University of Sheffield

SciFCheX: Developing a Scientific Fact-Checker with Hybrid Evidence Retrieval Approaches

Filip J. Cierkosz

Supervisor: Mark Stevenson

A report submitted in fulfilment of the requirements for the degree of BSc in Computer Science

in the

Department of Computer Science

May 22, 2024

Declaration

All sentences or passages quoted in this report from other people's work have been specifically acknowledged by clear cross-referencing to author, work and page(s). Any illustrations that are not the work of the author of this report have been used with the explicit permission of the originator and are specifically acknowledged. I understand that failure to do this amounts to plagiarism and will be considered grounds for failure in this project and the degree examination as a whole.

Name: Filip J. Cierkosz

Signature: Filip J. Cierkosz

Date: May 22, 2024

Abstract

Recently, fact-checking has become more important as unverified information spreads rapidly online, which was particularly evident during COVID-19. The pandemic exposed new risks of misinformation in healthcare, and emphasised the necessity of replacing traditional verification approaches with robust automated solutions.

Automated fact-checking is primarily concerned with assessing the veracity of claims based on evidence, while its subset, *scientific fact-checking*, focuses on science-related statements. State-of-the-art approaches for these tasks typically utilise Transformer models, implementing a three-stage pipeline of: *evidence retrieval*, *rationale selection*, and *verdict prediction*.

The project introduces SciFCheX, a fact-checking system designed for scientific content. This work primarily focuses on *evidence retrieval*, proposing a novel *hybrid* architecture that combines BM25 with a fine-tuned BERT-based classifier. It achieves an impressive F1-score of 0.73 for retrieval on the SciFact dataset, significantly outperforming the baseline TF-IDF method (0.26). Additionally, the development of the pipeline's *rationale selection* and *verdict prediction* is approached by fine-tuning domain-specific BERT models, following literature-based approaches. SciFCheX participated in the SciFact task, ranking 48th out of over 150 submissions.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

The widespread access to the Internet has led the development of our society in countless ways, notably by making knowledge-related resources more available and instantly searchable. In the blink of an eye, individuals might find any type of information, ranging from scientific publications to trending news stories. While there are many positives to this phenomenon, several studies highlight it has introduced certain negative societal implications, primarily caused by an increasing number of unverified sources spreading potentially harmful claims (Del Vicario et al., 2016, West and Bergstrom, 2021). Vosoughi et al. (2018) points that these might spread even six times faster than true information. Furthermore, Del Vicario et al. (2016) notes that such claims often include false scientific news, or conspiracy theories, which increase polarisation in worldwide communities. The issue of misinformation is particularly significant during major political events, or global crises, which was evident in the time of COVID-19 (West and Bergstrom, 2021). The pandemic exposed new risks of biomedicine-related misinformation on social media (e.g., Facebook). Roozenbeek et al. (2020) and Pennycook et al. (2020) emphasise that the spread of unverified claims in the light of such a healthcare emergency might lead individuals to experimenting with unconfirmed treatment methods, resulting in detrimental health decisions. The aforementioned issues, combined with the rapid pace of scientific advancements and the growing presence of AI in the digital ecosystem, highlight the need for robust fact-checkers.

1.2 Motivation

The outlined concerns have prompted a bold response in journalism, with the establishment of over 400 fact-checking initiatives specialised in debunking online misinformation, as per the Duke Reporters' Lab[1]. Although these efforts are beneficial, the manual verification methods frequently employed by these organisations are becoming impractical, as they are too labour-intensive and inefficient to keep up with the latest technological advancements (Graves, 2017, Vladika and Matthes, 2023). On top of that, verifying claims in science is particularly difficult due to its jargon, which often requires domain-specific expertise. These issues have driven recent advancements in Natural Language Processing, particularly in *automated fact-checking*, which aims to assess the veracity of claims based on evidence. This progress has led to the development of new architectures for verification pipelines that utilise powerful Transformer models (Zeng et al., 2021, Vaswani et al., 2017). Concurrently, these research efforts have resulted in the formation of new scientific datasets (e.g., SciFact (Wadden et al., 2020), PubHealth (Kotonya and Toni, 2020*b*)), and shared tasks (such as SciVer (Wadden and Lo, 2021)).

1.3 Aims & Contributions

The project aims to present SciFCheX, a fact-checking pipeline tailored to the scientific domain, building upon recent NLP findings. The system is designed to address the SciFact task, and the best-performing model configuration has been submitted to the official leaderboard[2], ranking 48th out of over 150 worldwide submissions (Wadden and Lo, 2021). Notably, the study refers the work of Wadden et al. (2020), which introduced the SciFact dataset and its respective verification model – VeriSci. The dataset is used for training and evaluation, while the model serves as a baseline of the presented SciFCheX.

The pipeline for automated claim verification typically consists of three main stages: *evidence retrieval*, *rationale selection*, and *verdict prediction* (Zeng et al., 2021). The high-level purpose of such a system is illustrated in Figure 1.1. The step of *evidence retrieval*

[1] As of May 22, 2024. The report is available at: https://reporterslab.org/fact-checking/

[2] As of May 22, 2024, SciFCheX ranks 48th out of 154 worldwide submissions. In comparison, VeriSci ranks out of top 100. The official leaderboard can be found at: https://leaderboard.allenai.org/scifact/submissions/public

can be identified as the primary focus of the research. The task is addressed by a thorough evaluation of multiple retrieval strategies, contrasting *sparse* and *hybrid* approaches; the latter combine *sparse* and *dense* techniques (Zhao et al., 2024). Notably, the proposed *hybrid* approach of BM25 (Robertson and Zaragoza, 2009) with a domain-adapted BERT-based (Devlin et al., 2019) classifier has the potential to bring some performance improvements. Thus, this study seeks to answer:

To what extent can scientific fact-checkers benefit from hybrid retrieval methods?

Furthermore, the remaining stages of the final pipeline, i.e., *rationale selection* and *verdict prediction*, utilise a separate BERT (Devlin et al., 2019) model, fine-tuned following literature-based approaches.

In summary, the main contributions of this research include:

- Establishment of SciFCheX pipeline, featuring the innovative BERT-based *hybrid retrieval* strategy, that substantially outperforms the VeriSci baseline.
- Examination of *evidence retrieval* methods, ranging from *sparse* (e.g., TF-IDF (Singhal et al., 2001)) to advanced *hybrid* strategies. The latter include not only the aforementioned approach, but also a general-domain BGE-M3 reranker (Chen et al., 2024), that has not been reported to be evaluated in this domain.
- Thorough review of recent advancements in the field of scientific fact-checking.

1.4 Relation to Degree Program

This work relates to the concepts learned during the previously completed course of *Text Processing (COM3110)*, which provided valuable foundations for NLP concepts. In particular, the module covered topics that are relevant in the context of data pre-processing, or information retrieval. It also briefly introduced topics related to Large Language Models, which was an incentive for a further development of skills required for this project.

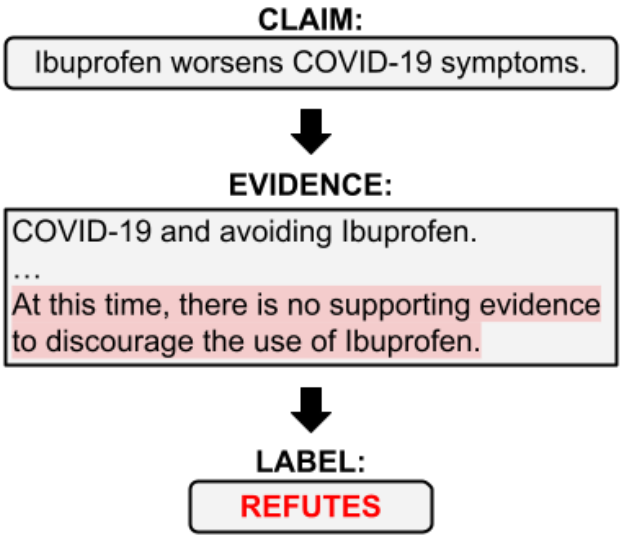

Figure 1.1: A figure illustrating a simple claim verification process for a COVID-related statement from the HealthVer dataset (Sarrouti et al., 2021). As it can be seen, the claim is refuted by deducing an appropriate rationale based on identified evidence.

1.5 Report Overview

Having introduced the main directions of the project, it is now appropriate to provide a brief overview of the report. Therefore, the remaining chapters of the paper cover:

- **Chapter 2 – Scientific Fact-checking Overview:** Defines scientific fact-checking, explores its state-of-the-art approaches, technology, datasets, and research directions.
- **Chapter 3 – SciFact Task Requirements:** Formulates the requirements of the task addressed in this project, discussing its objectives, dataset, and evaluation strategies.
- **Chapter 4 – Methods:** Details the project's methodology, also referring to BERT's architecture. It introduces SciFCheX's pipeline and details each of its component.
- **Chapter 5 – Implementation:** Provides a short overview of implementation details, covering the key technological aspects and codebase insights.
- **Chapter 6 – Experiments, Results & Discussion:** Details the experimental setup, parameters, and procedures. It also presents the results and discusses their significance

in reference to state-of-the-art.

- **Chapter 7 – Conclusions:** Summarises the outcomes of this work, provides conclusions, and highlights the potential future directions.

Chapter 2

Scientific Fact-Checking Overview

Before discussing the details of SciFCheX, it is crucial to gain a deeper understanding of automated fact-checking concepts. The aims of this chapter include defining the problem, related tasks, and the general approaches for scientific fact-checkers. Furthermore, the survey covers details and statistics for the existing science-related datasets. The section is succeeded by insights about the Transformer architecture, and review of the state-of-the-art models for scientific verification, analysing their performance and implementation. The chapter is concluded with a critical analysis that identifies research gaps.

2.1 Task Definition

General fact-checking is concerned with assessing the veracity of a factual claim made in a written or spoken language (Guo et al., 2022). This task is commonly conducted in journalism, typically achieved by a group of specialists, who are required to research a topic, find evidence, evaluate it, and finally come to a consensus related to the investigated thesis (Graves, 2017). As mentioned in §1.2., this job is performed by hundreds of international organisations, including PolitiFact[1]. Although beneficial, many of these institutions employ approaches that are not robust enough to keep up with the reality of the digital era, which brings an immense amount of data that requires verification. The traditional methods are too labour-intensive and time-consuming, sometimes taking up to several hours to deduce a

[1]Available at: https://www.politifact.com/.

final verdict for a topic (Hassan et al., 2015).

Fortunately, this issue has recently been addressed by NLP research, introducing new methodologies of *automated fact-checking*. These approaches typically utilise Transformer models, which have proven to be quite efficient in this task (Nakov et al., 2021, Vaswani et al., 2017). As depicted in Figure 2.1, the concept behind such a system involves deducing a verdict based on some background evidence fed to the model. The solutions are often modelled as a multi-step pipeline.

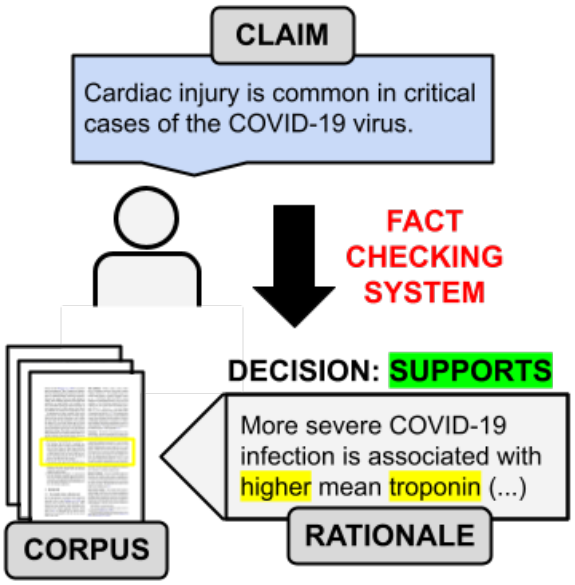

Figure 2.1: A diagram illustrating the process of a fact-checking system that valides a sample scientific statement. The model needs to be provided with background knowledge in order to deduce a verdict. In this example, it must recognise that "troponin", a cardiac muscle protein, indicates heart injuries when elevated.

Furthermore, *scientific fact-checking*, examined in this project, can be defined as a subset of the general verification task that is concerned with the assessment of veracity of science-related claims. Performing verification in this field includes additional challenges that are not solved with general fact-checkers used in, e.g., politics. As pointed by Vladika and Matthes (2023), scientific claims often include advanced terminologies, that require domain adaptation from the models, or an expert-level knowledge from a human. In addition, the growing complexity and specialised nature of scientific concepts call for advanced fact-checking systems that not only aid researchers in validating hypotheses but also help the public contextualise

the findings (Vladika and Matthes, 2023). Also, scientific data is often structured in a more complex way than just text (e.g. charts), which adds an extra layer of complexity to the process due to its multi-modality. Moreover, science evolves over time and so the solutions have to adapt to its dynamic nature. Finally, it is worth noting that the modelling approach for this task is conceptually the same as for *automated fact-checking* (Figure 2.1).

2.2 Related Tasks

The examined field identifies several related tasks that are critical for effective claim verification, collectively forming the backbone of automated systems for domain adaptation. One key task is *claim detection*, which involves identifying factual claims within texts to distinguish verifiable claims from opinions, or ambiguous assertions (Wührl and Klinger, 2021). This challenge has been also solved using Transformer-based approaches; for instance, Wührl and Klinger (2021) utilised a BERT variant to identify biomedical statements on social media, while Yuan and Yu (2019) applied a rule-based approach to recognise claims in news headlines. Furthermore, determining the *check-worthiness* of detected claims is pivotal, as not every statement might be verifiable. This aspect has been explored in the CLEF-CheckThat! shared task (Nakov et al., 2022). Moreover, Natural Language Inference (NLI), closely related to Recognizing Textual Entailment (RTE), plays pivotal role in automated fact-checking. This task helps assessing whether a premise supports or contradicts a hypothesis (Vladika and Matthes, 2023). This task is particularly challenging in scientific domains, where the interpretation of data is complicated by specialised vocabulary and metadata. Another related task of *question-answering* (QA) in science involves finding answers to questions in unstructured textual corpora, which mirrors the process of finding relevant evidence for claims in fact-checking (Karpukhin et al., 2020). Lastly, NLP research continues to address the challenge of *misinformation detection*, with recent focus on healthcare, which reflects the importance of accurate information in this field (Antypas et al., 2021).

2.3 Datasets

Selecting an appropriate data source is a crucial decision in implementing a reliable fact-checking system. Some of the well-established scientific datasets are presented in Table 2.1. As it can be seen, biomedicine and healthcare emerge as the most researched scientific fields in NLP, which might refer to the recent COVID-19 pandemic. In addition, the recent surveys (Rajpurkar et al., 2022, Guo et al., 2022) point that the insights retrieved from these particular domains can be beneficial for the development of models targeting other scientific fields, or open-domain topics.

Dataset (Authors)	**Claims**	**Domain**	**Origin**	**Evidence Source**	**Labels**
CoVERT (Mohr et al., 2022)	300	Biomedical	Twitter posts	Research, news	3
SciFact (Wadden et al., 2020)	1,409	Biomedical	Researchers	Research papers	3
Climate-Fever (Diggelmann et al., 2021)	1,535	Climate	News articles	Wikipedia articles	4
COVID-Fact (Saakyan et al., 2021)	4,086	COVID-19	Reddit posts	Research, news	2
PubHealth (Kotonya and Toni, 2020*b*)	11,832	Public health	Fact-checkers	Fact-checking sites	4
HealthVer (Sarrouti et al., 2021)	14,330	Health	Search queries	Research papers	3

Table 2.1: The datasets that are relevant for the task of scientific fact-checking (organised in ascending order, based on the number of claims).

