Knowledge-Intensive NLP for Real-World Information Needs

David Wadden

A dissertation

submitted in partial fulfillment of the

requirements for the degree of

Doctor of Philosophy

University of Washington
2023

Reading Committee:
Hannaneh Hajishirzi, Chair
Noah Smith
Sheng Wang

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Knowledge-Intensive NLP for Real-World Information Needs

David Wadden

Chair of the Supervisory Committee:

Hannaneh Hajishirzi
Department of Computer Science and Engineering

Knowledge-intensive NLP aims to help humans navigate and synthesize the information contained in massive textual corpora. The past few years have witnessed rapid progress on knowledge-intensive NLP for general-domain corpora like Wikipedia, which can be accessed conveniently and annotated at scale using crowd workers. However, approaches that perform well on general-domain text may fail when applied to specialized settings like scientific literature, or when asked to generalize to novel entities or concepts. Three key research challenges arise in these real-world settings: (1) Standard task formulations and evaluation metrics may not adequately capture user information needs, (2) Training data are often scarce and expensive, and (3) Novel linguistic phenomena — ranging from vocabulary shift to long-range semantic dependencies — may not be captured by existing modeling approaches.

In this thesis, we report progress on three knowledge-intensive NLP tasks addressing real-world information needs. *Scientific information extraction* aims to organize the findings reported in scientific literature into a structured knowledge graph. *Scientific claim verification* aims to assess the veracity of scientific claims against a corpus of research literature. Finally, *entity-centric query refinement* aims to help users navigate an open-ended space of entities to quickly understand a new domain or discover relevant information. Throughout this thesis, we will frequently encounter the three key research challenges outlined above. We hope that our proposed solutions will contribute toward the development of more robust, flexible, and capable systems for knowledge-intensive NLP.

Acknowledgements

Thanks first and foremost to my advisor, Professor Hannaneh Hajishirzi. Hanna has provided the flexibility to explore problems and applications that I find inspiring, while also offering direction and guidance to help me stay focused on important research questions. My writing and presentation skills have benefitted greatly Hanna's feedback, and she has been instrumental in helping me find collaborations that have enriched my PhD experience.

Thanks to my committee members Noah Smith, Sheng Wang, and Meliha Yetisgen, for valuable research feedback and career guidance. Thanks also to Neil Abernethy for serving as GSR for my thesis defense.

Thanks to the many collaborators with whom I've been lucky enough to work during my PhD. Thanks to Yi Luan for mentoring me on my first NLP project. At AI2, thanks to Arman Cohan, Iz Beltagy, Bailey Kuehl, Madeleine van Zuylen, and Tom Hope. Thanks especially to my internship hosts Kyle and Lucy, both for the many insightful conversations and for the editorial heroics before multiple *ACL deadlines. At UW, thanks to my collaborators Rahul Nadkarni, Aida Amini, Tim Althoff, Qisheng Li, Tal August, and Ashish Sharma. Thanks also to my mentees Ulme Wennberg and Peter Lin. Finally, thanks to my Google internship hosts Nikita Gupta, Kenton Lee, and Kristina Toutanova.

Thanks to my lab-mates, office-mates, and fellow UW NLP students for enjoyable discussions and valuable feedback over the years.

Finally, thanks to my friends and family.

DEDICATION

To Jiechen

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In the past few decades, massive online corpora have become available in domains ranging from science and medicine, to newswire, to social media. In the science domain, for example, the Semantic Scholar academic search engine[1] indexes over 200 million research articles at present. The availability of this massive repository of knowledge has the potential to empower users to make better decisions both in their personal lives (e.g. when parents decide whether the COVID-19 vaccine is safe for their children), and on a global scale (e.g. when lawmakers formulate policies in response to the effects of climate change). However, the volume of the available knowledge is far too vast for a single person to read and assimilate.

As a result, there is an urgent need for automated NLP systems to help us make use of the knowledge contained within massive textual corpora. Research in this area is broadly referred to as *knowledge-intensive NLP* [Petroni et al., 2021], and encompasses a wide variety of tasks, including: information retrieval, entity linking, knowledge base construction, question answering, and fact-checking. The last decade has witnessed rapid progress on knowledge-intensive NLP, particularly on tasks associated with Wikipedia and its knowledge resources (e.g. Wikidata [Vrandecic and Krotzsch, 2014] and YAGO [Suchanek et al., 2007]). However, systems developed for Wiki-domain text often under-perform when deployed in novel settings.

In this thesis, we study three knowledge-intensive NLP tasks motivated by the shared goal of facilitating scientific understanding and discovery, which are not well-addressed by standard approaches. The *scientific information extraction* task aims to extract and organize information contained in research literature. *Scientific*

[1] `https://www.semanticscholar.org/`

claim verification aims to verify scientific claims against the literature. Finally, *entity-centric query refinement* aims to assist users in open-ended exploration and concept discovery.

As we study these tasks, we will frequently encounter — and propose solutions to — three key research challenges:

1. **Non-standard task formulations**: Meeting a user information need requires formulating a new task and proposing novel evaluation metrics.
2. **Scarce labeled data**: Labeled in-domain demonstrations are either unavailable, or require domain experts.
3. **Novel linguistic phenomena**: Solving a task requires specialized linguistic capabilities.

The remainder of this thesis is organized is follows. In the rest of the Introduction, we provide background on each task and highlight our main contributions. In Chapter 2, we present our work on scientific information extraction. Chapters 3-5 report progress on scientific claim verification, and Chapter 6 studies entity-centric query refinement. We conclude in Chapter 7 by summarizing the work presented in this thesis, and identifying some key future directions for real-world knowledge-intensive NLP.

1.1 Scientific information extraction

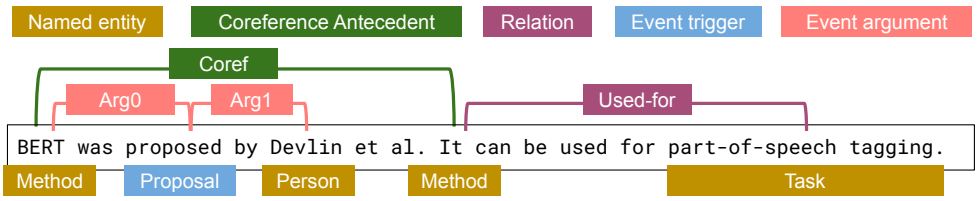


Figure 1.1: An example illustrating the four information extraction tasks studied in this work.

Information extraction, or *IE*, aims to organize free-text information into a structured *knowledge base*. In this thesis, we study four common IE sub-tasks, illustrated in Figure 1.1. **Named entity recognition** aims to extract mentions of *entities*, which are unique objects in the world. Entities can range from people or places, to genes or computer science algorithms. **Coreference resolution** aims to cluster mentions within a document which refer to the same underlying entity. **Relation extraction** aims to extract relationships between pairs of entities. Finally, **event extraction** aims to extract events that occur. Events include a *trigger*

(the predicate indicating that an event has occurred), associated with a number of *arguments* (the entities participating in the event).

Information extraction has a long history in NLP, dating back to the Message Understanding Conference (MUC) series beginning in the late 1980's [Sundheim, 1993]. Early attempts at IE relied on "expert systems" defined using hand-written templates, regular expressions, and finite state machines [Hobbs, 1993]. Since that time, a variety of datasets have been released in domains including (among others) newswire and the web [Doddington et al., 2004], molecular biology [Kim et al., 2003; Krallinger et al., 2017], and computer science [Luan et al., 2018a]. In the early 2010's, IE methodology was dominated by structured prediction approaches applied to syntactic, discourse, and other hand-engineered features [Li et al., 2013; Yang and Mitchell, 2016; Li and Ji, 2014]. Shortly prior to the work presented in this thesis, neural architectures for IE began to supersede structured prediction approaches on most tasks [Peng et al., 2017; Zhang et al., 2018; Sha et al., 2018; Christopoulou et al., 2018]. Grishman [2019] provides a review on the history of information extraction.

