previous step to decide which ones to include in the SR.

The traditional approaches to automate *living SRs* require reviewers active participation, to carry on the steps in the above workflow. The reasons for this active participation are multiple. First, the available bibliographic databases are many and etherogeneous, and each SR has its own relevant ones. In addition, the search queries used to find the interesting citations are inherently complex, and requires huge efforts to be developed and tested. Moreover, the algorithms used for screening citations and abstracts often compromise on efficiency to achieve near-perfect effectiveness, which requires to manually assess many irrelevant publications and to fine-tune the screening algorithms on the specific SR.

This traditional process for updating a single SR requires huge efforts and resources. Thus, it is usually applied periodically, when a SR requires to be updated and a project can be allocated to it. Hence, the reviewers efforts to

attend the SR update process are usually expected. However, this approach is too specific to its target SR to be fully useful in large *living evidences*. In fact, especially in life sciences, *living evidences* comprise thousands of SRs, with new SRs being released on a daily basis.

The expectation for a *living evidence* is that it runs automatically, and it provides to reviewers frequent recommendations about the new evidences to assess for inclusion in their SRs. If the *living evidence* system is not efficient, reviewers would receive too many inaccurate recommendations every day, and they would lose interest. On the other hand, compromising effectiveness to reduce the number of recommendations is not really an option.

This research proposes *ContReviews*, an automated system to manage *living evidences* in the health care domain. Specifically, *ContReviews* is based on an academic knowledge graph and a content-based recommendation model.

The **academic knowledge graphs** allow the quick identification of new publication for entire domains of research, without the need to identify SR-specific bibliographic databases. Specifically, OpenAlex [1] is used, though it is not restricted to the health care domain.

The **content-based recommendation model** generalizes the notion of relevance assessment, avoiding SR-specific models to infer new publications relevance to SRs. Specifically, a content-based recommendation model matches items to recommend (i.e., new publications) to user profiles (i.e., SRs) based on a formal representation of their content; moreover, a relevance assessment function is learnt from data, based on matching outcomes.

In the context of *ContReviews*, the content-based recommendation model leverages publication already included in the *living evidence* to learn a unique model to assess new publications relevance to each SR in the *living evidence*. In one hand, this lets to avoid designing and testing complex search queries over bibliographic databases. On the other hand, content-based matching is a general concept which applies to every SR in a *living evidence*, letting to avoid developing, training and evaluating SR-specific citation and abstract screening models.

Moreover, as SRs usually have few dozens of publications included (this is especially true in life sciences SRs), training SR-specific models is challenged by their data scarcity, leading to sub-optimal performance. On the contrary, as the content-based recommendation model is unique, it leverages the entire *living evidence* for training and evaluation.

To represent publications and SRs more faithfully, *ContReviews* lever-

ages multiple of their features, which includes titles, abstracts, citations and authors. *ContReviews* leverages both bag of words and embeddings to represent textual features (i.e., titles and abstracts), and combine them with the vector representations of entities (i.e., authors and citations). The system is extensible, in that additional publication features can be used.

These faithful representations aim to achieve efficiency with near-perfect effectiveness when assessing the relevance of new publications to SRs. Specifically, new publications and SRs are matched on all the available features, producing multiple 'likelihoods of relevance'. These are used by a binary classification model to infer the final likelihood of relevance. The classifier is based on machine learning, and is trained over the entire *living evidence*.

Among the others, textual features—such as titles and abstracts—are particularly useful to represent publications content. Pre-trained language models based on the transformer neural architecture (such as BERT and GPT) provide dense and contextual vectors to represent language expressions (*embeddings*). They are adapted to a specific application domain (such as *living evidences*) through fine-tuning, to provide more faithful embeddings. To achieve such adaptation, a fine-tuning dataset and a fine-tuning task are used.

This research proposes and evaluate two fine-tuning approaches. The first, is fine-tuning SciBERT for abstract screening, using a dataset based on the Cochrane Reviews.[1] The second one is fine-tuning LongFormer for semantic similarity, also using the Cochrane Reviews. Note that SciBERT is pre-trained on scientific publications, though not necessarily all of them belong to the health care domain as the Cochrane Reviews do. In addition, LongFormer supports longer input sequences than SciBERT, which better align to the average length of the Cochrane Reviews abstracts.

ContReviews has been evaluated over a large dataset of Cochrane Reviews, to assess its ability to correctly infer relevance of some new publications to each one of the Cochrane Reviews. Its performance was compared to two baseline models, which align to the traditional approaches for abstract screening. To infer relevance to SRs, the first baseline calculates the cosine similarity between the embeddings of SRs and publications; while the second baseline consists in training one binary classification model per SR, which

[1] Cochrane Reviews are SRs of research in health care and health policy, published in the Cochrane Database of Systematic Reviews (https://www.cochranelibrary.com/about/about-cochrane-reviews).

uses the embeddings of the included publications as features.

Following, the main research questions and results are reported.

- Independence on the target SR is an important design factor for a *living evidence* system because, as argued above, it should run without any active reviewers participation. *ContReviews* design and evaluation shows that the same content-based recommendation model is used to match all the new publications to all the SRs, without any reviewer intervention to either design queries or optimize SR-specific systems.

- While the previous factor is true by design, the most important aspect is *ContReviews* effectiveness, i.e., the system ability to correctly capture all the relevant publications for each SR. Traditional systems, which are based on search queries specifically designed for specific SRs, achieve near-perfect effectiveness. This is usually measured in terms of the classification metric of 'precision' and 'recall', where many researches aim for precision as high as possible with at least 95% recall. *ContReviews* evaluation results show it can achieve better precision than the traditional approaches with recall of 100%.

- The evaluation's ablation studies show the importance of the *ContReviews* content-based matching mechanism to achieve good precision with high recall. Specifically, when compared to the baseline methods, the greatest contribution to achieving high precision with high recall (i.e., precision above 97% with recall of 100%) is given by the content-based matching by itself. In addition, using multiple features to represent publications further helps to improve precision by a smaller factor.

- As mentioned, embedding models are helpful to represent textual features and complement the bag of words based representations: while the former focus on text semantics, the latter accounts for word statistics. The fine-tuning approaches introduced above beat the pre-trained model. Specifically, evaluation's ablation studies show that the most important factor is fine-tuning with the *living evidence* publications (i.e., Cochrane Reviews), to adapt to the domain specific distributional semantics. However, the specific fine-tuning task and base model (i.e., the supported size of the input sequences) did not leave a particular

footprint on performance, steering to fine-tune with computationally efficient models, rather than more sophisticated ones.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Literature reviews and evidence syntheses are important research practices, aiming to advance science based on the available knowledge. In this context, **Systematic Reviews** (SRs) have emerged as a distinctive approach in health sciences to provide a rigorous and comprehensive way of assessing the literature [3, 4]. SRs are characterized by being methodical, comprehensive, transparent, and replicable; this systematic approach aims to minimize subjectivity and bias [5]. Unlike SRs, traditional literature reviews are often driven by their authors experience, missing the same systematic approach.

Developing SRs and maintaining them up to date with the most current knowledge is an important challenge, given the size of the available research, the pace at which new studies are published, and ultimately the complexity of the task. To address these issues, **living evidences**—which are libraries of Systematic Reviews—have been recently proposed, aiming to provide researchers, practitioners, and policy makers with a current and comprehensive review of the available knowledge across entire domains of research. However, *living reviews* are relatively rare and there is not large-scale grant funding that would enable researchers and partners to build out living evidence across science.

This research focuses on the problem of maintaining *living evidences* current with the latest researches, studies, and publications (these terms will be used interchangeably). This chapter introduces SRs and *living evidences*, to clarify the application context of this study. Then, the existing methods for updating SRs are briefly reviewed and their limitations for dealing with living evidences explained. Finally, the specific research questions that this study aims to address are presented.

1.1 Application Context

Systematic Reviews (SRs) can be applied to any field that can benefit from evaluating the existing literature based on specific inclusion criteria, aiming to expand the current knowledge. SRs can assess different types of evidence, such as clinical trials, public health interventions, environmental interventions, social interventions, adverse effects, policy reviews, and economic evaluations. However, health care is the most mature domain where SRs are applied. Looking at the type of SRs produced by the Cochrane Collaboration,[1] the leading international network that creates and shares SRs in health care, provides a clear idea about the variety of the topics in the health domain that are addressed with SRs.

SRs are a recent development in the field of evidence-based medicine, which aims to base clinical decisions on the most reliable scientific data. Before SRs were developed, evidence-based medicine used to synthesise research by means of conventional literature reviews, which provided only a broad summary of a topic, rooted to the authors' perspective, without trying to cover all the available evidence or describing the methods used to choose and combine studies. Thus, conventional literature reviews may yield biased outcomes, subjective opinions, and untested protocols which do not help to inform clinical practices in the best way possible. To overcome these issues, researchers proposed to treat the review process as a scientific process in itself, which evolved into the SR process [6].

1.1.1 Systematic Reviews (SRs)

In Cochrane's words:

> *a systematic review attempts to identify, appraise, and synthesize all the empirical evidence that meets pre-specified eligibility criteria to answer a specific research question. Researchers conducting systematic reviews use explicit, systematic methods that are selected with a view aimed at minimizing bias, to produce more reliable findings to inform decision making.*

The standard process adopted by reviewers for developing, conducting, and reporting a SR in the health care domain is the following [6].

[1] https://www.cochranelibrary.com/about/about-cochrane-reviews

1. Formulate the research questions, and define precise inclusion and exclusion criteria.

2. Develop a search strategy aimed at covering the broadest possible range of sources which are relevant to the research questions.

3. Assess the studies identified by the search strategy to decide if they meet the inclusion criteria. This step is usually performed in two stages: a first stage where (often thousands of) titles and abstracts are screened, and a second stage where the full texts of studies not excluded in the first stage are evaluated.

4. Report and interpret results by using pre-defined methods to assess the quality of studies and to extract, analyse, and synthesise the data of interest from each included study.

Usually, two or more reviewers collaborate to perform the above tasks, thus, some procedures must be set beforehand to manage disagreements.

The SR process is extremely rigorous and specific, and the number of scientific publications to consider is huge; therefore, SRs often take years to complete, demand large-scale collaboration, and consume significant resources. Conducting SRs involves the following methodological and practical challenges, in addition to those intrinsic to their inevitable large-scale nature.

Search complexity. The initial search for relevant articles can be very long and difficult, and the precision of search is generally low; for example, in some researches [6], about 2% of the publications considered for a SR are ultimately included.

SR updating. A second major challenge is keeping SRs up to date: research does not stop after a SR is initially constructed, and new publications can quickly make the results of many SRs obsolete. With the increasing volume of new studies published every day, updating SRs can be as challenging and time-consuming as creating them from scratch, especially if they have not been updated for a long time. Moreover, as the timing of updating a SR may not be clear—since it is impossible to know how many new and relevant publication have been published [7]—SRs are usually updated when a compelling need arises and a project to do it is funded. Furthermore, there

may not be a consistent group of people being responsible for the SR and, therefore, SR updating would be even less homogeneous in these cases.

These type of complexities make SRs of poor quality, duplicative, and out of date as soon as they are published, or even before [8, 9].

1.1.2 From SRs to *living evidences*

Recently, health sciences have proposed **living evidences** to address the above problems [7, 10]. *Living evidences* are libraries of domain specific SRs, which are current, automated and centrally managed on behalf of an entire community of researchers, practitioners, and policy makers. For example, NICE maintains 350 'guidelines', i.e., recommendations on broad topics covering health, public health and social care in England.[2] In addition, the *Cochrane Database of Systematic Reviews*,[3] is the leading database for SRs of research in health care and health policy. A report from 2010 estimated that about 75 trials and 11 SRs were published every day to the Cochrane Database of Systematic Reviews [11], and a more recent study found an average of 10,000 SRs published annually in the past 22 years [12]. However, some of them might be old, or not actively maintained.

A *living evidence* requires an automated process which involves (i) continuously surveying the research across a whole domain of knowledge, (ii) automatically assessing the relevance of the new publications to the SRs and (iii) flagging the potentially relevant ones for inclusion. Note that, although a *living evidence* is largely automated, the final decision to include new publications in a SR is always a human responsibility. Indeed, a *living evidence* provides a current understanding of the state of the art, in terms of which new scientific publications 'could' be relevant to the owned SRs. This way, the new (potentially relevant) studies can be immediately notified to reviewers as soon as they are published, so that they can update their SRs sooner and faster, avoiding early obsolescence. This fastest pace of 'update operations' plays down the need for periodical large and expensive efforts to update deeply outdated SRs. However, some SRs might not be tightly owned, with reviewers being either not available or loosely committed to the SR update task. In these cases, some new (potentially relevant) publications would remain locked within the *living evidence* with nobody being available

[2] https://www.nice.org.uk/process/pmg20/chapter/glossary#recommendations
[3] https://www.cochranelibrary.com/

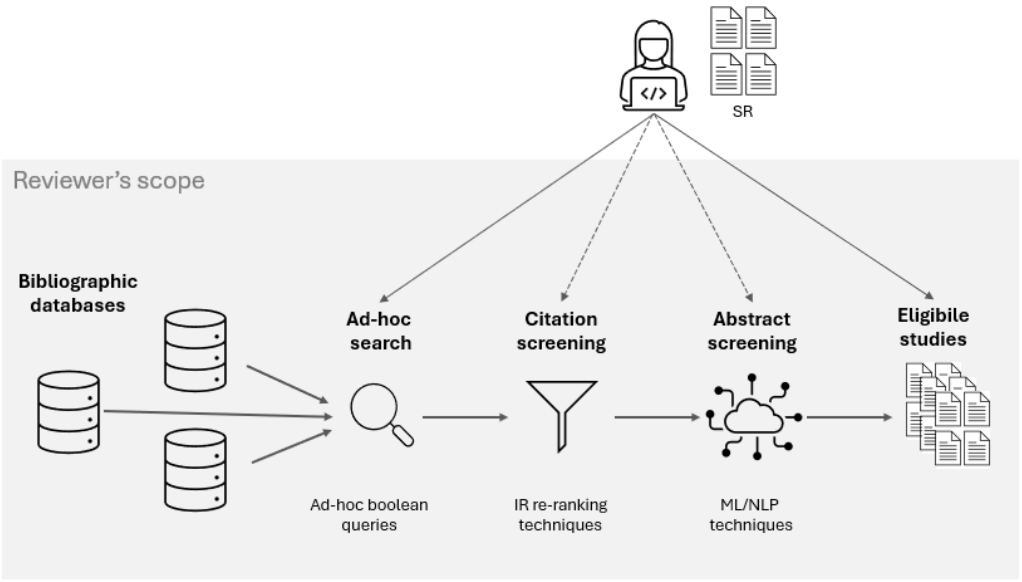

Figure 1.1: Workflow for *SR update*.

to judge for their inclusion.

While *living reviews* can be hugely effective, they require significant computational, organizational, and human resources. For these reasons, *living reviews* are relatively rare and there is not large-scale grant funding that would enable researchers and partners to build out living evidence across science. Creating these funding opportunities is a critical next step to optimizing and scaling this model for the research community, which is unfeasible with the current SR workflows [13].

1.2 Current Approaches to SR Updating

Given the large scale of the SR updating process, technology can certainly play a big role to ensure it can be conducted as much efficiently as possible: to this aim, information retrieval, machine learning, and natural language processing techniques are of great help [14]. Approaches at the state of the art focus on updating individual SRs (or small groups of SRs), rather than whole *living evidences*. One common approach is re-running the same procedure which is undertaken to create a SR: this way, from a technology perspective, there is a little difference between creating or updating a SR.

Creating and updating SRs involves performing the following steps, as represented in Figure 1.1.

- The most useful *data sources*, such as bibliographic databases and journals, are first manually identified. Usually, these data sources are specific to a SR and diverse in their scope, interface, search method and access policy.

- Adequate search queries are formulated and run on the selected data sources, to identify an initial list of *citations*. A common approach in the health care domain is to use Boolean queries, which combine keywords and other terms extracted from domain taxonomies (e.g., PICO, and MeSH terms, as explained in the next chapter). A great experience and domain expertise is required to formulate these queries effectively and efficiently.

- *Citation screening* is done to exclude the clearly irrelevant citations. When search queries are not well formulated, citation screening can perform sub-optimally: ineffective queries lead to missing important citations and inefficient ones expose many (useless) publications to citation screening. Information retrieval techniques help to re-rank all the citations returned by the search queries.

- The remaining publications are evaluated in more details within an eligibility assessment step, usually based on *abstracts screening*. This involves considering publication abstracts in light of the SR inclusion criteria. Natural language processing and machine learning techniques are involved.

- The resulting abstracts are considered by reviewers for the final analysis, to take a decision about their inclusion in SRs. This step is mostly manual and involves considering the whole documents to evaluate if they meet the SR criteria.

Updating SRs sporadically presents little technology advantages over creating them for the first time; for example, SRs are more stable after they have been created, meaning that all of their important components (such as the research questions and inclusion criteria) are established [15]. Apart from this, especially if the SRs remain unchanged for longtime, updating them presents

the same exact challenges than creating them for the first time. Thus, updating SRs periodically is hindered by the complexity of searching across heterogeneous data sources, the large number of citations that are involved in citation screening, the still large number of abstracts to be screened for eligibility assessment and, ultimately, the amount of full documents that must be reviewed manually. To overcome these limitations, *living SRs* [16, 17] have been proposed as a more efficient method to address the problem of updating SRs. Early examples of this new *living* approach are *EPPI-Reviewer*,[4] *Trialstreamer*,[5] and the *Human Behaviour-Change Project* [18]. The main component of a *living SR*, as shown in Figure 1.2, is the continuous surveillance of the research literature to identify new relevant publication as soon as they are available and to run the down-stream steps of citation and abstract screening over a smaller amount of data. The notion of *continuous surveillance* of data sources has been proposed in contrast to periodically producing Boolean queries to update existing SRs [19, 20]. However, beside continuous surveillance, the *living SR* process is still the same as *SR update*. Given this similarity, from now on the term *living SR* will be used to mean the SR updating process.

1.3 Updating *living evidences*

In principle, the *living SR* process could be run at scale over all the SRs comprised within a *living evidence*; however, this would be challenged by the *living evidences* size and the pace at which new SRs are published every day. Indeed, such a scalability issue is expected and the latest cloud technologies make it addressable from a cost and computing perspective. Beyond this 'scalability factor', the *living SR* process is inherently not well suited to support *living evidences*, and a different approach is necessary [13]. In fact, reviewers are actively involved in all the phases of the *living SR* process, as they develop complex search queries and they must manually assess many irrelevant publications. This type of participation of reviewers to the *living SR* process represents a too strong assumption in the case of *living evidences*; in fact, the expectation is that they run almost unattended, i.e., reviewers only receive daily or weekly notifications to check the latest potentially relevant evidences, instead of being actively involved in all of the SR phases.

[4] https://eppi.ioe.ac.uk/cms/Default.aspx?alias=eppi.ioe.ac.uk/cms/er4
[5] https://trialstreamer.ieai.robotreviewer.net/

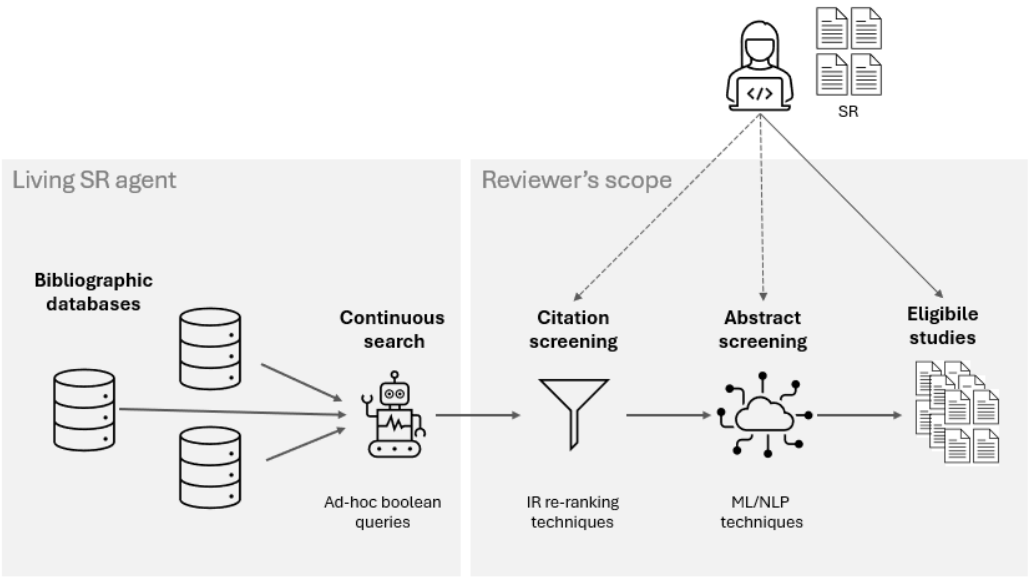

Figure 1.2: Workflow for a *living SR*.

Specifically, this research focuses on the two following important issues which, in the above sense, severely hinder the applicability of the *living SR* process to *living evidences*.

Complexity of searching. As introduced in the Section 1.2, searching for the new evidences is a complicated task due to the need to identify the relevant data sources and, especially, to the complexity of formulating adequate queries over them. Specifically, domain experts in health sciences blend keywords and other terms extracted from domain taxonomies into articulated and well tested Boolean queries. These must be both effective, to retrieve all the relevant evidences, and as efficient as possible to reduce the amount to useless ones. Moreover, leveraging any type of taxonomies introduces a potential bug in the overall retrieval system, because it assumes that all the new publications are correctly labeled by their authors, which might not be true. As *living evidences* must automatically survey whole research domains, which span thousands of SRs, it is not feasible to maintain one set of so complex search queries for each one of them.

Poor efficiency of screening models. To be really useful, *living evidences* should be highly effective in identifying all the relevant new publications and as efficient as possible to reduce the 'false positives' that are mistakenly deemed relevant [12]. In one hand, clearly, loosing any relevant publication would simply not be acceptable. In the other hand, a system that is too lenient in accepting 'false positives' to catch all the relevant new publications would burden human reviewers with many irrelevant publications that they have to assess manually.