Each presented dataset typically provides claims, evidence source, and veracity labels, that enable further evidence retrieval and verdict prediction. The process emulates the actions of a human expert. Furthermore, claims in these datasets can be categorised into two groups: *synthetic* and *natural* (Vladika and Matthes, 2023). *Synthetic* claims are often created by annotators who modify sentences from existing sources (Wright et al., 2022b). In contrast, *natural* claims are often sourced from real-world contexts, e.g., social media posts (Wadden et al., 2020). For instance, COVID-Fact (Saakyan et al., 2021) and Climate-Fever (Diggelmann et al., 2021) extracted their samples from online sources, i.e., Reddit

and Wikipedia respectively. On the other hand, SciFact (Wadden et al., 2020) comprises biomedical claims from a collection of research publications.

When it comes to labelling strategies, most of the datasets follow a classic three-label approach, popularised by Wikipedia-sourced FEVER (Thorne et al., 2018). Thereby, a given statement could be labelled as either: "Refuted", "Supported", or "NEI" ("not enough information").[2] In SciFact, a claim can be classified as "NEI" if no suitable evidence is found in the abstracts, whereas in other datasets, the label might indicate a lack of sufficient information to determine veracity. It is worth mentioning that PubHealth (Kotonya and Toni, 2020*b*) includes a fourth label, Mixed, implying that a claim has both refuting and supporting evidence. In contrast, COVID-Fact (Saakyan et al., 2021) applies a binary approach, having only two labels indicating "Supported" or "Refuted". The additional statistics for each dataset, including their distribution of labels, can be found in Appendix A, while the detailed review of SciFact is provided in §3.1.

2.4 Transformer Architecture

The introduction of Transformer (Vaswani et al., 2017) revolutionised various aspects of NLP research, including fact-checking. Thus, it is crucial to understand its architecture before further examination of its applications within the field.

Firstly, it is worth noting that previous NLP models, such as those based on Recurrent Neural Networks (RNNs), processed text in a sequential fashion (Grossberg, 2013). This approach inherently limited their ability to manage long-range dependencies. For instance, consider the sentence:

"The boy likes playing football and he also enjoys hanging out with his friends."

In this example, RNNs could struggle to link "he," "boy," and "his" in reference to the subject's actions, particularly as the complexity and length of the sentence increase. This limitation becomes more pronounced in sentences that are significantly longer or more complex.

[2]These three labels are meant to provide a uniform labelling idea that refer to verdicts. The exact names of labels in each presented dataset might slightly differ.

As introduced by Vaswani et al. (2017), Transformers resolve these limitations by utilising *self-attention* mechanisms, which allow simultaneous processing of all positions in the input sequence, thereby capturing dependencies without regard to distance within the text. This architecture, depicted in Figure 2.2, consists of two core components: *encoder* and *decoder*, each composed of layers that process data concurrently.

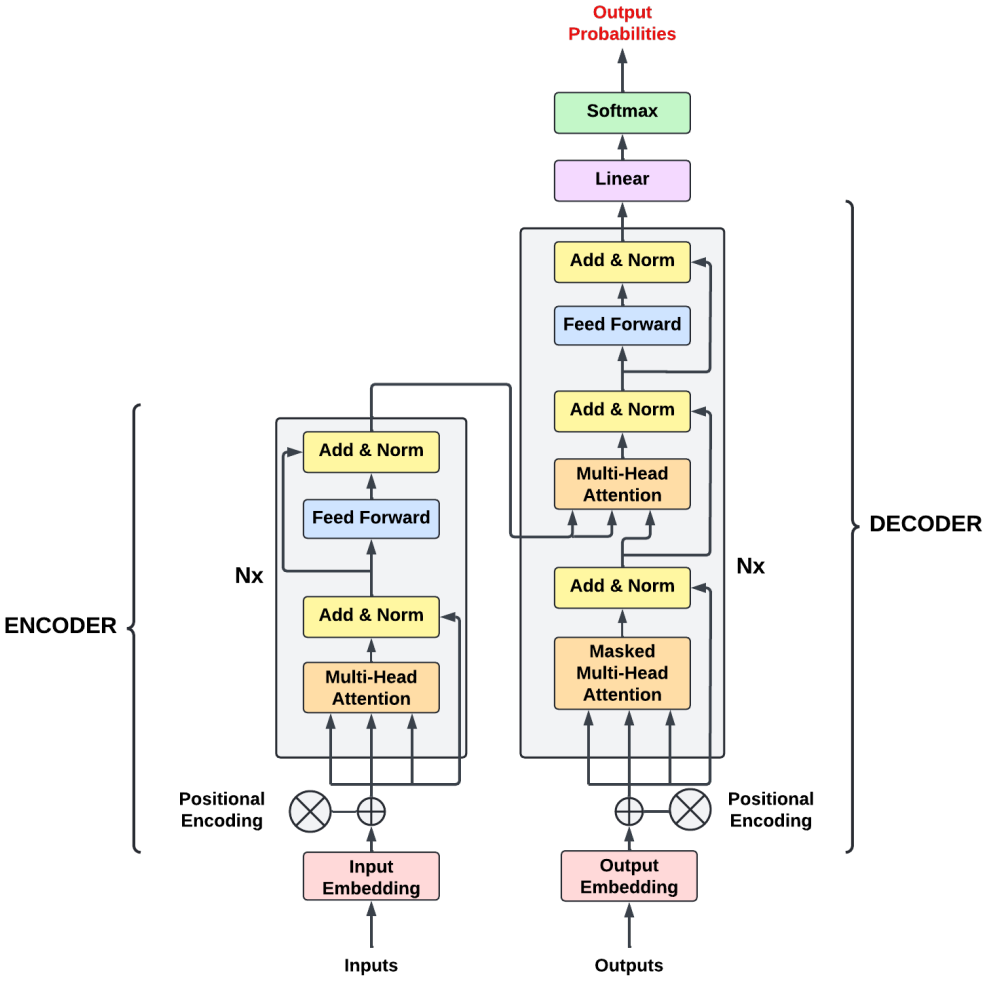

Figure 2.2: Schematic representation of the Transformer architecture. It illustrates the flow from input to output through stacked *encoder* and *decoder* layers.

Each encoder layer features a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. The encoder not only transforms each input word into rich contextual embeddings but also ensures that every word can directly influence every other

word in the sequence, a crucial capability for maintaining comprehensive contextual awareness (Vaswani et al., 2017). The decoder, utilising a similar multi-head attention mechanism, focuses both on the encoder's output and its own previous outputs to predict the next word in the sequence. This dual focus enables the decoder to align its predictions more accurately with the context provided by the encoder.

The capacity of the Transformer to evaluate all input words concurrently offers significant advantages in complex applications such as scientific fact-checking, where understanding the precise context and relationships within technical documents is essential (Soleimani et al., 2019). The architecture's ability to develop a deeper and more accurate contextual understanding, coupled with its parallel processing capabilities, which eliminate the computational inefficiencies of sequential models, makes Transformers particularly suitable for this domain.

2.5 Modelling Approaches

As briefly mentioned in §1.3, the standard modelling of scientific fact-checking pipeline consists of three main stages: *document retrieval*[3], *evidence selection*[4], and *verdict prediction*[5] (Zeng et al., 2021). The approach presented in Figure 2.2 skips the earlier mentioned steps of claim detection, or check-worthiness, by holding the assumption that the input claim is always valid.

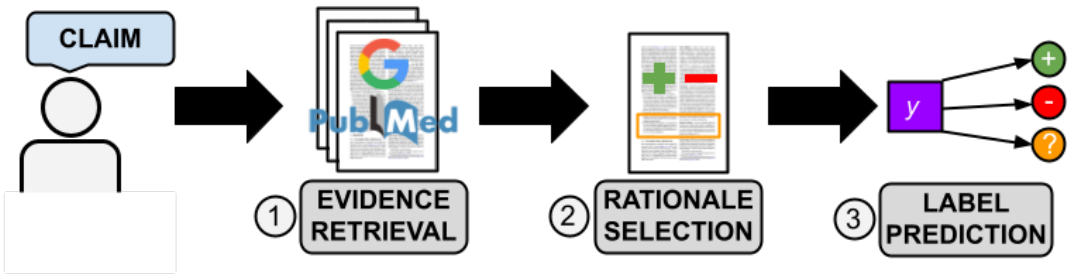

Figure 2.3: The standard three-stage pipeline for automated scientific fact-checking combining: (1) *evidence retrieval*, (2) *evidence selection*, and (3) *verdict prediction*. It is assumed that the provided input claim is always valid.

Overall, the implementation of models in a pipeline fashion, separating each of the three

[3] *Document retrieval* / *Evidence retrieval* refer to the same task and are used interchangeably.
[4] *Evidence selection* / *Rationale selection* refer to the same task and are used interchangeably.
[5] *Verdict prediction* / *Label prediction* refer to the same task and are used interchangeably.

steps (Figure 2.3), remains as the most common approach among modern fact-checkers (Vladika and Matthes, 2023). However, it is worth noting that the highest performance by the state-of-the-art systems is in fact achieved through a joint method that combines evidence selection and verdict prediction. This integration provides models with a wider context before deriving a final verdict (Wadden et al., 2022a). Moreover, modern verification approaches utilise various types of Transformers (Vaswani et al., 2017), which architecture was discussed in §2.4. Notably, these models include **B**idirectional **E**ncoder **R**epresentations from **T**ransformers (BERT) introduced in Devlin et al. (2019). Interestingly, the potential of "BERT-to-BERT" models for fact-checking was first explored in a paper from Soleimani et al. (2019). This work addressed the task on FEVER (Thorne et al., 2018), i.e., the most popular dataset for general fact-checking.

Table 2.2 provides a brief summary of the state-of-the-art models and their F1 performance on various datasets, which will be further discussed in subsequent sections. It can be observed that SciFact (Wadden et al., 2020) might be identified as the most popular dataset targeted by the highest number of systems, which was certainly motivated by the formation of its associated shared task – SciVer (Wadden and Lo, 2021) by *Allen Institute for AI*. The task is recognised as a robust performance benchmark for scientific fact-checkers and features an online leaderboard[6], which also includes participation from further proposed SciFCheX. Notably, MultiVerS (Wadden et al., 2022a) achieves a state-of-the-art result on the dataset with an F1-score[7] of approximately 0.67, significantly outperforming the baseline of VeriSci with 0.40 (Wadden et al., 2020).

2.5.1 Evidence Retrieval

Evidence retrieval (or *document retrieval*) is concerned with gathering k documents from a corpus (or any other source linked to data), that might be further used as evidence to support or refute an input claim (Wadden et al., 2020). The most common approaches to encounter this task refer to *sparse* and *dense* retrieval strategies, that emerged from

[6]The full list of submissions is available at https://leaderboard.allenai.org/scifact/submissions/public.

[7]F1-score is an evaluation metric used to assess systems by measuring their ability to correctly identify and label relevant abstracts or evidentiary sentences, balancing precision (the model's accuracy in identifying relevant items) and recall (the model's ability to find all relevant items). Further formalised in §3.3.

Dataset	Model (Author)	Document Retrieval	Rationale Selection	Verdict Prediction	F1-score
SciFact	VeriSci (baseline) (Wadden et al., 2020)	TF-IDF	RoBERTa	RoBERTa	0.395
	Paragraph-Joint (Li et al., 2021)	BioSentVec	BERT + MLP / BERT + KGAT	BERT + MLP	0.609
	VerT5erini (Pradeep et al., 2021)	BM25 + T5 (tuned on MS MARCO)	T5 (tuned on MS MARCO)	T5 (without fine-tuning)	0.634
	ARSJoint (Zhang et al., 2021)	BioSentVec	BioBERT+MLP	BioBERT+MLP	0.655
	MultiVerS (Wadden et al., 2022a)	BM25 + T5	Longformer (binary head)	Longformer (ternary head)	0.672
CoVERT	Zero-Shot MultiVerS (Wührl and Klinger, 2022)	BM25 + T5	Longformer (binary head)	Longformer (ternary head)	0.620
PubHealth	Baseline for PubHealth (Kotonya and Toni, 2020*b*)	(provided)	Sentence-BERT	SciBERT	0.705
Climate-FEVER	ClimateBERT (Webersinke et al., 2022)	(provided)	(provided)	ClimateBERT	0.757
HealthVer	Baseline for HealthVer (Sarrouti et al., 2021)	(provided)	(provided)	T5-base	0.796
COVID-Fact	Baseline for COVID-Fact (Saakyan et al., 2021)	Google Search API	Sentence-BERT	RoBERTa (fine-tuned on GLUE)	0.820

Table 2.2: State-of-the-art models in scientific fact-checking. Each presented model includes its pipeline components and performance metrics on a given dataset.

the field of Information Retrieval. The *sparse* methods (e.g. BM25, TF-IDF[8]) follow the "bag-of-words" approach, using an inverted index of all words from a corpus, and relying on term frequency measures (Thorne et al., 2018). Such methods often miss semantically relevant documents that do not contain query terms. For instance, this retrieval strategy was employed in VeriSci (Wadden et al., 2020) and it reflects in a drop of the overall system performance (F1: 0.395).

In contrast, *dense* retrieval methods use dense vector embeddings for queries, capturing a

[8]Term Frequency-Inverse Document Frequency.

broader semantic meaning (Karpukhin et al., 2020). In essence, it results in finding semantically relevant documents, that share the meaning, but might not necessarily contain the exact query terms. For instance, ARSJoint (Zhang et al., 2021) employs BioSentVec, which was trained on over 30 million biomedical sources (Chen et al., 2019). While robust, this approach is often computationally intensive, especially for larger data collections (Zhao et al., 2024). To address these challenges, the development of *hybrid* approaches has been implemented, balancing efficiency by initially filtering large document collections with sparse methods, followed by reranking the results using dense embeddings. VerT5erini (Pradeep et al., 2021) utilises such an approach initially retrieving k documents using sparse BM25, that are further neurally re-ranked with dense "text-to-text" T5 Transformer (Nogueira et al., 2020, Raffel et al., 2020). VerT5erini achieves the state-of-the-art for retrieval on SciFact, reporting an F1-score of 0.76 (Pradeep et al., 2021). Generally, it can be observed that including neural-based methods have a beneficial impact on full-pipeline performance of fact-checkers.

2.5.2 Evidence Selection & Label Prediction

Evidence selection (or *rationale selection*) is the task of selection of n relevant rationale sentences from the previously retrieved k documents, that can be used as evidence for the further step of label prediction (Vladika and Matthes, 2023). The process is often approached as a binary classification task, determining the relevance of sentences to a given statement (Guo et al., 2022). A popular method for this task includes using Sentence-BERT (Reimers and Gurevych, 2019), which was applied to find the five top-matching sentences in both PubHealth and COVID-Fact (Kotonya and Toni, 2020*b*, Saakyan et al., 2021). While this BERT variant might be intuitive to use, the model falls behind if the selected sentences use much different vocabulary than the input claim (Vladika and Matthes, 2023). To encounter this problem, Wright et al. (2022a) proposed fine-tuning the model on pairs of scientific statements and their respective news paraphrases, that led to better rationale selection in COVID-Fact and CoVERT (Saakyan et al., 2021, Mohr et al., 2022). In contrast, ARSJoint (Zhang et al., 2021) uses domain-adapted BioBERT (Lee et al., 2019), which was pre-trained on biomedical research papers.

Furthermore, *verdict prediction* (or *label prediction*) is the last stage of scientific claim

verification that returns a final verdict, often denoted by one of the three labels: "SUPPORTED", "REFUTED", "NEI". Notably, this process is usually modelled as a classification task, where the classifier is trained to predict one of the labels (Vladika and Matthes, 2023). Similarly as earlier, some models, such as the PUBHEALTH baseline, leverage this task by using Transformers that were pre-trained on domain-specific data (Kotonya and Toni, 2020*b*).

Interestingly, the best-performing models, including PARAGRAPHJOINT, MULTIVERS, ARSJOINT, and PARAGRAPHJOINT, adpoted multi-task learning to unify the processes of retrieving sentences and predicting labels (Vladika and Matthes, 2023). A shared representation of claims and abstract, preserved between the stages, make it possible to analyse a wider context. For instance, ARSJOINT pass sentence representations to an MLP classifier for evidence selection (Zhang et al., 2021), while MULTIVERS uses LONGFORMER capable of handling longer forms of text, taking up to 4096 tokens[9] (Beltagy et al., 2020). Notably, this model's zero-shot[10] application to COVERT, investigated by Wührl and Klinger (2022), unveils good adaptability capabilities, that might lead to robust performance in open-domains.