In Chapter 2 of this thesis, we present DyGIE++, which accomplishes the four IE tasks mentioned above by enumerating and scoring spans of text. This span-based approach is *simple* and *general*. Many previous works on neural IE developed bespoke architectures for a single task, which required annotations from other IE tasks as input — for instance, a relation extraction system might require "gold" entities. DyGIE++, by contrast, performs all tasks simultaneously and does not require any gold inputs. In addition, DyGIE++ is able to capture two key linguistic phenomena necessary to understand scientific text (Challenge 3). First, since DyGIE++ makes independent predictions for each text span, it can extract overlapping and nested entity mentions, which are quite frequent in scientific text. Second, DyGIE++ models cross-sentence context by constructing a graph of all entity mentions in a document, and updating the representations of each mention based on those of its graph neighbors. Understanding of cross-sentence context is crucial when performing IE over scientific documents, which make heavy use of acronyms and coreference.

1.2 Scientific claim verification

Automated claim verification (which we will refer to interchangeably as *fact checking*) aims to assess the veracity of a claim against a body of relevant knowledge. In this thesis, we formulate fact checking as follows.

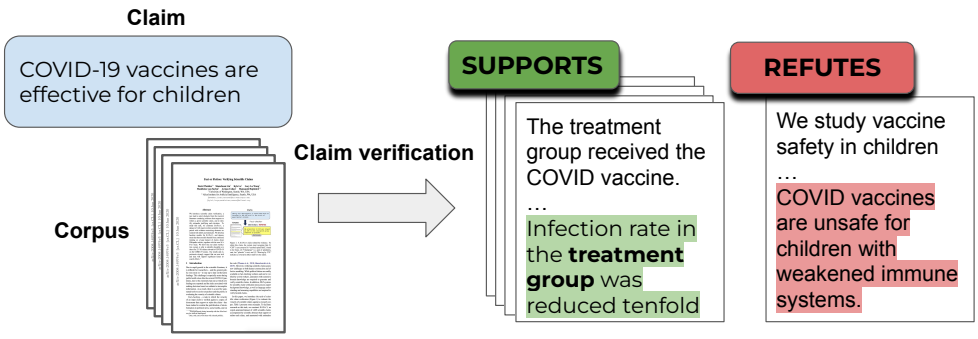


Figure 1.2: An example illustrating the claim verification task. Evidence supporting and refuting the input claim is identified from a corpus.

Given a claim c and a corpus of documents $\mathcal{A}$, the task is to: (1) Identify all documents $a \in \mathcal{A}$ that either SUPPORT or REFUTE c. (2) For each identified document, identify the *rationales* (also referred to in the literature as *highlights* or *evidence* [Wiegreffe and Marasović, 2021]) justifying the label. Figure 1.2 provides an example.

Fact checking has a short history compared to IE. The task was proposed in 2014 by Vlachos and Riedel [2014]. Between the time of its proposal and the work presented in this thesis, a number of fact checking datasets were released in domains including Wikipedia [Thorne et al., 2018, 2021] and newswire [Hanselowski et al., 2019; Wang, 2017; Augenstein et al., 2019]. Most modeling approaches follow the same general pipeline: (1) Retrieve potentially relevant documents, using e.g. lexical approaches like BM25 [Robertson and Zaragoza, 2009], or dense retrieval [Karpukhin et al., 2020]. (2) Select relevant sentences from the retrieved documents. (3) Make an entailment prediction based on the selected sentences. The sentence selection and entailment prediction steps can be performed using a variety of neural architectures, such as a CNN [Yin and Roth, 2018], or more recently pretrained transformers [Soleimani et al., 2019]. For a recent survey on automated fact-checking, see Guo et al. [2022].

Despite progress on this task, when we began work on claim verification there were no datasets or models available to verify claims in the scientific domain. In addition, as we will demonstrate in Chapter 4, models trained on text from the Wiki or news domains tend to struggle when deployed on scientific text. With the emergence of the COVID-19 pandemic in the winter of 2020 and the accompanying rise in scientific mis- and disinformation [Pennycook et al., 2020], the need for scientific claim verification resources became increasingly urgent. Chapters 3-5 describe our efforts to address this need.

As we developed methods and resources for scientific fact-checking, we encountered all three of the key research challenges outlined earlier in this Introduction. Regarding Challenge 1 (non-standard task formulations), prior work on fact-checking generally required a model to identify just a *single* source of evidence. This task formulation is inappropriate for science, both because there may be conflicting evidence, and because the *weight* of the evidence (i.e. number of documents supporting and refuting) is relevant when assessing the veracity of a scientific claim. Therefore, for scientific fact-checking, we require models to identify *all* available evidence in the reference corpus. This requirement for exhaustive evidence raises Challenge 2 (data scarcity): annotating scientific evidence is challenging and time-consuming, even for domain experts. In Chapters 3 and 5, we propose approaches to collect expert annotations at scale and with reasonable coverage. Finally, we find that verifying scientific claims often requires understanding the full context in which a piece of evidence appears, rather than just interpreting a single sentence in isolation (Challenge 3; novel linguistic phenomena). In Chapter 4, we propose a modeling approach which makes veracity predictions based on the full available context, and which is capable of leveraging weakly-labeled in-domain data to supplement human labels (Challenge 2).

1.3 Entity-centric query refinement

Given an input query from a system user, the goal of *query refinement* is to return a collection of refinements that help satisfy a user's information need. The nature of the desired refinements depends on the user's search intent. During *lookup search*, a user has a well-defined information need, and the goal of the query refinement is to clarify and resolve ambiguity. For instance, refinements to the query "Harry Potter release date" might include "Harry Potter release date in the United States", "Harry Potter release date in the UK", etc.[2]. By contrast, during *exploratory search* [Marchionini, 2006; White and Roth, 2009], the user does not have a specific information need, but instead is interested in discovering new concepts or gaining a better understanding of

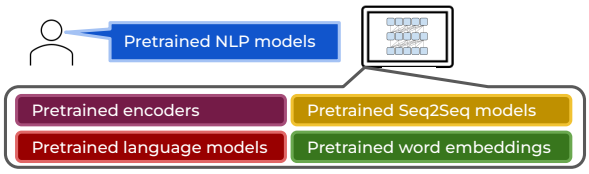


Figure 1.3: An example illustrating the query refinement task. Given an open-ended query whose answer is a list of entities (e.g. "Pretrained NLP models"), the task aims to provide a list of refinements which can help structure the answer space for the user, facilitating domain understanding and entity discovery.

[2]The Harry Potter example is borrowed from Min et al. [2020]

a new domain. For instance, a user who enters a query for "pretrained language models" probably would not be well-served by reading a list of every pretrained language model in existence. Query refinements can assist this user by categorizing pretrained language models into semantically meaningful groups. Figure 1.3 provides an example. In this work, we study query refinement in the exploratory setting. In addition, we focus specifically on *entity-centric queries*; these are queries whose answer is a list of entities.

Query refinement for exploratory search has a relatively long history, characterized by two main types of approaches. *Search results clustering* [Carpineto et al., 2009] generates query refinements by clustering the documents identified by an information retrieval system. Classic approaches relied on word count-based statistics to perform clustering [Zamir and Etzioni, 1999; Osinski and Weiss, 2005], while more recent work has relied on Seq2Seq models [Medlar et al., 2021] to generate cluster names. *Faceted search* [Tunkelang, 2009; Hearst, 2006] organizes the outputs of an IR system according to a faceted concept hierarchy. The hierarchy can be created by the system designers [Yee et al., 2003], adapted from an existing taxonomy [Li et al., 2010; Arenas et al., 2016], or constructed automatically [Stoica et al., 2007].

Importantly, both lines of work rely on external resources, in the form of a corpus from which to retrieve documents and (in the case of faceted search) a faceted taxonomy. Faceted search, in particular, has been limited by the need for pre-existing concept taxonomies; refinements are only available for concepts covered by the taxonomy. Given that large language models have been shown to store knowledge in their parameters [Petroni et al., 2019; Roberts et al., 2020a] the question we explore in this work is whether language models can be trained to generate entity-centric query refinements for exploratory search, *without access to external resources at inference time*. This approach enjoys the dual advantages of (1) Reducing computational burden by removing the need for document retrieval, and (2) Being able to generate refinements for *arbitrary* input queries, not just those covered by existing knowledge bases or taxonomies.