SRs in the same sub-domain are often similar to each other, supporting research questions whose differences lay in the details. For this reason, the typical citation and abstract screening models might be unable to clearly discriminate the evidences being relevant to a SR from the ones which are irrelevant; thus, to be effective, the criteria to discard the irrelevant evidences are made less selective. For this reason, the automation techniques employed to keep a *living SR* current, are tightly related to the SR itself: reviewers are encouraged to either optimize the screening models for their SRs, or supervise the entire process to complement the weakness of the system. An example is using *active learning* techniques [14]: machine learning models are used to classify abstracts as relevant and not relevant, however, as they are usually trained with scarce data (SRs in the health care space hold in average a few dozens of publications), they are not always accurate; thus, training steps and manual selection of new relevant publications can be alternated until they reach the inflection point. In the context of *living evidence* this participation of reviewers to the SR updating process is clearly unfeasible, due to the large number of SRs and to the fact that some of them might have not fully dedicated reviewers.

Complexity of queries and inefficient screening models are a challenge for any SR updating method, not only for *living evidences*. However, updating individual SRs usually happens in the context of a project, i.e., in a context where there is budget coverage, temporal boxing, and dedicated human resources. SR updating project stakeholders all expect to have to evaluate many potential data sources, formulate and test complex queries and manage a large amount of poorly screened documents. In fact, this is basically the main scope of their SR updating projects.

Instead, a *living evidence* is an on-going and semi-unattended system. Every week, or every day, new evidence recommendations will be notified to a full community of reviewers, practitioners, and policy makers: if these

recommendations are too many, due to an inefficient system, people will not be able to manage them; if they are too few, due to an ineffective system, people will not trust it. In all of these cases, the *living evidence* system would loose traction with the community. If domain experts were needed to engage on SR-specific projects, either for fine-tuning queries or optimizing screening models, the *living evidence* would loose control of its own update processes.

1.4 Research Questions

To progress in the field of updating entire *living evidences*, the focus of this research is twofold. First, a content-based recommendation system is proposed as a novel approach to identify the new publications to suggest to reviewers. Second, a new method for representing publications and SRs is introduced. In the next two paragraphs, the specific research questions concerning these objectives are presented.

Content-based recommendation system. A content-based recommendation system leverages the content of the publications already included in a SR for: (i) identifying the new potentially relevant publications, among all the newly published researches; and (ii) screening these new potentially relevant publications to find the most useful ones, to be considered by reviewers for manual assessment. To this aim, the main research problem is whether such a system can replace complex search queries (such as Boolean queries), citations screening, and abstracts screening. Specifically, the following research questions are relevant to this problem.

- Can a content-based recommendation system be applied uniformly to all the SRs in a *living evidence* or, instead, it still requires SR-specific optimizations? This is relevant for addressing the issue of the current *living SR* systems which depend too much on the specific SR, which means they need constant reviewer supervision.
- Can they achieve good efficiency with near-perfect effectiveness, at least as much as the current systems?

Representation of publications and SRs. For the content-based recommendation model to achieve good efficiency with near-perfect effectiveness,

it is crucial that the methods used to represent publications and SRs capture their key features. This research proposes two representation methods. First, multiple publication characteristics are considered (such as title, abstract, citation network, and authors). This is unlike most of the existing methods that rely on publication titles and abstracts only. Second, to generate embeddings of titles and abstracts, a language model fine-tuned for *living evidence* is proposed. This is unlike the traditional representation methods, which mostly use pre-trained language models, sometimes fine-tuned over general scientific datasets. The following research questions are relevant to these representation approaches, considering the notion of 'usefulness' in terms of system efficiency with near-perfect effectiveness:

- In the context of the content-based recommendation system introduced above, how useful are the proposed methods to represent publications and SRs?
- How useful is fine-tuning the embedding model using *living evidence* data, instead of a more generic scientific dataset,
- How useful is fine-tuning the embedding model through tasks which are compatible with *living evidence*, instead of the original pre-training tasks?
- How important is considering full abstracts (instead of a truncated version of them) when computing the embeddings, and how much it is worth paying for the additional computational costs?

1.5 Final Remarks

This chapter introduced the concept of *living SRs*, which reviewers update regularly to reflect the most recent research. *Living evidences* have also been introduced, as collections of 'continuously updated' SRs that cover entire fields of research. Unlike *living SRs*, which requires huge resources to maintain a set of search queries and screen an elevate number of citations and abstracts, *living evidences* must address the SRs update process in a more agile manner. To this aim, this research focuses on two contributions: one is to get after search queries; the other one is to improve the system efficiency to identify the relevant new publications for each SR in the *living evidence*,

while preserving near-perfect effectiveness. To do this, a content-based recommendation model is proposed, as well as a novel method to represent publications and SRs.

The next chapter reviews the application context from a technical perspective, and survey the state of the art. Next, Chapter 4 analyzes the embedding methods which are more commonly used for *living SRs* and their issues, and proposes a fine-tuning method for *living evidences*. Chapter 3 describes in details the proposed content-based recommendation model, which is named *ContReviews*, and the methods for representing publications and SRs. Chapter 5 is about the evaluation of *ContReviews*. Finally, before drawing the conclusions, Chapter 6 discusses the implementation of *ContReviews*.

Chapter 2

Application Context and State of the Art

This chapter first introduces the application context, based on the *living SR* and *living evidence* processes discussed in the previous chapter. In addition it surveys the existing literature: most of it is about *living SR*, while no structured works exist about *living evidences*. Finally, the last section describes the datasets, evaluation methods, and benchmarks available to compare researches in this space. However, it can be argued that they are not homogeneous, nor well suited to the *living evidence* problem.

2.1 Application Context

The previous chapter explained that the main method for updating SRs is to run the process again at regular intervals, with the *living SR* option trying to reduce the time between two consequential updates. This approach is highly customized for each SR, so it has to be done separately for every one of them. *Living evidences* have been recently proposed to overcome the main challenges of these traditional methods. Specifically, a *living evidence* is a large set of SRs in a specific domain, which are maintained current in a continuous manner—from this perspectives, SRs are 'living'. To do so, reviewers for each one of those SRs are notified as soon as new evidences which might be relevant to their SRs are published. They then can decide whether to include the recommended evidences in their SRs or not. The *living evidence* approach is substantially different from the more traditional

ones used for *living SR*, as motivated in Subsection 1.3.

2.1.1 Living SR

To update one specific SR through the *living SR* process—which is described in Subsection 1.2—domain experts first select the most important data sources, and develop and test search queries over them. Then, to conduct a *living SR*, the following steps are performed.

- *Search*: the search queries are automatically executed, to retrieve potentially relevant citations.
- *Citation screening*: the citations retrieved by the previous step are ranked, and the less relevant ones are eliminated.
- *Abstracts screening*: the abstract of the citations identified at the previous step are assessed to select the eligible ones.
- *Final inclusion*: the reviewers analyze the eligible abstracts to manually select those to include in the SR.

The search queries often make use of PICO [21] and MeSH terms.[1] **PICO** is a mnemonic standing for: patient, problem or population (P), intervention (I), comparison, control or comparator (C) and outcome (O). PICO are used to frame and answer a healthcare-related question, helping to identify keywords or search terms for efficiently searching the literature for the best evidences. The **Medical Subject Headings** (MeSH), is a controlled vocabulary thesaurus used for indexing articles for PubMed and other biomedical databases; particularly MeSH terms are useful to label and classify the content of publications so that they can be retrieved with search and other information retrieval methods.

PICO and MeSH terms, as well as other important keywords, are usually combined to construct **Boolean queries**, which are complex search strings formulated using Boolean operators (AND, OR, NOT)—an example of a Boolean query is shown in Figure 2.1). Formulating effective Boolean queries can be very challenging and time-consuming, and it is mainly a manual operation. Instead, citations and abstracts screening are greatly supported by

[1] https://www.nlm.nih.gov/mesh/intro_retrieval.html

Box 4.4.b Cochrane Highly Sensitive Search Strategy for identifying randomized trials in MEDLINE: sensitivity-maximizing version (2023 revision); Ovid format

1	exp randomized controlled trial/
2	controlled clinical trial.pt.
3	randomized.ab.
4	placebo.ab.
5	drug therapy.fs.
6	randomly.ab.
7	trial.ab.
8	groups.ab.
9	1 or 2 or 3 or 4 or 5 or 6 or 7 or 8
10	exp animals/ not humans.sh.
11	9 not 10

Ovid search syntax (for Box 4.4.b above):

exp denotes a Medical Subject Heading (MeSH) 'exploded';
/ denotes a Medical Subject Heading (MeSH);
.pt. denotes a Publication Type term;
.ab. denotes a word in the abstract;
.fs. denotes a 'floating' subheading, that is a subheading irrespective of the MeSH term to which it is attached;
.sh. denotes a MeSH term not 'exploded'.

Figure 2.1: Example of a Cochrane's search strategy, taken from the Cochrane training material available at https://training.cochrane.org/handbook/current.

technologies such as information retrieval, machine learning, and natural language processing [14]. However, all the above steps are usually very tailored to the specific SR, thus, it is not possible to re-use the same procedure to address different SRs. For example, abstract screening often uses SR-specific classification models based on machine learning, which are trained with SR data.

2.1.2 Living Evidence

The procedure for *living evidences*, which is based on the process introduced in Subsection 1.1.2, is not dissimilar from the one for *living SR* described above. The main difference is the need for one or more data sources which cover the entire domain research, instead of one specific SR. Academic knowl-

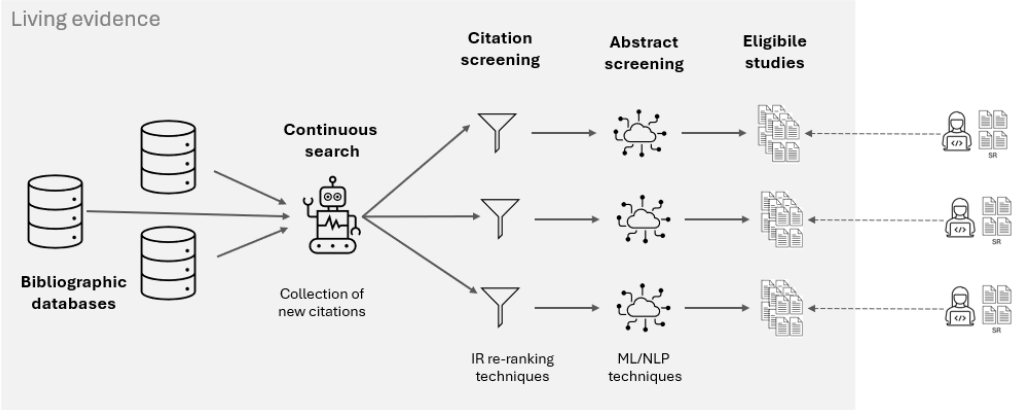

Figure 2.2: Workflow for a *living evidence*.

edge graphs are a convenient choice for these data sources, as discussed below in this chapter. The procedure for a *living evidence* is composed of the following steps, as represented in Figure 2.2.

- *Collection of new citations*: new citations are collected from an academic knowledge graph.
- *Citation screening*: citations are prioritized for each SR in the *living evidence*.
- *Abstract screening*: the abstracts of the most promising citations are assessed and prioritized for each SR in the *living evidence*.
- *Final inclusion*: reviewers consider the eligible abstracts to manually select those to include in their own SR.

As introduced in Subsection 1.3, both these procedures perform citations and abstracts screening, but the characteristics of the latter motivate the need for a different approach. The most important thing is that a *living evidence* comprises thousands of SRs (i.e., this is the case in health science), and they all must be considered by the updating procedure. Thus, to handle SRs collectively, a general method for searching, citation screening, and abstract is required, instead of using a specific method for each SR as in *living SRs*. Note, as introduced in the previous chapter, that the *living SR* characteristics that hinder most their applicability to *living evidences* are two: (i)

the complexity of formulating Boolean queries, which requires active involvement of reviewers into the update process; and (ii) the level of efficiency (i.e., usually too poor as motivated in 2.2.3) of traditional screening techniques, which again requires active reviewers participation.

2.2 State of the Art

The large majority of the state of the art regards the creation and update of SRs and, more recently, *living SRs*. A very few mentions of *living SRs* exist and they are mainly of methodological nature. There are three main research areas that support the creation and update of SRs automatically, either in a traditional or living style. One area—which regards citation screening—involves information retrieval techniques, which aim to find and rank citations effectively. The other area, which regards abstract screening, relates to machine learning and natural language processing to classify abstracts as relevant or not relevant to SRs. These techniques can work together: the results of search can be re-ranked with information retrieval techniques and then sent to a classifier for automated abstract screening. However, reviewers usually focus their efforts on the either of these techniques. Therefore, they are typically used independently and fall under distinct research domains, each with its own datasets and evaluation criteria.

The following subsections describe some of the useful works available in the literature, which regard the steps of the *living SR* process. As discussed, these techniques are usually applied to individual SRs.

2.2.1 Finding Useful Citations

Methods at the state of the art for identifying an initial, broad set of potentially useful citations are fundamentally rooted in using adequate queries for searching the relevant data sources. In an *SR update* these queries are usually developed and run every time an update project is in place. In *living SRs*, data sources and queries are formulated once and deployed within a continuous surveillance service. More recently, Academic Knowledge Graphs have emerged as more generalized, domain-independent data sources which are maintained up to date with all the new published researches.

Search

To search bibliographic data sources for useful citations effective queries must be developed by reviewers and domain experts. The quality of these queries is crucial, as it affects how many documents need to be checked for inclusion in the final SR by human reviewers. Developing a search strategy is an iterative process, involving continual assessment and refinement: as keywords or key terms are used in a search, their usefulness will be determined by the search results. Thus, searching for evidence is sometimes considered more of an art than a science; it is therefore unlikely that two people, whether they are clinicians or librarians, develop an identical search strategy or yield identical results from a search on the same review question [22].

In addition, in the health care domain, complex Boolean queries are often used to identify studies. Developing Boolean queries for this task requires the expertise of trained information specialists. For example, one challenge to creating an effective Boolean queries is the selection of adequate MeSH terms to include in the query [23]. However, even for expert searchers, query formulation can be difficult and lengthy: especially when dealing with areas of medicine that they may not be knowledgeable about [24].

The problem of creating effective Boolean queries to make SRs in health care has been tackled by several works. To cite a few of them, the challenge has been tackled with MeSH term suggestion methods [23], data-driven query generation from existing citations [25], pre-trained generative models [15, 26], and seeding methods [27].

In conclusion, the effectiveness of Boolean queries relies on the keywords, MeSH terms, and PICO that are used to formulate queries, and the correctness and completeness of the ones that were assigned to original publications by their authors. One drawback of this method—in addition to being time consuming, as highlighted above—is that it is prone to errors; for instance, a reviewer might omit an important MeSH term or PICO when formulating the search query; similarly, authors might omit to assign an important MeSH term when publishing their research. In any case, an important publication might be missed by a SR update process. As described in Chapter 3, an alternative method to find useful citations, is matching the new publications to the ones already included in a SR, by means of their content.

Continuous Surveillance

The notion of continuous surveillance of data sources was proposed in contrast to periodically producing Boolean queries to update existing SRs, as introduced in Section 2.1.1. For example, Trialstreamer [20] implements a continuous aggregated database search with push notifications, covering the majority of health care literature and focusing on randomized controlled trials. Note that this approach does not provide an alternative to the challenges due to the complexity of formulating Boolean queries; instead, it continuously runs Boolean queries to increment the frequency of discovering new publications. Thus, if the original Boolean queries are not well formulated, having them continuously run will not make them better.

Recently, freely-available academic knowledge graphs (AKG) [1, 28] have emerged; they maintain continuous, web-scale search over bibliographic databases and publicly available repositories in multiple domains. An AKG can be helpful to reduce the need for identifying compelling data sources, as it provides a unique repository of new publications; however, it still requires query techniques to extract the relevant publications. In addition, the new publications are provided continuously, as they are discovered as soon as they are published. However, it could be debatable if a unique, domain-independent repository covers all the relevant publications for a certain SR. Moreover, the issue is even more relevant when considering *living evidences*, that are made of thousands of SRs with new SRs being added daily. Literature is not abundant around this issue however, the Microsoft Academic Graph [28] was found to be sufficiently comprehensive to maintain a living map of COVID-19 research [29].

2.2.2 Citation Screening

One drawback of Boolean queries is that they tend to generate many results but only a few of them are pertinent to the target SR [30]. Therefore, to reduce the amount of manual labor, it is necessary to filter out the irrelevant citations automatically, without missing any important ones. This problem is particularly researched in the context of SR creation, though it also applies to *SR update*. In fact, the same problem arises when there is a long gap between two consecutive *SR updates*. To mitigate this issue, citation screening has been proposed by several studies to re-rank the set of citations initially found through Boolean queries, so that reviewers can focus on the most promising

ones [31].

Several methods have been proposed to re-rank citations [15, 30–34]. They match the representations of the citations initially found through the search to the representation of a revisited version of the Boolean query; a relevance function is finally used to re-rank citations based on their relevance to the revisited query. These methods use different approaches to revisit the Boolean query, to obtain representations used for matching and to re-rank citations after the matching. Earlier studies [32, 33] proposed to rewrite the Boolean query based on keywords extracted from the SR's title and from the Boolean query itself. In addition, they proposed to represent citations and the rewritten query through bag of words, and a lexical relevance function to re-rank citations. An issue of these early approaches is that the SR's title is usually unknown at the time the SR is constructed [15], thus, using keywords extracted only from the Boolean query has also been proposed [33]; however, this issue does not apply when updating SRs which already exist, because their title has been already established. Rank fusion techniques [35] have also been proposed to implement the relevance function [33]; specifically, rank fusion is used to rank the citations retrieved by means of each clause of the Boolean query.

Another variations to re-ranking is based on technology-assisted review (TAR), which refers to iterative active learning for citation review in high recall retrieval tasks [34]. TAR research have applied linear models to lexical features, such as bag of words, reporting better effectiveness than the traditional lexical rankers. Methods at the current state of the art are based on neural ranker [31]. These methods rely on pre-trained language models such as BERT [36] and have achieved similar performance as TAR methods based on lexical features. However, in contrast to lexical models, there are still challenges in using these neural rankers; in fact, the maximum input token length imposed by most BERT-based models (i.e., 512 tokens) does not allow processing longer language expressions, such as the full text of candidate citations and extensive revisited queries.

No research on continuous surveillance has addressed citation screening; however, they use methods which are similar to citation screening to regularly identify and select the pertinent studies. For example, one system that performs continuous surveillance is Trialstreamer [20]: it identifies randomized controlled trials (RCTs) and continuously updates some systematic reviews (SRs) with them. Trialstreamer still runs queries to retrieve candidate RCT documents and use a binary classification model to filter the

RCTs, which is a similar to what citation screening does.

2.2.3 Abstract Screening

Once new publications are retrieved by means of Boolean queries or other search strategies, their relevance to SRs can be checked through abstract screening techniques, which determine whether a publication is relevant to a SR by analyzing its abstract. As anticipated, citation screening and abstract screening can work together, where the former provides the inputs for the latter, though, they are more frequently used alternatively.

Abstract screening is often framed as a binary classification task, where various models have been proposed based on machine learning [19, 37–40] and, more recently, on deep learning [41–44]. These models extract their features from publication title and abstract, usually using embeddings (see Section 4.1), and use them to infer publications relevance to one single, model-specific SR. Note that this close connection between abstract screening models and SRs differs from *living evidences*, which handle multiple SRs simultaneously. The abstract screening problem has been addressed by several works in the literature; despite having a similar architecture, they differ significantly by various factors. These factors are explained in the following paragraphs.

Evaluation process. A common way to evaluate abstract screening models is to measure how well they perform against human reviewers over an observation period [19], where some new publications are fed to both an abstract screening method and a review group (i.e., humans!). These new publications are usually obtained from a research phase that uses search tools, as mentioned earlier. Another approach is to collect some SRs and put apart some publications for evaluation and testing; this approach is more standardized and easier to replicate in comparison efforts though, as explained in Section 5.1.2, it is still a challenge and fully replicable generalized datasets are missing. Finally, some works consider the 'review performance' of two competing groups of reviewers [39], where only one of them is given a semi-automated tool to assist during the review. The focus of these studies is to evaluate aspects like abstract screening speed, screening accuracy, characteristics of included texts, and user satisfaction. Thus, they have a methodological nature and they do not necessarily follow under either the categories of search, citation screening, and abstract screening.

Evaluation metrics. Several metrics are proposed in the literature to measure model performance [12]: a common choice is using the classification metrics based on the confusion matrix, such as *precision* and *recall*. The recall classification metric measures the fraction of all the truly relevant publications that are correctly captured by the model, while the precision classification metric measures how many of the publications judged as relevant by the model are actually so. On the one hand, requesting a recall of 95% or more is a common choice, aiming to identify all the relevant publications; in fact, in the *living SR* task, new relevant publications should never be lost. On the other hand, higher precision corresponds to lowering the manual effort of assessing new publications that are actually irrelevant. Usually, when recall rises, precision drops, that is, to identify more relevant publications it is possible that some irrelevant ones are also collected. Hence, a common criterion for evaluating abstract screening models is measuring precision while maintaining a recall of 95% or higher: to reduce the manual work of screening out irrelevant abstracts, precision should be as high as possible.

Some older works also consider the Work Saved over Sampling at r% recall (WSS) [45], that is, the percentage of papers that meet the original search criteria that the reviewers do not have to read, because they have been filtered out during abstract screening. Again, a common value for the WSS's recall is 95%. However, recent research has exposed limitations with WSS [46] and showed that it is equivalent to the *specificity*. The specificity classification metric, which is also known as True Negatives Rate, is based on the confusion matrix and measures the fraction of all the irrelevant publications predicted by the model which are actually so.

Classification performance Abstract screening models at the state of the art, when evaluated by means of precision and recall, often achieve high recall and relatively low precision (see Table 2.1 for a summary). This is a common and challenging issue that hinders the applicability of these models [47] to the *living evidence* problem.

One reason is that they are often trained in presence of highly imbalanced data, where the positive class (i.e., the relevant publications) has so fewer elements compared to the negative class. In fact, these models are often trained on individual SRs, which include only a few dozen publications (especially in health care) [12, 48], while the irrelevant publications are obviously many more. Active learning represents a broadly adopted approach [14, 49] to ad-

Table 2.1: Precision and recall of some abstract screening models.

Model	**Precision**	**Recall**
Binary classification over BERT embeddings [19]	0.55	1
LightGBM over different BERT encoders [38]	0.496	0.96
LR classifier using topics (LDA) as features [40]	0.559	0.987
Binary classifier over fastText word embeddings [41]	0.121	0.95
SVM classifier using auto-encoding [42]	0.167	0.95
CNN classifier over GloVe word embeddings [43]	0.135	0.95

dress classification performance in presence of imbalanced data. Specifically, it has been argued that active learning over a small subset of informative data can actually produce a better generalized model than one trained over large, randomly selected data [50].