2.6 Analysis

In conclusion, this in-depth examination of the state-of-the-art in scientific fact-checking emphasises that the recent NLP advancements give hope for the automation of claim verification in the digital ecosystem in the near future. In particular, the systems following wider context thanks to the multi-tasking approach (e.g. MULTIVERS by Wadden et al. (2022a)) show a promising performance in zero-shot testing that could be used to develop open-domain claim verification systems. In addition, utilising advanced neural retrieval strategies for evidence retrieval brings a potential to expand the search of evidence into larger collections of documents. Nonetheless, the current systems can be mostly applied into experimental environments, due to the major issues associated with their scalability. This problem was recently addressed with the release of a new large test collection of 500K documents in SCIFACT-OPEN (Wadden et al., 2022b), which unveiled a performance drop of over 15 points on final F1-scores for

[9] In comparison, BERT architecture allows the maximum of 512 tokens (Devlin et al., 2019).

[10] *Zero-shot* – refers to model's ability to accurately evaluate claims from domains it has not been trained on; the approach is a good benchmark to discover model's understanding and pre-existing knowledge to make informed judgements on new data (Wadden et al., 2020).

the models leading in SciVer (Wadden and Lo, 2021).

Furthermore, it is important to note that majority of the state-of-the-art solutions target exclusively textual data, which can be concerning, especially in the context of science. There is no doubt that scientific misinformation tends to spread through other forms of media, such as images, videos, or charts. Thus, it can be deduced that the formation of appropriate datasets targeting the aspect of *multi-modality* in science is essential to further improvements. This issue was recently addressed in the general domain through the releases of new multi-modal datasets, e.g., MuMiN by Nielsen and McConville (2022), but not yet in science. Finally, it is worth mentioning the lack of *explainability* in the current solutions. As correctly pointed by Kotonya and Toni (2020*a*), the current models, implementing various DNN[11] architectures (e.g., Transformer), tend to work as "black-boxes", which makes their reasoning unclear, or even prone to biases.[12] This issue has been thoroughly addressed in the open-domain setting, while the scientific domain has been only presented by the approach from (Kotonya and Toni, 2020*a*). Thus, it can be concluded that understanding of the models' internal reasoning is a crucial step for the further development of human-centred fact-checking in science.

2.7 Summary

In summary, the chapter provided the essential context and details associated with scientific fact-checking, along with the current state-of-the-art. The verification task and its general pipeline were clearly defined. The section also identified related topics, such as, claim detection. It could be seen that the impact of the global pandemic and the spreading misinformation encouraged NLP research to address the problem by releasing new biomedical datasets. Furthermore, the analysis included examples of state-of-the-art models, exploring both separate and joint modelling architectures. In particular, the latter (e.g. ParagraphJoint (Li et al., 2021)) presented an increased performance and potential for developing general domain fact-checkers. Ultimately, the conclusions addressed several challenges – including scalability and multi-modality – that suggest that while current solutions are effective in experimental settings, significant enhancements are still required before their real-world deployment.

[11] Deep Neural Networks.

[12] For instance, a model could be fine-tuned on a biased dataset to satisfy some agenda, that might spread undetected among general public, and therefore introduce potential risks (Kotonya and Toni, 2020*a*).

Chapter 3

SciFact Task Requirements

The chapter formalises the aim of this study, discussing the requirements of the addressed SciFact task[1] (Wadden et al., 2020). It begins by outlining the formation of its dataset, setting the stage for a further analysis of the task objectives, that are built upon this dataset. The final section of this chapter introduces the evaluation metrics, both in the contexts of abstract retrieval and full-pipeline performance.

3.1 Dataset

Fold / Label	**"Supports"**	**"NoInfo"**	**"Refutes"**	**Total**
Train	332	304	173	809
Dev	124	112	64	300
Test	100	100	100	300
Total	556	516	337	1409

Table 3.1: Distribution of claim labels in SciFact, based on Wadden et al. (2020).

The presented SciFact[2] contains a total of 1,409 claims, that are verified against a collection of 5,183 abstracts (Wadden et al., 2020). The dataset is split into three folds: *train* (809 samples), *dev* (300 samples), and *test* (300 samples). To construct SciFact, the authors utilised the S2ORC corpus (Lo et al., 2020), a public source of millions of scholarly articles.

[1] SciFact task is associated with a later introduced shared task called "SciVer" (Wadden and Lo, 2021). Most of the literature sources refer to it as "SciFact task", which refers to its dataset name. Overall, either of "SciFact task" / "SciVer task" names are valid, and can be used interchangeably.

[2] The SciFact dataset can be accessed at *HuggingFace*: https://huggingface.co/datasets/allenai/scifact.

The abstract candidates were chosen based on their citations, using a seed set of highly-cited papers from various scientific domains (Wadden et al., 2020).[3] Additionally, the corpus was enriched with *distractor* abstracts to further challenge the retrieval process (Wadden et al., 2020).

SciFact follows the earlier mentioned three-labelling system, where each claim can only have one label (Wadden et al., 2020). Their exact distributions can be found in Table 3.1. The claims are identified as both *natural*, since they are extracted from real publications, and *atomic*, indicating that they can only refer to one verifiable aspect of science (Wadden et al., 2020). An example of such a statement from *train* set can be found below:

"Immune responses in immune cells are geographically segregated."[4]

The annotation process for the SciFact dataset involved experts who adapted claims from citations within source articles. Each claim was then verified against abstracts selected from the corpus, which were labelled as "Supports", "Refutes", or "NoInfo" based on their relevance to the claim (Wadden et al., 2020). Furthermore, *evidence* supporting these labels was explicitly identified by marking specific sentences within the abstracts. These sentences, known as *rationales*, crucially justify each label. The authors limited the number of rationales to three per abstract and each of them consists of no more than three sentences.

Evidence-related counts	**0**	**1**	**2**	**3+**
Cited abstracts per claim	-	1278	86	45
Evidence abstracts per claim	516	830	37	26
Rationales per abstract	-	552	290	153
Sentences per rationale	-	1542	92	11

Table 3.2: Different aspects of evidence counts (Wadden et al., 2020). For instance, column 2 of the row "Sentences per rationale" means that 92 claims are supported by two-sentence rationales.

[3]The dataset authors note that articles with fewer than 10 citations were excluded to maintain quality.
[4]Sample from SciFact *train* set with the ID of 559 (Wadden et al., 2020).

3.2 Task Objectives

The SciFact task (or SciVer) is designed to evaluate the system's ability to assess the veracity of scientific claims using abstracts (Wadden et al., 2020, Wadden and Lo, 2021). As defined in Wadden et al. (2020), the task's inputs are a claim c and a corpus of abstracts $\mathcal{A}$, where all abstracts $a \in \mathcal{A}$ are denoted with a label $y(c, a) \in \{\text{“Supports”}, \text{“Refutes”}, \text{“NoInfo”}\}$ in relation to some claim c. Furthermore, $\mathcal{E}(c)$ defines *evidence* abstracts that *support* or *refute* a claim c (Wadden et al., 2020). Each single evidence abstract $a \in \mathcal{E}(c)$ contains annotated rationales, where each rationale R_i forms a set of sentences $\{r_1(c, a), \ldots, r_m(c, a)\}$.[5] Finally, all rationales with respect to a and c are defined as: $\mathcal{R}(c, a) = \{R_1(c, a), \ldots, R_n(c, a)\}$.

Therefore, given a claim c and corpus $\mathcal{A}$, the fact-checker must predict:

- The set of evidence abstracts $\hat{\mathcal{E}}(c)$.
- The label $\hat{y}(c, a)$ mapping a claim c and an abstract a to one of:

$$\{\text{“Supports”}, \text{“Refutes”}, \text{“NoInfo”}\}$$

- The set of rationales $\hat{\mathcal{R}}(c, a)$, provided that:

$$\hat{y}(c, a) \in \{\text{“Supports”}, \text{“Refutes”}\}$$

3.3 Evaluation

Having introduced the objectives of the task, this part clarifies the evaluation approaches used to assess the effectiveness of the system. It introduces the assessment techniques for both abstract retrieval and pipeline. The former is introduced in this research as an introductory step that assesses retrieval methods in isolation. In contrast, the latter describes the *abstract-level* and *sentence-level* evaluations, as established in the works of Wadden et al. (2020) and Wadden and Lo (2021).

[5] m denotes the number of sentences in rationale R_i.

3.3.1 Primary Metrics

Before detailing the evaluation methods at different levels, it is essential to understand the concepts of *precision*, *recall*, and *F1-score*. These metrics are pivotal in assessing the performance across all evaluations in this study. They are defined as follows:

- ***Precision*** – measures the ratio of correctly identified positive results to the total number of results predicted by a model. It can be calculated as:

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}} \tag{3.1}$$

- ***Recall***[6] – quantifies the proportion of the actual positives that were correctly identified by a model. It can be expressed as:

$$\text{Recall} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}} \tag{3.2}$$

- ***F1-score*** – is the harmonic mean of precision and recall, offering a balance between the two by considering both false positives and false negatives. It can be defined as:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{3.3}$$

3.3.2 Abstract Retrieval Evaluation

As briefly noted, the *abstract retrieval*[7] evaluation is introduced as an additional layer of assessment in this study. It primarily uses the previously defined metrics of *precision*, *recall*, and *F1-score* to assess the efficacy of retrieval methods. In addition, *recall@k* is utilised for initial analyses to determine the effectiveness of retrieval within the top k ranked abstracts, which helps to tune this parameter for retrievers. Also, it provides additional insights about the dataset statistics.

[6] Also known as *sensitivity*.
[7] In the context of SciFact, *evidence retrieval* is termed as *abstract retrieval*.

Furthermore, AR evaluation is expanded by including the retrieval-specific metrics of *hit-one* and *hit-all*:

- ***Hit-one*** – this metric assesses whether at least one relevant abstract is retrieved among the top k results, reflecting initial retrieval success.
- ***Hit-all*** – contrasts with hit-one by evaluating whether all relevant abstracts are included within the top k results, offering a stringent test of retrieval completeness.

3.3.3 Pipeline Evaluation

Building upon the task theory outlined in §3.2, *pipeline* evaluation examines the model's effectiveness at both the *abstract-level* and *sentence-level*. Given the ground truth label $y(c, a)$, the set of gold abstracts $\mathcal{E}(c)$, and the set of gold rationales $\mathcal{R}(c, a)$, predictions are assessed at two levels, following the foundations from Wadden et al. (2020):

- ***Abstract-level*** – evaluates the system's ability to find abstracts that support or refute the input claim c. It consists of $Abstract_{\text{LABEL-ONLY}}$ and $Abstract_{\text{LABEL+RATIONALE}}$ evaluations.
  - $Abstract_{\text{LABEL-ONLY}}$ – $a \in \hat{\mathcal{E}}(c)$ is *correctly labelled* if: (1) $a \in \mathcal{E}(c)$, and (2) $\hat{y}(c, a) = y(c, a)$ requirements are both satisfied.
  - $Abstract_{\text{LABEL+RATIONALE}}$ – $a \in \hat{\mathcal{E}}(c)$ is *correctly rationalised* if it satisfies the above, and also the predicted rationales have a gold rationale, so that there exists some gold rationale $R_i(c, a)$ such that: $R_i(c, a) \subseteq \hat{\mathcal{S}}(c, a)$.
- ***Sentence-level*** – evaluates the system's ability to identify rationales that are relevant to justify the abstract-level results. It consists of $Sentence_{\text{SELECTION-ONLY}}$ and $Sentence_{\text{SELECTION+LABEL}}$ evaluations.
  - $Sentence_{\text{SELECTION-ONLY}}$ – the predicted rationale $\hat{f}(c, a)$ is *correctly selected* if: (1) it can be found in the gold rationale set $R_i(c, a)$, (2) all other rationales of the same gold $R_i(c, a)$ are among the results $\hat{\mathcal{S}}(c, a)$, and (3) $\hat{y}(c, a) \neq$ "NoInfo".
  - $Sentence_{\text{SELECTION+LABEL}}$ – it is *correctly labelled* if it satisfies the former, and also $\hat{y}(c, a) = y(c, a)$.

Finally, it is worth mentioning that *pipeline* evaluation is performed in two retrieval settings:

- ***"Open" setting*** – inspired by FEVER (Thorne et al., 2018); it requires the retrieval of abstracts using the examined retrieval strategies.
- ***"Oracle" setting*** [8] – inspired by ERASER (DeYoung et al., 2019); there is no actual retrieval, i.e., the gold abstracts are directly provided.

This additional step for retrieval setting is expected to provide valuable insights when compared with the tested retrievers.

3.4 Summary

This chapter detailed the formation of SciFact, which helped to contextualise the objective of SciVer (i.e., SciFact task). Furthermore, it introduced the primary evaluation metrics used throughout this study, which was followed by discussions about the evaluation methods, referring to abstract retrieval and pipeline respectively. It also formulated the abstract-level and sentence-level evaluations for the latter.

[8] It essentially imitates a *perfect retrieval* scenario, where: P=1.0, R=1.0, and F1=1.0.

Chapter 4

Methods

The chapter introduces the key methodology associated with SciFCheX and its multi-stage pipeline design. It starts with a brief discussion about BERT (Devlin et al., 2019), covering its architecture, fine-tuning, and relevance to fact-checking. The further sections of the chapter detail SciFCheX pipeline, starting with insights about abstract retrieval approaches, which include the novel hybrid approach. The final part covers the system's rationale selection and verdict prediction, which build upon the the approaches from VeriSci (Wadden et al., 2020).

4.1 BERT Overview

This section expands the basic concept of *Transformer* covered in §2.4 by introducing the details of ***B**idirectional **E**ncoder **R**epresentations from **T**ransformers* (BERT) from Devlin et al. (2019). The presented model plays a pivotal role in the context of the SciFCheX pipeline, thus it is essential to understand its concepts, architecture, and fine-tuning processes.

4.1.1 Architecture

The architecture of BERT is built upon the Transformer's *encoder stack*. The model has two core versions: *base* and *large* with 110M and 340M parameters respectively. Furthermore, BERT-base consists of 12 layers, while BERT-large includes 24 layers of a multi-layer bidirectional Transformer encoder (Devlin et al., 2019). Notably, BERT does not leverage the *decoder* component from the original Transformer architecture (Figure 2.3). Each layer

of BERT transforms an input sequence of *token embeddings* into a sequence of *contextual embeddings*, as presented in Figure 4.1. For each given token, this kind of representation integrates information from both the left and right context of the token, which makes it *deeply bidirectional* and *far-reaching* (Devlin et al., 2019). This kind of *attention* mechanism can be expressed by the equation:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{4.1}$$

Where: Q, K, and V represent the *queries*, *keys*, and *values*, respectively, with d_k denoting the dimensionality of the keys and queries. This mechanism dynamically adjusts focus across words, significantly enhancing contextual understanding (Devlin et al., 2019).

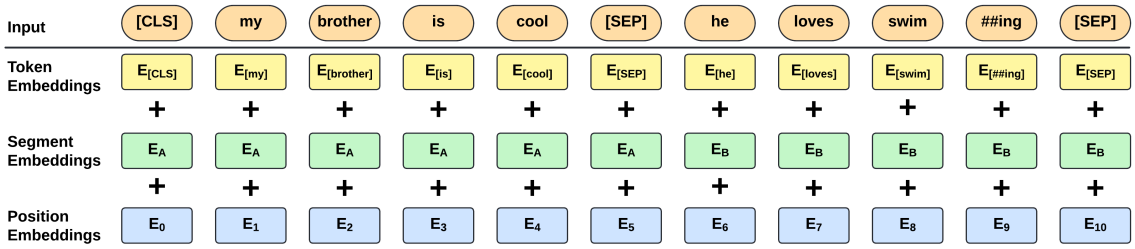

Figure 4.1: An example of representation of BERT model input (Devlin et al., 2019). The *input embeddings* consist of *token embeddings*, *segment embeddings*, and *position embeddings*.