We propose an approach for this task in Chapter 6. We formally define the entity-centric query refinement task, and establish both automated and human evaluation criteria (Challenge 1). Since there are no labeled datasets for this task (Challenge 2), we create one using programmatic weak supervision based on the Wikipedia category hierarchy. We show that a model trained on our weakly-supervised demonstrations can generate refinements for queries not covered by the training taxonomy.

Chapter 2

Scientific Information Extraction with DyGIE++

This chapter contains material that was originally published in Wadden et al. [2019].

2.1 Introduction

Many information extraction tasks – including named entity recognition, relation extraction, event extraction, and coreference resolution – can benefit from incorporating global context across sentences or from non-local dependencies among phrases. For example, knowledge of a coreference relationship can provide information to help infer the type of a difficult-to-classify entity mention. In event extraction, knowledge of the entities present in a sentence can provide information that is useful for predicting event triggers.

To model global context, previous works have used pipelines to extract syntactic, discourse, and other hand-engineered features as inputs to structured prediction models [Li et al., 2013; Yang and Mitchell, 2016; Li and Ji, 2014] and neural scoring functions [Nguyen and Nguyen, 2019], or as a guide for the construction of neural architectures [Peng et al., 2017; Zhang et al., 2018; Sha et al., 2018; Christopoulou et al., 2018]. Recent end-to-end systems have achieved strong performance by dynamically constructing graphs of spans whose edges correspond to task-specific relations [Luan et al., 2019; Lee et al., 2018; Qian et al., 2018].

Meanwhile, contextual language models [Dai and Le, 2015; Peters et al., 2017, 2018; Devlin et al., 2018]

have proven successful on a range of natural language processing tasks [Bowman et al., 2015; Sang and De Meulder, 2003; Rajpurkar et al., 2016]. Some of these models are also capable of modeling context beyond the sentence boundary. For instance, the attention mechanism in BERT's transformer architecture can capture relationships between tokens in nearby sentences.

In this chapter, we study different methods to incorporate global context in a general multi-task IE framework, building upon a previous span-based IE method [Luan et al., 2019]. Our DyGIE++ framework, shown in Figure 2.1, enumerates candidate text spans and encodes them using contextual language models and task-specific message updates passed over a text span graph. Our framework achieves state-of-the results across three IE tasks, leveraging the benefits of both contextualization methods.

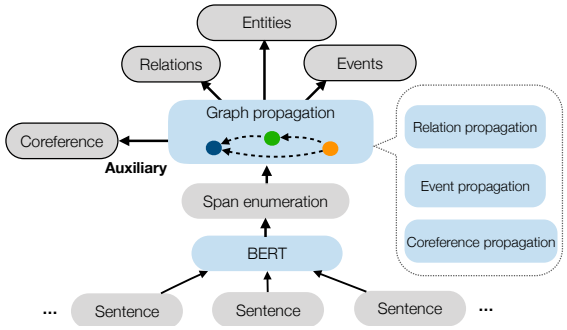


Figure 2.1: Overview of DyGIE++. Shared span representations are constructed by refining contextualized word embeddings via span graph updates, then passed to scoring functions for three IE tasks.

We conduct experiments and a thorough analysis of the model on named entity, relation, and event extraction. Our findings are as follows: (1) Our general span-based framework produces state-of-the-art results on all tasks and all but one subtasks across four text domains, with relative error reductions ranging from 0.2 - 27.9%. (2) BERT encodings are able to capture important within and adjacent-sentence context, achieving improved performance by increasing the input window size. (3) Contextual encoding through message passing updates enables the model to incorporate cross-sentence dependencies, improving performance beyond that of BERT alone, especially on IE tasks in specialized domains.

2.2 Task and Model

Our DyGIE++ framework extends a recent span-based model for entity and relation extraction [Luan et al., 2019] as follows: (1) We perform event extraction as an additional task and propagate span updates across a graph connecting event triggers to their arguments. (2) We build span representations on top of multi-sentence BERT encodings.

2.2.1 Task definitions

The input is a document represented as a sequence of tokens D, from which our model constructs spans $S = \{s_1, \ldots, s_T\}$, the set of all possible within-sentence phrases (up to a threshold length) in the document.

Named Entity Recognition involves predicting the best entity type label e_i for each span s_i. For all tasks, the best label may be a "null" label. Relation Extraction involves predicting the best relation type r_{ij} for all span pairs (s_i, s_j). For the data sets studied in this chapter, all relations are between spans within the same sentence. The coreference resolution task is to predict the best coreference antecedent c_i for each span s_i. We perform coreference resolution as auxiliary task, to improve the representations available for the "main" three tasks.

Event Extraction involves predicting named entities, event triggers, event arguments, and argument roles. Specifically, each token d_i is predicted as an event trigger by assigning it a label t_i. Then, for each trigger d_i, event arguments are assigned to this event trigger by predicting an argument role a_{ij} for all spans s_j in the same sentence as d_i. Unlike most work on event extraction, we consider the realistic setting where gold entity labels are not available. Instead, we use predicted entity mentions as argument candidates.

2.2.2 DyGIE++ Architecture

Figure 2.1 depicts the four-stage architecture. For more details, see [Luan et al., 2019].

Token encoding: DyGIE++ uses BERT for token representations using a "sliding window" approach, feeding each sentence to BERT together with a size-L neighborhood of surrounding sentences.

Span enumeration: Spans of text are enumerated and constructed by concatenating the tokens representing their left and right endpoints, together with a learned span width embedding.

Span graph propagation: A graph structure is generated dynamically based on the model's current best guess at the relations present among the spans in the document. Each span representation $\mathbf{g}_j^t$ is updated by integrating span representations from its neighbors in the graph according to three variants of graph propagation. In coreference propagation, a span's neighbors in the graph are its likely coreference antecedents. In relation propagation, neighbors are related entities within a sentence. In event propagation, there are event trigger nodes and event argument nodes; trigger nodes pass messages to their likely arguments, and arguments pass messages back to their probable triggers. The whole procedure is trained end-to-end, with the model

learning simultaneously how to identify important links between spans and how to share information between those spans.

More formally, at each iteration t the model generates an update $\mathbf{u}_x^t(i)$ for span $s^t \in \mathbb{R}^d$:

$$\mathbf{u}_x^t(i) = \sum_{j \in B_x(i)} V_x^t(i,j) \odot \mathbf{g}_j^t, \tag{2.1}$$

where $\odot$ denotes elementwise multiplication and $V_x^t(i,j)$ is a measure of similarity between spans i and j under task x – for instance, a score indicating the model's confidence that span j is the coreference antecedent of span i. For relation extraction, we use a ReLU activation to enforce sparsity. The final updated span representation $\mathbf{g}_j^{t+1}$ is computed as a convex combination of the previous representation and the current update, with weights determined by a gating function.

Multi-task classification: The re-contextualized representations are input to scoring functions which make predictions for each of the end tasks. We use a two-layer feedforward neural net (FFNN) as the scoring function. For trigger and named entity prediction for span $\mathbf{g}_i$, we compute $\text{FFNN}_{\text{task}}(\mathbf{g}_i)$. For relation and argument role prediction, we concatenate the relevant pair of embeddings and compute $\text{FFNN}_{\text{task}}([\mathbf{g}_i, \mathbf{g}_j])$.

2.3 Experimental Setup

Data We experiment on four different datasets: ACE05, SciERC, GENIA and WLPC. The **ACE05** corpus provides entity, relation, and event annotations for a collection of documents from a variety of domains such as newswire and online forums. For named entity and relation extraction we follow the train / dev / test split from Miwa and Bansal [2016]. Since the ACE data set lacks coreference annotations, we train on the coreference annotations from the OntoNotes dataset [Pradhan et al., 2012]. For event extraction we use the split described in Yang and Mitchell [2016]; Zhang et al. [2019]. We refer to this split as ACE05-E in what follows. The **SciERC** corpus [Luan et al., 2018b] provides entity, coreference and relation annotations from 500 AI paper abstracts. The **GENIA** corpus [Kim et al., 2003] provides entity tags and coreferences for 1999 abstracts from the biomedical research literature with a substantial portion of entities (24%) overlapping some other entity. The **WLPC** dataset provides entity, relation, and event annotations for 622 wet lab

protocols [Kulkarni et al., 2018]. Rather than treating event extraction as a separate task, the authors annotate event triggers as an entity type, and event arguments as relations between an event trigger and an argument.