In an active learning setting, abstract screening classifiers are trained interactively by providing labels for publications the classifier thinks will be most informative [25, 49, 51, 52]. Specifically, few seeding publications that are known to be relevant (i.e., they might have been identified in full manual way) are used to train an abstract screening classifier that, in turn, it is applied to infer the relevance of a group of new publications (i.e., discovered by means of a search phase). The model is trained again considering the publications that scored best with the current version of the model and were judged as relevant by users. The process is iterated until reviewers think it is not providing any better results. However, the decision about stopping to train the model is critical and not fully supported by clear metrics. The performance of the active learning process is evaluated in terms of Yield and Burden [49]: the former determines the percentage of eligible studies identified by the active learner, while burden represents the percentage of studies that are manually labelled. It is common to aim for Yield of 95% and Burden as low as possible.

Another factor which affects low precision is that some SRs cover very similar research topics, thus, many publications can use similar terminologies and address similar problems, but they might not be relevant to the same SRs. Few works tackle the challenge by focusing on representing publications in a more informative manner. To do so, they consider additional publication aspects besides their title and abstract [53, 54], such as bibliographic information [47], citation network, and context [55], MeSH terms, and UMLS concepts [51].

2.3 Datasets and Evaluation Metrics

In recent years there has been a growing interest in automating the process of SR update, especially in health care, through the application of information retrieval, machine learning, and natural language processing techniques. Most works at the state of the art focus on methods for identifying the new publications and automating the screening of citations and abstracts. However, the systematic comparison of different methods is challenged by the lack of standardised datasets and common evaluation criteria [12].

The evaluation of the proposed methods is not uniform, not only because a common dataset is missing, but also because of the adoption of different evaluation methodologies. Methods that focus on Boolean queries and citation screening usually report their performance in terms of information retrieval metrics; on the other hand, abstract screening methods report their performance in terms of classification metrics. The latter usually focus on the classification matrix, i.e., precision with recall at 95% or more [19, 37]. Some older works adopts the Work Saved over Sampling at r% recall [45], which is still based on the classification matrix, but measures the amount of work that reviewers do not need to do thanks to the model.

Regardless of the specific evaluation metrics used, these methods offer recommendations to reviewers, who then complete the task by manually examining the abstracts and full documents. Other works focus on methods where humans and models jointly perform the task of updating SRs in a semi-automated setting. As mentioned in Subsection 2.2.3, examples of this approach are active learning [49] and assistant tools for reviewers [39]. In the former case, to report evaluation performance authors mostly focus on Yield and Burden, which are still based on the confusion matrix, but they discriminate whether publications are successfully considered as relevant by a model or by a reviewer. In the latter case, a semi-automated tool is given to one of two groups, and both groups are compared on the same task of updating a systematic review; usually, the typical classification metrics of precision and recall are used for this comparison.

To the best of our knowledge, *ContReviews* is the first attempt to address the problem of *living evidence*. Common datasets and benchmarks for this problem, thus, are not available. Although they are not homogeneous, as mentioned above, various datasets have been published in relation to the different studies that have tackled the problem of SR currency. These datasets contain only a few SRs, which are insufficient to assess a *living ev-*

idence system. Moreover, many of the available datasets suffer from either data leakage and overlapping, miss canonical train/test splits, lack common evaluation criteria and have limited applicability [12].

2.4 Final Remarks

This chapter first discussed the *living SR* and *living evidence* procedures, aiming to clarify the application context that this research addresses. The state of the art related to *living SRs* has been discussed: it comprises information retrieval, machine learning and natural language processing techniques to address the main phases of *living SRs*, i.e., searching for new citations which might be relevant to a SR, usually through complex Boolean queries; re-ranking these citations to discard the less relevant ones; and classify abstracts for relevance to SRs.

This chapter explained the reasons why the traditional methods for *living SRs* are not well suited to address *living evidences*, arguing that they are fundamentally too specific to one single SR. Instead, *living evidences* would take benefit from a more general approach to address all the SRs in the *living evidence* simultaneously. The main aspects of *living SRs* which hinder their applicability to *living evidences* are the complexity of Boolean queries and the usually poor efficiency of the screening methods. These factors motivate the active involvement of reviewers, to handcraft the update process to the specific SR. The next chapter introduces *ContReviews*, a content-based recommendation model for *living evidences*, aiming to overcome the challenges of the *living SR* update procedures.

Chapter 3

ContReviews: a Content-based Recommendation System for *living evidences*

Section 1.3 discussed the limitations of the conventional *living SR* methods which hinder their applicability to *living evidences*, and motivated the need for a novel approach. In summary, the *living SR* approach is well suited to update one SR or a few ones, but is not general enough to address entire domains of SRs, as required in *living evidence*. In fact, *living SRs* are hand-crafted for specific target SRs, and do not generalize to cover other different SRs. Instead, a *living evidence* system would benefit from a more general approach, that can be applied to any SR being part of it.

This chapter introduces **ContReviews**, which is the proposed system for addressing the *living evidence* problem. Although *ContReviews* could be in principle applied to SRs in any domain, it was specifically developed for the health care domain and tested over a large subset of the *Cochrane Reviews*.

3.1 Requirements and Approach

The ideal approach to *living evidence* is be based on the following requirements.

- The procedures to discover all the new publications that might be relevant to any of the SRs in the *living evidence* is automated and runs

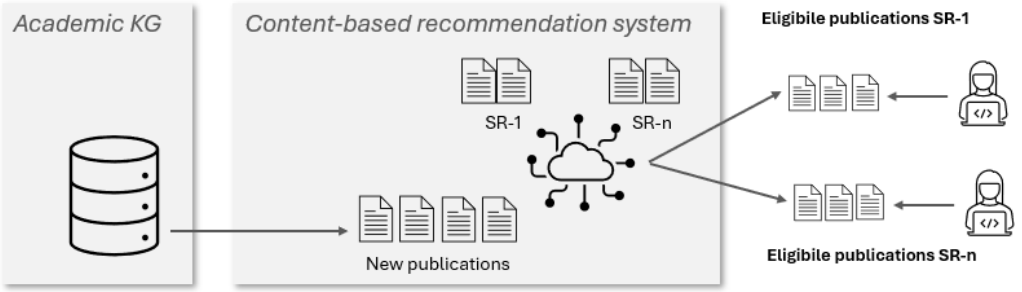

Figure 3.1: Workflow for *living evidence*.

continuously, without reviewers active involvement to design and maintain complex search queries.

- The new (potentially relevant) publications discovered by the system in the previous step are automatically screened, and the most relevant ones are identified for each SR and notified to reviewers. The screening methods should be automatic and general enough to not require the active involvement of reviewers or any SR-specific optimizations.
- For their SRs, reviewers only assess the publications that were identified in the previous step, when notified to them. They do not participate to any SR-specific design, preparation, or fine-tuning activity.
- The success of a *living evidence* lays in its ability to recommend to reviewers all the truly relevant publications (effectiveness) and a limited number of irrelevant ones (efficiency).

ContReviews, an SR-independent system for *living evidence*, is introduced to realize the approach described above. *ContReviews* is based on the three following design principles, which are represented in Figure 3.1.

Academic knowledge graphs (AKG). As proposed in [29], *ContReviews* leverages academic knowledge graphs (AKG) for continuous surveillance. AKGs are domain-independent, unified sources of new publications which are continuously updated by agents searching the Web and bibliographic databases. They provide a periodic feed where the new publications

are organized over a common schema, which can replace multiple and heterogeneous SR-specific data sources. AKGs generalize the problem of identifying data sources for each SR.

Content-based recommendation model. *ContReviews* is based on a content-based recommendation model that leverages the publications already included in the *living evidence*'s SRs to infer the relevance of new publications. Specifically, it provides a relevance assessment function which is based on content-based matching of new publications and SRs. This approach lets avoiding to formalize one set of search queries for each SR, and represents a general method that can be used with all the SRs in a *living evidence*, avoiding SR-specific models.

Multi-property representation of publications and SRs. Vector representations are used by the content-based recommendation model to match publications and SRs. *ContReviews* leverages multiple publication features to construct these representations. Specifically, the features used in this research are the publication titles, abstracts, citations network and authors; however, the system is not limited to them. By doing so, *ContReviews* aims to increase meaningfulness of representation and, ultimately, to achieve better recommendation efficiency with almost-perfect effectiveness.

3.2 Intuition of *ContReviews*

Very informally, *ContReviews* is based on a very simple concept: if a new publication 'looks like' those included in a SR then it might be relevant to the SR. Such a 'resemblance' between the new publication and the SR might be because they have similar abstracts, similar citations networks, similar features, or a combination of them. Moreover, their 'extent of resemblance' can be used to calculate a likelihood of relevance between the new publication and the SR, and decide whether to recommend it to the SR's reviewers based on its value. This section explains this idea, before formalizing it in the next two sections.

ContReviews is a content-based recommendation system. In general terms, a content-based recommendation system is used to recommend new items to users based on their past item preferences, formally described in terms of

item features. The following quote helps to define a content-based recommendation model [56]:

> *In the real world, it would be straightforward to recommend the new Harry Potter book to Alice, if we know that (a) this book is a fantasy novel and (b) Alice has always liked fantasy novels. An electronic recommender system can accomplish this task only if two pieces of information are available: a description of the item features and a user profile that somehow describes the (past) interests of a user, maybe in terms of preferred item features. The recommendation task then consists of determining the items that match the user's preferences best. This process is commonly called content-based recommendation.*

In the *living evidence* context, the new publications act as the items to recommend and the SRs as the user profiles; moreover, the publications already included in the SRs describe the user past preferences. Instead of using Boolean queries to find potentially useful citations, and applying screening models to refine the set of citations and abstracts, *ContReviews* leverages the publications already included in the *living evidence*'s SRs to identify the new relevant ones for each SR.

The *ContReviews* system comprises an academic knowledge graph (AKG) and a content-based recommendation model, working together to manage a *living evidence*. The AKG, which is OpenAlex [1],[1] provides a periodic stream of domain-independent new publications to the content-based recommendation model. The latter, in turns, assesses their relevance to each SR in the *living evidence*, based on publications and SRs content.

The *ContReviews*'s content-based recommendation model relies on a formal representation of publications, which is based on their features,[2] such as title, abstract, citations, journal, authors, venue. Note that the *ContReviews*'s AKG conveniently provides several of these features in a structured manner. The experiments conducted to evaluate *ContReviews* used only some of these properties (i.e., title, abstract, citation network, and authors); however, the model can be instantiated with any combination of publication properties, provided by either the AKG or any other system. For example, PICO and MeSH terms are viable options, which are commonly employed in Boolean queries.

[1] https://openalex.org/

[2] In *ContReviews* parlance, publication features are also called *publication properties*

ContReviews uses publication properties to provide vector representations of the new publications provided by the AKG and the publications already included in the *living evidence*. SRs vector representations are obtained by aggregating the ones of their included publications. For both new publications and SRs, *ContReviews* creates one vector representation for each publication property. For example, a publication (or, similarly, a SR) will have one vector representation based on authors, one based on abstract, and another one based on citations. *ContReviews* uses these property-specific vector representations to obtain property-specific *likelihoods of relevance* between SRs and new publications.

To assess the relevance of a new publication to a SR, their property-specific vector representations are constructed and one likelihood of relevance for each of them is calculated. A relevance assessment function uses the property-specific likelihoods of relevance to estimate how relevant the new publication is to the SR. *ContReviews* considers such a relevance assessment function as a binary classification model, based on machine learning. A unique binary classification model for the entire *living evidence* can be used, and trained using a supervised dataset which is based on all the *living evidence* publications. This let to overcome the data imbalance problem which is typical in *living SRs*, where the SR-specific citation and abstract screening models are trained using their target SR data only.

The supervised dataset for training the *living evidence* binary classification model consists of records, each of which corresponds to a SR and a publication. The records have a binary label that shows whether the publication is relevant to the SR, and a set of likelihoods of relevance derived from their vector representations. To construct such supervised dataset from the *living evidence* publications and SRs, the following procedure is used (see Figure 3.2):

- The publications in the *living evidence* are represented with one vector per publication property.
- SRs are represented with one vector per publication property, by averaging the vectors of their included publications.
- Likelihoods of relevance between publications and SRs are obtained (the $\triangle$ in the picture), by computing and re-scaling the cosine similarity between their property-specific vector representations (this is described in details in the next sections).

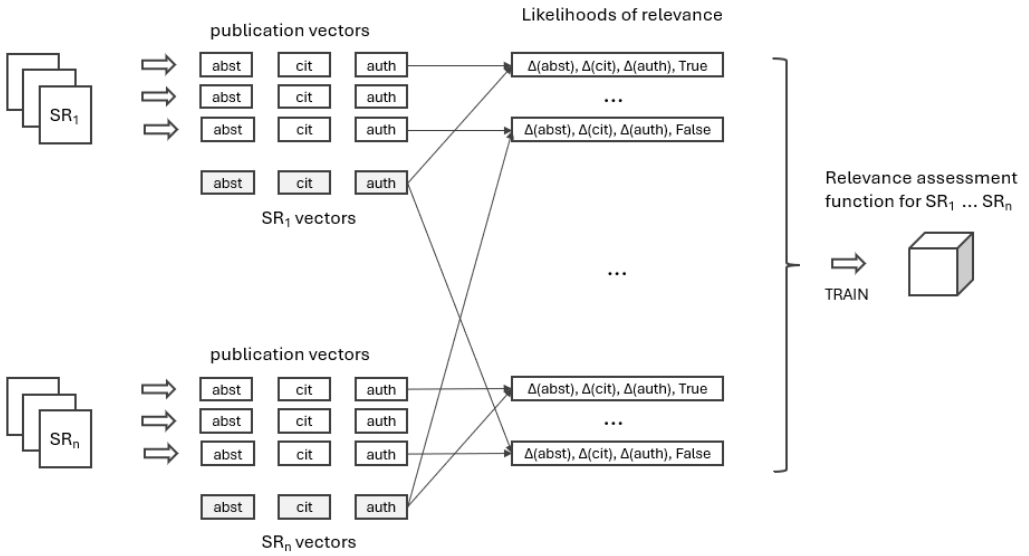

Figure 3.2: *ContReviews*'s relevance assessment function training method.

- For pairs of one publication and one SR a vectors of property-specific likelihoods of relevance is obtained. If the publication is included in the SR, a positive label is associated to it (a negative label otherwise).

Te supervised dataset obtained as described above is used to train a binary classification model, which provides a unique relevance assessment function for all the SRs in the *living evidence*. In addition, it can be used at inference time to score the new publications for relevance to each of the SRs in the *living evidence*. To this aim, new publications are provided by the AKG: *ContReviews* constructs their property-specific vector representations for each SR, calculates the property-specific likelihoods of relevance and applies the binary classification model to obtain the final likelihoods of relevance. The latter, provide the relevance for each new publication and each SR in the *living evidence*, and it is used to notify reviewers about useful new publications.

3.3 Problem Statement

Let $\mathcal{P}$ be the domain of *scientific publications*, and $\mathcal{R}$ the domain of systematic reviews *SRs*, where each SR $r \in \mathcal{R}$ is formally represented as the subset of the publications it includes ($r \subseteq \mathcal{P}$). In other words, a SR is characterised by the set of publications that are *relevant* to it, and hence included in it.

Let Pubs be the function returning all the publications included in a subset R of SRs: $\mathsf{Pubs}(R) = \bigcup_{r \in R} r$. That is, R is a *subset* of SRs and $\mathsf{Pubs}(R)$ denotes all the publications that appear in at least one of these SRs. Moreover, let η be the Boolean function returning $True$ if a publication is relevant to a SR and $False$ otherwise. Note that $p \in \mathsf{Pubs}(r) \Rightarrow \eta(p, r)$ but $p \notin \mathsf{Pubs}(r) \not\Rightarrow \neg\eta(p, r)$; in fact, new publications are not already included in the *living evidence*.

Let $\hat{R} \subseteq \mathcal{R}$ be a a *living evidence*, i.e., a subset of SRs all belonging to the same scientific domain. Moreover, let $\hat{P} \subseteq \mathcal{P}$ be a set of *new* publications which are not included in the *living evidence*; that is, $\mathsf{Pubs}(\hat{R}) \cap \hat{P} = \emptyset$. For each SR $r \in \hat{R}$ the *living evidence* task is to determine $\hat{P}_r \subseteq \hat{P}$, which is the subset of new publications being relevant to r:

$$\hat{P}_{r \in \hat{R}} = \{p \in \hat{P} \mid \eta(p, r)\} \tag{3.1}$$

3.4 Formal Definition of *ContReviews*

This section formalizes the *ContReviews*'s content-based recommendation model. To do so, first Subsection 3.4.1 introduces the formal representation method for publications and SRs; then Subsection 3.4.2 illustrates the matching method between publications ans SRs and the calculation of the likelihoods of relevance; finally, subsections 3.4.3 and 3.4.4 describe the relevance assessment function and its implementation as a binary classification model based on machine learning (i.e., training and inference). Note that this binary classification model provides an estimation of equation 3.1.

3.4.1 Formal Representation of Publications and SRs

As previously outlined, we formally represent both publications and SRs through what we name their *publication properties*. In the scope of this work we consider, as properties of a publication, its title, abstract, authors, and

citations. However, this does not preclude that any set of publication properties can be considered. Specifically, each publication $p \in \mathcal{P}$ is represented by a set of vectors $v^{\pi}(p)$, one for each publication property π. Formally, the representation $v(p)$ of the publication $p \in \mathcal{P}$ is defined as the following set:

$$v(p) = \{v^{\pi_1}(p), \ldots, v^{\pi_n}(p) \mid \pi_n \in \Pi\} \tag{3.2}$$

where $\Pi = \{\pi_1, \ldots, \pi_n\}$ is the set of considered properties. Analogously, each SR r is represented by a set of vectors $v^{\pi}(r)$ that summarise all the publications it contains. In this case, each $v^{\pi}(r)$ is obtained by averaging the vector representations of the publications included in it. For each property $\pi \in \Pi$:

$$v^{\pi}(r) = avg(\{v^{\pi}(p) \mid p \in Pubs(r)\}) \tag{3.3}$$

where avg is the function which averages numeric vectors, i.e., $avg(v^i, v^j) = [avg(v^i_1, v^j_1), ..., avg(v^i_n, v^j_n)]$.

The specific construction of each vector $v^{\pi}(p)$ depends on whether the property π represents an entity (such as authors and citations) or a language expression (such as title and abstract).

Representation of entities. Binary vector representations are considered, which are based on vocabularies of entity instances extracted from the entire *living evidence*. For example, considering authors: a vocabulary of n distinct authors is constructed by extracting them from the *living evidence*; hence, a publication p can be represented by a binary vector $v^{auths}(p)$ of length n, having 1 as the i-th element if p is authored by the i-th author in the vocabulary, and 0 otherwise. Citations or any other entity would behave exactly in the same manner.

Note that new publications may contain new entities that are not included in the relevant vocabularies. Therefore, it may be difficult to construct the publication representations using these new entities. This is known as the 'out of vocabulary' issue. For example, a new publication may reference citations that are not referenced by any of the existing publications in the *living evidence*; therefore, these citations would not be part of the relevant vocabulary. *ContReviews* simply ignore these out of vocabulary entities, for two reasons. In one way, if a new entity was added to the relevant vocabulary prior to constructing the representations of the new publications, it would not contribute to rise any measure of relevance between the new publications

and the SRs in the *living evidence*: in fact, no one of them has a '1' in their representation for that entity. Thus, it is not useful to update the vocabulary. In addition, whenever the vocabularies are updated all the representations must also be updated and the entire *ContReviews* system should in turn be updated. Thus, adding entities to the vocabularies is a computationally expensive operation that should be carefully planned.

Representation of language expressions. Two distinct vector representations are considered, which may carry different aspects of the text. The first vector representation corresponds to a bag of words, using TF-IDF (see Subsection 4.1). The other vector representation is based on contextual embeddings, obtained by means of pre-trained language models, such as SciBERT [57]. There are various language models that have been pre-trained on specific datasets. These models can be further fine-tuned for specific tasks and datasets.

3.4.2 Computing Likelihoods of Relevance

Likelihoods of relevance is an important notion in *ContReviews*, as introduced in Section 3.2. These likelihoods of relevance are employed in two cases: to construct the supervised dataset for training the binary classification model, that estimates the relevance assessment function; and to assess the relevance of a new publication to a SR, by inferring it through the binary classification model. In the first case, likelihoods of relevance are computed for pairs of one SR and one publication which are already known to be relevant or not. In the second case, the likelihoods of relevance are used for infer relevance of new publications.

This subsection describes how likelihoods of relevance are computed, through the following two steps: first, publication and SR representations (which are calculated exactly as explained in Subsection 3.4.1) are matched and similarity scores are computed through their cosine distance; then, similarity scores are re-scaled.

Computing similarity scores. The calculation of the similarity between a publication p and a SR $r \in \hat{R}$ is an intermediate step, either to construct the supervised training dataset or to assess the relevance of p to r. In either case, one similarity score for each publication property is computed as the

cosine similarity between the vector representations of the SR and the vector representations of the publication p.

A set $\hat{R}$ of SRs (each of them described as the set of included publications $P_r = \mathsf{Pubs}(r), r \in \hat{R}$) and a set of properties $\Pi = \pi_1, ...\pi_n$ are considered. For each publication $p \in \mathsf{Pubs}(\hat{R})$ one vector representation $v^\pi(p)$ for each property $\pi \in \Pi$ is obtained (equation 3.2). Similarly, for each SR $r \in \hat{R}$, one vector representation $v^\pi(r)$ for each property $\pi \in \Pi$ is obtained as well (equation 3.3). The property-specific similarity scores between p and r corresponds to the set $\phi(p, r) = \{\kappa(v^\pi(p), v^\pi(r) \mid \pi \in \Pi\}$, where κ denotes the cosine similarity function.

Re-scaling similarity scores. The similarity scores computed at the previous step could be directly used as measures of the property-specific likelihoods of relevance. However, as shown by the experiments conducted to evaluate *ContReviews*, re-scaling them is more effective. For each SR $r \in \hat{R}$, the set of publications $S_r = P \cup N$ is sampled from $\mathsf{Pubs}(\hat{R})$, where P comprises a sample of the publications included in r (i.e., relevant to r) and N comprises some publications randomly sampled from other SRs (i.e., irrelevant to r):

$$S_r = P \cup N = \{p \in \mathsf{Pubs}(r)\} \cup \{p \in \mathsf{Pubs}(\hat{R}) - \mathsf{Pubs}(r)\} \tag{3.4}$$

note that as data is imbalanced (i.e., for each SR the irrelevant publications are many more than the relevant ones), N is downsampled to be in the right proportion to P.