In addition, BERT incorporates special tokens to effectively manage input sequences and output features. The `[CLS]` token is placed at the start of each input sequence, serving as an aggregate representation for classification tasks; its final hidden state is typically used as the classification layer's input. On the other hand, `[SEP]` token separates distinct text segments within the input, which is essential for multi-sequence tasks, e.g., question answering.

Furthermore, it is worth noting that BERT is pre-trained on a large corpus of data in *unsupervised* manner using two primary tasks: (1) *Masked Language Model* (MLM) and (2) *Next Sentence Prediction* (NSP) (Devlin et al., 2019). The first task (MLM) involves predicting the original tokens from the masked tokens, which forces the model to learn a deep understanding of the linguistic context. Devlin et al. (2019) applied the process randomly to 15% of tokens of the input sequence. Consequently, the latter (NSP) enables BERT to capture relationships between consecutive sentences, which is pivotal for such tasks as NLI

(Natural Language Inference) related to fact-checking (Devlin et al., 2019). Given a sentence A and its next sentence B, they provided B as the actual successor of A 50% of time, while the other trials used a random sentence, that was not A's successor, which tested the model's ability to recognise logical connections between sentences.

4.1.2 Fine-Tuning

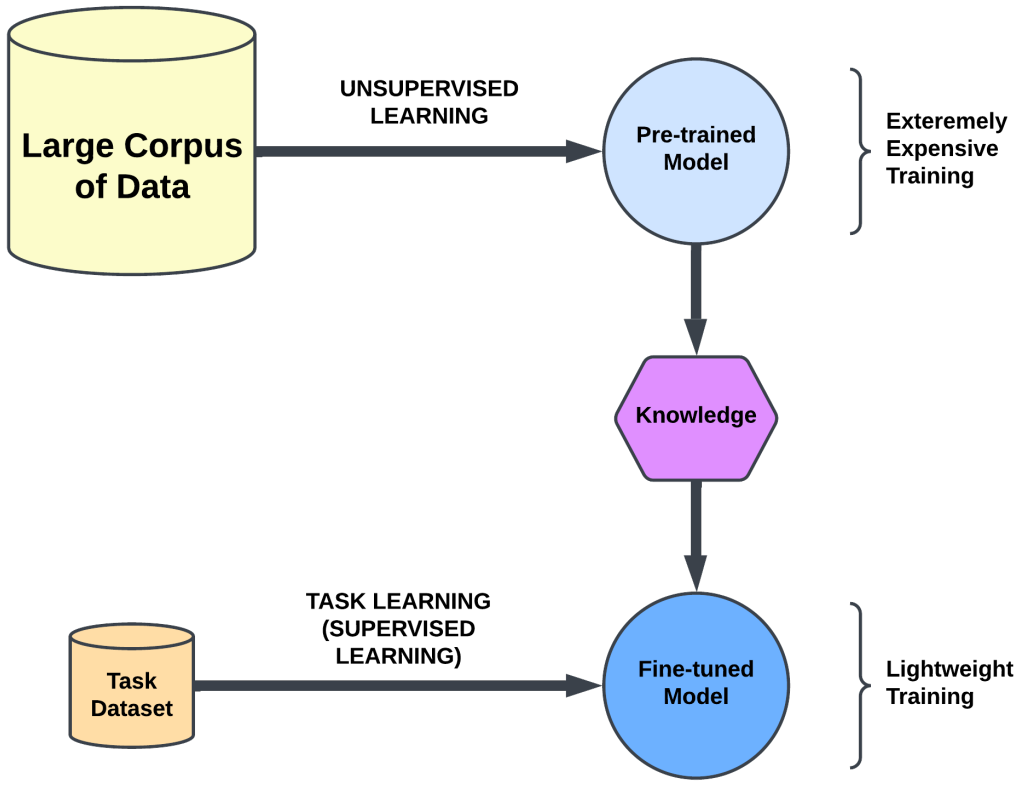

Figure 4.2: Schematic representation of the concept of *transfer learning* for BERT, where the knowledge from pre-training is transferred to the fine-tuned model (Zhang et al., 2020).

Due to its robust design, fine-tuning BERT for specific tasks (e.g., classification) involves minimal adjustments to its architecture (Zhang et al., 2020). While the pre-training process is computationally expensive and resource-demanding, fine-tuning can be executed much more rapidly using a significantly smaller dataset associated with a particular task. This phase typically involves adding a single neural network layer on top of the pre-trained model's output

layer. Once it is added, the model undergoes end-to-end fine-tuning, allowing the pre-trained knowledge to be adapted to the task-specific requirements. This process is underpinned by the principles of *transfer learning* (Figure 4.2), where a model developed for one task can be easily repurposed as the starting point for a model on another task (Zhang et al., 2020).

4.1.3 Domain-Related Variants

The promising performance of BERT models, including their capability of bidirectional context, make them exceptionally suitable for the task of fact-checking, which was first examined in the work of Soleimani et al. (2019). This study demonstrated that BERT's nuanced understanding of language and context significantly enhances its ability to identify and verify facts, which was further examined in science (Wadden et al., 2020).

Thus, this research explores several BERT variants tailored to the scientific and biomedical contexts that are relevant to this study:

- BioBERT (Lee et al., 2019) – extends BERT's architecture by continuing pre-training on a large corpus of biomedical data, which included the data from *PubMed*[1] and *PMC*[2]. This domain-oriented training enabled the model to excel in biomedical NLP tasks, such tasks as disease name recognition, biomedical data mining, or relation extraction in medical texts. It is worth noting that the model is available in two main configurations: BioBERT-base and BioBERT-large, which have 110M and 340M respectively (similarly as BERT).

- SciBERT (Beltagy et al., 2019) – is an adaptation of BERT that was trained on a corpus comprising over 1 million scientific papers from *Semantic Scholar*[3]. It has been specifically designed to better understand the domain-specific language of science.

- RoBERTa (Liu et al., 2019) – an optimised version of BERT by removing the earlier mentioned NSP task, training with larger mini-batches and learning rates, and training on larger data corpus. It improves results, e.g., for capturing the entailment relationship

[1] Available at: https://pubmed.ncbi.nlm.nih.gov/.
[2] Available at: https://www.ncbi.nlm.nih.gov/pmc/.
[3] Available at: https://www.semanticscholar.org/.

between entities. Overall, its size is slightly larger than BERT with 125M parameters for its *base* version (RoBERTa-base), and 355M for the *large* one (RoBERTa-large).

4.2 SciFCheX Pipeline

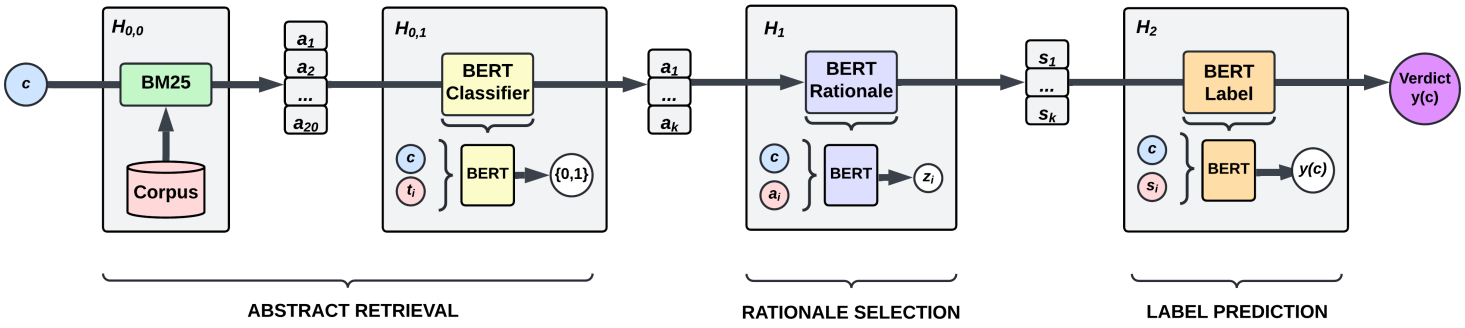

Figure 4.3: Illustration of the SciFCheX pipeline design, that consists of: *evidence retrieval*, *rationale selection*, and *label prediction*. The presented pipeline refers to the optimal solution that approaches AR using hybrid approach, that combines sparse (BM25) and dense (BERT classifier) retrievals. Further described in §4.3.3.

To address the task of scientific claim verification outlined in §3, the SciFCheX pipeline employs the "BERT-to-BERT" modelling strategy (Soleimani et al., 2019) with the structure of a three-stage pipeline (Thorne et al., 2018). This approach includes the three main stages of *evidence retrieval*, *rationale selection*, and *label prediction*; in this work, *evidence retrieval* is termed as *abstract retrieval*. The optimal system configuration, with a further breakdown of components, is illustrated in Figure 4.3 (where: H_0 refers to the approach from §4.3.3).

The components of SciFCheX are designed to accomplish the following tasks:

1. H_0**: Abstract Retrieval** — Given a claim c, the retriever identifies *at most* the top k abstracts from the corpus $\mathcal{A}$.

2. H_1**: Rationale Selection** — For each of the retrieved top k abstracts a, this stage determines the set of rationale sentences $\hat{S}(c, a)$, given the claim c.

3. H_2**: Label Prediction** — Given the claim c and the rationale sentences $\hat{S}(c, a)$, it

predicts the final label $\hat{y}(c, a)$ (i.e., verdict).

4.3 H_0: Abstract Retrieval

Given a claim c and a corpus of abstracts A, the task of AbstractRetrieval module is to identify *at most* top k relevant abstracts from $\mathcal{A}$. To do so, the study examines several approaches, including *sparse* and *hybrid* settings. This section begins with the fundamental concepts of *sparse* retrieval, to further discuss the details of the optimal *hybrid* strategy, which combines *sparse* BM25 with a task-adapted *dense* BERT-like classifier (§4.3.3). Also, the examination includes an extra *hybrid* approach using a general BGE-M3 reranker (Chen et al., 2024), which is covered in §4.3.4.

4.3.1 Sparse Retrieval: TF-IDF

The baseline model of VeriSci addresses the AR task by using the *sparse* TF-IDF[4] approach, which always returns some static k number of abstracts (Wadden et al., 2020). This strategy follows the "bag-of-words" concept, where each document is represented as a vector of *term frequencies*. Terms are independently scored according to their frequency in a document against their inverse frequency in the entire corpus of documents. This strategy emphasises words that are frequent in a document, but not across the collection (Singhal et al., 2001).

The TF-IDF score for a term t in a document d within a corpus D can be calculated as:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \times \text{IDF}(t, D) \tag{4.2}$$

where:

$$\text{TF}(t, d) = \frac{\text{frequency of term } t \text{ in document } d}{\text{total number of terms in document } d} \tag{4.3}$$

$$\text{IDF}(t, D) = \log\left(\frac{\text{total number of documents in collection } D}{\text{number of documents containing term } t}\right) \tag{4.4}$$

This calculation reflects a term's significance within a document compared to the entire collection, which enables efficient keyword-based document retrieval (Singhal et al., 2001).

[4]Term Frequency-Inverse Document Frequency

Nonetheless, TF-IDF has many limitations, including the inability to recognise synonyms.

4.3.2 Sparse Retrieval: BM25

Furthermore, BM25 (Robertson and Zaragoza, 2009) is another well-established sparse retrieval method, which also returns abstracts for some static k. It is often favoured over TF-IDF for its improved handling of term frequency and document length normalisation. Unlike TF-IDF, BM25 mitigates the risk of overemphasising frequent terms through a *saturation function*. It also adjusts for document length, which makes the assessment of document relevance more balanced. The BM25 equation is expressed with slightly different notations than TF-IDF. Namely, it explicitly focuses on the relationship of query Q and document D. The scoring function for BM25 can be defined as in Robertson and Zaragoza (2009):

$$\text{Score}(D, Q) = \sum_{i=1}^{n} \text{IDF}(t_i) \cdot \frac{f(t_i, D) \cdot (k_1 + 1)}{f(t_i, D) + k_1 \cdot (1 - b + b \cdot \frac{|D|}{\text{avgdl}})} \tag{4.5}$$

where:

- $f(t_i, D)$ — represents the term frequency of term t_i in document D.
- $|D|$ — is the length of the document.
- avgdl — is the average document length in the text collection.
- $\text{IDF}(t_i)$ — is the inverse document frequency of term t_i.

4.3.3 Hybrid Retrieval: BERT-Like Binary Classifier

Having established the fundamentals, it is now appropriate to discuss the proposed optimal method for the AbstractRetrieval module, which implements a *hybrid* strategy.

The initial stage of the hybrid retrieval process, denoted as $H_{0,0}$ in Figure 4.3, employs the BM25 algorithm to generate a preliminary ranking of abstracts. This sparse retrieval step quickly narrows down the corpus to the top k_0 most relevant abstracts, where k_0 is optimally set to 20, which is empirically justified by *recall@k* analyses in §6.1.1.

The second stage, $H_{0,1}$, is approached using *dense* retrieval. Specifically, it involves fine-tuning a BERT-like model for binary classification, where each candidate abstract from

the initial BM25 retrieval is further evaluated for its relevance to the claim. $H_{0,1}$ must be preceded with a sparse method, since it would be infeasible computationally to classify all documents from corpus, especially in the case of larger corpora.

Unlike further stages of the pipeline, the input for AbstractRetrieval is designed to take minimal context by only considering the abstracts' titles, as it was found to work quite effectively (Zeng and Zubiaga, 2021). Therefore, given a claim c and a candidate abstract with its title t, the model processes the input sequence as:

"`[CLS]` *claim* c `[SEP]` *title* t `[SEP]`"

Model Fine-Tuning

Crucially, fine-tuning is performed on SciFact's *train* set, implementing a *linear classification head* over the Transformer's output to predict the relevance of the abstract. In essence, the model is trained to return a classification of 1 for relevant abstracts, and 0 otherwise.

To build the full training set, each of the 809 claims from SciFact undergoes an initial retrieval step using BM25, where each claim is matched with a preliminary set of k_0 relevant abstracts, which directly imitates $H_{0,0}$ of the realistic retrieval scenario. Based on this, *positive* samples for training[5] are directly drawn from abstracts cited as evidence within SciFact, which certainly affirms their relevance. On the other hand, *negative* samples, are extracted from two pools:

- *Cited but non-evidential abstracts* — these are the abstracts that the claims cite, but are not part of the evidence. They train the model to differentiate between related documents, and those that directly support or refute.
- *High-ranking but un-cited abstracts* – these include abstracts that appear in the top k_0 results of BM25 retrieval, but are not actually cited (i.e., false positives from $H_{0,0}$). They train the model to ignore keyword-matching abstracts that are contextually irrelevant.

Inference Process

To sum up, the model operates with the following strategy at inference time:[5]

[5]Appendix B provides code snippets showcasing this method's training and behaviour at inference.

1. **Tokenization:** The process begins by tokenizing the corpus using the BM25 tokenizer, which removes stop words and tokenizes each document for subsequent retrieval.

2. **BM25 Retrieval:** BM25 then ranks the abstracts, identifying the top $k_0 = 20$ abstracts for a claim c. They are initially considered relevant based on keyword matching.

3. **BERT Encoding:** The top $k_0 = 20$ abstracts are then processed by the fine-tuned model. Each claim-title pair is encoded using the model tokenizer, ensuring the maximum token limit (512).

4. **BERT Classification:** The model outputs logits for each abstract, which are then transformed into binary relevance decisions. The system classifies each abstract a as: relevant (1), or not relevant (0), based on these predictions.

5. **Final Output:** The retrieval system compiles the results, selecting *at most* the top $k = 3$ relevant abstracts as determined by BERT model. In case it exceeds the value of k (which is very unlikely), the surplus of abstracts is randomly truncated. Finally, the selections are passed to RationaleSelection.