Evaluation We follow the experimental setups of the respective state-of-the-art methods for each dataset: Luan et al. [2019] for entities and relations, and Zhang et al. [2019] for event extraction. An entity prediction is correct if its label and span matches with a gold entity; a relation is correct if both the span pairs and relation labels match with a gold relation triple. An event trigger is correctly identified if its offsets match a gold trigger. An argument is correctly identified if its offsets and event type match a gold argument. Triggers and arguments are correctly classified if their event types and event roles are also correct, respectively.

Model Variations We perform experiments with the following variants of our model architecture. **BERT + LSTM** feeds pretrained BERT embeddings to a bi-directional LSTM layer, and the LSTM parameters are trained together with task specific layers. **BERT Finetune** uses supervised fine-tuning of BERT on the end-task. For each variation, we study the effect of integrating different task-specific message propagation approaches.

Comparisons For entity and relation extraction, we compare DYGIE++ against the DYGIE system it extends. DYGIE is a system based on ELMo [Peters et al., 2018] that uses dynamic span graphs to propagate global context. For event extraction, we compare against the method of Zhang et al. [2019], which is also an ELMo-based approach that relies on inverse reinforcement learning to focus the model on more difficult-to-detect events.

Implementation Details Our model is implemented using AllenNLP [Gardner et al., 2017]. We use $\text{BERT}_{\text{BASE}}$ for entity and relation extraction tasks and use $\text{BERT}_{\text{LARGE}}$ for event extraction. For BERT finetuning, we use BertAdam with the learning rates of 1×10^{-3} for the task specific layers, and 5.0×10^{-5} for BERT. We use a longer warmup period for BERT than the warmup period for task specific-layers and perform linear decay of the learning rate following the warmup period. Each of the feed-forward neural networks has two hidden layers and ReLU activations and 0.4 dropout. We use 600 hidden units for event extraction and 150 for entity and relation extraction.

2.4 Results and Analyses

State-of-the-art Results Table 2.1 shows test set F1 on the entity, relation and event extraction tasks. Our framework establishes a new state-of-the-art on all three high-level tasks, and on all subtasks except event argument identification. Relative error reductions range from 0.2 - 27.9% over previous state of the art models.

Benefits of Graph Propagation Table 2.2a shows that Coreference propagation (`CorefProp`) improves named entity recognition performance across all three domains. The largest gains are on the computer science research abstracts of SciERC, which make frequent use of long-range coreferences, acronyms and abbreviations. `CorefProp` also improves relation extraction on SciERC.

Relation propagation (`RelProp`) improves relation extraction performance over pretrained BERT, but does not improve fine-tuned BERT. We believe this occurs because all relations are within a single sentence, and thus BERT can be trained to model these relationships well.

Dataset	Task	SOTA	Ours	Δ%
ACE05	Entity	88.4	**88.6**	1.7
	Relation	63.2	**63.4**	0.5
ACE05-Event*	Entity	87.1	**90.7**	27.9
	Trig-ID	73.9	**76.5**	9.6
	Trig-C	72.0	**73.6**	5.7
	Arg-ID	**57.2**	55.4	-4.2
	Arg-C	52.4	**52.5**	0.2
SciERC	Entity	65.2	**67.5**	6.6
	Relation	41.6	**48.4**	11.6
GENIA	Entity	76.2	**77.9**	7.1
WLPC	Entity	79.5	**79.7**	1.0
	Relation	64.1	**65.9**	5.0

Table 2.1: DyGIE++ achieves state-of-the-art results. Test set F1 scores of best model, on all tasks and datasets. We define the following notations for events: *Trig*: Trigger, *Arg*: argument, *ID*: Identification, *C*: Classification. * indicates the use of a 4-model ensemble for trigger detection.

Our best event extraction results did not use any propagation techniques (Table 2.2c). We hypothesize that event propagation is not helpful due to the asymmetry of the relationship between triggers and arguments. Methods to model higher-order interactions among event arguments and triggers represent an interesting direction for future work.

Benefits of Cross-Sentence Context with BERT Table 2.3 shows that both variations of our BERT model benefit from wider context windows. Our model achieves the best performance with a 3-sentence window across all relation and event extraction tasks.

Pre-training or Fine Tuning BERT Under Limited Resources Table 2.2 shows that fine-tuning BERT generally performs slightly better than using the pre-trained BERT embeddings combined with a final LSTM layer.[1] Named entity recognition improves by an average of 0.32 F1 across the four datasets tested, and relation extraction improves by an average of 1.0 F1, driven mainly by the performance gains on SciERC. On event extraction, fine-tuning decreases performance by 1.6 F1 on average across tasks. We believe that this is due to the high sensitivity of both BERT finetuning and event extraction to the choice of optimization hyperparameters – in particular, the trigger detector begins overfitting before the argument detector is finished training.

	ACE05	SciERC	GENIA	WLPC
BERT + LSTM	85.8	69.9	78.4	**78.9**
+RelProp	85.7	70.5	-	78.7
+CorefProp	86.3	**72.0**	78.3	-
BERT Finetune	87.3	70.5	78.3	78.5
+RelProp	86.7	71.1	-	78.8
+CorefProp	**87.5**	71.1	**79.5**	-

(a) F1 scores on NER.

	ACE05	SciERC	WLPC
BERT + LSTM	60.6	40.3	65.1
+RelProp	61.9	41.1	65.3
+CorefProp	59.7	42.6	-
BERT FineTune	**62.1**	44.3	65.4
+RelProp	62.0	43.0	**65.5**
+CorefProp	60.0	**45.3**	-

(b) F1 scores on Relation.

	Entity	Trig-C	Arg-ID	Arg-C
BERT + LSTM	90.5	68.9	**54.1**	**51.4**
+EventProp	**91.0**	68.4	52.5	50.3
BERT FineTune	89.7	**69.7**	53.0	48.8
+EventProp	88.7	68.2	50.4	47.2

(c) F1 scores on ACE05-E.

Table 2.2: Comparison of contextualization methods. All ablations are performed on the dev set except for ACE05-E, where the precedent in the literature is to ablate on test.

Pretrained BERT combined with an LSTM layer and graph propagation stores gradients on 15 million parameters, as compared to the 100 million parameters in $\text{BERT}_{\text{BASE}}$. Since the BERT + LSTM + Propagation approach requires less memory and is less sensitive to the choice of optimization hyperparameters, it may be appealing for non-experts or for researchers working to quickly establish a reasonable baseline under limited resources. It may also be desirable in situations where fine-tuning BERT would be prohibitively slow or memory-intensive, for instance when encoding long documents like scientific articles.

Importance of In-Domain Pretraining We replaced BERT [Devlin et al., 2018] with SciBERT [Beltagy et al., 2019] which is pretrained on a large multi-domain corpus of scientific publications. Table 2.4 compares the results of BERT and SciBERT with the best-performing model configurations. SciBERT

[1] Pre-trained BERT without a final LSTM layer performed substantially worse than either fine-tuning BERT, or using pre-trained BERT with a final LSTM layer.

significantly boosts performance for scientific datasets including SciERC and GENIA. These results indicate that introducing unlabeled text of similar domains for pre-training can significantly improve performance.

Qualitative Analysis To better understand the mechanism by which graph propagation improved performance, we examined all named entities in the SciERC dev set where the prediction made by the BERT + LSTM + `CorefProp` model from Table 2.2a was different from the BERT + LSTM model. We found 44 cases where the `CorefProp` model corrected an error made by the base model, and 21 cases where it introduced an error. The model without `CorefProp` was often overly specific in the label it assigned, labeling entities as *Material* or *Method* when it should have given the more general label *Other Scientific Term*. Figure 2.2 shows an example where span updates passed along a coreference chain corrected an overly-specific entity identification for the acronym "CCRs".