The set of similarity scores $\Phi^\pi_{S_r}$ between r and the publications in S_r is computed for each property $\pi \in \Pi$, yielding the following set of similarity scores:

$$\Phi^\Pi_{S_r} = \{\kappa(v^\pi(p), v^\pi(r)) \mid p \in S_r, \pi \in \Pi\} \tag{3.5}$$

Let p be one publication, which could be either a new publication or a known one that is used to construct a supervised training record. To obtain the likelihoods of relevance between p and a SR $r \in \hat{R}$, with respect to the set of properties Π, we first calculate—for each property $\pi \in \Pi$—the similarity scores $\kappa(v^\pi(p), v^\pi(r))$, and re-scale them with respect to $\Phi^\pi_{S_r}$. This is achieved by computing the fractions of publications in the sample S_r that are *less similar* to r than p, with respect to $\pi \in \Pi$ ($|\cdot|$ denotes the cardinality

of a set):

$$\theta_\pi(p, r) = \frac{|\{p' \in S_r : \kappa(v^\pi(p'), v^\pi(r)) < \kappa(v^\pi(p), v^\pi(r))\}|}{|S_r|} \tag{3.6}$$

Thus, the vector whose elements are likelihoods of relevance, as defined by equation 3.6, can be either used to infer relevance of p to r (see Subsection 3.4.4), or to construct one supervised training record, where the relevance is already known (see Subsection 3.4.3).

The reason for using re-scaled similarity scores, rather than absolute ones, stands on the fact that many SRs in health care are related to similar research questions, or are compiled for the same research topic. Thus, some publications and SRs might be similar from the angle of certain properties just because of that. For example, 'smoking cessation' is a group of several SRs in the 'human behaviour' domain: clearly the language used in these SRs is homogeneous, and potentially also authors and citations are so. Thus, publications from such sub-domains might have high absolute similarity scores with all the 'smoking cessation' SRs, even if they are irrelevant. Re-scaling provides a statistical dimension for evaluating relevance, allowing the extent of similarity for relevant publications to be emphasised compared to irrelevant (but plausible) ones.

3.4.3 Learning the Relevance Assessment Function

The relevance assessment function is estimated by a model $\Theta_\Pi^{\hat{R}}$ that infers relevance to SRs $\hat{R}$ based on a set of publication properties Π. Such a model can be conveniently learnt from data through machine learning. Specifically, we consider a binary classification model using the likelihoods of relevance formalized in equation 3.6 as its features. To train $\Theta_\Pi^{\hat{R}}$ a supervised dataset can be constructed, where each element represents one training record being relative to one publication $p \in \mathsf{Pubs}(\hat{R})$ and one SR $r \in \hat{R}$: Each one of these records is made of some features and a label, defined as follows:

- *label*: $True$ if p is relevant to r (i.e., $p \in \mathsf{Pubs}(r)$), $False$ otherwise;
- *features*: the likelihoods of relevance $\theta_\pi(p, r), \pi \in \Pi$ (as per equation 3.6).

The relevance assessment function is learnt for the entire *living evidence* by considering the contribution of all the SRs together. In fact, for each

SR in the *living evidence*, a sample of relevant publications and irrelevant publications is considered. Irrelevant publications for a SR r are sampled from the 'other SRs' given by $\hat{R}-r$. Any publication can be sampled multiple times: at least once as a relevant case for the SRs it is included into, and potentially several other times as irrelevant cases for different SRs. The likelihoods of relevance will be different every time, because they are obtained for different SRs.

3.4.4 Inference

Let $\Theta_{\Pi}^{\hat{R}}$ be a relevance assessment function as defined in Subsection 3.4.3; and $S = \{S_r \mid r \in \hat{R}\}$ be the union of the sets S_r, as defined by equation 3.4. Given some *new publications* $\hat{P}$, such that $\hat{P} \cap \mathsf{Pubs}\hat{R} = \emptyset$, the goal is to rank these new publications for relevance to each SR $r \in \hat{R}$, using the relevance assessment function $\Theta_{\Pi}^{\hat{R}}$. For every new publication $p \in \hat{P}$ and SR $r \in \hat{R}$ the goal can be achieved by means of the following steps:

- Calculate the new publication's vector representations $\{v^{\pi}(p),\ \pi \in \Pi\}$ (equation 3.2).
- Calculate the similarity scores $\{\kappa(v^{\pi}(p), v^{\pi}(r)), \pi \in \Pi\}$ (equation 3.5).
- By using using S_r (equation 3.4), calculate the vector of likelihoods of relevance $\theta_{\Pi}(p, r) = [\theta_{\pi_0}(p, r), ..., \theta_{\pi_n}(p, r)]$ (equation 3.6).
- Score the vector of likelihoods of relevance associated to the new publication p and the SR r by using the relevance assessment function $\Theta_{\Pi}^{\hat{R}}(\theta_{\Pi}(p, r))$.

The latter provides a score that can be used to rank p for relevance to r and, ultimately, to estimate $\hat{P}_{r \in \hat{R}}$ (equation 3.1):

$$\hat{P}_{r \in \hat{R}} = \{p \in \hat{P} \mid \Theta_{\Pi}^{\hat{R}}(\theta_{\Pi}(p, r)) \geq \tau_r\} \tag{3.7}$$

where τ_r is a threshold which depends on the SR r.

3.5 Final Remarks

This chapter described the *ContReviews* system, which is used to infer relevance of new publications to each SR in a *living evidence*. The new publications are sourced through an academic knowledge graph, which can replace

developing adequate search queries for each SR. In addition, a content-based recommendation model is used to match these new publications to each SR in the *living evidence*, based on their content. A content-based recommendation model provides a general approach which can replace developing one abstract screening model per SR. Finally, to match the new publications and the SRs, multiple publication features are used to construct multiple vector representations.

The vector representations are based on bag of words and embeddings for the textual features (such as title and abstract), and on binary vectors for entities (such as citations network and authors). Before describing the *ContReviews* evaluation results, the next chapter describes the state of the art of using embeddings in the context of SRs, and the fine-tuning approch used for *ContReviews*.

Chapter 4

Representation of Publication Titles and Abstracts

Representing language expressions is crucial to any SR updating method, including *living SRs* and *living evidences*. In fact, publications and SRs provide multiple textual descriptors which can be useful to infer relevance of new publications to SRs. For example, publications have a title and an abstract, while SRs provide their inclusion criteria in text. Thus, capturing and representing the most meaningful text features from these descriptors become crucial to effective and efficient SR updating models.

Section 4.1 surveys the most common text representation techniques, which includes bag of words and embeddings, and how they are used in SR updating models at the state of the art. Section 4.2 introduces the embedding models considered for *ContReviews*.

4.1 Representation of Language Expressions

Citation screening models, abstract screening models, and the *ContReviews* content-based recommendation model all use publication textual features (such as title and abstract) to drive their inference process. They require to transform raw language expressions to meaningful numeric representations, prior to being able to process them.

A traditional method is based on *bag of words*, which describes the occurrence of words within a document, disregarding syntax and word order but maintaining word frequency. The bag of words representation technique is

well known and established in the domain of information retrieval and, more in general, in the natural language processing community. However, as it is fundamentally based on word statistics, it is challenged by representing the semantic of language expressions independently of the specific words which are used. In the last decade, bag of words have been compensated with embeddings that are able to capture the meaning of language expressions by means of contextual vectors.

4.1.1 Bag of Words Representations with TF-IDF

Bag of words representations often use TF-IDF [58] scores, where TF stands for 'term frequency' and IDF stands for 'inverse document frequency'. TF-IDF is a numerical statistic based on word counts, that reflects how important a term is to a document with respect to a collection of documents.

For example the term 'and' could be considered important in one document, as it certainly occurs many times; however, given the same can be assumed for many more documents, the term 'and' cannot be considered important to characterize any document. The opposite holds for the term 'Paris' with respect to a corpus of documents about capital towns (i.e., it must occur within documents about France); however, it would not be distinctive with respect to a corpus of documents about tourism in France, as all of them would mention Paris multiple times.

In the context of systematic reviews, a publication can be represented through a vector of TF-IDF scores, with respect to a set of SRs. Each element of the vector corresponds to a term in a given vocabulary (usually, initialized from the set of SRs itself). Such an element is a real number obtained as the product between the TF and the IDF statistics. There are various ways for determining the exact values of both statistics, however, TF is the relative frequency of a term within a publication, and IDF is a measure of how much information the term provides (i.e., if it is common or rare across the set of SRs).

The majority of the latest citation and abstract screening models, which have been previously introduced in Chapter 2, shifted their attention to embeddings and only a few of them used bag of words representations. Embeddings have received, in fact, a lot of attention due to the well known weaknesses of bag of words based representations: they lose the ordering of the words and they ignore semantics of the words [59].

In the specific application context of SRs, the supremacy of embeddings

in regard to bag of words is debatable and both can actually be useful: in fact, scientific publications are fundamentally technical documents, where some specific keywords have their own importance that goes beyond semantics. Examples of such keywords are the name of proteins, diseases, medical treatments or procedures.

The representation power of bag of words with respect to embeddings has been addressed by studies such as morbidity identification in clinical notes [60], automation in job interview grading [61] and fine-grained event classification [62]. These studies all reported better results using bag of words comparing to embeddings. Specifically, the first of the cited works is in the medical domain. However, there is no work that provides a structured and motivated indication of the type of representations to use and most results are empirical.

4.1.2 Embeddings

Embeddings are vector representations of language expressions that capture their semantics. In fact, their vector space is designed so that words with similar meanings are closer together. Word embeddings [63], which provide static embeddings of words, have been initially proposed; then, *transformers* [64] introduced the notion of contextualized embeddings, where the meaning of a words depends on the words around it.

Word embeddings. Word embeddings are obtained by means of shallow neural networks that learn word features from a large corpus of generic data [65–67]. Word embedding models usually leverage a fixed vocabulary of words, which associates an embedding to each of them. These word embeddings can be applied to a multitude of downstream tasks (including updating SRs); to this aim, **task-specific architectures** [36], which include the pre-trained representations as features, are used. Some studies involving *SR update* undertook this approach [41, 43], using word embeddings to provide representations of publication titles and abstracts.

Transformers. A relatively recent advancement in the field of embeddings is the use of transformers [64], which are a deep learning architectures for language modeling that provide contextualized embeddings. Instead of having a fixed mapping between words in a vocabulary and their embeddings,

transformers provide representations that take into account the context of words. Specifically, embeddings of words are influenced by the embeddings of the surrounding words, thus, the same word will have different embeddings when used in different contexts.

Transformers have a notion of *token*, which is a sub-word. The main reason is that by using a vocabulary of words, like word embedding models do, there is a risk of following in 'out-of-vocabulary' errors (i.e., the word for which the embedding is requested is not available in the vocabulary). Tokens can be learnt from data and used to embed any input sequence.

Two major transformer-based language models are BERT [36] and GPT [68]. Both of them are pre-trained over large unsupervised data-sets of general data, requiring extensive processing capabilities (for example, training GPT required 8 GPUs for 30 days.[1]) GPT is a causal language model, which means it predicts the next word based on the context of the words coming before it. BERT is a masked language model, that predicts some masked words based on the context of the full input language expression. GPT, traditionally, is used in generative tasks given its causal modeling nature, while BERT is more used in tasks where it is important to obtain meaningfully representations of language expressions.

A common way to use transformer-based language models for downstream applications is **fine-tuning** [36]. This means optimizing the pre-trained model parameters on a specific dataset, and for a desired task. This approach represents the current state of the art, and the most recent works related to abstract screening and citation screening are based on it (see Section 2.2.3). Specifically, fine-tuning a language model is based on the following concepts:

- the fine-tuned model uses a pre-trained model as its *base model*; this means that the fine-tuned model architecture and its parameters are initialized using the pre-trained model;
- the fine-tuning model uses a dedicated final neural layer and a dedicated loss function (usually called the model's *head*), which are specific to the target down-stream applications;
- the fine-tuning dataset is used to train the fine-tuned model, that is, the base model and the model's head are used to process the training input sequences, and the loss function is used to fine-tune the model.

[1] https://openai.com/research/language-unsupervised

The fine-tuned model absorbs the domain specific language, thanks to the fine-tuning dataset: for example, using the Cochrane Reviews to fine-tune an embedding model lets it to understand the medical language and semantics. In addition, it learns to capture the language expression nuances which are meaningful to the specific task it has been fine-tuned for. Note that, although this task does not necessarily need to be exactly the same as the target down-stream application–also because the same fine-tuning model could be used for a family of down-stream applications—it can be designed to be significant to it. For example, learning to predict semantic similarity between pairs of sentences is useful to down-stream tasks like semantic search and semantic clustering. Finally, note that this fine-tuning approach is more advantageous compared the task-specific architectures usually employed with word embeddings: in fact, it is highly standardized and does not need to train from scratch a neural network, but just to optimize the pre-trained weights for the new scenario.

Several variants of BERT have been fine-tuned on scientific or health care datasets, such as SciBERT [57], BioMed-RoBERTa [69], and PubMed-BERT [70]. Most of the recent works in *living SR* that leverages the concept of fine-tuning are based on these type of models. In fact, compared to the original BERT model, they have been exposed to data which is closer to the specific domain of interest. For example, SciBERT [57] is a fine-tuned language model based on the BERT's architecture, i.e., it uses BERT as a base model for encoding the input sequences, and it uses the same BERT's pre-training tasks. SciBERT provides a convenient baseline for health care applications, because it is fine-tuned with scientific publications from Semantic Scholar.[2] Specifically, its fine-tuning corpus comprises 1.14M full text of papers. Semantic Scholar aligns to the *living evidence* problem in health care as it provides scientific publications, however, Semantic Scholar is cross-domain and it covers health care partially. Note that the fine-tuned SciBERT, in turn, is provided as a pre-trained model to anyone willing to use it for down-stream tasks.

4.1.3 Paragraph Embeddings

Systems for updating SRs need representations of publication titles and abstracts, which means obtaining one embedding for each title and abstract.

[2] https://scholar.google.com/

These are known as *paragraph embeddings*, because they represent a full paragraph of text. Usually, embedding models returns one embedding for each token in the input sequence. Thus, a common approach to obtaining paragraph embeddings, is to average all the token embeddings. In addition, some models, also return one embedding for the full input sequence. For example, BERT and its variants provide the *pooler output*, that is the embedding of a special token (named [CLS]) which is trained with a sentence-scoped task (i.e., next sentence prediction).

Input sequence embeddings of this type seem an ideal approach to producing paragraph embeddings, although they often pose two challenges for the downstream applications that use them. In one hand, fine-tuning is usually required to achieve adaptation to the specific downstream application. In the other hand, the embedding model maximum length of the input sequence might be too limited in regard to the typical application's domain documents. These challenges are especially important for health care *living evidences*, where the document abstracts tend to be lengthy and the task of determining their relevance to SRs is difficult in itself.

Fine-tuning paragraph embeddings. As discussed above, fine-tuning an embedding model has two parts: one is the dataset used for fine tuning, which should be representative of the specific domain so that the model can learn the relevant distributional semantics; the other one is the fine-tuning task, that is the objective function used to optimize the neural network parameters. A common approach that is relevant to paragraph embedding is the one adopted by BERT and its variants. They pre-train the [CLS] token for the 'next sentence prediction' task [36], which involves feeding two input sequences to the network and checking if the second sequence is the immediate successor of the first one. To do this, BERT uses the [CLS] token to cross-encode the two input sequences, and train it for binary classification.

Some applications are challenged by this cross-encoding approach and using the above methods for paragraph embeddings does not usually provide satisfactory results for them. Examples of these applications are large-scale semantic similarity, clustering, and semantic search. Specifically, they are challenged by the quadratic complexity of cross-encoding [71]; and updating SRs is also challenged the same way. In fact, considering the task of determining the similarity between n new abstracts and the m abstracts already included in a SR, it would be needed to cross-encode $m \times n$ sequence-pairs

to obtain a similarity score. Instead, it would be desirable to compute $m+n$ embeddings and use any distance function (e.g., the cosine similarity) to obtain a similarity score.

To this aim, Sentence-BERT [71] uses a siamese BERT-network to encode two input sequences independently, pool their outputs (through either the [CLS] token or the average of the token embeddings) and fine-tune for semantic similarity the shared BERT weights and the pooler weights. Specifically, as a loss function, the cosine distance is used. A similar approach is proposed by GPT [68].

These methods usually approach fine-tuning with sentence pairs which are either fully similar or completely dissimilar. Instead, when updating SRs, the desired notion of similarity is more subtle. The main issue is that some SRs belong to the same family of SRs, while other SRs belong to completely different sub-domains (and even different domains!). Thus, useful embeddings are those which are able to discriminate the relevance of publications to SRs in the same sub-domain. In other words, the relationship between a publication and a SR can follow in the following categories: (i) they publication is relevant to the SR; (ii) the publication is irrelevant to the entire SR's sub-domain; and (iii) the publication is irrelevant to the SR, but it could be relevant to other SRs in the same sub-domain. In other words, the embedding model should not get confused by sub-domains and, instead, it should be able to capture enough nuances to discriminate between SRs in the same sub-domain. The fine-tuning procedure is crucial to address this issue.

To this aim, SPECTER [72] suggests an interesting approach. SPECTER focuses on the scientific domain, though not specifically on the health care domain, where publications have a network of citations. Thus, the degree of similarity between two publications can be determined by their distance on the citations network. Specifically, SPECTER employs three siamese networks (based on SciBERT) to encode three scientific publication abstracts, and fine-tunes the SciBERT shared parameters for sentence similarity with a triplet loss function. To do so, out of the three publications, one is designated as the 'query document', and the other two have varying degrees of similarity to it. The similarity degree depends on the citation network, such that a direct reference from the 'query document' gives a similarity of 1, and an indirect reference gives a similarity of 0.5. The dissimilar publication has a similarity degree of 0. The triplet loss function incorporates these similarity scores and learns embeddings that capture a semantic similarity based on

citations.

Input sequence size. The most popular transformer based architectures support input sequences of limited size: for example, all the BERT variants are limited to 512 input tokens (i.e., all the tokens following the 512th one are ignored). Some empirical statistics on the dataset of Cochrane Reviews used for this research, show that the average length of an abstract is 2000 tokens and a few of them overcome 4000. Paragraph embedding models using BERT variants as their base model, would truncate Cochrane Reviews abstracts when calculating their embeddings.

One challenge of embedding long sequences depends on the *attention mechanism*, which is the building block of transformers [64]. Specifically, transformers employ several neural layers of *self-attention*, where the internal representation of each token is matched to the one of all the others. For example, given an input sequence of n tokens, each self-attention layer perform $n \times n$ operations, to update each token embedding. In other words, each token *attends* to all the other tokens. This quadratic complexity of the original attention mechanism is not well suited to represent long sequences.

Recently, language models supporting longer input sequences have been introduced, such as LongFormer [2] and others. They use variants of the original self-attention mechanism, where a sub-set of all the attention operations are performed. For example, LongFormer, as well as other similar pre-trained language models, uses a variant of the self-attention [64] mechanism where each token attends only to the ones in a surrounding window (this is named 'local attention'), instead of attending to all the tokens in the input sequence. In addition, some selected tokens attend to all the tokens in the sequence (this is named 'global attention'). Global attention can be configured, as the tokens attending globally depend on the fine-tuning task.

4.2 *ContReviews* Embeddings

As discussed in the previous chapter, the *ContReviews*'s embedding model is used to represent publications and SRs, and these representations are used by the content-based recommendation model to match them. In this context, the following requirements are crucial to fine-tune an adequate embedding model: (i) the corpus covers the health care domain and comprises documents similar to the ones in the *living evidence* (i.e., Cochrane Reviews); and (ii) the

fine-tuning task is enough meaningful in regard to the downstream problem of determining if a publication is relevant to a SR.

The first requirement can be simply achieved by fine-tuning the *ContReviews* embedding model using the abstracts available in the Cochrane Reviews. The fine-tuning task, instead, is less obvious and several approaches can be considered. The following subsections describe the two of them used with *ContReviews*.

4.2.1 Fine-tuning for Abstract Screening

The first approach to fine-tuning the *ContReviews* embedding model is using the abstract screening task. To this aim, the neural network shown in Figure 4.1 is proposed. It encodes an input sequence using SciBERT [57] as the base model and applies a classification head to the pooler output. The network is auto-regressive in that it is applied to Cochrane Reviews abstracts to learn predicting their relevance to the Cochrane Reviews. The classifier is 'multi-class' and 'multi-label', because Cochrane Reviews comprise multiple SRs, and each abstract can be relevant to multiple SRs.

The linear layer in the above neural network outputs a tensor of length equal to the number of classes (i.e., SRs): the *sigmoid* activation function maps each of those output elements to the $[0, 1] \in \mathbf{R}$ range, so that they can be interpreted as class probabilities. The network is trained to minimize the Binary Cross Entropy (BCE) loss between the network's output class probabilities and the true class probabilities. The BCE is typically used as a loss function for binary classification models: note that a multi-label, multi-class classification model can be seen as a series of binary classification models (i.e., one model per class). For each class, the BCE computes the loss as $bcs(x, y) = ylog(x) + (1 - y)log(1 - x)$, where x is the predicted class probability and y is the true class probability (i.e., it is either 0 or 1). For a multi-class, multi-label classification model, the BCE returns the average loss over all classes, i.e., $bce(X, Y) = mean(bce(X_1, Y_1), ..., bce(X_n, Y_n))$, where X and Y are vectors of probabilities.

ContReviews uses the fine-tuned SciBERT pooler output as a representation of publication abstracts. *ContReviews* evaluation in Chapter 5.4.4 shows that the fine-tuned SciBERT model provides better results than its corresponding pre-trained model.