4.3.4 Hybrid Retrieval: BGE-M3 Neural Reranker

In addition, this research implements another *hybrid* approach that combines BM25 (Robertson and Zaragoza, 2009) with a lightweight BGE-M3 reranker model (Chen et al., 2024), which could be treated as an interesting alternative to the classifier presented in §4.3.3. Notably, this reranker model does not require any fine-tuning, and has not yet been evaluated on a scientific verification task, due to its recent release. Similarly as in §4.3.3, the process starts with BM25 which identifies the top k_0 abstracts from the corpus of abstracts. Furthermore, BGE-M3, the pre-trained neural reranker, assesses and reranks these candidate abstracts. Unlike BERT-classifier, it evaluates *claim-abstract* pairs using wider context. Hence, given a claim c and an abstract a, it calculates the scores using the following input:

"`[CLS]` *claim* c `[SEP]` *abstract* a `[SEP]`"

At inference time, the presented approach operates in the following way:

1. **Tokenization & BM25 Retrieval:** The first two steps are identical, as in §4.3.3.

2. **BGE-M3 Neural Reranking:** The top $k_0 = 20$ abstracts identified by BM25 are then passed to the BGE-M3 reranker. Each abstract's full text, concatenated with the claim, forms a pair that is encoded using the BGE tokenizer. This prepares each pair for neural processing.

3. **Score Normalisation and Selection:** The raw scores from BGE-M3 are first normalised into the range of 0–1. Furthermore, they are analysed to identify a score drop-off[6], which indicates a natural cutoff point for selecting the most relevant abstracts. At least $k = 1$ and up to $k = 3$ abstracts are chosen based on these scores. If the model exceeds the limit of k, the surplus of abstracts is randomly truncated.

4. **Final Output:** The final set of relevant abstracts is passed to RationaleSelection.

4.4 H_1: Rationale Selection

The next stage of the pipeline, RationaleSelection, aims to select rationale sentences $\hat{S}(c, a)$ from each abstract a for each of the top k abstracts $\hat{\mathcal{E}}(c)$ identified by AbstractRetrieval module. For this stage, the project mainly follows the foundations from VeriSci (Wadden et al., 2020), with slight adjustments to experiment with different models.

Similarly as §4.3.3, the approach leverages a model from the BERT family. The rationale selection model operates by encoding and classifying sentences in a sequence-dependent manner, utilising the contextual embeddings provided by BERT. This allows the model to effectively capture and utilise the language-specific nuances associated with scientific domain, which is crucial for accurate identification of abstracts. It aims to classify individual sentences within an abstract as either *supporting* or *refuting* in regards to a claim c. The encoding scheme concatenates each sentence a_i from an abstract a with the claim c, denoted as in Wadden et al. (2020):

$$w_i = [\textit{sentence } a_i \text{ from abstract } a, \texttt{[SEP]}, \textit{claim } c] \tag{4.6}$$

Furthermore, this sequence is processed through the Transformer model, and a $\tilde{z}_i$ score is

[6]The parameter for *drop-off* is tuned to 0.05.

computed for each sentence (Wadden et al., 2020):

$$\tilde{z}_i = \sigma[f(CLS(w_i)] \tag{4.7}$$

Where:

- σ — denotes the sigmoid function, which transforms the output of the linear layer into a probability score between 0–1 (Vaswani et al., 2017); the score indicates the likelihood of a sentence in an abstract being a rationale in relation to the claim.
- f — represents a linear layer, which can be defined as a type of neural network layer that applies a linear transformation, combining the input features into a single output score through learned weights and biases (Vaswani et al., 2017).
- $CLS(w_i)$ — is the output corresponding to the `[CLS]` token from the sequence's encoding.

Model Fine-Tuning

The model is trained on SciFact's *train* fold, particularly using pairs of claims and their cited abstracts. *Positive* examples for training are denoted by the sentences within these abstracts that are marked as the corresponding evidence. In contrast, the *negative* ones include sentences from abstracts that do not contribute to determining the stance associated with the claim. More specifically, sentences from abstracts that are cited but labelled as "NoInfo", and non-evidence sentences from abstracts labelled "Supports" and "Refutes" are used as negative samples. This approach ensures the model learns to make a distinction between sentences that might be critical for claim validation, and those that might not.

Inference Process

At inference, the model evaluates each sentence in the context of the associated claim c. Sentences are scored, and those with a score $\tilde{z}_i$ exceeding a predefined threshold[7] are selected as *rationales* (Wadden et al., 2020).

[7]The threshold t is tuned as 0.5 for SciFact, as in Wadden et al. (2020).

During the inference stage, the model processes sentences retrieved from the abstracts that were identified as potentially relevant in AbstractRetrieval[8]. Each sentence from these abstracts is concatenated with the claim and encoded using the BERT tokenizer. The model then predicts the likelihood of each sentence being a rationale based on its relevance to the claim. Sentences that meet the threshold t are selected as rationales $\hat{S}(c, a)$. Finally, the results are passed to LabelPrediction.

4.5 H_2: Label Prediction

Given a claim c, abstract a and the associated set of rationale sentences $\hat{S}(c, a)$, LabelPrediction is the final stage of SciFCheX, which predicts the final label $\hat{y}(c, a)$ so that:

$$\hat{y}(c, a) \in \{\text{“Supports”}, \text{“Refutes”}, \text{“NoInfo”}\} \tag{4.8}$$

Yet again, this stage utilises a BERT-based model to address the task. For each claim c and abstract a, the identified rationales $\hat{S}(c, a)$ received from the RationaleSelection are concatenated with the claim c. This concatenated sequence is represented as introduced in Wadden et al. (2020):

$$u = [\hat{s}_1(c, a), \ldots, \hat{s}_\ell(c, a), \texttt{[SEP]}, c] \tag{4.9}$$

Where: $\hat{s}_i(c, a)$ — are the respectively predicted rationales.

It is worth mentioning, the rationale input is subject to truncation in order to fit within the BERT's limit of 512 tokens. This key adjustment ensures that the actual claim c is never cut from the sequence u. That is unfortunately a downside of using BERT for this task, that was overcame by MultiVerS (Wadden et al., 2022a), that used the Longformer Transformer with the limit of 4096 tokens (Beltagy et al., 2020).

Furthermore, the model uses the encoded sequence to predict the probability of each label $\hat{y}(q, a)$ from {“Supports”, “NoInfo”, “Refutes”} using a linear layer f with three corresponding outputs, and a *softmax* function ϕ:

$$\tilde{y}(c, a) = \phi[f(CLS(u)] \tag{4.10}$$

[8]Provided that any abstracts are classified.

Wadden et al. (2020) notes that this training strategy aims to minimise the cross-entropy loss[9] between the predicted labels $\tilde{y}(c, a)$ and the true labels $y(c, a)$.

Model Fine-Tuning

The model is trained using pairs of claims and their corresponding cited abstracts with gold rationale sentences (Wadden et al., 2020). Certainly, the gold rationales are utilised as *positives*, while the *negative* samples consist of:

- Sentences from abstracts labelled as "NoInfo".
- Non-rationale sentences from abstracts labelled as "Refutes" or "Supports".

This training strategy teaches the model to distinguish between different types of evidence provided in the text, to capture the entailment relationship between claims and abstracts.

Inference Process

At inference time, the model evaluates the concatenated sequence of rationale sentences $\hat{S}(c, a)$ and the claim c. It computes the scores for each of the three labels, and the final verdict $\hat{y}(c, a)$ is denoted by:

$$\hat{y}(c, a) = \text{argmax } \tilde{y}(c, a) \tag{4.11}$$

If no rationale sentences are provided, then the model predicts "NoInfo", as in Wadden et al. (2020).

4.6 Summary

In summary, this chapter introduced the key concepts behind the BERT architecture, outlining its *encoder stack*, and presenting its variants that are relevant to this research; e.g., BioBERT (Lee et al., 2019), SciBERT (Beltagy et al., 2019). Furthermore, the chapter introduced the pipeline design for SciFCheX and clarified the strategies applied for each

[9] *Cross-entropy loss* — quantifies the difference between predicted probabilities and true class labels, penalising the incorrect predictions to improve model accuracy.

of the stages of the verification process. Notably, it featured the proposed retrieval strategy utilising a BERT-based classifier. It also described the method using BGE-M3 reranker (Chen et al., 2024), that might bring interesting insights when compared with the latter.

Chapter 5

Implementation

This short chapter aims to cover the key implementation details associated with the development of the SciFCheX pipeline, based on methodologies from §4. It begins by outlining the essential technologies utilised within the project, followed by a brief examination of its codebase, particularly referring to the baseline of VeriSci from Wadden et al. (2020).

5.1 Technology

SciFCheX was exclusively developed using the *Python* programming language, chosen for its comprehensive support for Natural Language Processing tools. The project employed various libraries, from general scientific ones, such as *numpy* (Van Der Walt et al., 2011) and *scikit-learn* (Pedregosa et al., 2011), to deep learning-specific frameworks, that included such packages as *PyTorch* (Paszke et al., 2019) and *HuggingFace Transformers* (Wolf et al., 2019). *Transformers* made it more accessible to import Transformer models at different checkpoints, whereas *PyTorch* tools were mainly utilised to implement the models' fine-tuning.

Any training processes involving Transformer models required access to powerful graphics processing units. Thus, all the experiments for this project were conducted on the Google Colaboratory Pro+ platform[1], utilising its powerful GPUs: Nvidia A100 and Nvidia L4.

[1]Available at: https://colab.research.google.com/.

5.2 Codebase Insights

Initially, the development of SciFCheX leveraged the VeriSci codebase[2] as a foundation. However, throughout the project, every file within this codebase has been replaced or underwent extensive modifications to satisfy the objectives of SciFCheX.

Initial challenges included resolving issues related to dependency configuration. For SciFCheX, a new *conda*[3] environments were established, incorporating newer versions of *PyTorch*, compatible with the latest *CUDA* drivers. Furthermore, the integration of the BGE-M3 reranker (Chen et al., 2024) required an introduction of dual-environmental setup, that was handled implementing BASH scripts that dynamically adjust the environments based on the experiment configuration.

As noted in earlier chapters, VeriSci leverages a simple TF-IDF retrieval. In contrast, SciFCheX involved the examination of multiple approaches, including advanced hybrid strategies. To facilitate this, the initial package for AbstractRetrieval was discarded, as it required different architectural solutions. A new package was engineered from the ground up, featuring the Retriever interface class. This class serves as a blueprint, instantiated by each specific retrieval strategy, such as BioBERTRetriever, or BM25Retriever. Equally, the scripts for AR model fine-tuning were introduced to implement the methodologies from the former chapter. In addition, there was no retrieval-specific evaluation. Therefore, the techniques from §3.3.2 were developed "from scratch", and further verified for their consistency with the results reported in literature sources.

Overall, the SciFCheX codebase implements a significantly higher level of automation when compared to VeriSci. It is primarily facilitated by extensive use of BASH scripts. These scripts automate various tasks including project setup, model training, or writing evaluation results into files, among others. For example, the RationaleSelection and LabelPrediction training scripts have been re-designed to allow the selection of different model variants. Certainly, some modules retain functionalities similar to those in Wadden et al. (2020). Nonetheless, they have been thoroughly reorganised into classes and refactored into more atomic components, referring to the SOLID principles of software engineering

[2] VeriSci codebase is publicly available on GitHub at: https://github.com/allenai/scifact.
[3] *Conda* is a Python environment/package manager. Available at: https://docs.conda.io/en/latest/.

(Hofmeister et al., 2000).

Unlike many projects identified within the fact-checking literature, SciFCheX is well-documented and maintains high quality (Appendix B). Every file, class, and function includes detailed comments explaining its purpose, which significantly enhances the project's potential for a future expansion, or recognition within the NLP community. These aspects clearly contrast with VeriSci, which suffered from poor documentation, thereby hindering its readability.

5.3 Summary

Overall, this chapter outlined the *Python*-based setup for SciFCheX, that included the key DL tools (*PyTorch, Transformers*). Furthermore, it provided insights associated with its codebase modules and highlighted its quality, referring to high-level of automation, reusability, and documentation.

Chapter 6

Experiments, Results & Discussion

This chapter presents all experimental results that led to the final formation of the SciFCheX pipeline. For each experiment, it details the parameter settings, model selection, and experimental procedures, followed by discussion of the results. The experiments are structured as follows: (1) analysis of AbstractRetrieval methods as standalone components, (2) analysis of RationaleSelection and LabelPrediction using the "Oracle"setting (i.e., passing gold abstracts), and (3) full-pipeline performance analysis in the "Open" setting, building upon the findings from (1) & (2).

6.1 Abstract Retrieval Experiments

This section covers the details of the experiments for the AbstractRetrieval module, treated a separate component. The results are reported for all methods presented in §4.3. The evaluation strategies, as defined in §3.3.2, are employed to assess the effectiveness of each method based on their performance on the SciFact development set.

6.1.1 Experimental Settings

Sparse Retrieval

Initially, the sparse retrieval methods are examined. They are utilised to conduct a preliminary experiment measuring Recall@K, providing an initial assessment of parameter settings.

Method	R@1	R@3	R@5	R@10	R@20
Oracle	0.900	0.976	1.000	1.000	1.000
TF-IDF	0.536	0.694	0.756	0.837	0.895
BM25	0.651	0.780	0.828	0.914	0.919

Table 6.1: Comparison of "Oracle" and sparse retriever's Recall@K on SciFact's *dev* set.

Method	R@1	R@3	R@5	R@10	R@20
Oracle	0.895	0.991	1.000	1.000	1.000
TF-IDF	0.569	0.732	0.810	0.881	0.917
BM25	0.667	0.810	0.879	0.918	0.945

Table 6.2: Comparison of "Oracle" and sparse retriever's R@K on SciFact's *train* set.

Additionally, the "Oracle"[1] is adjusted to analyse further dataset statistics, which might be particularly useful to understand how many evidential abstracts are linked with a claim.

The R@K experiments, conducted for $K \in \{1, 3, 5, 10, 20\}$ (Tables 6.1 & 6.2), assess performance on both the training and development sets. This is the only exception of a test on the training set, since the assessed methods did not require training.

As shown in Tables 6.1 and 6.2, the "Oracle" test indicates that approximately 90% of claims, both in train and dev sets, contain no more than one evidence abstract. The majority of cases fall within R@3 with 98% and 99% for the dev and train sets respectively. This suggests that setting $k = 3$ is an optimal choice for standalone sparse retrieval (i.e., TF-IDF, BM25), as higher values might lead to an inflated rate of *false positives*.

Furthermore, it can be seen that BM25 consistently outperforms the TF-IDF approach used in the baseline VeriSci (Wadden et al., 2020), showing nearly a 10% improvement at R@3. It also demonstrates superior performance at higher recall levels, such as R@10 and R@20, with a gain of about 2-3% per metric.

These findings thereby confirm BM25 as a more effective choice for initial stage in hybrid models. While R@10 and R@20 show minimal difference on the development set, there is a noticeable 3% improvement at R@20 on the larger training set (809 samples). Hence, this study selects $k_0 = 20$ for the preliminary stage of all hybrid retrievers.

[1] As mentioned earlier, "Oracle" imitates a *perfect retrieval* scenario, passsing the gold evidence from the dataset.

Hybrid Retrieval: BGE-M3

Following the foundation that sets $k_0 = 20$ for the initial retrieval phase, the BGE-M3 ranker is employed directly, requiring no further fine-tuning. The study uses the model checkpoint: "BAAI/bge-reranker-v2-m3"[2] Additionally, after analyses in Tables 6.1 and 6.2, the upper threshold for selecting candidate abstracts is established at $k = 3$. A *drop-off* threshold t of 0.05 is applied to manage the selection process effectively, as in §4.4.4. This particular parameter was derived by manual testing, after considering: $t \in \{0.01, 0.05, 0.10, 0.20\}$

Hybrid Retrieval: BERT Binary Classifier

BM25 with $k_0 = 20$ is consistently applied to the novel hybrid retrieval approach, both in the contexts of training and inference phases. This method is further explored using various BERT-like models, as detailed in §4.1.3. These models include:

- BERT (Devlin et al., 2019) — *base* variant; checkpoint: "bert-base-uncased"[3].
- SciBERT (Beltagy et al., 2019) — *base* variant; checkpoint: "allenai/scibert_scivocab_uncased"[4].
- BioBERT (Lee et al., 2019) — *base* variant; checkpoint: "dmis-lab/biobert-base-cased-v1.1"[5].