Task	Variation	1	3
Relation	BERT+LSTM	59.3	**60.6**
	BERT Finetune	62.0	**62.1**
Entity	BERT+LSTM	90.0	**90.5**
	BERT Finetune	88.8	**89.7**
Trigger	BERT+LSTM	**69.4**	68.9
	BERT Finetune	68.3	**69.7**
Arg Class	BERT+LSTM	48.6	**51.4**
	BERT Finetune	**50.0**	48.8

Table 2.3: Effect of BERT cross-sentence context. F1 score of relation F1 on ACE05 dev set and entity, arg, trigger extraction F1 on ACE05-E test set, as a function of the BERT context window size.

Coreference propagation updated the span representations of all but one of 44 entities, and in 68% of these cases the update with the largest coreference "attention weight" came from a text span in a different sentence that was itself a named entity.

	SciERC		GENIA
	Entity	Relation	Entity
Best BERT	69.8	41.9	78.4
Best SciBERT	**72.0**	**45.3**	**79.5**

Table 2.4: In-domain pre-training: SciBERT vs. BERT

2.5 Conclusion

In this chapter, we provide an effective plug-and-play IE framework that can be applied to many information extraction tasks. We explore the abilities of BERT embeddings and graph propagation to capture context relevant for these tasks. We find that combining these two approaches improves performance compared to using either one alone, with BERT building robust multi-sentence representations and graph propagations imposing additional structure relevant to the problem and domain under consideration. Future work could

extend the framework to other NLP tasks and explore other approaches to model higher-order interactions like those present in event extraction.

1: This paper summarizes the formalism of **Category Cooccurrence Restrictions (CCRs)** and describes two parsing algorithms that interpret it .

2: **CCRs** are Boolean conditions on the cooccurrence of categories in local trees which allow the statement of generalizations which can not be captured in other current syntax formalisms .

(a) The green span *CCRs* in sentence 2 is updated based on its predicted coreference antecedent.

2: **CCRs** are Boolean conditions on the cooccurrence of categories in local trees which allow the statement of generalizations which can not be captured in other current syntax formalisms .

3: The use of **CCRs** leads to syntactic descriptions formulated entirely with restrictive statements .

(b) The mention of *CCRs* in sentence 2 serves as a bridge to propagate information from sentence 1 to the mention of *CCRs* in sentence 3

(c) Coreference link strength. Red is strong.

Figure 2.2: `CorefProp` enables a correct entity prediction. In each subplot, the green token is being updated by coreference propagation. The preceding tokens are colored according to the strength of their predicted coreference links to the green token. Tokens in **bold** are part of a gold coreference cluster discussing *CCRs*. During the `CorefProp` updates, the span *CCRs* in sentence 2 is updated based on its antecedent *Category Cooccurrence Restrictions*. It then passes this information to the span *CCRs* in sentence 3. As a result, the model changes its prediction for *CCRs* in sentence 3 from *Method* to the correct answer *Other Scientific Term*.

Chapter 3

Scientific Claim Verification: Task Definition and SciFact Dataset Construction

This chapter contains material that was originally published in Wadden et al. [2020].

3.1 Introduction

Due to rapid growth in the scientific literature, it is difficult for researchers – and the general public even more so – to stay up to date on the latest findings. This challenge is especially acute during public health crises like the current COVID-19 pandemic, due to the extremely fast rate at which new findings are reported and the risks associated with making decisions based on outdated or incomplete information. As a result, there is a need for automated tools to assist researchers and the public in evaluating the veracity of scientific claims.

Fact-checking – a task in which the veracity of an

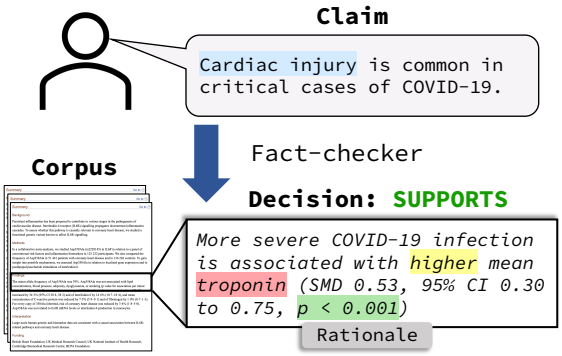


Figure 3.1: A scientific claim, supported by evidence identified by our system. To correctly verify this claim, the system must possess background knowledge that *troponin* is a protein found in cardiac muscle and that elevated levels of *troponin* are a marker of *cardiac injury*. In addition, it must be able to reason about directional relationships between scientific processes: replacing *higher* with *lower* would cause the rationale to REFUTE the claim rather than SUPPORT it. Finally, the system should interpret $p < 0.001$ as an indication that the reported finding is statistically significant.

Claim 1: Lopinavir / ritonavir have exhibited favorable clinical responses when used as a treatment for coronavirus.
Supports: . . . *Interestingly, after lopinavir/ritonavir (Kaletra, AbbVie) was administered, β-coronavirus viral loads significantly decreased and no or little coronavirus titers were observed.*
Refutes: *The focused drug repurposing of known approved drugs (such as lopinavir/ritonavir) has been reported failed for curing SARS-CoV-2 infected patients.* It is urgent to generate new chemical entities against this virus . . .
Claim 2: The coronavirus cannot thrive in warmer climates.
Supports: *...most outbreaks display a pattern of clustering in relatively cool and dry areas...This is because the environment can mediate human-to-human transmission of SARS-CoV-2, and unsuitable climates can cause the virus to destabilize quickly...*
Refutes: *...significant cases in the coming months are likely to occur in more humid (warmer) climates, irrespective of the climate-dependence of transmission and that summer temperatures will not substrantially limit pandemic growth*.

Table 3.1: Evidence identified by our system as supporting and refuting two claims concerning COVID-19.

input *claim* is verified against a corpus of documents that *support* or *refute* the claim – has been studied to combat the proliferation of misinformation in political news, social media, and on the web [Thorne et al., 2018; Hanselowski et al., 2019]. However, verifying scientific claims poses new challenges to both dataset construction and effective modeling. While political claims are readily available on fact-checking websites and can be verified by crowd workers, annotators with extensive domain knowledge are required to generate and verify scientific claims.

In addition, NLP systems for scientific claim verification must possess additional capabilities beyond those required to verify factoid claims. For instance, to verify the claim shown in Figure 3.1, a system must have the ability to access scientific background knowledge, reason over increases and decreases in quantities or measurements, and make sense of specialized statistical language.

In this chapter, we introduce the task of scientific claim verification to evaluate the veracity of scientific claims against a scientific corpus. Table 3.1 presents some examples. To facilitate research on this task, we construct SciFact, an expert-annotated dataset of 1,409 scientific claims accompanied by abstracts that support or refute each claim, and annotated with rationales [Lei et al., 2016] justifying each Supports / Refutes decision. To create the dataset, we develop a novel annotation protocol in which annotators re-formulate naturally occurring claims in the scientific literature – *citation sentences* – into atomic scientific claims. Using citation sentences as a source of claims both speeds the claim generation process and guarantees that the topics discussed in SciFact are representative of the research literature. In addition, citation links indicate the exact documents likely to contain evidence necessary to verify a given claim.

We establish performance baselines on SciFact with an approach similar to DeYoung et al. [2020a],

which achieves strong performance on the FEVER claim verification dataset [Thorne et al., 2018]. Our baseline is a pipeline system which retrieves abstracts related to an input claim, uses a BERT-based [Devlin et al., 2019] sentence selector to identify rationale sentences, and labels each abstract as SUPPORTS, REFUTES, or NOINFO with respect to the claim. We demonstrate that our baseline can benefit from training on claims from domains including Wikipedia articles and politics.