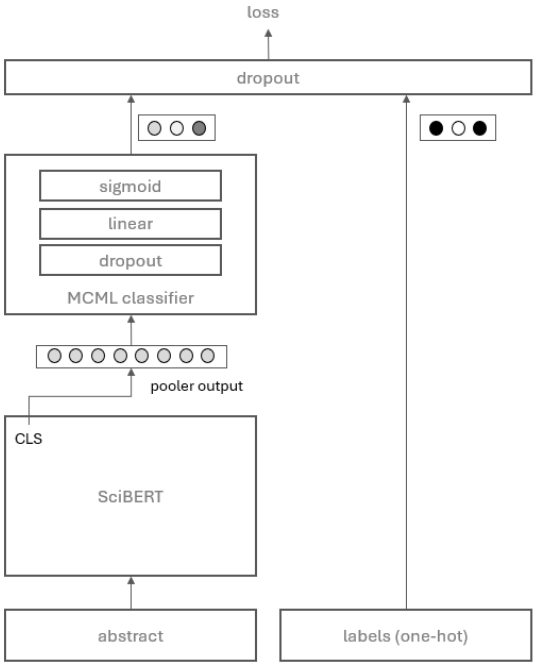

Figure 4.1: Neural architecture to fine-tune SciBERT for abstract screening, through a multi-class, multi-label (MCML) classifier. The pooler output is used as an abstract embedding at inference time.

4.2.2 Fine-tuning with Semantic Similarity.

Fine-tuning for abstract screening is challenged by the following issues. First, as discussed in Section 2.2.3, there are few abstracts available for each SR, especially in an health care related *living evidence*; thus, each class is trained with scarce data. In addition, SciBERT supports relatively short input sequences, if compared to the average size of abstracts in Cochrane Reviews, as introduced in the previous section.

To assess weather the above two challenges have a tangible impact on the quality of the generated embeddings, an alternative embedding model is proposed. Specifically, instead of SciBERT, LongFormer [2] is used as a base pre-trained model; in fact, LongFormer supports longer input sequences (i.e., up to 4096 tokens). In addition, to make use of more training data for each inference case, semantic similarity is chosen as the fine-tuning task. In fact, given two input sequences, the output of the network should be either

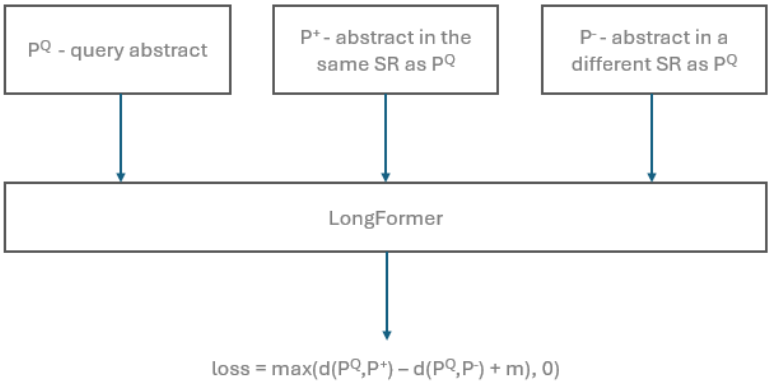

Figure 4.2: Neural architecture to fine-tune LongFormer for semantic similarity. The LongFormer pooler output is used as an abstract embedding at inference time.

'similar' or 'not similar'; thus, a large amount of supervised data can be used to train those two cases.

The proposed embedding model, which is represented in Figure 4.2, follows the same approach as SPECTER [72] with the two following differences: (i) LongFormer is used as the base model (instead of SciBERT); and (ii) the abstracts' similarity is determined by the SRs that contain them, not by their citation network as in the original SPECTER model. Likewise SPECTER, the pooler output is used as the input sequence embedding, and it is trained by means of a triplet loss function. As represented in Figure 4.3 the pooler output leverages 'global attention', i.e., the one token which feeds the pooler output attends to all the other tokens in the input sequence.

A fine-tuning dataset can be constructed from the Cochrane Reviews, where each element comprises three abstracts (note that Cochrane Reviews has a notion of 'family', which represents a research sub-domain comprising many SRs which are close to each other):

- the first abstract is randomly sampled from all the Cochrane Reviews, and is named the 'query abstract';
- the second abstract is selected to be similar to the 'query abstract', i.e, it is randomly selected from the same SR where the 'query abstract' is included;
- the third one is selected to be not similar to the 'query abstract', i.e., it is randomly sampled from either all the other SRs in the same family

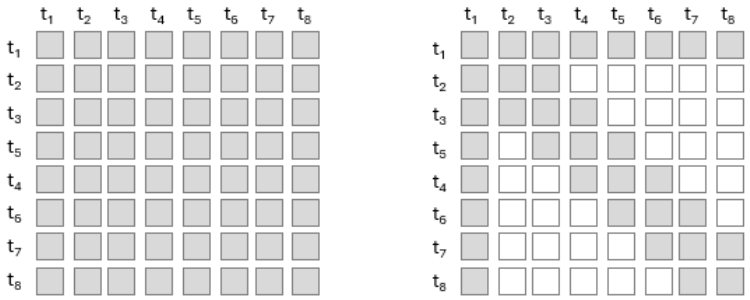

Figure 4.3: A representation of full self-attention (left) and local/global self-attention (right) as proposed by LongFormer [2]. With full self-attention every token attends to every other token, leading to quadratic complexity. With local/global self-attention, each token attends to their immediate context (in the picture this context is a window of 3 tokens) and some selected tokens attend to all the other tokens (in the picture the first token).

as the 'query abstract', or from all the other families, or from outside the Cochrane Reviews.

Discriminating the source of the third abstract ensures that the model is exposed to different extent of dissimilarity. In fact, abstracts coming from the same family are 'less dissimilar' compared to abstracts coming from different families or even from outside the Cochrane Reviews. This training approach aims to expose the model to subtle inference cases, so that it learns the difficult task of addressing relevance to SRs in the same family.

To train the network, the loss function is the same used for SPECTER [72]. Let A^Q be the 'query abstract', A^+ an abstract similar to A^Q and A^- an abstract not similar to A^Q:

$$loss = max(\|A^Q - A^+\|_2 - \|A^Q - A^-\|_2 + m), 0)$$

where, following the SPECTER settings, m is a constant set empirically to 1 and $\|\cdot\|_2$ is the $l^2 - norm$, which is defined by $\|x\|_2 = \sqrt{\sum_{i=0..n} x_i^2}$. The above loss function steer the network to learn embeddings as vectors being close to each other for abstracts included in the same SR (and far otherwise).

4.3 Final Remarks

This chapter examined different ways of representing publication titles and abstracts, which are likely the most important features for the SR updating

task. Two methods, bag of words and embeddings, were compared, with the latter being more prevalent in recent research on SR updating. However, bag of words can also offer useful representations, as they can capture key phrases that have significance beyond semantics in the scientific literature. *ContReviews* uses both methods.

In addition, this chapter summarized the current literature on fine-tuning embeddings for SR updating, and introduces two novel methods. The first method fine-tunes SciBERT for abstract screening, and the second method uses LongFormer for semantic similarity. Both methods leverage the Cochrane Reviews publications to adapt to the domain-specific language. Moreover, the Cochrane Reviews offer valuable information to create the supervised dataset, which consists of the relationships between publications, SRs and SR families.

In summary, this chapter introduced how to create embeddings for the publication abstracts, which can be used as an input for the *ContReviews* content-based recommendation model. Based on this outcomes and the formal definition of *ContReviews*, the following chapter presents the evaluation results.

Chapter 5

Evaluation of *ContReviews* over Cochrane Reviews

This chapter reports the *ContReviews* evaluation results. Section 5.1 discusses data preparation issues and construction of train and test datasets; Section 5.2 regards training the relevance assessment function model, including calculation of the likelihoods of relevance; finally, Section 5.3 discusses the *ContReviews* evaluation results and compares it to a baseline. Specifically, the model's ability to correctly infer the relevance of the new publications (i.e., publications in the test dataset) to SRs is measured in terms of classification metrics of precision with high recall.

5.1 Data Preparation

As motivated in Section 2.3, datasets and evaluation metrics to assess *living evidence* performance are either missing or not homogeneous. For this reason, *ContReviews* performance is evaluated through a dedicated dataset, which is extracted from a large subset of *Cochrane Reviews* (CRs).

5.1.1 Pre-processing

The subset of the Cochrane Reviews (CRs) used for evaluating *ContReviews* contains thousands of SRs; each of them includes a set of relevant publications. As anticipated, the publications included in a SR can be used to construct positive (i.e., with a positive label) training pairs of one publication

and one SR; moreover, negative ones (i.e., with a negative label) can be constructed as pairs of one SR and one publication which is not included in it. In addition, each training example must be enriched with one set of features for each publication property; these features are the likelihoods of relevance. To evaluate *ContReviews* the publication properties used are: title, abstract, authors and citations network (as previously discussed in Section 3.4.2, the set of publication properties is not limited to these). However, these properties are provided by CRs in text format, which is not ideal to construct a robust dataset for training and testing. In fact, while title and abstract are inherently textual properties, citations and authors must be managed as entities. The following cases could apply:

- The same author is reported on different studies with a different name or different authors are reported on different studies with the same name. For example, it is subtle to determine if 'J Thomas' and 'JD Thomas' refer to the same author, which requires explicit disambiguation (e.g., considering the e-mail or the associated University).

- Citations often exist in different versions, e.g., the same work could have been first presented to a conference, then released as a pre-print and finally published in a journal. The research is the same, but it might be subtle disambiguate references to it. In fact, for example, the title might have been slightly changed.

To disambiguate citations and titles the AKG could be used. In fact, it provides de-duplicated and indexed entities and a schema with formal relationships between them. For example, the author 'James Thomas' from the 'University College London' is an entity and publications authored by him reference that entity in a non ambiguous manner. From another angle, all the publications included in the AKG are indexed and, among others, have a title, an abstract, cited publication references, and author references. Thus, the AKG is an ideal resource to collect in a robust manner the publication properties required by *ContReviews* (i.e, title, abstract, authors, and citations network). To do so, the publications in CRs are matched to those maintained in the AKG. i.e., OpenAlex [1]. OpenAlex provides APIs [1] which make it easy to run structured queries against its content. Specifically, some properties of the publications in the CRs (i.e., title, journal, authors, volume,

[1] https://docs.openalex.org/#access

Table 5.1: **Statistics about train and test datasets**: number of SRs and their included publications.

	Train dataset		Test dataset	
	CR-5	**CR-40**	**CR-5**	**CR-40**
Num. SRs	6395	699	6395	699
Num. sub-domains	52	52	52	52
Tot num. publications per SR	141240	45636	22955	4800
Min num. publications per SR	3	3	2	2
Avg num. publications per SR	22	65	3	6
Max num. publications per SR	596	363	10	10

issue, pages and DOI) are used to formalize and run OpenAlex queries, and retrieve corresponding publications.

As a consequence of matching publications from CRs to OpenAlex, a new set of publications is obtained. Each of them is associated to the SRs it is included into, its properties and the publication date (which is useful to split train and test publications). From this new set of publications, two distinct datasets are obtained; they are denoted as **CR-5** and **CR-40**, corresponding to all the SRs from CRs including respectively at least 5 publications and 40 publications (Table 5.1 reports some statistics). Specifically, CR-5 is useful to evaluate *ContReviews* with both small and large SRs, and CR-40 to compare *ContReviews* to state of the art methods. In fact, as previously discussed in Section 2.2.3, state of the art methods for abstract screening leverage one model per SR, requiring enough publications to support model training.

5.1.2 Train and Test Datasets

To evaluate *ContReviews*, CR-5 and CR-40 are split in two datasets, one used for training and one for testing (Table 5.1). The test dataset is used to simulate the set of new publications that, in reality, would be provided by the academic knowledge graph (AKG). In fact, it is not feasible to get test publications from the AKG directly, because determining their actual 'label' would require excessive manual work.

To discriminate which publications to use for training and which one for testing, the date of publication is used: the most recently published publications in each SR are used to construct the test dataset, while the others are used for training the relevance assessment function model. The

rationale to consider the most recent publications for testing is preserving the temporal order of publications. This is particularly relevant for the citation network. In fact, the older publications do not reference the newer ones, thus, having them in the test dataset would cause missing key references to the *living evidence* in the training dataset.

The elements of each dataset are relative to one publication and one SR; moreover, they comprise the features (i.e., the likelihoods of relevance, as introduced in Section 3.4.2) and the binary label. The latter holds *True* if the publication is relevant to the SR (*positive publications*), and *False* otherwise (*negative publications).*

The date of publication can be extracted from the AKG together with the other publication properties: for each SR, the *most recent* 10% of publications are used for testing, and the rest is used for training. Positive publications can be easily generated by simply considering the publications included in the SRs. Instead, CR-5 and CR-40 do not comprise any negative publication; thus, they are artificially generated by simply considering that publications included in one SR can be taken as negative publications for other SRs.

To have enough differentiation an additional source of negative publications has been used, that is a set of health care related publications not included in any of the considered SRs in CRs (*external publications*). The rationale behind using external publications is that the considered SRs from the CRs do not cover the entire domain of health care, and so do all the CRs. Thus, there will be some health care related publications which are not relevant to any of the SRs in the *living evidence*. The relevance assessment function model must correctly manage the inference for those profiles of research which are not comprised by the considered SRs.

The test datasets for CR-5 and CR-40 are generated as follows.

- Up to the 10% newest publications included in each SR are considered as positive publications; and, to be realistic, they are constrained to be at least 2 and at maximum 10 per SR. Each of them, together with the SR they are taken from, constitutes an element in the test dataset with a positive label.

- For each positive publication, 25 negative ones are sampled from the external publications; the SR which includes the positive publication together with each one of the external publications constitute an element in the test dataset with a negative label.

The rationale of this setting is to simulate a realistic stream of new publications, where most of them are irrelevant to the SRs in the *living evidence* (in fact, they are sampled from external publications) and only few of them are relevant for each SR.

The training datasets are generated in a similar manner. The positive publications are constrained to be at maximum 75 per SR, to streamline computational efficiency; moreover, 10 negative publications every positive one are also sampled from the other SRs. Conversely to the test dataset, where negative publications are external to CRs, the negative publications in the train dataset are sampled from other SRs within CR-5 and CR-40: this is specifically purposed to support the calculation of likelihoods of relevance. We briefly recall that, as detailed in 3.4.2, a likelihood of relevance reflects the degree of relevance of one publication to a SR *relatively* to the relevance of other publications (positive and negative). To make likelihoods of relevance effective, their calculation should involve a set of publications that are similar to each other, hence, we have sampled them from within CR-5 and CR-40.

5.2 Learning the Relevance Assessment Function

To learn a relevance assessment function for the content-based recommendation model, a binary classification model is trained using the dataset introduced in the previous section. The elements of this dataset correspond to pairs of one publication and one SR, and they comprise the model's features (i.e., the likelihoods of relevance) and the binary label. In this section the actual calculation of likelihoods of relevance and the training procedure are discussed.

5.2.1 Calculating the Likelihoods of Relevance

Calculating the likelihoods of relevance involves obtaining property-specific vector representations of publications and SRs, as formalized in Section 3.4.2. These vector representations are obtained as follows, depending on the specific publication property they are based on:

- *Binary vector representations based on authors and citations*: vocabularies of authors and citations are constructed from de-duplicated au-

thors and citations provided by OpenAlex. The vocabularies are directly used to construct the binary vector representations.

- *Bag of words of titles and abstracts*: text pre-processing operations are applied to titles and abstracts, i.e., lower-casing language expressions, removing stop-words, and tokenizing over uni-grams. All the uni-grams obtained from all the publications in all the SRs are used to construct a vocabulary of terms, which is directly used to construct the vectors of TF-IDF scores.

- *Title and abstract embeddings*: different language models are used, i.e., SciBERT pre-trained, SciBERT fine-tuned for abstract screening, and LongFormer fine-tuned for semantic similarity.

5.2.2 Training the Relevance Assessment Function Model

A binary classification model is used to learn a relevance assessment function for the content-based recommendation model. The classification model is based on LightGBM [73], which is an implementation of the Gradient Boosting Decision Tree (GBDT) ML algorithm [74]. We trained a LightGBM model.[2] The classification model is trained over the training set previously introduced in this chapter: its elements regard pairs of one publication and one SR, and comprise their likelihoods of relevance and a binary label representing relevance. As discussed in Section 5.1.2, the train dataset holds 10 negative publications for each positive one: to avoid any issue related to class imbalance, the negative publications are downsampled as suggested by [19].

5.3 Evaluation

The evaluation was conducted using the datasets and the training methodology described in Section 5.1.2. *ContReviews* performance is compared to two baseline models in terms of precision and recall, as described in details below. The evaluation results are reported in Table 5.2.

[2] https://lightgbm.readthedocs.io/en/stable/

5.3.1 Baseline

As motivated in Section 2.3, common evaluation metrics and benchmarks in the context of *living evidence* are not available; thus, two baseline models was developed over the CR-40 dataset. The first baseline model is inspired by the one proposed in [12], which is based on the similarity between the embeddings of publications and the embeddings of SR eligibility criteria (which are still language expressions). Instead of the eligibility criteria, the first baseline uses publication abstracts: specifically, the embedding of a SR is constructed as the average of the embeddings of its included publication abstracts. This is coherent with the content-based recommendation model that the baseline is meant to be compared with, which represents SR with the same strategy.

The second baseline model follows the abstract screening methods proposed in several studies at the state of the art [19, 38, 41, 43], where relevance is seen as the result of a binary classification model over the embeddings of publication abstracts. Specifically, the second baseline uses one binary classification model for each SR, which leverages the embeddings of abstracts as features to infer relevance to its target SR. To be coherent with the relevance assessment function model, based on the LightGBM algorithm as discussed in Section 5.2, the second baseline model also uses LightGBM.

5.3.2 Evaluation Metrics

To assess evaluation performance the classification metric of precision with recall of (at least) 97% is used, as proposed in [19, 38, 41, 43] (note that these works mostly consider 95% as the evaluation recall threshold; however, 97% is a more challenging objective). This is because the priority for SR updating is to capture *all* the relevant publications (that is, recall should be as high as possible); and, although precision can be sacrificed in the name of high recall, achieving good precision is still desirable to lower the number of new irrelevant publications recommended to reviewers and, ultimately, to reduce the amount of manual work.

To calculate precision and recall for SRs, the following procedure is applied to each individual SR:

- for each publication in the SR's test dataset, the likelihood of relevance to the target SR is calculated;
- initially, all the test publications are considered included in the target

SR, regardless of their likelihood of relevance;

- all the new publications are sorted ascending by their likelihood of relevance to the target SR;
- starting from the one with the lowest likelihood of relevance, the new publications are excluded from the target SR one by one;
- precision and recall are computed and recorded at each step of the above process;
- a threshold of 97% over the recall is considered to chose the best performing step; and
- the actual precision and recall (which will be between 97% and 100%) are recorded for the target SR.

The rationale behind this approach is that a *living evidence* model should perform well for all the SRs it contains; thus, it is interesting to monitor the distribution of precision and recall over all the SRs, rather than one single value aggregating all the SRs. In more detail, the latter scenario would correspond to having one single confusion matrix, aggregating all the prediction operations over all the SRs. This would make it difficult to analyze the model's performance in relation to SR characteristics. The proposed approach, instead, is based on one confusion matrix per SR which means the following for each SR: (i) the SR test publications are scored with a model, and a likelihood of relevance is obtained; (ii) all the most relevant test publications are taken until the recall threshold (i.e., 97%) is reached; and (iii) the precision value provides an indication of the fraction of the taken publications that are irrelevant. This approach supports decision making, such as:

- for each SR, find the best threshold in the test phase and use it for inference;
- obtaining a measure of the model's accuracy for each SR, to assess the reliability of recommendations at inference time;
- identify the low-performing SRs and take correction actions (e.g., manually seed additional publications);

- profile the characteristics of the good performing SRs (e.g., the number of included publications) and use them to profile the ones which are ready to be included in the *living evidence*; and
- report and monitor the quality of the *living evidence*.

5.3.3 *ContReviews* Evaluation

Evaluation results are reported in Table 5.2, in terms of the evaluation metrics introduced in the previous subsection:

- *ContReviews@lGBM* refers to the *ContReviews* content-based recommendation model, using LightGBM as the relevance assessment function model; as model features it uses the bag of words and the embeddings of publication textual properties (title and abstract) and binary vector representations of entities (authors and citations);
- *similarity@eABST* refers to the first baseline model, i.e., the one using cosine similarities between embeddings of publication and SR abstracts);
- *classification@eABST* refers to the second baseline model, i.e., the one using SR-specific binary classification models for abstract screening.

The baseline models leverage both the pre-trained SciBERT language model and its fine-tuned version. The notations *pt* and *ft* are used to express when the pre-trained or the fine-tuned version of the embedding model is used. Specifically, the fine-tuned embedding model is trained with the abstract screening task (i.e., SR prediction from abstracts), that is described in Section 4.2.1.

The *ContReviews* content-based recommendation model is trained and tested with both the CR-40 and CR-5 datasets. Instead, the *classification@eABST* baseline results are only reported for the *CR-40* dataset. In fact, to train one abstract screening model per SR (which is the case for the baseline using LightGBM), a minimal number of publications should be available: to this aim, 40 publications per SR are empirically considered as that minimal number, and both the baseline models have been reported for the *CR-40* dataset only.

Table 5.2: Average precision and recall over all the SRs. The closest recall to 97% is specifically considered. The standard deviation refers to precision.

	Dataset	Avg precision	Avg recall	Std Dev
classification@eABST (pt)	CR-40	0.115	0.981	0.202
classification@eABST (ft)	CR-40	0.211	0.982	0.356
similarity@eABST (pt)	CR-40	0.065	0.981	0.092
similarity@eABST (ft)	CR-40	0.227	0.981	0.372
ContReviews@lGBM	CR-40	0.974	1	0.086
ContReviews@lGBM	CR-5	0.981	1	0.086

Description of Evaluation Results

Results show that *ContReviews* achieves average precision above 97.4% with recall of 100% with both datasets; instead, both the baseline models (*classification@eABST (pt)* and *classification@eABST (ft)*) achieve recall greater than 97% as *ContReviews* does, but at the price of precision being significantly lower. As discussed, lower precision rates correspond to an higher amount of irrelevant publications recommended to reviewers, triggering additional human labour.

Moreover *ContReviews* shows better standard deviation over SR precision values, meaning that the average precision is more regular compared to the base models. Note that the latter have much higher standard deviation, except *similarity@eABST (pt)* which, however, achieves the lowest precision.

In addition, Table 5.3 reports the statistics about the gap in precision between *ContReviews* and the baseline models. It shows that for a small fraction of SRs (i.e., the *% worst* column is less than 4%) the baseline models perform better than *ContReviews* in terms of precision with recall grater than 95%. However, *ContReviews* still achieves precision values above 86% (*Avg precision* column), though the gap with the baseline model is consistent (*Avg gap* column is less than 12%). Moreover, shifting the threshold over recall from 95% to 97%, *ContReviews* beats the baseline models for all the SRs.