Furthermore, the training procedure has been followed accordingly for each of the presented BERT-like model, training on SciFact's train set. The study performed manual search for the parameters, experimenting with the following hyperparameters:

- Learning rate for the *transformer base*: 1×10^{-5}, 1×10^{-4}.
- Learning rate for the *linear layer*: 1×10^{-4}, 1×10^{-3}.
- Batch size: 256.
- Number of *epochs*: 1, 5, 10, 20.

[2]BGE-M3 checkpoint at *HuggingFace*: https://huggingface.co/BAAI/bge-reranker-v2-m3
[3]BERT checkpoint available on *HuggingFace*: https://huggingface.co/google-bert/bert-base-uncased
[4]SciBERT checkpoint available on *HuggingFace*: https://huggingface.co/allenai/scibert$_s civocab_u ncased$
[5]BioBERT checkpoint available on *HuggingFace*: https://huggingface.co/dmis-lab/biobert-base-cased-v1.1

The optimal parameters were determined to be a learning rate of 1×10^{-5} for the transformer base and 1×10^{-3} for the linear layer, with the number of epochs set to 10, as extending to 20 epochs did not imply significant improvements.

6.1.2 Results & Discussion

Retrieval Method	Hit-one	Hit-all	Precision	Recall	F1-score
TF-IDF (where: $k = 3$)	0.847	0.833	0.161	0.694	0.262
BM25 (where: $k = 3$)	**0.903**	**0.880**	0.181	**0.780**	0.294
BM25 + BGE-M3 reranker (where: $k_0 = 20$; $0 < k \leq 3$)	0.900	0.877	0.401	0.756	0.524
BM25 + BERT classifier (where: $k_0 = 20$; $k \leq 3$)	0.737	0.717	0.801	0.560	0.659
BM25 + SciBERT classifier (where: $k_0 = 20$; $k \leq 3$)	0.790	0.770	0.816	0.636	0.715
BM25 + BioBERT classifier (where: $k_0 = 20$; $k \leq 3$)	0.783	0.763	**0.862**	0.627	**0.726**

Table 6.3: Performance comparison for all retrieval methods on the dev set of SciFact.

Table 6.3 provides the results for all retrieval configurations investigated in this project.

Firstly, the transition from TF-IDF to BM25 showcases a notable improvement, with BM25 achieving the highest recall of nearly 80%. This enhancement substantiates the use of BM25 as a foundation for hybrid retrieval methods.

Furthermore, it might be seen that BM25 and reranker achieve the top results for hit metrics and recall, which implies that binary classifiers might occasionally suffer from over-filtering of their final results. It is also important to note that while the hit metrics effectively measure retrieval success, they do not penalise the inclusion of irrelevant abstracts, potentially leading to an overestimation of system performance.

The implementation of hybrid methods, especially the BioBERT-based classifier, significantly outperforms the sparse retrieval methods, denoted by an F1-score of 0.73 compared to 0.26 for the baseline TF-IDF method used in VeriSci (Wadden et al., 2020). This marked improvement can be attributed to the immense difference in precision, where classifiers achieve between 80-86%, and the reranker reaches about 40%, compared to only 16-18% reported for

sparse methods. Undoubtedly, the dynamic adjustment of k in hybrid methods contributes to the more balanced precision-recall ratio. In addition, the dense embeddings of Transformer's architecture clearly leads to enhanced context analysis, whereas the "bag-of-words" approach solely relies on term similarities.

Furthermore, all classifiers demonstrate superior performance compared to the BGE-M3 reranker (F1: 0.52). This improvement underscores the positive impact of the task-specific fine-tuning. Moreover, the advantages of BioBERT (F1: 0.73) and SciBERT (F1: 0.72) over the generic BERT (F1: 0.66) suggest that pre-training on domain-specific corpora substantially enhances the retrieval tasks. Undoubtedly, this process impacts the model's contextual understanding of the domain-related linguistics.

To sum up, the results affirm that employing fine-tuned classifiers within a hybrid retrieval framework significantly enhances the retrieval performance over the traditional sparse methods. In addition, the domain-adaptation plays a pivotal role in balancing the metrics. Finally, the general domain reranker presents a balanced option. Although it lacks the precision of fine-tuned classifiers, it significantly surpasses the baseline approaches without a need for in-domain training.

6.2 Rationale Selection & Label Prediction Experiments

The section reports the details of the experiments for RationaleSelection and LabelPrediction in isolation from AbstractRetrieval. In order to do so, the AR module is replaced with "Oracle".

In contrast to other studies within the scientific fact-checking (Wadden et al., 2020, 2022a, Pradeep et al., 2021), this project does not separate and test RationaleSelection and LabelPrediction into further standalone setups. This is mainly caused by the focus on AbstractRetrieval stage, and limited scope of the project. Therefore, the main purpose of this section is to derive the best performing combination of modules, that will be further used for full-pipeline integration.

6.2.1 Experimental Settings

For RationaleSelection and LabelPrediction, the experiments examine the following BERT models:

- SciBERT (Beltagy et al., 2019) — the same as in §6.1.1; used only for RS.
- RoBERTa (Lee et al., 2019) — *large* variant; checkpoint: "roberta-large"[6].
- BioBERT (Lee et al., 2019) — *large* variant; checkpoint: "dmis-lab/biobert-large-cased-v1.1"[7].

Furthermore, the training procedure has been followed accordingly for each of the presented BERT-like model, training on SciFact's train set. This part of the study did not involve hyperparameters' search due to the hardware-related limitations. Thus, it follows the recommendations from Wadden et al. (2020) study conducted on similar models:

- Learning rate for the *transformer base*: 1×10^{-5} (for RS/LP).
- Learning rate for the *linear layer*: 1×10^{-4} (for RS), 1×10^{-3} (for LP).
- Batch size: 256 (LP).
- Number of *epochs*: 20.

In addition, this part also examines the impact of RoBERTa-large model for LabelPrediction accessed from the resources of Wadden et al. (2020). This particular model was initially pre-trained on FEVER (Thorne et al., 2018), to further undergo SciFact fine-tuning. The experiments will reveal if the impact of the initial training benefits performance.

6.2.2 Results & Discussion

The results for all assessed combinations of RationaleSelection and LabelPrediction in "Oracle" retrieval setting are reported in Table 6.4. Furthermore, Table 6.5 provides averages of their F1-performance. As it might be observed the overall performance of the models, that were exclusively fine-tuned on SciFact, is very comparable averaging between 0.61–0.64 for F1-scores (Table 6.5).

	Abstract-Level						Sentence-Level					
	Label-Only			Label+Rationale			Selection-Only			Selection+Label		
Method	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
RS: SciBERT **LP:** BioBERT	0.756	0.608	0.674	0.702	0.565	0.626	0.779	0.645	0.706	0.591	0.489	0.535
RS: SciBERT **LP:** RoBERTa	0.716	0.579	0.640	0.669	0.541	0.598	0.796	0.661	**0.722**	0.566	0.470	0.513
RS: RoBERTa **LP:** RoBERTa	0.729	0.617	0.668	0.684	0.579	0.627	0.740	0.669	0.703	0.520	0.470	0.494
RS: RoBERTa **LP:** BioBERT	0.744	0.612	0.672	0.692	0.569	0.625	0.740	0.653	0.694	0.533	0.470	0.499
RS: BioBERT **LP:** BioBERT	0.753	0.656	0.701	0.687	0.598	0.639	0.729	0.675	0.701	0.537	0.497	0.516
RS: BioBERT **LP:** RoBERTa	0.724	0.603	0.658	0.655	0.545	0.595	0.734	0.656	0.693	0.523	0.467	0.494
RS: BioBERT **LP:** RoBERTa*	0.902	0.708	**0.793**	0.817	0.641	0.718	0.772	0.593	0.671	0.704	0.541	0.612
RS: SciBERT **LP:** RoBERTa*	0.894	0.684	0.775	0.844	0.646	**0.731**	0.790	0.617	0.693	0.723	0.566	**0.635**
RS: RoBERTa **LP:** RoBERTa*	0.910	0.675	0.775	0.851	0.632	0.725	0.794	0.590	0.677	0.713	0.530	0.608

Table 6.4: Comparison of different combinations of RationaleSelection / LabelPrediction models on the SciFact's dev set. All results reported in "Oracle" retrieval setting. RoBERTa* denotes LP model applied from VeriSci, fine-tuned on FEVER+SciFact.

Notably, there is a significant gain when applying LabelPrediction that undergoes the initial training on the FEVER dataset. For instance, the combination of SciBERT (RS) with RoBERTa-large (LP) shows an increase in average F1 from approximately 0.64 to as high as 0.71 when pre-trained on FEVER. Despite FEVER not being a science-specific dataset, its training appears to enhance LP's ability to discern entailment relationships, which is crucial for effective label prediction. This task does not necessarily favour the domain-related knowledge. Conversely, using SciBERT or BioBERT as RS generally yields superior performance on sentence-level selection tasks, benefiting from their strengths in analysing biomedical and scientific context.

To sum up, this part of experimentation leads to a conclusion that a combination of two domain-specific models does not necessarily improve the performance of verdict prediction.

[6] RoBERTa checkpoint at *HuggingFace*: https://huggingface.co/FacebookAI/roberta-large.
[7] BioBERT checkpoint at *HuggingFace*: https://huggingface.co/dmis-lab/biobert-large-cased-v1.1.

Method	Average F1
RS: SciBERT **LP:** BioBERT	0.635
RS: SciBERT **LP:** RoBERTa	0.618
RS: RoBERTa **LP:** RoBERTa	0.623
RS: RoBERTa **LP:** BioBERT	0.623
RS: BioBERT **LP:** BioBERT	0.639
RS: BioBERT **LP:** RoBERTa	0.610
RS: BioBERT **LP:** RoBERTa*	0.699
RS: SciBERT **LP:** RoBERTa*	**0.708**
RS: RoBERTa **LP:** RoBERTa*	0.696

Table 6.5: Average F1-scores for each RS / LP model combination. Based on full-results from Table 6.4.

The use of scientific-oriented RS certainly improves the performance. Nonetheless, the latter stage of LP is highly reliant on the size of dataset. Considering this, the study proceeds to full-pipeline integration with two combinations that use: BioBERT-large and SciBERT as sentence selector combined with the FEVER-tuned RoBERTa-large (from Wadden et al. (2020)), as both combinations lead to a very similar performance of approximately 0.70.

6.3 Full-Pipeline Experiments

The final section of the chapter seeks to determine the final setup for the SciFCheX pipeline. Based on findings from §6.1 & §6.2, it narrows down its selections to the following candidates for each module:

- AbstractRetrieval — uses: TF-IDF (as a baseline), BGE-M3 reranker, and both BioBERT-base and SciBERT binary classifiers, as their performance was similar.
- RationaleSelection — uses: BioBERT-large and SciBERT, since their performance was closely aligned.

- LabelPrediction — RoBERTa-large from Wadden et al. (2020), as it notably outperformed the implemented models trained on SciFact-only.

6.3.1 Results & Discussion

The results of the final pipeline formation are reported in Table 6.6, with averages provided in Table 6.7. Analysis of these results reveals that utilising a SciBERT-based rationale selector yields a slight improvement for binary classifiers—approximately a 0.01 increase in F1 scores. Notably, when this Transformer is used across both AR and RS modules, it achieves the highest downstream performance within the pipeline, averaging about 0.60 in F1-scores.

	Abstract-Level						**Sentence-Level**					
	Label-Only			**Label+Rationale**			**Selection-Only**			**Selection+Label**		
RS: SciBERT LP: RoBERTa*	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
AR: TF-IDF	0.595	0.493	0.539	0.572	0.474	0.518	0.580	0.464	0.516	0.532	0.426	0.473
AR: BM25 + BGE-M3	0.657	0.531	0.587	0.621	0.502	0.556	0.617	0.502	0.554	0.557	0.454	0.500
AR: BM25 + SciBERT	0.875	0.502	**0.638**	0.833	0.478	**0.608**	0.790	0.484	**0.600**	0.723	0.443	**0.549**
AR: BM25 + BioBERT	0.875	0.502	**0.638**	0.825	0.474	0.602	0.787	0.484	0.599	0.716	0.440	0.545
RS: BioBERT LP: RoBERTa*	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
AR: TF-IDF	0.562	0.522	0.541	0.510	0.474	0.491	0.519	0.456	0.485	0.475	0.418	0.445
AR: BM25 + BGE-M3	0.684	0.560	0.616	0.626	0.512	0.563	0.615	0.497	0.550	0. 554	0.448	0.495
AR: BM25 + BioBERT	0.863	0.483	0.619	0.829	0.464	0.595	0.799	0.467	0.590	0.724	0.423	0.534
AR: BM25 + SciBERT	0.864	0.488	0.624	0.822	0.464	0.593	0.794	0.464	0.586	0.724	0.423	0.534

Table 6.6: Comparison of different retrieval methods for the two final RS / LP configurations evaluated on SciFact dev set. Again, RoBERTa* denotes LP model applied from VeriSci (Wadden et al., 2020), fine-tuned on FEVER + SciFact.

It is noteworthy that BioBERT, despite having the highest performance for retrieval in isolation (Table 6.3), exhibits a slight decrease in impact when integrated into the full pipeline compared to SciBERT. This difference may be attributed to the superior recall of SciBERT, which could be more critical in the context of a complete pipeline, or the fact that it was pre-trained on data that was closer aligned to the task.

In general, the inclusion of binary classifiers in AbstractRetrieval consistently deliv-

RS: SciBERT **LP: RoBERTa***	**Average Precision**	**Average Recall**	**Average F1**
AR: TF-IDF	0.570	0.464	0.512
AR: BM25 + BGE-M3	0.613	0.497	0.549
AR: BM25 + SciBERT	**0.805**	0.477	**0.599**
AR: BM25 + BioBERT	0.801	0.475	0.596
RS: BioBERT **LP: RoBERTa***	**Average Precision**	**Average Recall**	**Average F1**
AR: TF-IDF	0.517	0.468	0.491
AR: BM25 + BGE-M3	0.620	**0.504**	0.556
AR: BM25 + BioBERT	0.804	0.459	0.585
AR: BM25 + SciBERT	0.801	0.460	0.584

Table 6.7: Average metrics for final model combinations, based on the full-pipeline results from Table 6.6.

ers the top performance on full-task evaluations, with results tightly grouped between 0.58 and 0.60. This underscores the advantage of accessing in-domain data for scientific fact-checking, which substantially outperforms traditional TF-IDF methods, as these typically average around 0.50 in F1 score, approximately 0.1 points lower. It is equally important to note that even the best-performing BERT-based classifier combination shows a decline of nearly 0.1 in F1 score compared to the "Oracle" setting (Table 6.5), indicating that even slight errors in retrieval can still significantly propagate to the subsequent pipeline stages.

The results also reveal that classification-based retrieval tends to create an imbalance between precision and recall, continuing the patterns observed during retrieval evaluations. Specifically, hybrid models combining BM25 with BERT-like classifiers consistently achieve high average precision of over 80%, whereas their recall rates typically range between 45-50%. Notably, the BGE-M3 hybrid approach stands out by achieving a slightly higher average recall, just over 50%, indicating a more balanced performance.

In addition, it is worth noting that the BGE-M3 hybrid approach holds its own, positioning itself between TF-IDF and domain-adapted classifiers. This indicates potential for future exploration with general-domain retrievers, particularly for challenging datasets like SciFact which are rich in specialised scientific terminology.

All in all, the most optimal configuration for the SciFCheX pipeline integrates:

- AbstractRetrieval: SciBERT.
- RationaleSelection: SciBERT.
- LabelPrediction: RoBERTa-large.

This combination leverages SciBERT's robust performance in scientific content understanding for initial stages and utilises the strong entailment and classification capabilities of RoBERTa-large for accurate verdict prediction.