We showcase the ability of our model to verify expert-written claims concerning the novel coronavirus COVID-19 against the newly-released CORD-19 corpus [Wang et al., 2020]. Expert annotators judge retrieved evidence to be plausible for 23 of 36 claims.[1] Our results and analyses demonstrate the importance of the new task and dataset to support significant future research in this domain.

In summary, our contributions include: (1) We introduce and formalize the scientific claim verification task. (2) We develop a novel annotation protocol to generate and verify 1.4K naturally-occurring claims about scientific findings. (3) We establish strong baselines on this task, and identify substantial opportunities for improvement at all stages of the modeling pipeline. (4) We demonstrate the efficacy of our system in a real-world case study verifying claims about COVID-19 against the research literature.

3.2 Background and task definition

As illustrated in Figure 3.1, scientific claim verification is the task of identifying evidence from the research literature that SUPPORTS or REFUTES a given scientific claim. Table 3.1 shows the results of our system applied to claims about the novel coronavirus COVID-19. For each claim, the system identifies relevant scientific abstracts, and labels the relation of each abstract to the claim as either SUPPORTS or REFUTES. Verifying scientific claims is challenging and requires domain-specific background knowledge – for instance, in order to identify the evidence supporting Claim 1 in Table 3.1, the system must determine that a reduction in coronavirus viral load indicates a favorable clinical response, even though this fact is never mentioned.

Scientific claims In SCIFACT, a scientific claim is an *atomic verifiable statement* expressing a finding about one aspect of a scientific entity or process, which can be verified from a single source.[2] For instance, "*The R_0 of the novel coronavirus is 2.5*" is valid, but opinion-based statements like "*The government should require*

[1] We emphasize that our model is a *research prototype* and should not be used to make any medical decisions whatsoever.

[2] Requiring annotators to search multiple sources increases cognitive burden and decreases annotation quality.

people to stand six feet apart to stop coronavirus" are not. Compound claims like "*Aerosolized coronavirus droplets can travel at least 6 feet and can remain in the air for 3 hours*" should be split into two atomic claims.

Claims in SCIFACT are *natural* – they are derived from citation sentences, or *citances* [Nakov et al., 2004], that occur naturally in scientific articles. This is similar to political fact-checking datasets such as UKP Snopes [Hanselowski et al., 2019], which use political fact-checking websites as a source of natural claims. On the other hand, claims in the popular FEVER dataset [Thorne et al., 2018] are *synthetic*, since they are created by annotators by mutating sentences from the Wikipedia articles that will serve as evidence.

Supporting and refuting evidence In most fact-checking work, claims are assigned a global truth label based on the entirety of the available evidence. For example in FEVER, the claim "*Barack Obama was the* 44^{th} *President of the United States*" can be verified using Wikipedia as an evidence source.

While SCIFACT claims are indeed verifiable assertions about scientific findings, accurately assigning a global truth label to a scientific claim (given a fixed scientific corpus) requires a systematic review by a team of experts. In this chapter we focus on the simpler task of assigning SUPPORTS or REFUTES relations to individual *claim-abstract pairs*.

Each SUPPORTS or REFUTES relation between claim and abstract must be justified by at least one *rationale*. A rationale is a minimal collection of sentences which, taken together as premises in the context of the abstract, can reasonably be judged by a domain expert as implying the claim. Rationales facilitate the development of *interpretable models* which not only have the ability to make label predictions, but can also identify the exact sentences that are necessary for their decisions.

3.3 The SCIFACT dataset

The SCIFACT dataset consists of 1,409 scientific claims[3] verified against a corpus of 5,183 abstracts. Abstracts that support or refute each claim are annotated with rationales. We describe our corpus creation and annotation process.

[3] SCIFACT is comparable in size to recent scientific datasets for tasks such as QA (e.g. PubMedQA [Jin et al., 2019] has 1,000 questions), and information extraction (e.g. SciERC [Luan et al., 2018a] has 500 annotated abstracts).

3.3.1 Data source and corpus construction

To construct SciFact, we use S2ORC [Lo et al., 2020a], a publicly-available corpus of millions of scientific articles. To ensure that documents in our dataset are of high quality, we randomly sample articles from a manually curated collection of well-regarded journals spanning domains from basic science (e.g., *Cell*, *Nature*) to clinical medicine (e.g., *JAMA*, *BMJ*). We restrict to articles with at least 10 citations. The resulting collection is referred to as our *seed* set. We use the S2ORC citation graph to sample *source citances* from *citing articles* which cite these seed articles. If a citance cites other articles not in the seed set, we refer to these as *co-cited* articles and add them to the corpus, as depicted in Figure 3.2. The content of the cited abstracts encompasses a diverse array of topics within biomedicine, as shown in Figure 3.3. The majority of citances used for SciFact cite only the seed article (no co-cited articles), as we found in initial annotation experiments that these citances tended to yield specific, easy-to-verify claims.

To expand the corpus, we identify five papers cited in the same paper as each source citance but in a different paragraph, and add these to the corpus as *distractor abstracts*. These abstracts often discuss similar topics to the evidence documents, increasing the difficulty of abstract retrieval and making our metrics more accurately reflect the system's performance on a large research corpus.

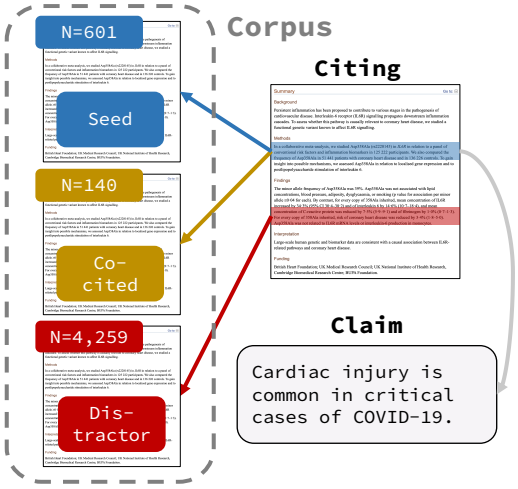


Figure 3.2: Corpus construction. Citing abstracts are identified for each seed document. A claim is written based on the source citance in the citing abstract.

3.3.2 Claim writing

Annotation Annotators are shown a source citance in the context of an article, and are asked to write up to three claims based on the content of the citance. This results in *natural* claims because the annotator does not see the cited article's abstract – the *cited abstract* – at the time of claim writing. Annotators are asked to skip citances that do not make statements about specific scientific findings.

The claim writers included four experts with background in scientific NLP, fifteen undergraduates studying

the life sciences, and four graduate students (doctoral or medical) in the life sciences.

Claim negation Unless the authors of the source citance were mistaken, cited articles should provide supporting evidence for the claims made in a citance. To obtain examples where an abstract REFUTES a claim, an NLP expert wrote negations of existing claims, taking precautions not to bias the negations by using obvious keywords like "not" [Schuster et al., 2019; Gururangan et al., 2018]. In §3.6.1, we demonstrate that a "claim-only" verification model performs poorly, suggesting that the negation process did not introduce severe artifacts.

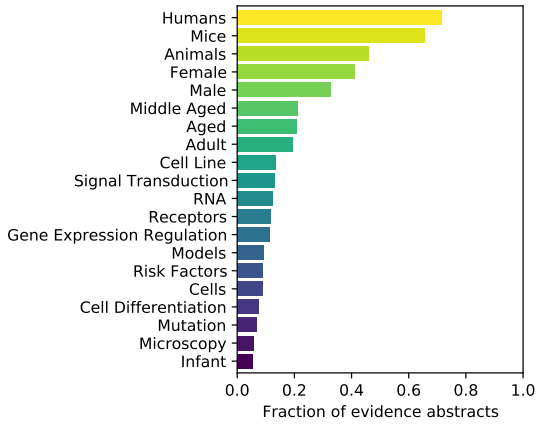


Figure 3.3: Most frequently occurring Medical Subject Headings (MeSH) terms (y-axis) among cited abstracts. MeSH is a controlled vocabulary used for indexing articles in PubMed. Topics range from clinical trial reports ("Humans", "Risk Factors") to molecular biology ("Cell Line", "RNA").