Figure 5.1 shows descriptive statistics about the results summarized in Table 5.3. Specifically, it uses candlesticks[3] to display the precision values for each SR, and represent them for both *ContReviews@LightGBM* (*precision_y*) and the best baseline model (*precision_x*). In addition, their gap (*delta_p*)

[3] https://matplotlib.org/stable/api/_as_gen/matplotlib.pyplot.boxplot.html#matplotlib.pyplot.boxplot

Table 5.3: Statistics about the gap in precision between the baseline models and *ContReviews*, with the CR-40 dataset. *% worst* of all the SRs perform better with the baseline models; for them *ContReviews* achieve average precision as of *Avg precision* with a gap to baseline models as of *Avg gap*.

Model	Threshold	% worst	*Avg precision*	Avg gap
classification@eABST (ft)	0.95	3.15%	0.867	-0.113
similarity@eABST (ft)	0.95	3.78%	0.870	-0.106
classification@eABST (ft)	0.97	0%	na	na
similarity@eABST (ft)	0.97	0%	na	na

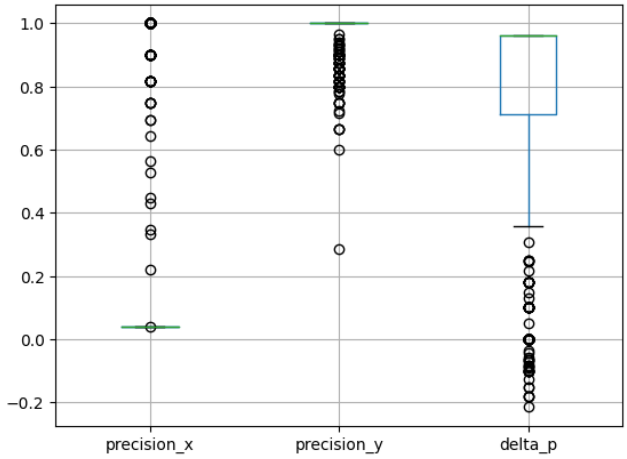

Figure 5.1: Candlesticks of precision with recall of 95%, for *similarity@eABST (ft)* (leftmost), *ContReviews@LightGBM* (center) and their gaps (rightmost). The box extends from the first quartile to the third quartile of the precision points, with a line at the median; and the whiskers extend from the box to the farthest data point lying within 1.5x the inter-quartile range from the box. Outlier points are those past the end of the whiskers.

is also shown. The candlesticks show that the inter-quartile range of both models (that is, the precision points between the first quartile and the third quartile) collapsed around their median value; and that the SRs achieving better precision with the baseline model are outliers.

Compared to the bare mean values in Table 5.2, the candlesticks provide a better intuition about the performance of *ContReviews* compared to the baseline models. However, it is required to formally confirm that the difference between the means is statistically significant. To this aim a statistical significance test was used, to check the default expectation ('null hypothesis'

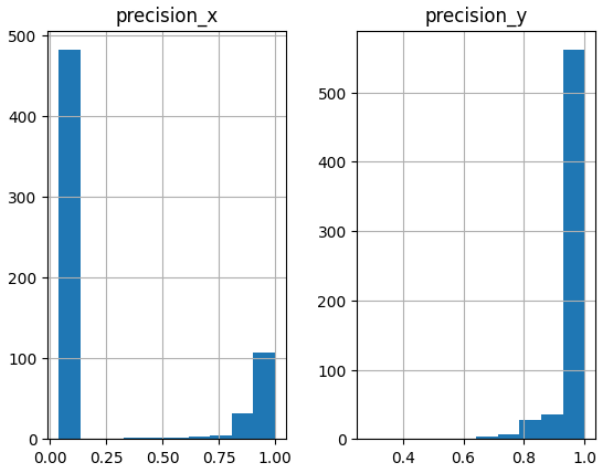

Figure 5.2: The distributions of the values of precision with recall of 95%, for the baseline model *similarity@eABST (ft)* (*precision_x*) and *ContReviews* (*precision_y*.

or 'status quo') that both samples of precision values were drawn from the same population. Failing to reject this hypothesis means that there is no significant difference between the means, i.e., they are different by chance, and so it is the observed difference in performance between the baseline models and *ContReviews*. Thus, the objective of this test is to reject the null hypothesis.

Figure 5.2 displays the distribution of the precision values for the base model (left) and *Contreviews* (right), which intuitively shows that the means come from different populations and their difference is meaningful. To provide a statistical proof, two steps were undertaken. First, a statistical test[4] rejected the null hypothesis that the data samples follow a normal distribution. Second, given there is not normal distribution of data samples, the Kolmogorov-Smirnov[5] two-sided test was used to check the null hypothesis that two independent samples are drawn from the same continuous distribution. The test rejected the null hypothesis, hence, the means of the baseline model and *ContReviews* are drawn from different populations, and their difference is statistically meaningful.

An interesting question would be if there is any variable affecting this

[4]Python function 'normaltest', https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.normaltest.html

[5]Python function 'ks_2samp', https://docs.scipy.org/doc/scipy/reference/generated/scipy.stats.ks_2samp.html

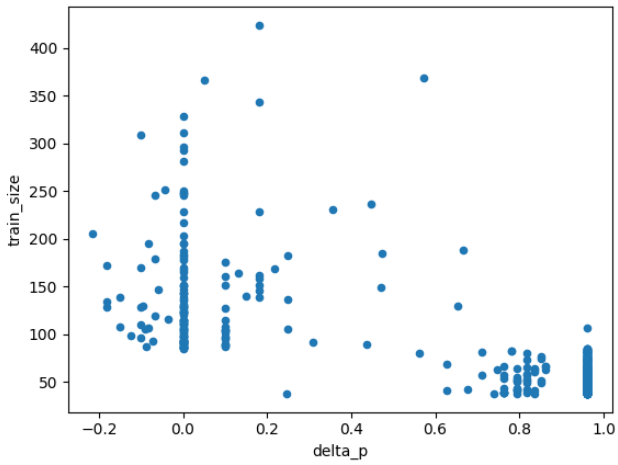

Figure 5.3: Scatter plot of SRs, by their gap in precision (between *ContReviews* and *similarity@eABST (ft)*) and their size (i.e., number of included publications).

difference between the means. Specifically, the question is whether the size of the SR affects the precision of the inference. To this aim the chart in Figure 5.3 was plotted, which displays one point per SR by the coordinates of the gap in precision at 95% recall (*delta_p*) and the SR size (*train_size*). The chart empirically shows no correlation exists between the observed variables.

The same conclusion can be drawn by calculating the *Kendall's Tau*,[6] which is a measures of the non-parametric rank correlation between to variables. The value of *Kendall's Tau* between the SR size (*train_size*) and the gap in precision (*delta_p*) is -0.5176, which confirms the two variables are not dependent. Note that the Kendall correlation is similar to the Spearman correlation, however, it measures the dependence of two variables, which is a broader concept than correlation. On one hand, dependence refers to the relationship between two variables, where the value of one variable directly affects the value of the other. On the other hand, correlation measures the strength and direction of the linear relationship between two variables.

Finally, *ContReviews* shows good adaptation to large datasets with small SRs, i.e., CR-5 holds 6000+ SRs with 22 publications included in average, with some of them having a few publications included. The chart in Figure 5.4 shows the trend of precision with recall of 100% for SRs with up to

[6] https://pandas.pydata.org/pandas-docs/stable/reference/api/pandas.DataFrame.corr.html

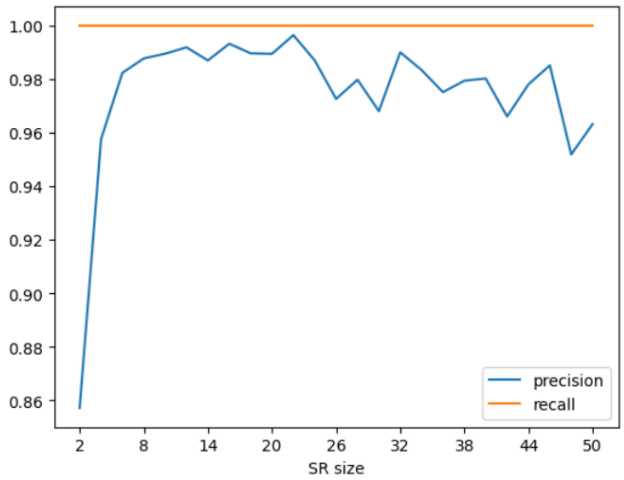

Figure 5.4: *ContReviews* precision with recall of 100%, obtained with the CR-5 dataset. Data points are plotted for SRs with 50 publications included at maximum, in bins of size 5. For each bin, the average precision is shown.

50 publications included, being above 86%. The trend of precision leans to be even better with small SRs and, considering the SRs with more than 50 publications included (which are not shown in the chart), the trend tends to be less regular.

A possible explanation is twofold. Large SRs occur less often and those with unusual precision values stand out more clearly. On the other hand, extremely large SRs (i.e., those with hundreds of publications included) tends to be more generic, and the task of inferring relevant publications becomes intrinsically more difficult. Hence, a possible interpretation of this analysis is that *ContReviews* performs well where the SRs are truly more selective, and inevitably less well where the SRs are more generic. On the contrary, the traditional models (which are based on SR-specific machine learning models, i.e., they are trained on each SR) prefers large SRs, which is where the very efficient inference is truly more difficult to obtain.

5.4 Ablation Studies

To assess the evaluation results introduced above, we focus on (i) the contribution of re-scaling the cosine similarities when calculating likelihoods of relevance, as illustrated in Section 3.4.2; (ii) the contribution of using mul-

tiple publication vector representations; and (iii) using different pre-trained language models and fine-tuning techniques. To this aim, a simplified version of *ContReviews*—that is called *single property model*—is introduced.

5.4.1 Definition of Single-property Models

Single-property models are simpler *ContReviews* content-based recommendation model, which are based on a single publication property. Thus, they can only count on a single likelihood of relevance, which corresponds to the considered publication property. This likelihood of relevance is directly used to assess the relevance to SRs.

For example, the single property model using embeddings of abstracts would infer one publication relevance to one SR as follows: the embedding of the publication abstract is obtained; it is then matched to the one of the SR, and a similarity score is obtained through the cosine similarity function (recall that the SR's embedding is obtained as the average of the abstract embeddings of its included publications); the cosine similarity is re-scaled and one likelihood of relevance is obtained; then the likelihood of relevance is immediately used as the final likelihood of relevance between the publication and the SR.

5.4.2 Re-scaling Cosine Similarities

To assess the importance of re-scaling the cosine similarities two models are compared: one (*ContReviews@eABST (ft)* in Table 5.4) is a single property model based on embeddings of abstracts; the other one is the baseline model *similarity@eABST* reported in Table 5.2). Note that these two models only differ because the former re-scales cosine similarities to obtain likelihoods of relevance, while the latter uses absolute cosine similarities. *ContReviews@eABST (ft)* largely outperforms *similarity@eABST* (with or without fine-tuning), that is, re-scaling similarity scores is effective. Moreover, we also report in Table 5.4 the evaluation results for other single property models (i.e., *sABST*, *sTITL*, *sCITA*, *sAUTH*): they all outperform the baseline models in Table 5.2, showing effectiveness of re-scaling.

Table 5.4: Evaluation results for models using one single property, in terms of precision with recall of 0.95 or greater. The notion of *P(R) is used*, where P and R respectively are the precision and the recall. *ContReviews@feat* uses one single property, using the identity deterministic relevance assessment function. *feat* can be embeddings of abstracts ($eABST$), bag of words representation of abstracts ($sABST$), embeddings of titles ($eTITL$) or binary representations of citations ($sCITA$) and authors ($sAUTH$).

Model	Dataset	Mean - P(R)	Std dev - P(R)
ContReviews@eABST (ft)	CR40	0.964 (0.981)	0.11 (0.036)
	CR5	na	na
ContReviews@sABST	CR40	0.944 (0.981)	0.131 (0.036)
	CR5	0.919 (0.995)	0.175 (0.02)
ContReviews@sTITL	CR40	0.875 (0.981)	0.216 (0.364)
	CR5	0.87 (0.995)	0.232 (0.19)
ContReviews@eTITL (ft)	CR40	0.953 (0.981)	0.132 (0.036)
	CR5	0.881 (0.995)	0.222 (0.02)
ContReviews@sCITA	CR40	0.517 (0.981)	0.470 (0.036)
	CR5	0.623 (0.995)	0.461 (0.02)
ContReviews@sAUTH	CR40	0.191 (0.981)	0.349 (0.036)
	CR5	0.173 (0.995)	0.334 (0.02)

5.4.3 Using Multiple Properties

To assess the effectiveness of using multiple vector representations, the *ContReviews* content-based recommendation model (*ContReviews@lGBM (ft)* in Table 5.2) is compared to the single property models (*ContReviews@eABST (ft)* in Table 5.4). The difference between these two models is clearly that the former uses all the available publication properties and a relevance assessment function is learnt from their corresponding likelihoods of relevance. Instead, as introduced above, the single property model only uses one single publication property. The former achieves the best results over all the evaluation tests, showing that using an ensemble of vector representations is effective.

Indeed, single property models, especially those using textual features, achieve competitive average results and are more computationally efficient. However, their standard deviation is considerably higher, suggesting they are less regular across all the SRs.

Table 5.5: Evaluation results for different embedding models.

		LightGBM		**Cosine similarity**	
Model	Method	Precision	Recall	Precision	Recall
SciBERT pre-trained	Pooler	0.073	0.979	0.046	0.979
	Average	0.141	0.979	0.213	0.979
SciBERT fine-tuned	Pooler	0.209	0.979	0.213	0.979
	Average	**0.213**	0.979	**0.218**	0.979
LongFormer pre-trained	Pooler	0.134	0.976	0.072	0.976
	Average	0.129	0.979	0.058	0.979
LongFormer fine-tuned	Pooler	**0.212**	0.979	**0.212**	0.979
	Average	0.167	0.979	0.091	0.979

5.4.4 Using Different Embedding Models

Multiple embedding models have been evaluated on their ability to infer new publications relevance to SRs in a *living evidence*, as reported in Table 5.5. SciBERT and LongFormer are used as base models. For both of them, the CR-40 dataset was used to provide fine-tuning data. Moreover, SciBERT was fine-tuned for SR prediction (multi-class/multi-label classification), while LongFormer was been fine tuned for semantic similarity. To obtain abstract embeddings two methods have were used, respectively denoted as *pooler* and *average* in Table 5.5: the first one considers the token trained to represent the entire input sequence; the second one averages the embeddings of all the input sequence's tokens. Two inference algorithms were considered: the first one is a binary classification model (i.e., LightGBM, similarly to the previous evaluations) using the input sequence embeddings as features; the second one simply uses the cosine similarity between the SR's embeddings and the publications embeddings and a threshold for classification.

Based on the evaluation results reported in Table 5.5, the following observations can be made.

- Almost all fine-tuned models achieve consistent performance: the base-model, the fine-tuning task and the evaluation model do not seem to be important to determine the quality of the obtained embeddings, at least in regard to the SR updating task.

- The fine-tuned model based on LongFormer and the average method for calculating sentence embeddings is the only one that performs worse

than the other fine-tuned models, but it still does better than the pre-trained ones. The averaged embeddings, in this case, are affected by a local attention mask. That is: each token only attends to its neighborhood tokens, not to the whole sequence. In other words, attending to whole sequence seems to be an important factor.

- All the fine-tuned models significantly out-perform their pre-trained versions, showing the importance of fine-tuning on the *living evidence* data.

Based on these observations, the conclusion that can be drawn is that the most important factors to obtain high quality embeddings, in relation to the SR update task, is to fine-tune with SRs data and using global attention. Surprisingly, the ability to process the entire input sequence did not appear to be relevant. Certainly, LongFormer with global and local masks is much more computational expensive compared to the smaller SciBERT, although the latter obtained comparable performance.

5.5 Final Remarks

This chapter discussed the evaluation results of *ContReviews*, based on two large datasets obtained from 6000+ Cochrane Reviews. The first dataset (named CR-5) comprises all the Cochrane Reviews with at least 5 publications included. The other one (named CR-40) uses Cochrane Reviews with at least 40 publications included.

ContReviews was evaluated over both datasets, using all the available publication representations, i.e.: embeddings and bag of words of title, embeddings and bag of words of abstracts, binary representations of authors, binary representations of citations. *ContReviews* reported precision above 97% and near-perfect recall with both datasets.

These performance metrics were compared to two baseline models: one using semantic similarity between publications and SRs embeddings of their abstracts, and the other one using a SR-specific binary classification model based on the embeddings of abstracts of the included publications. For the same level of recall (i.e., above 97%), *ContReviews* reported much better precision.

Finally, ablation studies were also reported: (i) re-scaling of likelihoods of relevance provides the greatest improvement over precision; (ii) representing

publications by means of multiple features of themselves provides a modest improvement over precision; (iii) the best contribution to embeddings is given by fine-tuning over the *living evidence* data, while using more sophisticated fine-tuning tasks and encoding longer input sequences does not provide any benefits.

The next chapter provides some details about the system architecture and implementation.

Chapter 6

ContReviews Implementation and Architecture

ContReviews is implemented in Python and provided as a Python package for simple installation and usage. Unit tests are provided to simplify maintenance and change management. Integration tests are also provided. The *ContReviews* package can be run on a laptop computer for development and test purposes, provided the size of data is reasonable. For example, integration tests use 5 SRs with about 50 publications included and a full run takes some minutes. Instead, as argued below, running with full sized *living evidences* requires parallel processing over cloud resources and can take many hours.

The *ContReviews* package provides several classes, the most important ones are the *Encoder* class and the *VEM* class (Vectors Ensemble Model). The former provides the publication vector representations, which are based on bag of words, embeddings and binary vectors; the latter provides the *grounding of a SR*, which comprises the SR's vector representations and the supporting data needed to calculate the likelihoods of relevance (i.e., the cosine similarities for the SR's positive publications and for a sample of negative ones).

Input data consists of publication records, which comprise the publication ID, the SR ID, the title, the abstract, a list of comma-separated author IDs, a list of comma-separated citation IDs and the date of publication. These publication records are extracted from OpenAlex and fed to the *ContReviews* classes as a TSV file. Note that this file can be constructed in any way, although the use of academic knowledge graphs was motivated in Section 5.1.1.

The *ContReviews* package leverages several typical Python packages and frameworks; among the others, the following are particularly relevant: the *Scikit Learn* package[1], to construct publication vector representations based on entities and bag of words; the Hugging Face *transformers* package[2], to work with different pre-trained and fine-tuned embedding models; the Pandas package[3], to construct the various involved datasets for training and testing.

6.1 Operating *ContReviews*

Running *ContReviews* over a *living evidence* requires to training the system once and run it for inference every time new publications are provided by the academic knowledge graph. Sometimes, tests must also be conducted as described in Chapter 5.

To train *ContReviews* over a *living evidence*, the following operations are necessary:

- match the SRs in the *living evidence* to OpenAlex to obtain, for each SR, the disambiguated publications;
- ground the SRs in the *living evidence* by means of the VEM class, as introduced above;
- generate features for the relevance assessment function model (i.e., likelihoods of relevance);
- train the relevance assessment function model.

As discussed in more details in Section 6.3, running *ContReviews* over large *living evidences* require cloud resources to support parallel processing. To this aim, the Azure Machine Learning[4] platform has been used. This cloud platform provides (among the others) cloud storage, cloud compute resources (i.e., CPUs, GPUs and clusters), convenient abstraction layers and tools to run data science software in a reliable, repeatable and observable manner.

[1] https://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

[2] https://huggingface.co/docs/transformers/model_doc/auto

[3] https://pandas.pydata.org/

[4] https://learn.microsoft.com/en-us/azure/machine-learning

To use *ContReviews* for inference, the following operations are necessary:

- collect the new (disambiguated) publications from OpenAlex;
- perform citation screening for each SR in the *living evidence* to obtain a reduced list of new candidate publications for each SR;
- obtain likelihoods of relevance, for each SR and its candidate publications;
- predict relevance using the relevance assessment function model.

The citation screening step is required to reduce the amount of new publications passed to the content-based recommendation model. In fact, the academic knowledge graph provides domain-independent new publications which, in large part, are completely irrelevant to the entire *living evidence*. This first layer of screening, does not actually require the complexity of the full *ContReviews* system and the computation of likelihoods of relevance for multiple publication features. To this aim, each new publication is scored in two steps for relevance to SRs: (i) first, *citation screening* is applied through the 'single property model' based on bag of words of title and abstract (see Section 5.4.1), (ii) then, *abstract screening* is applied to the abstracts resulting from the previous step, through the *ContReviews* relevance assessment function model. Note that only the latter scoring operation requires computing all the likelihoods of relevance. On the other hand, the former scoring operation is much less computational expensive as it simply concerns calculating the likelihood of relevance for one single property and filter out publications by means of a convenient threshold.

The objective of citation screening is achieving recall of 100% without any specific requirement for precision. In fact, its aim is simply to discard all the completely and clearly irrelevant publications for each SR (which are the majority), leaving to the *ContReviews* content-based recommendation model the task to perform the actual recommendations.

A more interesting approach to citation screening would make use of a vector database: (i) the embeddings of all the new publication abstracts are stored and indexed into the vector database; (ii) SRs are used to query the vector database by means of their aggregated abstract embeddings, which is available as part of their grounding data (i.e., their associated VEM object); (iii) finally, the top k results are taken for each query. Different SRs would

benefits from a different value of k. In fact, the most recent and active SRs (for example, those relevant for 'the Covid') would benefit from an higher value compared to the oldest and slower ones which might have very few relevant studies in a whole year. A good reason of adopting this approach would be to permanently store publication abstract embeddings for multiple and different purposes. For example, older publications could be searched anytime to start new SRs, clusters of publications could be compared to existing SRs to monitor their alignment, clusters of publications could be used to monitor topics and their evolution over time. More generally speaking, organizations maintaining *living evidences* are actually information driven organizations and their ability to conduct analysis on their information assets (i.e., the scientific publications) is important.