6.3.2 Shared Task Performance

As previously discussed, the top-performing configuration of SciFCheX[8] has been submitted into the official leaderboard for the SciFact task. Table 6.8 details the performance comparison between the model developed in this study, the current state-of-the-art, MultiVerS[9] (Wadden et al., 2022a), and the baseline model, VeriSci[10] (Wadden et al., 2020). Furthermore, Table 6.9 provides averages across all metrics.

	Abstract-Level						**Sentence-Level**					
	Label-Only			**Label+Rationale**			**Selection-Only**			**Selection+Label**		
Model	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**	**P**	**R**	**F1**
MultiVerS #1 (Wadden et al., 2022a)	0.725	**0.738**	**0.712**	0.711	**0.724**	**0.698**	0.743	**0.745**	**0.741**	**0.672**	**0.674**	**0.670**
SciFCheX #48 (Cierkosz, 2024)	**0.804**	0.500	0.617	**0.783**	0.487	0.600	**0.758**	0.508	0.608	0.669	0.449	0.537
VeriSci #103 (Wadden et al., 2020)	0.475	0.473	0.474	0.466	0.464	0.465	0.450	0.473	0.461	0.386	0.405	0.395

Table 6.8: Comparison of SciFCheX performance with MultiVerS (Wadden et al., 2022a) and VeriSci (Wadden et al., 2020) evaluated on the test set of SciFact.

It can be analysed that SciFCheX demonstrates competitive performance, falling short of the state-of-the-art model, MultiVerS, by just over 0.1 in average F1-score. Notably, SciFCheX surpasses MultiVerS in precision in three out of four evaluation metrics, showcasing its efficacy in accurately identifying relevant information. However, the recall-precision

[8]Official report for SciFCheX: https://leaderboard.allenai.org/scifact/submission/cp4hdtfitdc6sv0kle60.
[9]Official report for MultiVerS: https://leaderboard.allenai.org/scifact/submission/c2sscs37q4q5ci4so890.
[10]Official report for VeriSci: https://leaderboard.allenai.org/scifact/submission/c07qugcginojr93itb5g.

imbalance, highlighted in §6.3.1, still remains as a challenge due to the classifier-based approach.

Model	Average Precision	Average Recall	Average F1
MultiVerS #1 (Wadden et al., 2022a)	0.713	**0.720**	**0.705**
SciFCheX #48 (Cierkosz, 2024)	**0.754**	0.486	0.591
VeriSci #103 (Wadden et al., 2020)	0.444	0.454	0.449

Table 6.9: Average metrics for the full shared task performance from Table 6.8.

Overall, transitioning from sparse to hybrid retrieval for initial AR significantly enhances performance, with SciFCheX surpassing its baseline, VeriSci, by nearly 0.15 F1 on average. While the recall gain is very minor, the overall precision is almost doubled, scoring 0.75 (SciFCheX) compared to 0.44 (VeriSci). Despite these advancements, SciFCheX still falls behind the top-notch pipelines, such as MultiVerS, which leverages extended context through the usage of joint pipeline, using Longformer that sets new benchmarks in scientific claim verification. Lastly, it could be an interesting experiment to combine SciFCheX's AR with MultiVerS' joint RS / LP modules, however it was beyond the scope of this research.

6.4 Error Analysis

The manual examination of errors led to identification of two main sources of errors for SciFCheX, which can undermine its effectiveness in specific scenarios. Firstly, the pipeline occasionally struggles with processing detailed statistical information or comparing numerical values. For instance, the model lacks the capability to decisively determine whether certain quantitative data could support or refute a claim, especially when contextual scientific benchmarks are required for interpretation.

Secondly, another significant challenge arises with niche scientific vocabulary, including the use of specialized terms and unknown acronyms that are not part of the model's training data. This vocabulary gap can hinder the model's ability to fully comprehend and accurately evaluate the content of scientific texts. Consequently, this can lead to errors when the system encounters terms or phrases that have not been sufficiently represented in the training phase,

affecting the reliability of the verification process.

6.5 Summary

In summary, this chapter covered all experiments that led to the final selection of models for SciFCheX. It began with AR-based experiments that confirmed the superior performance of all hybrid methods, particularly favouring the featured binary classifier. Furthermore, it detailed the aspect of RS / LP models derivation, that finally led to the analysis on the full-task performance. The chapter was concluded with a reference to state-of-the-art, and a brief error analysis, highlighting the high precision associated with the SciFCheX pipeline.

Chapter 7

Conclusions

In conclusion, the introduction of SciFChex within this study made a positive contribution into the field of scientific fact-checking, which addresses the emerging issue of misinformation. The study began with a review of recent advancements in the field, which set a clear framework for the further research objectives associated with the SciFact task.

The project approached the main research question by the formation of two innovative approaches for hybrid retrieval, demonstrating their superiority over traditional sparse retrieval methods. In particular, the latter clearly lack a deeper contextual understanding, while the models leveraging BERT embeddings showcased exceptional performance. They performed well not only in isolated retrieval scenarios but also across the full-pipeline, with a significant highlight of the method's precision. Also, the binary classification approach underscores the significant impact of domain-specific fine-tuning, which enhances the effectiveness of fact-checking in science, which often uses quite specialised linguistics. Moreover, the application of BGE-M3 reranker suggests the potential for further development of neural rerankers that could be optimised for open-domain tasks, as it might not always be feasible to perform in-domain training for all applications. Thus, BGE-M3 might be considered as an interesting alternative for BERT-based classifiers.

Furthermore, it is worth noting that the integration of a hybrid retrieval approach significantly boosted the overall performance of the baseline pipeline without the need for extensive modifications to the RationaleSelection and LabelPrediction components. The effectiveness of this finding has been also supported by the success associated with SciFCheX's

performance on the official SciFact shared task leaderboard.

While all the goals of this project were successfully met, there is still a significant potential for further expansion. Firstly, the investigation into the model architectures was narrowed down to BERT-like models, due to hardware limitations. Future studies could explore a wider array of architectures, including larger Transformers, that could potentially lead to new insights. For instance, there exists a list of larger variants of BGE rerankers, that might be interesting to apply. Moreover, the computational-related restrictions also limited the scope of training datasets to primarily use SciFact. Expanding this research to test retrieval methods on multiple datasets, or a larger collection of data such as SciFact-Open (Wadden et al., 2022a), could potentially provide further insights for the solutions. In particular, it would be a robust way to assess their ability to generalise across different domains. Thus, it could be verified if, for instance, model fine-tuned on biomedical data are able to transfer this domain-specific knowledge into other domains, as suggested by Vladika and Matthes (2023).

Lastly, it can be concluded that, in today's digital era, when misinformation can spread rapidly, the role of fact-checkers become more critical than ever. Automated tools, such as SciFCheX, are designed to enhance people's ability to filter through the vast amounts of data and ultimately verify claims effectively. Nonetheless, while such automated solutions are essential, we, as the society, must also invest in critical thinking and evaluation skills to ensure a well-informed public discourse.

Bibliography

Antypas, D., Rogers, D., Preece, A. and Camacho-Collados, J. (2021), Covid-19 and misinformation: A large-scale lexical analysis on twitter, *in* 'Proceedings of the Joint Conference of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop', Association for Computational Linguistics, Cardiff University, United Kingdom, pp. 119–126. Available at: https://aclanthology.org/2021.acl-srw.13/.

Beltagy, I., Lo, K. and Cohan, A. (2019), 'Scibert: A pretrained language model for scientific text', *arXiv preprint arXiv:1903.10676* . Available at: https://arxiv.org/abs/1903.10676.

Beltagy, I., Peters, M. E. and Cohan, A. (2020), 'Longformer: The long-document transformer', *arXiv preprint arXiv:2004.05150* . Available at: https://arxiv.org/abs/2004.05150.

Chen, J., Xiao, S., Zhang, P., Luo, K., Lian, D. and Liu, Z. (2024), 'Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation'. Available at: https://arxiv.org/abs/2402.03216.

Chen, Q., Peng, Y. and Lu, Z. (2019), Biosentvec: creating sentence embeddings for biomedical texts, *in* 'IEEE/ACM Transactions on Computational Biology and Bioinformatics'. Available at: https://arxiv.org/abs/1810.09302.

Del Vicario, M., Bessi, A., Zollo, F., Petroni, F., Scala, A., Caldarelli, G., Stanley, H. E. and Quattrociocchi, W. (2016), 'The spreading of misinformation online', *Proceedings of the National Academy of Sciences* **113**(3), 554–559. Available at: https://www.pnas.org/doi/abs/10.1073/pnas.1517441113.

Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. (2019), 'Bert: Pre-training of deep bidirectional transformers for language understanding', *arXiv preprint arXiv:1810.04805* . Available at: https://arxiv.org/abs/1810.04805.

DeYoung, J., Jain, S., Rajani, N. F., Lehman, E., Xiong, C., Socher, R. and Wallace, B. C. (2019), 'Eraser: A benchmark to evaluate rationalized nlp models', *arXiv preprint arXiv:1911.03429* . Available at: https://arxiv.org/abs/1911.03429.

Diggelmann, T., Boyd-Graber, J., Bulian, J., Ciaramita, M. and Leippold, M. (2021), 'Climate-fever: A dataset for verification of real-world climate claims', *arXiv preprint arXiv:2012.00614* . Available at: https://arxiv.org/abs/2012.00614.

Graves, L. (2017), 'Anatomy of a fact check: Objective practice and the contested epistemology of fact checking', *Communication, Culture and Critique* **10**(3), 518–537. Available at: https://academic.oup.com/ccc/article/10/3/518/4662968.

Grossberg, S. (2013), 'Recurrent neural networks', *Scholarpedia* **8**(2), 1888. Available at: http://scholarpedia.org/article/Recurrent$_n eural_n etwork$.

Guo, Z., Schlichtkrull, M. and Vlachos, A. (2022), 'A survey on automated fact-checking', *Transactions of the Association for Computational Linguistics* **10**, 178–206. Available at: https://direct.mit.edu/tacl/article/doi/10.1162/tacl$_{a0}$0454/109469/$A - Survey - on - Automated - Fact - Checking$.

Hassan, N., Li, C. and Tremayne, M. (2015), Detecting check-worthy factual claims in presidential debates, ACM, pp. 1835–1838. Available at: https://ranger.uta.edu/ cli/pubs/2015/claimbuster-cikm15-hassan.pdf.

Hofmeister, C., Nord, R. and Soni, D. (2000), *Applied software architecture*, Addison-Wesley Professional.

Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D. and tau Yih, W. (2020), 'Dense passage retrieval for open-domain question answering', *arXiv preprint arXiv:2004.04906* . Available at: https://arxiv.org/abs/2004.04906.

Kotonya, N. and Toni, F. (2020*a*), 'Explainable automated fact-checking: A survey', *arXiv preprint arXiv:2011.03870* . Available at: https://aclanthology.org/2020.coling-main.474/.

Kotonya, N. and Toni, F. (2020*b*), Explainable automated fact-checking for public health claims, Association for Computational Linguistics, pp. 7740–7754. Available at: https://arxiv.org/abs/2010.09926.

Lee, J., Yoon, W., Kim, S., Kim, D., Kim, S., So, C. H. and Kang, J. (2019), 'Biobert: a pre-trained biomedical language representation model for biomedical text mining', *Bioinformatics* **36**(4), 1234–1240. Available at: https://doi.org/10.1093/bioinformatics/btz682.

Li, X., Burns, G. A. and Peng, N. (2021), A paragraph-level multi-task learning model for scientific fact-verification., *in* 'SDU@ AAAI'. Available at: https://ceur-ws.org/Vol-2831/paper8.pdf.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L. and Stoyanov, V. (2019), 'Roberta: A robustly optimized bert pretraining approach', *arXiv preprint arXiv:1907.11692* . Available at: https://arxiv.org/abs/1907.11692.

Lo, K., Wang, L. L., Neumann, M., Kinney, R. and Weld, D. S. (2020), 'S2orc: The semantic scholar open research corpus', *arXiv preprint arXiv:1911.02782* . Available at: https://arxiv.org/abs/1911.02782.

Mohr, I., Wuhrl, A. and Klinger, R. (2022), 'Covert: A corpus of fact-checked biomedical covid-19 tweets', *arXiv preprint arXiv:2204.12164* . Available at: https://arxiv.org/abs/2204.12164.

Nakov, P., Barrón-Cedeño, A., Da San Martino, G., Alam, F., Míguez, R., Caselli, T., Kutlu, M., Zaghouani, W., Li, C., Shaar, S., Mubarak, H., Nikolov, A. and Kartal, Y. (2022), Overview of the clef-2022 checkthat! lab task 1 on identifying relevant claims in tweets, *in* 'CLEF 2022: Conference and Labs of the Evaluation Forum', Vol. 3180, pp. 368–392. Available at: https://research.rug.nl/en/publications/overview-of-the-clef-2022-checkthat-lab-task-1-on-identifying-rel.

Nakov, P., Corney, D., Hasanain, M., Alam, F., Barron-Cedeño, A., Papotti, P., Shaar, S. and

Martino, G. D. S. (2021), 'Automated fact-checking for assisting human fact-checkers', *arXiv preprint arXiv:2103.07769* . Available at: https://arxiv.org/abs/2103.07769.

Nielsen, D. S. and McConville, R. (2022), Mumin: A large-scale multilingual multimodal fact-checked misinformation social network dataset, *in* 'Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval', ACM. Available at: https://dl.acm.org/doi/10.1145/3477495.3531744.

Nogueira, R., Jiang, Z. and Lin, J. (2020), 'Document ranking with a pre-trained sequence-to-sequence model', *arXiv preprint arXiv:2003.06713* . Available at: https://arxiv.org/abs/2003.06713.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L. et al. (2019), 'Pytorch: An imperative style, high-performance deep learning library', *Advances in neural information processing systems* **32**. Programming library available at: https://pytorch.org/.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V. et al. (2011), 'Scikit-learn: Machine learning in python', *the Journal of machine Learning research* **12**, 2825–2830. Programming library available at: https://scikit-learn.org/stable/.

Pennycook, G., McPhetres, J., Zhang, Y., Lu, J. G. and Rand, D. G. (2020), 'Fighting covid-19 misinformation on social media: Experimental evidence for a scalable accuracy-nudge intervention', *Psychological Science* **31**(7), 770–780. Available at: https://doi.org/10.1177/0956797620939054.

Pradeep, R., Ma, X., Nogueira, R. and Lin, J. (2021), Scientific claim verification with vert5erini, *in* 'Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis', pp. 94–103. Available at: https://ceur-ws.org/Vol-2831/paper8.pdf.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W. and Liu, P. J. (2020), 'Exploring the limits of transfer learning with a unified text-to-text transformer', *Journal of Machine Learning Research* **21**, 1–67. Available at: https://dl.acm.org/doi/abs/10.5555/3455716.3455856.

Rajpurkar, P., Chen, E., Banerjee, O. and Topol, E. J. (2022), 'Ai in health and medicine', *Nature Medicine* **28**, 31–38. Available at: https://www.nature.com/articles/s41591-021-01614-0.

Reimers, N. and Gurevych, I. (2019), 'Sentence-bert: Sentence embeddings using siamese bert-networks', *arXiv preprint arXiv:1908.10084* . Available at: https://arxiv.org/abs/1908.10084.

Robertson, S. and Zaragoza, H. (2009), 'The probabilistic relevance framework: BM25 and beyond', *Foundations and Trends® in Information Retrieval* **3**(4), 333–389. Available at: http://dx.doi.org/10.1561/1500000019.

Roozenbeek, J., Schneider, C., Dryhurst, S., Kerr, J., Freeman, A., Recchia, G., van der Bles, A. and van der Linden, S. (2020), 'Susceptibility to misinformation about covid-19 around the world', *R. Soc. Open Sci.* **7**, 201199. Available at: https://royalsocietypublishing.org/doi/full/10.1098/rsos.201199.