3.3.3 Claim verification

Annotation For each claim, all of the claim's cited abstracts are annotated for evidence. Annotators are shown a single claim - cited abstract pair, and asked to label the pair as SUPPORTS, REFUTES, or NOINFO. Although our task definition allows for a single claim to be both supported and refuted (by different abstracts) – an occurrence we observe on real-world COVID-19 claims (§3.6.3) – this never occurs in our dataset. Each claim has a single label. Counts for each label are shown in Table 3.2a. Overall, the annotators found evidence in 63% of cited abstracts. If the annotator assigns a SUPPORTS or REFUTES label, they must also identify all rationales as defined in §3.2. Table 3.2b provides statistics on the number of sentences per rationale, the number of rationales per claim / abstract pair, and the number of evidence abstracts per claim. No abstract has more than 3 rationales for a given claim, and all rationales consist of at most three sentences. Rationales in SCIFACT are mutually exclusive. 28 rationales contain non-contiguous sentences.

The verifiers included three NLP experts, five life science undergraduates, and five graduate students studying life sciences. Annotators verified claims that they did not write themselves.

SCIFACT claims are verified against abstracts rather than full articles since (1) abstracts can be annotated

more scalably, (2) evidence is found in the abstract in more than 60% of cases, and (3) previous attempts at full-document annotation suffered from low annotator agreement (§3.7).

Quality We assign 232 claim-abstract pairs for independent re-annotation. The label agreement is 0.75 Cohen's κ, comparable with the 0.68 Fleiss' κ reported in Thorne et al. [2018], and 0.70 Cohen's κ reported in Hanselowski et al. [2019]. To measure rationale agreement, we treat each sentence as either classified as "part of a rationale" or "not part of a rationale" and compute sentence-level agreement. The resulting Cohen's κ is 0.71.

Fold	SUPPORTS	NOINFO	REFUTES	All
Train	332	304	173	809
Dev	124	112	64	300
Test	100	100	100	300
All	556	516	337	1409

(a) Distribution of claim labels in SCIFACT.

	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

(b) Evidence counts at various levels of granularity. For example, Column 2 of the row "Rationales / abstract" indicates that 290 claim / abstract pairs are supported by 2 distinct rationales.

Table 3.2: Statistics on claim labels, and the number of evidence abstracts and rationales per claim.

3.4 The SCIFACT task

Task Formulation The inputs to our task are a scientific claim c and a corpus of abstracts $\mathcal{A}$. All abstracts $a \in \mathcal{A}$ are labeled as $y(c, a) \in \{\text{SUPPORTS}, \text{REFUTES}, \text{NOINFO}\}$ with respect to a claim c. The abstracts that either SUPPORT or REFUTE c are referred to as *evidence abstracts* for c, denoted as $\mathcal{E}(c)$. Each evidence abstract $a \in \mathcal{E}(c)$ is annotated with rationales. A single rationale R_i is a collection of sentences $\{r_1(c, a), \ldots, r_m(c, a)\}$, where m is the number of sentences in rationale R_i. We denote the set of all rationales as $\mathcal{R}(c, a) = \{R_1(c, a), \ldots, R_n(c, a)\}$.

Given a claim c and a corpus $\mathcal{A}$, the system must predict a set of evidence abstracts $\widehat{\mathcal{E}}(c)$. For each abstract $a \in \widehat{\mathcal{E}}(c)$, it must predict a label $\widehat{y}(c, a)$, and a collection of *rationale sentences* $\widehat{S}(c, a) = \{\widehat{s}_1(c, a), \ldots, \widehat{s}_\ell(c, a)\}$. Note that although the gold annotations may contain multiple separate rationales, to simplify the prediction task we only require the model to predict a single collection of *rationale sentences*; these sentences may encompass multiple gold rationales.

Task Evaluation We evaluate the task at two levels of granularity. For *abstract-level* evaluation, we assess the model's ability to identify the abstracts that support or refute the claim. For *sentence-level* evaluation,

we evaluate the model's performance at identifying the sentences sufficient to justify the abstract-level predictions. We conduct evaluations in both the "Open" FEVER-style [Thorne et al., 2018] setting where the evidence abstracts must be retrieved, and the "Oracle abstract" ERASER-style [DeYoung et al., 2020a] setting where the gold evidence abstracts $\mathcal{E}(c)$ are provided.

Abstract-level evaluation is inspired by the FEVER score. Given a claim c, a predicted evidence abstract $a \in \widehat{\mathcal{E}}(c)$ is *correctly labeled* if (1) a is a gold evidence abstract for c, and (2) The predicted label is correct: $\widehat{y}(c, a) = y(c, a)$. It is *correctly rationalized* if, in addition, the predicted rationale sentences contain a gold rationale, i.e., there exists some gold rationale $R_i(c, a) \subseteq \widehat{S}(c, a)$.

Like FEVER, which limits the maximum number of predicted rationale sentences to five, SCIFACT limits to three predicted rationale sentences. Overall performance is measured by the micro-F1 of the precision and recall over the correctly-labeled and correctly-rationalized evidence abstracts. We refer to these evaluations as $\text{Abstract}_{\text{Label-Only}}$ and $\text{Abstract}_{\text{Label+Rationale}}$, respectively.

Sentence-level evaluation measures performance in identifying individual rationale sentences. Unlike the abstract-level metrics, this evaluation penalizes the prediction of extra rationale sentences.

A predicted rationale sentence $\widehat{s}(c, a)$ is correctly *selected* if (1) It is a member of some gold rationale $R_i(c, a)$, (2) all other sentences from the same gold rationale $R_i(c, a)$ are among the predicted $\widehat{S}(c, a)$, and (3) $\widehat{y}(c, a) \neq$ NOINFO[4]. It is correctly *labeled* if, in addition, the abstract a is correctly labeled: $\widehat{y}(c, a) = y(c, a)$.

Overall performance is measured by the micro-F1 of the precision and recall of correctly-selected and correctly-labeled rationale sentences, denoted $\text{Sentence}_{\text{Selection-Only}}$ and $\text{Sentence}_{\text{Selection+Label}}$. For sentence-level evaluation, we do not limit the number of predicted rationale sentences, since the evaluation penalizes models that over-predict.

[4]Condition (3) eliminates rationale sentences which were identified by the rationale selector, but proved insufficient to justify a final SUPPORTS / REFUTES decision

3.5 VERISCI: Baseline model

We develop a baseline (referred to as VERISCI) that takes a claim c and corpus $\mathcal{A}$ as input, identifies evidence abstracts $\widehat{\mathcal{E}}(c)$, and predicts a label $\widehat{y}(c, a)$ and rationale sentences $\widehat{S}(c, a)$ for each $a \in \widehat{\mathcal{E}}(c)$. Following the "BERT-to-BERT" model presented in DeYoung et al. [2020a]; Soleimani et al. [2019], VERISCI is a pipeline of three components:

1. ABSTRACTRETRIEVAL retrieves k abstracts with highest TF-IDF similarity to the claim.
2. RATIONALESELECTION identifies rationale sentences $\widehat{S}(c, a)$ for each abstract.
3. LABELPREDICTION makes the final label prediction $\widehat{y}(c, a)$.

Rationale selection Given a claim c and abstract a, we train a model to predict $z_i \triangleq \mathbb{1}[a_i \text{ is a rationale sentence}]$ for each sentence a_i in a. For each sentence, we encode the concatenated sequence $w_i = [a_i, \text{SEP}, c]$ using a BERT-style language model and predict a score $\tilde{z}_i = \sigma[f(\text{CLS}(w_i))]$, where σ is the sigmoid function, f is a linear layer and $\text{CLS}(w_i)$ is the CLS token from the encoding of w_i. We train the model on pairs of claims and their cited abstracts and minimize cross-entropy loss between z_i and $\tilde{z}_i$. For each claim, we use cited abstracts labeled NOINFO, as well as non-rationale sentences from abstracts labeled SUPPORTS and REFUTES as negative examples. To make predictions, we select all sentences a_i with $\tilde{z}_i > t$ as rationale sentences, where $t \in [0, 1]$ is tuned on the dev set.