6.2 Reproducibility

The above operating workflows to train *ContReviews* and perform inference are run within reproducible **pipelines**, as shown in Figures 6.1 and Figure 6.2. A pipeline makes it easy to monitor execution, to scale it over compute clusters, to re-run pipelines with different parameters and to log results. Using pipelines is an important ability within the 'ML-Ops'[5] strategy. 'ML-Ops' platforms aim to achieve **reproducibility of results** in several manners, one of which is being able to re-run pipelines multiple times with reproducible results and reliable execution with changing parameters. For example, when a new version of a *living evidence* is available, for example because some SRs have been updated, the pipeline to train *ContReviews* is re-run with the new input data and it will produce new outputs. Re-running a pipeline is assured to run without any accidental modification or side effect compared to the previous runs. Another example is performing evaluation, when the system is run multiple times with different model parameters: pipelines allows to quickly set new parameters and run the system in a reliable manner.

In addition, pipelines log results within a MLFlow[6] database, as shown in Figure 6.3 and Figure 6.4. This is particularly useful for **observability**, meaning that results of pipelines must be discoverable with ease, for the

[5] 'ML-Ops' is a shortcut for 'machine learning operations', which is conceptually derived from the acronym 'Dev-Ops', i.e., 'development operations'.

[6] https://mlflow.org/

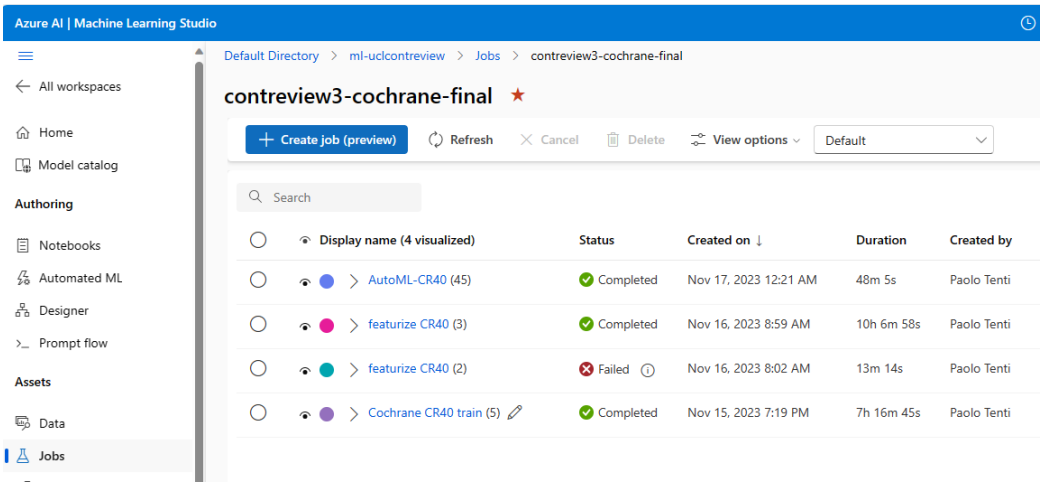

Figure 6.1: A list of executed pipeline jobs to support the lifecycle of the *ContReviews* system.

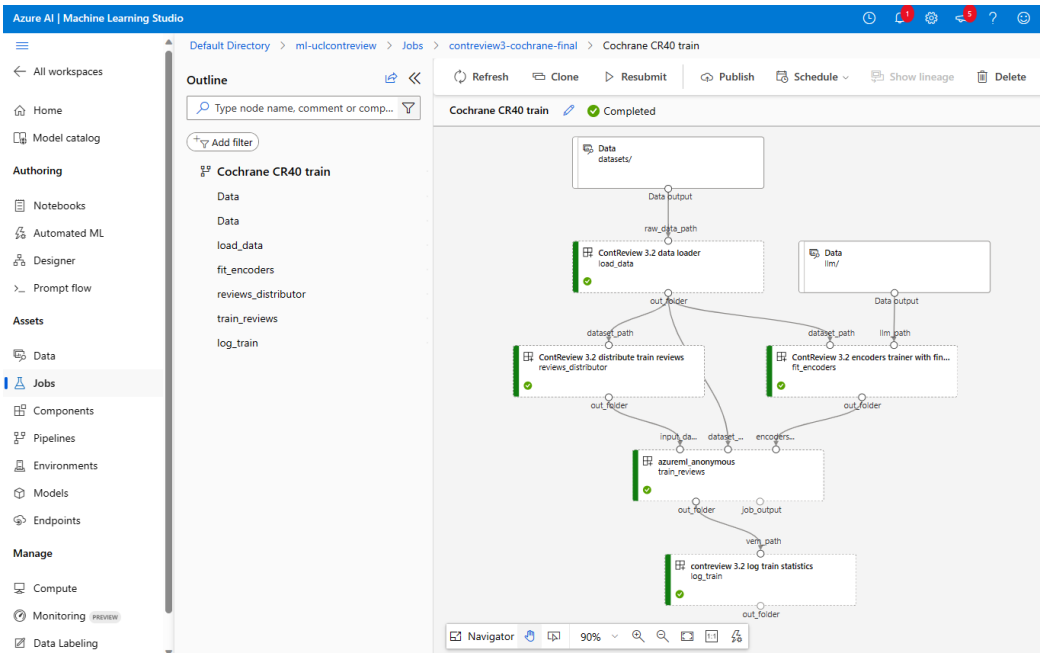

Figure 6.2: A pipeline workflow for instantiating the *ContReviews* system.

purpose of consulting older evaluations, compare them with newer evaluations or simply share them. Observability provides an important aspect of

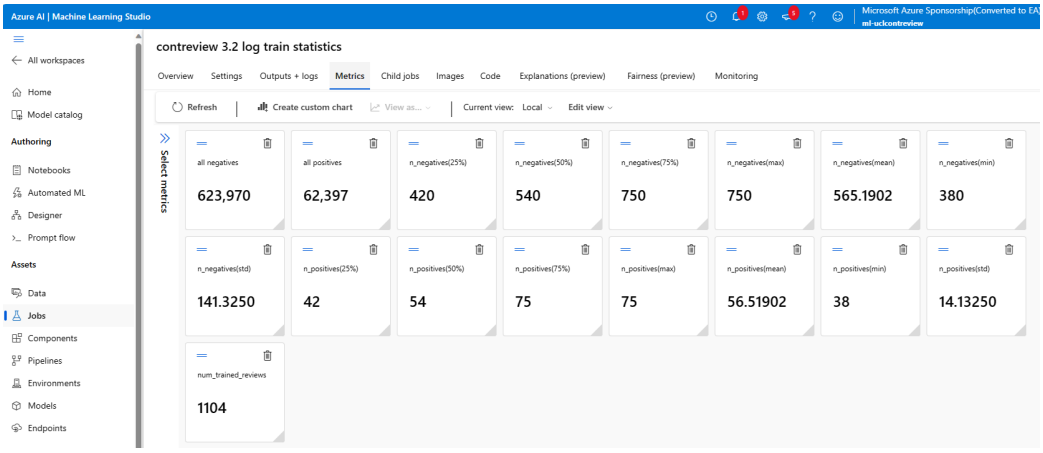

Figure 6.3: Pipeline stages report their metrics, for reproducibility and observability. Metrics are stored in a query-able MLFlow database for discoverability.

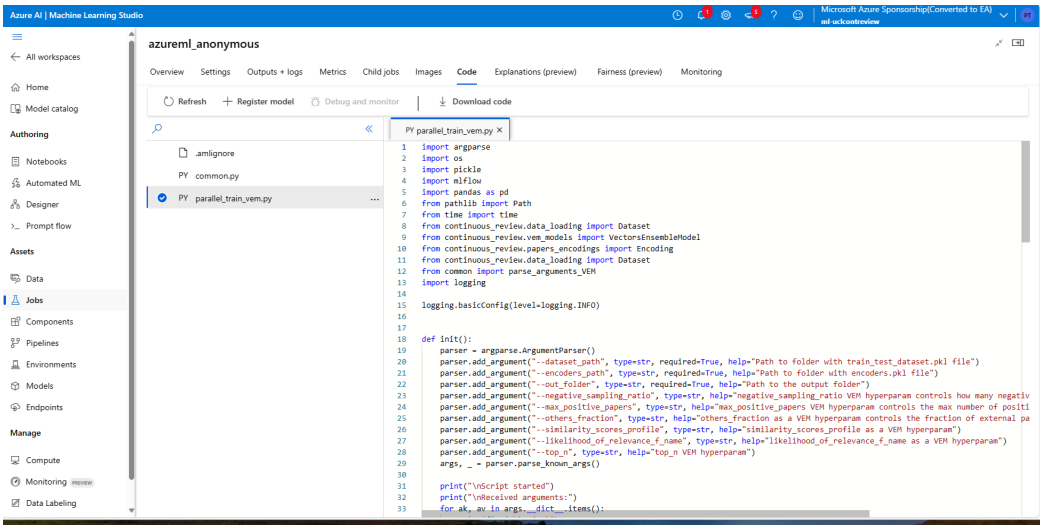

Figure 6.4: Pipeline stages track the code they had executed, for reproducibility and observability.

reproducibility of results.

6.3 Scalability and Computational Efficiency

Running *ContReviews* is resource intensive, especially with large *living evidences*. The 'VEM class' participates to all the *ContReviews* phases: it grounds SRs and their supporting data; it constructs the likelihoods of relevance to train and test a relevance assessment function model; and it computes the likelihoods of relevance for the new publications to run inference. In all of these cases, for each pair of a publication and a SR, a VEM object performs some core operations, which consist in constructing publication vector representations, computing cosine similarities between them and re-scaling these cosine similarities to compute likelihoods of relevance. This core computational engine needs to be scalable over the SRs in a *living evidences*, which can comprise tens of thousands of them.

For example, grounding SRs in the evaluation experiments illustrated in Chapter 5 over the CR-40 dataset, the core operations are performed for about 6000 SRs, up to 825 publications per SR (positive and negatives) and 6 vector representations. This leads to about 30M sets of core operations, which produce about 30GB as an aggregated memory footprint (i.e., about 5MB per SR). Table 6.1 reports some statistics about the execution time and the computing infrastructure. Note that publication vector representations could be reused multiple times, i.e., at least once for positive publications and several times as negative publications. Thus, a caching strategy is of great help to improve computational efficiency; however, caching is hindered by the large memory footprint, which could cause run time errors if not managed properly.

A VEM object grounds one single SR, thus, different SRs can be run in parallel to scale out *ContReviews* operations. In addition, each SR operates independently on all the others, that is, there is no need for one SR to access to other SRs' grounding data. In terms of parallel processing this means that there is one thread for each SR and threads do not need to synchronize with each other to exchange grounding data. Thus, *ContReviews* scales linearly with the number of SRs. Provided enough computational resources are available, *ContReviews* latency for grounding, training, testing or inference operations is the one of its largest SR. However, new publications are provided by academic knowledge graphs on a weekly or bi-weekly basis, thus, it is debatable how much it is worth using extremely large clusters to finish a computation quickly.

Some additional efforts might lead to better computational efficiency.

Size of *living evidence*	
Num. SRs	6395
Num. included publications in total	169790
Num. included publications per SR (mean)	25
Grounding	
Num. positive grounding publications per SR (mean)	22
Num. negative grounding publications per SR (mean)	220
Fitting vectorizers (exec. time)	52 min
Grounding SRs (exec. time)	2h 50 min
Relevance assessment function model	
Computing likelihoods of relevance w/ embeddings (exec. time)	4h 16 min
Training (exec. time)	36 min
Testing	
Num. positive test publications per SR (mean)	2
Num. negative test publications per SR (mean)	50
Computing likelihoods of relevance w/ embeddings (exec. time)	4h 44 min
Computing likelihoods of relevance no embeddings (exec. time)	1h 42 min

Table 6.1: Execution time and publication statistics for the evaluation experiments reported in Chapter 5 over the dataset named CR-5. A 8-nodes cluster was used, each node being a virtual machine with 16 CPU cores and 32 GB RAM. Training the relevance assessment function model was done with an Auto-ML platform.

Specifically, vector representations of entities are large and highly sparse: for example, there are tens of thousands of authors and even more citations in the considered *living evidence*, and each publication has, say, about less than 10 authors and 100 citations. On the contrary, embeddings and bag of words based vectors are much denser and less sparse. Thus, adopting a solution for embedding authors and citations might help to improve computational efficiency. In addition, the publication vector representations are stored within Python lists and dictionaries, and loops are used to calculate the cosine similarities. A potentially more efficient solution, especially if paired with embedding entity vector representations, would be to store vector representations as *numpy arrays*[7] and use optimized matrix operations. However, beside computational efficiency, using embeddings might lead to loosing precious information, thus, they need extensive testing.

6.4 Embeddings

PyTorch[8], a machine learning framework to build and train neural networks, is used to implement the *ContReviews* embedding model. Moreover, the Hugging Face *transformers* library[9] is used as an abstraction to instantiate neural language models and tokenize language expressions. Hugging Face also provides a community hub[10] to share many pre-trained and fine-tuned language models, including SciBERT and LongFormer.

As discussed in Chapter 4, *ContReviews* considers the following embedding models:

- SciBERT [57] pre-trained;
- SciBERT fine-tuned for the abstract screening task over Cochrane Reviews;
- LongFormer [2] fine-tuned for the semantic similarity task over Cochrane Reviews.

Compared to the SciBERT-based models, the LongFormer-based model requires more resources: first, LongFormer can process longer input se-

[7] https://numpy.org/
[8] https://pytorch.org/
[9] https://huggingface.co/docs/transformers/index
[10] https://huggingface.co/models

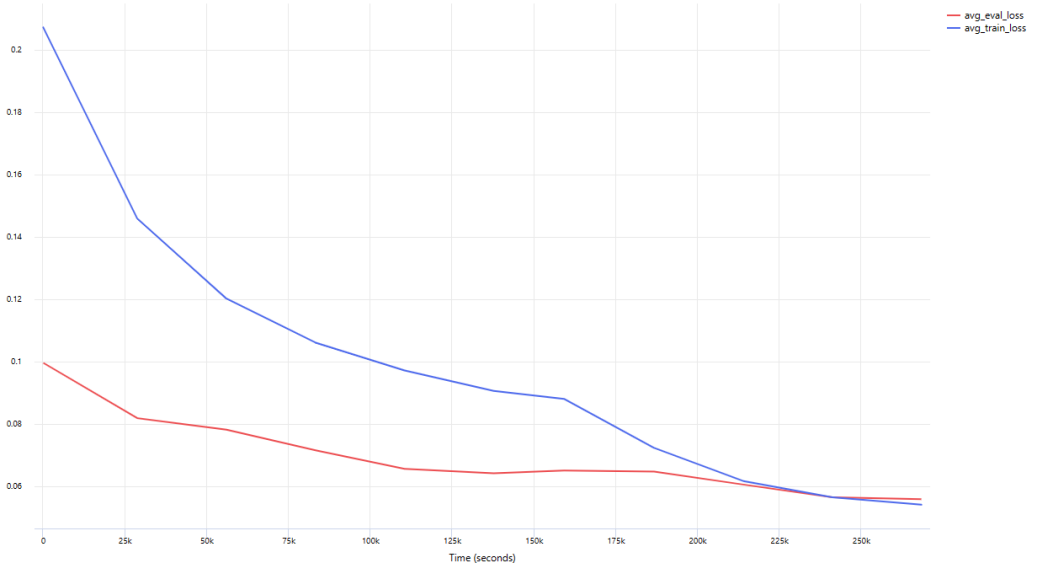

Figure 6.5: Training and evaluation loss for fine-tuning the LongFormer pre-trained language model for the semantic similarity task.

quences, which leads to an higher number of model parameters to train; second, each training case involves three runs of the LongFormer base model (i.e., as discussed in Section 4.2.2, the query abstract A^Q and the similar and dissimilar abstract instances A^+ and A^-), while the other models only need one run of SciBERT. Training the third model took two days for two epochs on a single GPU that had a NVIDIA Tesla V100 card—Figure 6.5 shows the training and evaluation loss. The SciBERT based models also required the same GPU based resources, but it required less training time, using the GPU smoothly. Without an adequate training strategy, the LongFormer model can run out of the GPU's memory due to its higher number of trainable parameters and the need to encode three input sequences instead of one.

Training a neural network is made of three main steps: (i) the *forward step* applies the neural network to a mini-batch of input sequences, produces a loss value and calculates the gradients of the loss value for each trainable parameter in the network; (ii) the *back-propagation step* allocates the loss value to the network's trainable parameters, based on the gradients; (iii) the *optimization step* fine-tune the network's trainable parameters proportionally to their gradients and their loss values. Particularly, the forward step

allocates a large data structure of gradients in the GPU's memory (i.e., about 2GB per abstract was observed with LongFormer, leading to about 6GB for each element in the input mini-batch), which is then released by the back-propagation step. To avoid running out of the GPU memory, the following training strategies have been implemented (see Figure 6.6):

- reducing the size of mini-batches, to reduce the amount of memory required to store gradients;
- using gradient accumulation, to accumulate a few mini-batches before optimization;
- freezing some layers on the network, to reduce the number of trainable parameters;

Note that using, say, 4 mini-batches with 8 training cases is equivalent to using one single mini-batch with 32 training cases. However, the former uses a quarter of the GPU's memory, thought, it is less effective in terms of optimization of parameters.

Overall, fine-tuning the SciBERT based model was faster, easier and allows to fine-tune the full base-model. Instead, LongFormer's complexity only allowed to fine-tune the last three attention layers and the pooling layer. Certainly, more sophisticated strategies to train the LongFormer based model could be considered, such as parallel training; however it would lead to additional complexity and it would require more GPUs. As argued in Section 5.4.4, in the specific context of *living evidences*, the LongFormer based model did not clearly outperform the SciBERT based model.

```
train_loss_values, eval_loss_values = [], []
for epoch in range(max_epochs):
    # Train loop
    model.train()
    train_loss = 0
    for step, batch in enumerate(train_dataloader):
        train_step = step+1
        loss = model(**batch)
        train_loss += loss.item()
        loss = loss / num_accumulation_steps
        loss.backward()
        torch.nn.utils.clip_grad_norm_(model.parameters(), 1.0)

        if train_step%num_accumulation_steps==0 \
                or train_step==train_epoch_size \
                or train_step%train_steps_before_eval==0:
            optimizer.step()
            lr_scheduler.step()
            optimizer.zero_grad()

        if train_step%train_steps_before_eval==0 or train_step==train_epoch_size:
            # Eval loop
            model.eval()
            eval_loss = 0
            for eval_step, batch in enumerate(eval_dataloader):
                with torch.no_grad():
                    loss = model(**batch)
                eval_loss += loss.item()
                if (eval_step+1) == eval_epoch_size:
                    break
```

Figure 6.6: Training and evaluation loop for fine-tuning LongFormer

Chapter 7

Conclusions

Accordingly to Cochrane,[1] "Systematic Reviews (SRs) attempt to identify, appraise and synthesize all the empirical evidences that meet pre-specified eligibility criteria to answer a specific research question. Researchers conducting SRs use explicit, systematic methods that are selected with a view aimed at minimizing bias, to produce more reliable findings to inform decision making". SRs use rigorous methods to find, assess, and synthesize studies that meet certain quality standards. They also check for potential sources of bias that could affect the results or conclusions of the review.

One critical challenge for SRs is their currency. SRs should be regularly updated to incorporate the new studies and reflect the current state of knowledge. To this aim, SR owners periodically drive a project to update their SRs. However, updating a SR requires almost the same level of complexity as creating it from scratch.

For this reason, *living evidences* have been recently proposed as a method to manage large collections of SRs efficiently, offering up to date and rigorous evidence syntheses to reviewers, researchers, practitioners and policy makers. Maintaining *living evidences* up to date with the huge amount of new publications released daily is a big enterprise. This work introduces **ContReviews**, a system to address the problem of keeping a *living evidence* current by updating its SRs as soon as new publications are available.

The *living evidence* process is based on the following steps:

- new research studies are continuously identified, by surveying the appropriate data sources;

[1] https://www.cochranelibrary.com/about/about-cochrane-reviews

- the research studies identified in the previous step are assessed for relevance to each SR in the *living evidence*;
- reviewers are continuously notified about the new research studies which might be interesting to their SRs;
- the final decision to include a new study in a SR is taken by the SR's reviewers in a manual process.

This work focuses on the health care domain: a Cochrane Review is a SR of research in health care and health policy, useful for those providing or receiving care, policy-makers, and researchers who want to make informed decisions based on reliable evidence. Cochrane Reviews are published in the Cochrane Database of Systematic Reviews, which is a *living evidence* that keeps Cochrane Reviews up to date with the latest studies as soon as they become available. Although *ContReviews* was evaluated on a large collection of Cochrane Reviews, it is not restricted to them and it is a general-purpose tool that can be used in different domains and fields of study.

7.1 Main Research Questions

The challenges related to applying the traditional SR updating methods to *living evidences* are introduced in Section 1.3. These traditional methods are specific to the SR they are meant to update, and they are not easily scalable across an entire domain of research. In fact, the typical activities coordinated by reviewers and domain experts consist on (i) identifying the most appropriate data sources, (ii) manually designing and testing sophisticated search queries, (iii) training SR-specific models for citation and abstract screening, and (iv) manually assess irrelevant studies, which are many due to poor efficiency of automated citation and abstract screening models. Clearly, a *living evidence* process, which aims to continuously recommend new publications to reviewers, cannot be that SR-specific and should run at a different level of automation.

ContReviews, the proposed system to address *living evidences*, leverages an academic knowledge graph to identify all the most recent publications, and a content-based recommendation model to match them to the available SRs. In addition, *ContReviews* leverages a method to formally represent publications and SRs, which take into consideration multiple publication

features (i.e., authors, citations network, embedding of abstracts, embedding of titles, bag of words of abstracts and bag of words of titles).

Following, the research questions introduced in Section 1.4 are reconsidered in light of the *ContReviews* system definition and the evaluation results.

7.1.1 Independence of the Relevance Assessment Model from SRs

The traditional SR updating methods are SR-specific, as introduced above, requiring the direct and active involvement of reviewers into the update process. This approach is not adequate to address the currency of entire *living evidences*, in fact, the expectation is that they run in an unattended manner, and provide frequent and precise recommendations to reviewers. *ContReviews* addresses this issue by means of a content-based recommendation model which, for each SR in the *living evidence*, leverages the content of the included publications to identify the new ones to recommend to reviewers. This approach let to avoid the reviewers involvement to manually designing and testing complex search queries.

ContReviews matches the new publications to SRs through their vector representations, and obtain a likelihood of relevance which is used to prioritize recommendations to reviewers. Specifically, the vector representations of SRs are obtained by averaging the vector representations of their included publications. The evaluation results show that this approach is effective, i.e., all the new publications which are relevant to a SR are correctly identified by the system. The effectiveness of the content-based recommendation model was measured through the classification metric of recall, which was in average greater than 97% over all the SRs in the test dataset.