Saakyan, A., Chakrabarty, T. and Muresan, S. (2021), 'Covid-fact: Fact extraction and verification of real-world claims on covid-19 pandemic', *arXiv preprint arXiv:2106.03794* . Available at: https://arxiv.org/abs/2106.03794.

Sarrouti, M., Abacha, A. B., Mrabet, Y. and Demner-Fushman, D. (2021), 'Evidence-based fact-checking of health-related claims', *Findings of the Association for Computational Linguistics: EMNLP 2021* pp. 3499–3512. Available at: https://aclanthology.org/2021.findings-emnlp.297/.

Singhal, A. et al. (2001), 'Modern information retrieval: A brief overview', *IEEE Data Eng. Bull.* **24**(4), 35–43. Available at: http://www1.cs.columbia.edu/ gravano/Qual/Papers/singhal.pdf.

Soleimani, A., Monz, C. and Worring, M. (2019), 'Bert for evidence retrieval and claim verification', *arXiv preprint arXiv:1910.02655* . Available at: https://arxiv.org/abs/1910.02655.

Thorne, J., Vlachos, A., Christodoulopoulos, C. and Mittal, A. (2018), 'Fever: a large-scale dataset for fact extraction and verification', *arXiv preprint arXiv:1803.05355* . Available at: https://arxiv.org/abs/1803.05355.

Van Der Walt, S., Colbert, S. C. and Varoquaux, G. (2011), 'The numpy array: a structure for efficient numerical computation', *Computing in science & engineering* **13**(2), 22–30. Programming library available at: https://numpy.org.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, and Polosukhin, I. (2017), Attention is all you need, *in* 'Advances in Neural Information Processing Systems', Curran Associates, Inc., pp. 5998–6008. Available at: https://arxiv.org/abs/1706.03762.

Vladika, J. and Matthes, F. (2023), 'Scientific fact-checking: A survey of resources and approaches', *arXiv preprint arXiv:2305.16859* . Available at: https://arxiv.org/abs/2305.16859.

Vosoughi, S., Roy, D. and Aral, S. (2018), 'The spread of true and false news online', *Science* **359**(6380), 1146–1151. Available at: https://www.science.org/doi/pdf/10.1126/science.aap9559.

Wadden, D., Lin, S., Lo, K., Wang, L. L., van Zuylen, M., Cohan, A. and Hajishirzi, H. (2020), 'Fact or fiction: Verifying scientific claims', *CoRR* **abs/2004.14974**. Available at: https://arxiv.org/abs/2004.14974.

Wadden, D. and Lo, K. (2021), 'Overview and insights from the sciver shared task on scientific claim verification', *arXiv preprint arXiv:2107.08188* . Available at: https://arxiv.org/abs/2107.08188.

Wadden, D., Lo, K., Kuehl, B., Cohan, A., Beltagy, I., Wang, L. L. and Hajishirzi, H. (2022b), Scifact-open: Towards open-domain scientific claim verification, *in* 'Findings of the Association for Computational Linguistics: EMNLP 2022', Association for Computational Linguistics, pp. 4719–4734. Available at: https://aclanthology.org/2022.findings-emnlp.347/.

Wadden, D., Lo, K., Wang, L. L., Cohan, A., Beltagy, I. and Hajishirzi, H. (2022a), 'Multivers: Improving scientific claim verification with weak supervision and full-document context', *arXiv preprint arXiv:2112.01640* . Available at: https://arxiv.org/abs/2112.01640.

Webersinke, N., Kraus, M., Bingler, J. A. and Leippold, M. (2022), 'Climatebert: A pretrained language model for climate-related text', *arXiv preprint arXiv:2110.12010v3* . Available at: https://arxiv.org/abs/2110.12010.

West, J. D. and Bergstrom, C. T. (2021), 'Misinformation in and about science', *Proceedings of the National Academy of Sciences* **118**(15), e1912444117. Available at: https://www.pnas.org/doi/abs/10.1073/pnas.1912444117.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M. et al. (2019), 'Huggingface's transformers: State-of-the-art natural language processing', *arXiv preprint arXiv:1910.03771* . Programming library available at: https://github.com/huggingface/transformers.

Wright, D., Pei, J., Jurgens, D. and Augenstein, I. (2022a), Modeling information change in science communication with semantically matched paraphrases, *in* 'Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing', Association for Computational Linguistics. Available at: https://aclanthology.org/2022.emnlp-main.117/.

Wright, D., Wadden, D., Lo, K., Kuehl, B., Cohan, A., Augenstein, I. and Wang, L. L. (2022b), 'Generating scientific claims for zero-shot scientific fact checking', *arXiv preprint arXiv:2203.12990* . Available at: https://arxiv.org/abs/2203.12990.

Wührl, A. and Klinger, R. (2021), 'Claim detection in biomedical twitter posts', *arXiv preprint arXiv:2104.11639* . Available at: https://arxiv.org/abs/2104.11639.

Wührl, A. and Klinger, R. (2022), Entity-based claim representation improves fact-checking of medical content in tweets, *in* 'Proceedings of the 9th Workshop on Argument Mining co-located with the 29th International Conference on Computational Linguistics'. Available at: https://aclanthology.org/2022.argmining-1.18/.

Yuan, S. and Yu, B. (2019), 'Hclaime: A tool for identifying health claims in health news headlines', *Information Processing and Management* **56**, 1220–1233.

Zeng, X., Abumansour, A. S. and Zubiaga, A. (2021), 'Automated fact-checking: A survey', *Language and Linguistics Compass* **15**(10), e12438. Available at: https://compass.onlinelibrary.wiley.com/doi/full/10.1111/lnc3.12438.

Zeng, X. and Zubiaga, A. (2021), 'Qmul-sds at sciver: Step-by-step binary classification for scientific claim verification', *arXiv preprint arXiv:2104.11572* .

Zhang, T., Wu, F., Katiyar, A., Weinberger, K. Q. and Artzi, Y. (2020), 'Revisiting few-sample bert fine-tuning', *arXiv preprint arXiv:2006.05987* . Available at: https://arxiv.org/abs/2006.05987.

Zhang, Z., Li, J., Fukumoto, F. and Ye, Y. (2021), 'Abstract rationale stance: A joint model for scientific claim verification', *arXiv preprint arXiv:2110.15116* . Available at: https://arxiv.org/abs/2110.15116.

Zhao, W. X., Liu, J., Ren, R. and Wen, J.-R. (2024), 'Dense text retrieval based on pretrained language models: A survey', *ACM Trans. Inf. Syst.* **42**(4). Available at: https://doi.org/10.1145/3637870.

Appendices

Appendix A

Expanded Statistics for Scientific Datasets

Dataset / Label	"Supports"	"Refutes"	"NEI"[1]	Other	Total
CoVERT (Mohr et al., 2022)	198	66	36	-	300
SciFact (Wadden et al., 2020)	556	337	516	-	1,409
Climate-Fever (Diggelmann et al., 2021)	655	253	474	153	1,535
COVID-Fact (Saakyan et al., 2021)	1,296	2,790	-	-	4,086
PubHealth (Kotonya and Toni, 2020*b*)	6,176	3,570	299	1,526	11,832
HealthVer (Sarrouti et al., 2021)	4,986	3,227	6,117	-	14,330

Table A.1: The distribution of labels in each of the scientific datasets covered in §2.3

[1] "NEI" stands for "Not Enough Information".

Appendix B

Experimental Code Examples

The section includes some basic code snippets associated with (1) SciFCheX's top-performing AbstractRetrieval, i.e., BM25 + BERT Binary Classifier (where the classifier uses BioBERT-base model), and (2) the custom-implemented AbstractRetrievalEvaluator class. These snippets showcase both the innovation and quality of SciFCheX.

```python
class BioBERTRetriever(Retriever):
    def __init__(self: 'BioBERTRetriever') -> None:
        # BM25: Build tokenizer, tokenize corpus, and load the model.
        super().__init__(retrieval_strategy='BIOBERT')
        self.tokenizer_bm25 = CountVectorizer(stop_words='english').build_analyzer()
        self.corpus_tokens = [
            self.tokenizer_bm25(doc['title'] + ' ' + ' '.join(doc['abstract']))
            for doc in self.corpus
        ]
        self.model_bm25 = BM25Okapi(self.corpus_tokens)
        # Rank TOP K abstract docs for each claim from the dataset fold.
        self.top_k_docs_for_claims = self.rank_bm25()
        # Instantiate the SciFact dataset instance. Required for torch.DataLoader.
        self.scifact_dataset = SciFactDataset(
            dataset=self.args.dataset,
            corpus=self.CORPUS_PATH,
            top_k_docs_for_claims=self.top_k_docs_for_claims
        )
        # Set up the device. Run the model on GPU (if available).
        self.device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')
        print(f'\nUsing device: "{self.device}"\n')
        # Load the BIOBERT-base transformer fine-tuned on SCIFACT.
        self.biobert_tokenizer = AutoTokenizer.from_pretrained(self.args.model)
        self.biobert_model = AutoModelForSequenceClassification.from_pretrained(self.args.model).to(self.device).eval()

    def rank_bm25(self: 'BioBERTRetriever', K=20) -> list:
        '''
        Perform BM25 ranking of abstract docs for each sample from
        the dataset fold. In nutshell, it finds the TOP K=20 docs for
        each claim from the dataset fold.

            Parameters:
            ---------------------------
            K : int, default=20
                Number of top docs for BM25.

            Returns:
            ---------------------------
            top_doc_ids_res : list
                Top K=20 docs listed for each sample from the dataset fold.
        '''
        top_doc_ids_res = []
        # Perform ranking of abstract docs for each of the claim sample.
        for data in self.dataset:
            # Set the current claim and tokenize for BM25.
            claim = data['claim']
            claim_tokens = self.tokenizer_bm25(claim)
            doc_scores = self.model_bm25.get_scores(claim_tokens)
            # Prepare K=20 best docs found for the BM25 stage.
            top_indices_bm25 = np.argsort(doc_scores)[::-1][:K]
            # Assign the corpus IDs of the identified top docs.
            top_doc_ids = [self.corpus[i]['doc_id'] for i in top_indices_bm25]
            # Update the results with new entry.
            top_doc_ids_res.append(top_doc_ids)
        return top_doc_ids_res
```

Figure B.1: The image presents the initial stage of BioBERTRetriever at inference time, that initialises the class, builds the dataset, performs BM25, etc.

```
46  class BioBERTRetriever(Retriever):
152     def classify(self: 'BioBERTRetriever', K=3) -> None:
153         '''
154         Perform classification of BM25 doc candidates for each claim
155         from the dataset fold. The classification is performed using
156         the loaded BioBERT-base model that was fine-tuned on SciFact.
157 
158             Parameters:
159             -------------------------
160             K : int, default=3
161                 The maximum number of classified docs. It is the
162                 upper threshold, which means that IFF the output
163                 contains more than 3 scores of 1, then only the
164                 first 3 will be considered.
165         '''
166         classify_res = []
167         self.biobert_model.eval()
168         with torch.no_grad():
169             for i, batch in enumerate(DataLoader(self.scifact_dataset, batch_size=self.BATCH_SIZE_GPU)):
170                 encoded_dict = self.encode(batch['claim'], batch['title'])
171                 logits = self.biobert_model(**encoded_dict)[0]
172                 output = logits.argmax(dim=1).long().tolist()
173                 # Classification step - keep only the results with score 1.
174                 # If more than K results computed, then truncate at K-1 index.
175                 if 1 in output:
176                     classified_indices = np.argwhere(np.array(output) == 1)
177                     top_docs_biobert = [
178                         self.top_k_docs_for_claims[i][ix[0]]
179                         for ix in classified_indices
180                     ]
181                     # Allow at most K=3 results.
182                     classify_res.append(top_docs_biobert[:K])
183                 else:
184                     classify_res.append([])
185         return classify_res
```

Figure B.2: The image presents the later stage of BioBERTRetriever at inference time, that performs classification using the earlier fine-tuned Transformer model.

```
60  class SciFactAbstractRetrievalDataset(Dataset):
149     def build_samples(self: 'SciFactAbstractRetrievalDataset') -> list:
153         Use matching docs as positive samples, and the other
154         unmatching ones as negatives.
155
156             Returns:
157             ------------------------
158             samples : list
159                 The list of sampled dataset.
160         '''
161         samples = []
162         for i, data in enumerate(jsonlines.open(self.dataset)):
163             # Extract top 20 docs for the current claim (data item).
164             data_top_docs = self.top20_docs_for_claims[i]
165             # Set correctly identified abstract IDs.
166             pos_abstract_ids = [
167                 int(doc_id) for doc_id in list(data['evidence'].keys())
168                 if int(doc_id) in data_top_docs
169             ]
170             cited_ids = [
171                 int(doc_id) for doc_id in data['cited_doc_ids']
172                 if doc_id in data_top_docs
173             ]
174             # Extract the set of uncorrect abstract IDs.
175             neg_abstract_ids = set(cited_ids) - set(pos_abstract_ids)
176             # Add POSITIVE samples.
177             for doc_id in pos_abstract_ids:
178                 doc = self.corpus[doc_id]
179                 # For token size, include claim+title only.
180                 samples.append({
181                     'claim': data['claim'],
182                     'title': doc['title'],
183                     'evidence': 1
184                 })
185             # Add NEGATIVE samples for wrong identifications.
186             if len(neg_abstract_ids) > 0:
187                 for doc_id in neg_abstract_ids:
188                     doc = self.corpus[doc_id]
189                     samples.append({
190                         'claim': data['claim'],
191                         'title': doc['title'],
192                         'evidence': 0
193                     })
194             # Use the remaining BM25 retrievals, as also NEGATIVE samples.
195             unmatched_docs = [
196                 doc_id for doc_id in data_top_docs
197                 if doc_id not in cited_ids
198             ]
199             for doc_id in unmatched_docs:
200                 doc = self.corpus[doc_id]
201                 samples.append({
202                     'claim': data['claim'],
203                     'title': doc['title'],
204                     'evidence': 0
205                 })
206         return samples
```

Figure B.3: The image presents a code snippet from the fine-tuning script for AbstractRetrieval, which handles building samples for the training set.

```python
class AbstractRetrievalEvaluator(Evaluator):
    @classmethod
    def calc_hit_all(cls, predict_docs: set, true_docs: set) -> int:
        '''
        Calculate HIT-ALL, which: requires all retrievals to be consistent
        with golden (true) retrieval docs to be successful.

            Parameters:
            ------------------------
            predict_docs : set
                Set of the predicted docs.
            true_docs : set
                Set of all the true docs.

            Returns:
            ------------------------
            int
                1 - if hits one. 0 - otherwise.
        '''
        if predict_docs.issuperset(true_docs):
            return 1
        return 0

    def eval(self: 'AbstractRetrievalEvaluator') -> None:
        '''
        Perform evaluation for AR using all standard metrics over the
        dataset. They include: precision, recall, F1, hit-one, hit-all.
        '''
        # Calculate standard eval metrics for predictions for each sample.
        for prediction in self.predict_docs:
            claim_id = prediction['claim_id']
            # Set the predicted docs and the true ones, as in the dataset.
            predict_doc_ids = prediction['doc_ids']
            true_doc_ids = set(self.gold_docs[claim_id])
            # Update hit metrics.
            self.hit_one += self.calc_hit_one(set(predict_doc_ids), true_doc_ids)
            self.hit_all += self.calc_hit_all(set(predict_doc_ids), true_doc_ids)
            # Update the counter object for further P, R, F1 calculations.
            self.counter['relevant'] += len(true_doc_ids)
            for prediction in predict_doc_ids:
                self.counter['retrieved'] += 1
                if prediction in true_doc_ids:
                    self.counter['correct'] += 1
        # Finally, calculate P, R, F1 cumulative metrics (using parent's method).
        (self.precision, self.recall, self.f1) = self.calc_f1(self.counter)
        # Normalize hits metrics, by division over dataset size.
        self.hit_one = self.hit_one / len(list(self.gold_docs.keys()))
        self.hit_all = self.hit_all / len(list(self.gold_docs.keys()))
```

Figure B.4: The image presents a code snippet from the custom-implemented AbstractRetrievalEvaluator class.