In addition, an academic knowledge graph is proposed as a unified data source of new publications, to avoid the reviewers involvement to manually identify SR-specific data sources. This approach is convenient, because it provides new publications in a structured, normalized, de-duplicated, centralized and programmatically accessible manner. However, the proposed content-based recommendation model works independently of its data sources.

7.1.2 Efficiency of Inference

Updating SRs is a so called *total-recall task*, meaning that the inference system must be almost perfectly effective to avoid loosing any relevant new pub-

lication. To achieve such an high level of effectiveness, current systems often sacrifice efficiency, which leads to many (ultimately irrelevant) new publications requiring manual assessment. Poor efficiency forces reviewers to take an active part in the SR update process: while this is natural when conducting a specific project to update a SR, it would have challenging implications in a *living evidence*. In fact, reviewers would receive many frequent recommendations for mostly irrelevant new publications. In this context, the research question was whether the proposed method can achieve better efficiency than the traditional systems, while maintaining near-perfect effectiveness.

ContReviews was tested with a large set of 6000+ Cochrane Reviews, with a minimum of 3 publications included, and the system performance was measured in terms of the classification metrics of precision and recall. Specifically, precision with recall higher than 97% was measured, that is, efficiency with almost perfect effectiveness. Note that precision and recall were measured for each SR and their statistics (such as mean and standard deviation) reported. With such an evaluation setting, *ContReviews* achieved average precision of 98.1% with perfect recall of 100%, and the 95% of all the SRs achieved average precision greater than 87.5%.

As motivated in Section 2.3, a common dataset and evaluation method is not available for neither the *living evidence* task nor SR updating more in general. For this reason two baselines have been proposed to compare *ContReviews* to canonical methods. The first baseline model considers the cosine distance between the abstract-based vector representations of new publications and SRs, and use it to infer their relevance. The second baseline method is based on one binary classification model for each SR, which uses the embeddings of the included publication abstracts as features to learn a relevance assessment function for each SR. As the second baseline requires a minimum number of publications included in each SR to train the relevance assessment function model, only the SRs with at least 40 publications included were considered in the comparison (where 40 was empirically determined). The baseline models achieved no more than 22.7% precision with 98.1% recall, reinforcing that the proposed content-based recommendation model is efficient and effective.

7.1.3 Representation of Publications and SRs

Most existing methods for determining how relevant new publications are to SRs rely on the textual properties of the publications, such as their titles and

abstracts. Some recent works use embeddings to represent these properties. However, to make the inference process more efficient, it may be helpful to use more precise representations of the publications. This section focuses on the research questions of whether using multiple publication features and more sophisticated embedding models can lead to better efficiency.

ContReviews uses multiple publication properties—such as title, abstract, citations network and authors—to construct publication vector representations. To assess this approach, the accuracy of *ContReviews* to infer publications relevance to SRs was evaluated in two settings: the first one uses all the available features (title, abstract, citations network and authors), while the second one uses only the textual ones (i.e., title and abstract). Using multiple features contributes to increase both average effectiveness (i.e., from 98.1% to 100%) and average efficiency (i.e., from 96.4% to 97.4%).

The embedding model used to represent publication textual features, is another component that can affect efficiency. Several design factors can distinguish different embedding models; this research focuses on the base model (i.e., SciBERT and LongFormer), the fine-tuning dataset (i.e., Cochrane Reviews), and the fine-tuning task (i.e., semantic similarity and abstract screening).

Note that the majority of the most recent works use variants of BERT, such as SciBERT, to obtain embeddings of abstracts. In fact, these embedding models are pre-trained on scientific or medical data, which is usually relatively close to the type of publications included in the SRs to update. This research, by fine-tuning over Cochrane Reviews, aims to better align the embedding model to the *living evidence* distributional semantics. In addition, base models support different sizes for the input sequence; for example, the SciBERT's one is about 4x shorter than the average length of the abstracts in the Cochrane Reviews. Instead, LongFormer supports a longer maximum input sequence length, which is aligned to the average length of abstracts in the Cochrane Reviews. Finally, the fine-tuning task lets the embedding model learning language nuances more precisely, which can be important to the downstream task. Both abstract screening and semantic similarity are more aligned to the task of SR updating than the original pre-training ones (i.e., next sentence prediction and masked language modeling).

Based on the evaluation reported in Subsection 5.4.4, the following conclusions can be drawn: the crucial factor is to fine-tune embeddings on the target *living evidence* data (i.e., abstracts from Cochrane Reviews in this research), while the fine-tuning tasks and the input sequence length are not

very significant factors. This implies that the most computationally efficient embedding models should be used for fine-tuning, together with the *living evidence* data. Note that a *living evidence* provides sufficient relationships between publications and SRs to let easily construct meaningful and large fine-tuning datasets, and that this is not trivial when working with individual SRs.

7.2 Future work

This research demonstrates that a content-based recommendation model can successfully infer the relevance of new publications to SRs in a *living evidence*, instead of using a separate inference model for each SR. The following are open research topics.

Improving computational efficiency. *Living evidences* are computationally intensive, because they comprise thousands of SRs, and because of the large number of new studies which are published every day. Therefore, improving the computational efficiency of the current model is an important goal. To this aim, several approaches could be considered.

In the one hand, a lightweight model could be used prior to the content-based recommendation model, aiming to filter out the clearly irrelevant publications for each SR. It is not clear what this new model should look like, though it should be computationally very efficient, and it should achieve perfect recall regardless of the precision.

In the other hand, the current content-based recommendation model uses highly sparse vectors. For example, a *living evidence* can have tens of thousands of unique authors, yielding extremely sparse representations (in fact, publications have only a few authors). To achieve more dense representations, embeddings of entities (i.e., authors and citations) should be evaluated. However, this approach might increase the overall system complexity.

Improving contribution of entities. Using multiple publication features to represent the SRs and the publications themselves provided a positive contribution in terms of inference efficiency and effectiveness. However, the contribution of entities (such as authors and citations) was not as huge as expected, with the textual features steering the performance of inference. One

possible reason is that their binary representations are quite raw: for example, two publications with an indirect connection in their citations network are considered completely dissimilar through their binary representations, as much as two publications without any connection in their citations network. A similar consideration holds for authors. Exploring more sophisticated techniques to represent publications based on their relationships might help to improve this aspect.

Covering *living evidences* in different domains. This research focus on the Cochrane Reviews, which are significant in the field of life sciences. In principle, the proposed model could be applied to any*living evidences* in any domain, however, these *living evidences* may greatly vary from those in life sciences. For example, a preliminary analysis of a *living evidence* in the education domain, revealed much fewer SRs with many more publications included and, therefore, the publication texts, authors and citations were less selective than those in the Cochrane Reviews. A more thorough evaluation of the *ContReviews* model is needed across different domains, and the most relevant publication properties for each domain should be determined.

Bibliography

[1] Jason Priem, Heather Piwowar, and Richard Orr. Openalex: A fully-open index of scholarly works, authors, venues, institutions, and concepts, 2022.

[2] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[3] Rory J Piper. How to write a systematic literature review: a guide for medical students. *National AMR, fostering medical research*, 1:1–8, 2013.

[4] JPT Higgins, J Thomas, J Chandler, M Cumpston, T Li, MJ Page, and VA Welch, editors. *Cochrane Handbook for Systematic Reviews of Interventions version 6.3 (updated February 2022)*. John Wiley & Sons, 2022. Chapter 1: Starting a review. https://training.cochrane.org/handbook/current/chapter-01#section-1-1.

[5] Andy P Siddaway, Alex M Wood, and Larry V Hedges. How to do a systematic review: a best practice guide for conducting and reporting narrative reviews, meta-analyses, and meta-syntheses. *Annual review of psychology*, 70:747–770, 2019.

[6] Guillaume Lame. Systematic literature reviews: An introduction. In *Proceedings of the design society: international conference on engineering design*, volume 1, pages 1633–1642. Cambridge University Press, 2019.

[7] Julian Elliott, Rebecca Lawrence, Jan C Minx, Olufemi T Oladapo, Philippe Ravaud, Britta Tendal Jeppesen, James Thomas, Tari Turner,

Per Olav Vandvik, and Jeremy M Grimshaw. Decision makers need constantly updated evidence synthesis, 2021.

[8] Steve McDonald, Simon Turner, Matthew J Page, and Tari Turner. Most published systematic reviews of remdesivir for covid-19 were redundant and lacked currency. *Journal of clinical epidemiology*, 146:22–31, 2022.

[9] Kaveh G Shojania, Margaret Sampson, Mohammed T Ansari, Jun Ji, Steve Doucette, and David Moher. How quickly do systematic reviews go out of date? a survival analysis. *Annals of internal medicine*, 147(4):224–233, 2007.

[10] Tari Turner, Julian Elliott, Britta Jeppesen, Joshua Vogel, Sarah Norris, Rhiannon Tate, and Sally Green. The australian living guidelines for the clinical care of people with covid-19: What worked, what didn't and why, a mixed methods process evaluation. *PLOS ONE*, 17:e0261479, 01 2022.

[11] Hilda Bastian, Paul Glasziou, and Iain Chalmers. Seventy-five trials and eleven systematic reviews a day: how will we ever keep up? *PLoS medicine*, 7(9):e1000326, 2010.

[12] Wojciech Kusa, Oscar E. Mendoza, Matthias Samwald, Petr Knoth, and Allan Hanbury. CSMed: Bridging the dataset gap in automated citation screening for systematic literature reviews. In *Thirty-seventh Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2023.

[13] Paul Garner, Sally Hopewell, Jackie Chandler, Harriet MacLehose, Elie A Akl, Joseph Beyene, Stephanie Chang, Rachel Churchill, Karin Dearness, Gordon Guyatt, Carol Lefebvre, Beth Liles, Rachel Marshall, Laura Martínez García, Chris Mavergames, Mona Nasser, Amir Qaseem, Margaret Sampson, Karla Soares-Weiser, Yemisi Takwoingi, Lehana Thabane, Marialena Trivella, Peter Tugwell, Emma Welsh, Ed C Wilson, and Holger J Schünemann. When and how to update systematic reviews: consensus and checklist. *BMJ*, 354, 2016.

[14] Iain Marshall and Byron Wallace. Toward systematic review automation: a practical guide to using machine learning tools in research synthesis. *Systematic Reviews*, 8, 12 2019.

[15] Shuai Wang, Harrisen Scells, Martin Potthast, Bevan Koopman, and Guido Zuccon. Generating natural language queries for more effective systematic review screening prioritisation. *arXiv preprint arXiv:2309.05238*, 2023.

[16] Julian H. Elliott, Tari Turner, Ornella Clavisi, James Thomas, Julian P. T. Higgins, Chris Mavergames, and Russell Lindsay Gruen. Living systematic reviews: An emerging opportunity to narrow the evidence-practice gap. *PLoS Medicine*, 11, 2014.

[17] James Thomas, Anna Noel-Storr, Iain Marshall, Byron Wallace, Steven McDonald, Chris Mavergames, Paul Glasziou, Ian Shemilt, Anneliese Synnot, Tari Turner, et al. Living systematic reviews: 2. combining human and machine effort. *Journal of clinical epidemiology*, 91:31–37, 2017.

[18] Susan Michie, James Thomas, Marie Johnston, Pol Mac Aonghusa, John Shawe-Taylor, Michael P. Kelly, Léa A. Deleris, Ailbhe N. Finnerty, Marta M. Marques, Emma Norris, Alison O'Mara-Eves, and Robert West. The human behaviour-change project: harnessing the power of artificial intelligence and machine learning for evidence synthesis and interpretation. *Implementation Science*, 12(121), 2017.

[19] Iain J Marshall, Thomas A Trikalinos, Frank Soboczenski, Gregory Kell Hye Sun Yun, Rachel Marshall, and Byron C Wallace. Systematic review updates in near real-time.

[20] Iain J Marshall, Benjamin Nye, Joël Kuiper, Anna Noel-Storr, Rachel Marshall, Rory Maclean, Frank Soboczenski, Ani Nenkova, James Thomas, and Byron C Wallace. Trialstreamer: A living, automatically updated database of clinical trial reports. *Journal of the American Medical Informatics Association*, 27(12):1903–1912, 2020.

[21] Mehrdad Amir-Behghadami and Ali Janati. Population, intervention, comparison, outcomes and study (picos) design as a framework to formulate eligibility criteria in systematic reviews. *Emergency Medicine Journal*, 2020.

[22] Edoardo Aromataris and Dagmara Riitano. Constructing a search strategy and searching for evidence. *Am J Nurs*, 114(5):49–56, 2014.

[23] Shuai Wang, Hang Li, and Guido Zuccon. Mesh suggester: A library and system for mesh term suggestion for systematic review boolean query construction. In *Proceedings of the Sixteenth ACM International Conference on Web Search and Data Mining*, pages 1176–1179, 2023.

[24] Harrisen Scells, Guido Zuccon, and Bevan Koopman. A comparison of automatic boolean query formulation for systematic reviews. *Information Retrieval Journal*, 24:3–28, 2021.

[25] Angelo D'Ambrosio, Hajo Grundmann, and Tjibbe Donker. An open-source integrated framework for the automation of citation collection and screening in systematic reviews. *arXiv preprint arXiv:2202.10033*, 2022.

[26] Shuai Wang, Harrisen Scells, Bevan Koopman, and Guido Zuccon. Can chatgpt write a good boolean query for systematic review literature search? *arXiv preprint arXiv:2302.03495*, 2023.

[27] Shuai Wang, Harrisen Scells, Ahmed Mourad, and Guido Zuccon. Seed-driven document ranking for systematic reviews: A reproducibility study. In *European Conference on Information Retrieval*, pages 686–700. Springer, 2022.

[28] Arnab Sinha, Zhihong Shen, Yang Song, Hao Ma, Darrin Eide, Bo-June (Paul) Hsu, and Kuansan Wang. An overview of microsoft academic service (mas) and applications. In *Proceedings of the 24th International Conference on World Wide Web*, pages 243–246, 2015.

[29] Ian Shemilt, Anneliese Arno, James Thomas, Theo Lorenc, Claire Khouja, Gary Raine, Katy Sutcliffe, Irene Kwan, Kath Wright, Amanda Sowden, et al. Cost-effectiveness of microsoft academic graph with machine learning for automated study identification in a living map of coronavirus disease 2019 (covid-19) research. *Wellcome Open Research*, 6(210):210, 2021.

[30] Grace E Lee and Aixin Sun. Seed-driven document ranking for systematic reviews in evidence-based medicine. In *The 41st international ACM SIGIR conference on research & development in information retrieval*, pages 455–464, 2018.

[31] Shuai Wang, Harrisen Scells, Bevan Koopman, and Guido Zuccon. Neural rankers for effective screening prioritisation in medical systematic review literature search. In *Proceedings of the 26th Australasian Document Computing Symposium*, pages 1–10, 2022.

[32] Amal Alharbi and Mark Stevenson. Ranking studies for systematic reviews using query adaptation: University of sheffield's approach to clef ehealth 2019 task 2. In *Clef (working notes)*, 2019.

[33] Harrisen Scells, Guido Zuccon, and Bevan Koopman. You can teach an old dog new tricks: Rank fusion applied to coordination level matching for ranking in systematic reviews. In *Advances in Information Retrieval: 42nd European Conference on IR Research, ECIR 2020, Lisbon, Portugal, April 14–17, 2020, Proceedings, Part I 42*, pages 399–414. Springer, 2020.

[34] Eugene Yang, Sean MacAvaney, David D Lewis, and Ophir Frieder. Goldilocks: Just-right tuning of bert for technology-assisted review. In *European Conference on Information Retrieval*, pages 502–517. Springer, 2022.

[35] Edward Fox and Joseph Shaw. Combination of multiple searches. *NIST special publication SP*, pages 243–243, 1994.

[36] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina N. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2018.

[37] Özge Kart, Alexandre Mestiashvili, Kurt Lachmann, Richard Kwasnicki, and Michael Schroeder. Emati: a recommender system for biomedical literature based on supervised learning. *Database*, 2022, 12 2022. baac104.

[38] Xuan Qin, Jiali Liu, Yuning Wang, Yanmei Liu, Ke Deng, Yu Ma, Kang Zou, Ling Li, and Xin Sun. Natural language processing was effective in assisting rapid title and abstract screening when updating systematic reviews. *Journal of Clinical Epidemiology*, 133:121–129, 2021.

[39] Sara Perlman-Arrow, Noel Loo, Niklas Bobrovitz, Tingting Yan, and Rahul K. Arora. A real-world evaluation of the implementation of nlp technology in abstract screening of a systematic review. *medRxiv*, 2022.

[40] A. Bannach-Brown, P. Przybyła, J. Thomas, A. Rice, S. Ananiadou, J. Liao, and Macleod M.R. Machine learning algorithms for systematic review: reducing workload in a preclinical review of animal studies and reducing human screening error. *Systematic Reviews*, 2019.

[41] Wojciech Kusa, Allan Hanbury, and Petr Knoth. Automation of citation screening for systematic literature reviews using neural networks: A replicability study. In Matthias Hagen, Suzan Verberne, Craig Macdonald, Christin Seifert, Krisztian Balog, Kjetil Nørvåg, and Vinay Setty, editors, *Advances in Information Retrieval*, pages 584–598, Cham, 2022. Springer International Publishing.

[42] Georgios Kontonatsios, Sally Spencer, Peter Matthew, and Ioannis Korkontzelos. Using a neural network-based feature extraction method to facilitate citation screening for systematic reviews. *Expert Systems with Applications: X*, 6:100030, 2020.

[43] Raymon van Dinter, Cagatay Catal, and Bedir Tekinerdogan. A multi-channel convolutional neural network approach to automate the citation screening process. *Applied Soft Computing*, 112:107765, 2021.

[44] Georgios Kontonatsios, Sally Spencer, Peter Matthew, and Ioannis Korkontzelos. Using a neural network-based feature extraction method to facilitate citation screening for systematic reviews. *Expert Systems with Applications: X*, 6:100030, 2020.

[45] A.M. Cohen, W.R. Hersh, K. Peterson, and Po-Yin Yen. Reducing workload in systematic review preparation using automated citation classification. *Journal of the American Medical Informatics Association*, 13(2):206–219, 2006.

[46] Wojciech Kusa, Aldo Lipani, Petr Knoth, and Allan Hanbury. An analysis of work saved over sampling in the evaluation of automated citation screening in systematic literature reviews. *Intelligent Systems with Applications*, 18:200193, 2023.

[47] Babatunde Kazeem Olorisade, Pearl Brereton, and Peter Andras. The use of bibliography enriched features for automatic citation screening. *Journal of biomedical informatics*, 94:103202, 2019.

[48] I Chalmers, M Enkin, and MJ Keirse. Preparing and updating systematic reviews of randomized controlled trials of health care. *The Milbank quarterly*, 71(3):411—437, 1993.

[49] Makoto Miwa, James Thomas, Alison O'Mara-Eves, and Sophia Ananiadou. Reducing systematic review workload through certainty-based screening. *Journal of Biomedical Informatics*, 51:242–253, 2014.

[50] David Cohn, Les Atlas, and Richard Ladner. Improving generalization with active learning. *Machine learning*, 15:201–221, 1994.

[51] Byron C Wallace, Thomas A Trikalinos, Joseph Lau, Carla Brodley, and Christopher H Schmid. Semi-automated screening of biomedical citations for systematic reviews. *BMC bioinformatics*, 11(1):1–11, 2010.

[52] Kazuma Hashimoto, Georgios Kontonatsios, Makoto Miwa, and Sophia Ananiadou. Topic detection using paragraph vectors to support active learning in systematic reviews. *Journal of Biomedical Informatics*, 62:59–65, 2016.

[53] Jason Portenoy and Jevin D West. Constructing and evaluating automated literature review systems. *Scientometrics*, 125(3):3233–3251, 2020.

[54] Christopher W Belter. Citation analysis as a literature search method for systematic reviews. *Journal of the Association for Information Science and Technology*, 67(11):2766–2777, 2016.

[55] Jingwen Hou, Xiaochen Wang, Jean-Jacques Dubois, R. Byron Rice, Amanda Haddock, and Yue Wang. Extreme systematic reviews: A large literature screening dataset to support environmental policymaking. *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, 2022.

[56] Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Content-based recommendation*, page 51–80. Cambridge University Press, 2010.

[57] Iz Beltagy, Kyle Lo, and Arman Cohan. Scibert: Pretrained language model for scientific text. In *EMNLP*, 2019.

[58] Gerard Salton and Michael McGill. Introduction to modern information retrieval. McGraw-Hill, 1983.

[59] Quoc Le and Tomas Mikolov. Distributed representations of sentences and documents. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 1188–1196, Bejing, China, 22–24 Jun 2014. PMLR.

[60] Danilo Dessi, Rim Helaoui, Vivek Kumar, Diego Reforgiato Recupero, and Daniele Riboni. Tf-idf vs word embeddings for morbidity identification in clinical notes: An initial study. *arXiv preprint arXiv:2105.09632*, 2021.

[61] Annalisa Wahyu Romadon, Kemas M Lhaksmana, Isman Kurniawan, and Donni Richasdy. Analyzing tf-idf and word embedding for implementing automation in job interview grading. In *2020 8th International Conference on Information and Communication Technology (ICoICT)*, pages 1–4. IEEE, 2020.

[62] Jakub Piskorski and Guillaume Jacquet. Tf-idf character n-grams versus word embedding-based models for fine-grained event classification: a preliminary study. In *Proceedings of the Workshop on Automated Extraction of Socio-political Events from News 2020*, pages 26–34, 2020.

[63] Samy Bengio and Georg Heigold. Word embeddings for speech recognition. In *Proceedings of the 15th Conference of the International Speech Communication Association, Interspeech*, 2014.

[64] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017.

[65] Jeffrey Pennington, Richard Socher, and Christopher Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*

(EMNLP), pages 1532–1543, Doha, Qatar, October 2014. Association for Computational Linguistics.

[66] Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In Marilyn Walker, Heng Ji, and Amanda Stent, editors, *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics.

[67] Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information, 2017.

[68] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding with unsupervised learning. 2018.

[69] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[70] Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. Domain-specific language model pretraining for biomedical natural language processing, 2020.

[71] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks, 2019.

[72] Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*, 2020.

[73] Guolin Ke, Qi Meng, Thomas Finely, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. Lightgbm: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems 30 (NIP 2017)*, December 2017.

[74] Jerome H. Friedman. Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29:1189–1232, 2001.