Label prediction Sentences identified by the rationale selector are passed to a separate BERT-based model to make the final labeling decision. Given a claim c and abstract a, we concatenate the claim and the predicted rationale sentences $u = [\widehat{s}_1(c, a), \ldots \widehat{s}_\ell(c, a), \text{SEP}, c]$[5], and predict $\tilde{y}(c, a) = \phi[f(\text{CLS}(u))]$, where ϕ is the softmax function, and f is a linear layer with three outputs representing the {SUPPORTS, REFUTES, NOINFO } labels. We minimize the cross-entropy loss between $\tilde{y}(c, a)$ and the true label $y(c, a)$.

We train the model on pairs of claims and their cited abstracts using gold rationales as input. For cited abstracts labeled NOINFO, we choose the k sentences from the cited abstract with highest TF-IDF similarity to the claim as input rationales. For prediction, we use the predicted rationale sentences $\widehat{S}(c, a)$ as input and predict $\hat{y}(c, a) = \text{argmax } \tilde{y}(c, a)$. NOINFO is predicted for abstracts with no rationale sentences.

[5] We truncate the rationale input if it exceeds the BERT token limit. c is never truncated.

We experimented with a label prediction model which encodes entire abstracts via the Longformer [Beltagy et al., 2020], and makes predictions using the document-level CLS token. Performance was not competitive with our pipeline setup, likely because the label predictor struggles to identify relevant information when given full abstracts.

3.6 Experiments

In our experiments, we (1) analyze the performance of each individual component of VERISCI, (2) evaluate full task performance in both the "Oracle abstract" and "Open" settings, (3) present promising results verifying claims about COVID-19 using VERISCI, and (4) discuss some modeling challenges presented by the dataset.

3.6.1 Pipeline components

We examine the effects of different training datasets, sentence encoders, and model inputs on the performance of the RATIONALESELECTION and LABELPREDICTION modules. The RATIONALESELECTION module is evaluated on its ability to select rationale sentences given gold abstracts[6]. The LABELPREDICTION module is evaluated on its 3-way label classification accuracy given gold rationales from cited abstracts. Cited abstracts labeled NOINFO are included in the evaluation. These abstracts have no gold rationale sentences; as in §3.5, we provide the k most similar sentences from the abstract as input.

Training Data We train on (1) FEVER, (2) UKP Snopes, (3) SCIFACT, and (4) FEVER pretraining followed by SCIFACT fine-tuning. RoBERTa-large [Liu et al., 2019] is used as the sentence encoder.

Sentence encoder We fine-tune SCIBERT [Beltagy et al., 2019], BioMedRoBERTa [Gururangan et al., 2020a], RoBERTa-base, and RoBERTa-large. SCIFACT is used as training data.

Model Inputs We examine the performance of "claim-only" and "abstract-only" models trained on SCIFACT, using RoBERTa-large as the sentence encoder. The claim-only model makes label predictions based on

[6]Our FEVER-trained RATIONALESELECTION module achieves 79.9 sentence-level F1 on the FEVER test set, virtually identical to 79.6 reported in DeYoung et al. [2020a].

			Sentence-level						Abstract-level					
			Selection-Only			Selection+Label			Label-Only			Label+Rationale		
Retrieval	Model		P	R	F1	P	R	F1	P	R	F1	P	R	F1
Oracle abstract	Oracle rationale	1	100.0	80.5	$89.2_{2.1}$	89.6	72.2	$79.9_{3.0}$	90.1	77.5	$83.3_{2.4}$	90.1	77.5	$83.3_{2.4}$
	Zero-shot	2	42.5	45.1	$43.8_{2.0}$	36.1	38.4	$37.2_{2.3}$	86.9	53.6	$66.3_{3.1}$	67.9	41.9	$51.8_{3.4}$
	VERISCI	3	76.1	63.8	$69.4_{2.6}$	66.5	55.7	$\mathbf{60.6}_{3.1}$	87.3	65.3	$74.7_{2.8}$	84.9	63.5	$\mathbf{72.7}_{2.9}$
Open	Oracle rationale	4	100.0	56.5	$72.2_{3.3}$	87.6	49.5	$63.2_{3.7}$	88.9	54.1	$67.2_{3.2}$	88.9	54.1	$67.2_{3.2}$
	Zero-shot	5	28.7	37.6	$32.5_{2.3}$	23.7	31.1	$26.9_{2.3}$	56.0	42.3	$48.2_{3.3}$	42.3	32.0	$36.4_{3.3}$
	VERISCI	6	45.0	47.3	$46.1_{3.0}$	38.6	40.5	$\mathbf{39.5}_{3.0}$	47.5	47.3	$47.4_{3.1}$	46.6	46.4	$\mathbf{46.5}_{3.1}$

Table 3.4: Test set performance on SCIFACT, according to the metrics from §3.4. For the "Oracle abstract" rows, the system is provided with gold evidence abstracts. "Oracle rationale" rows indicate that the gold rationales are provided as input. "Zero-shot" indicates zero-shot performance of a verification system trained on FEVER. Additionally, standard deviations are reported as subscripts for all F1 scores.

the claim text alone, without access to evidence abstracts. The abstract-only model selects rationale sentences and makes label predictions without access to the claim.

Results The results are shown in Table 3.3. For LABELPREDICTION, the best performance is achieved by training first on the large FEVER dataset and then fine-tuning on the smaller in-domain SCIFACT training set. To understand the benefits of FEVER pretraining, we examined the claim / evidence pairs where the FEVER + SCIFACT- trained model made correct predictions but the SCIFACT- trained model did not. In 36 / 44 of these cases, the SCIFACT- trained model predicts NOINFO. Thus pretraining on FEVER appears to improve the model's ability to recognize textual entailment relationships between evidence and claim – particularly relationships indicated by non-domain-specific cues like "is associated with" or "has an important role in".

	RATIONAL-SELECT.			LABEL-PRED.
Training data	P	R	F1	ACC.
FEVER	41.5	57.9	48.4	67.6
UKP Snopes	42.5	62.3	50.5	71.3
SCIFACT	73.7	70.5	**72.1**	75.7
FEVER + SCIFACT	72.4	67.2	69.7	**81.9**
Sentence encoder	P	R	F1	ACC.
SCIBERT	74.5	74.3	**74.4**	69.2
BioMedRoBERTa	75.3	69.9	72.5	71.7
RoBERTa-base	76.1	66.1	70.8	62.9
RoBERTa-large	73.7	70.5	72.1	**75.7**
Model inputs	P	R	F1	ACC.
Claim-only	-	-	-	44.5
Abstract-only	60.1	60.9	60.5	53.3

Table 3.3: Comparison of different training datasets, encoders, and model inputs for RATIONALESELECTION and LABELPREDICTION, evaluated on the SCIFACT dev set. The claim-only model cannot select rationales.

For RATIONALESELECTION, training on SCIFACT alone produces the best results. We examined the rationales that the SCIFACT- trained model identified but the FEVER- trained model missed, and found that

they generally contain science-specific vocabulary. Thus, training on additional out-of-domain data provides little benefit.

RoBERTa-large exhibits the strongest performance on label prediction, while SciBERT has a slight edge on rationale selection. The "claim-only" model exhibits very poor performance, which provides some reassurance that the claim negation procedure described in §3.3.2 does not introduce obvious statistical artifacts. Similarly, the poor performance of the "abstract-only" model indicates that the model needs access to the claim being verified in order to identify relevant evidence.

3.6.2 Full task

Experimental setup Based on the results from §3.6.1, we use the RationaleSelection module trained on SciFact only, and the LabelPrediction module trained on Fever + SciFact for our final end-to-end system VeriSci. Although SciBERT performs slightly better on rationale selection, using RoBERTa-large for both RationaleSelection and LabelPrediction gave the best full-pipeline performance on the dev set, so we use RoBERTa-large for both components. For the AbstractRetrieval module, the best dev set full-pipeline performance was achieved by retrieving the top $k = 3$ documents.

Model comparisons We report performance of three model variants. For the "Oracle rationale" setting, the RationaleSelection module is replaced by an oracle which outputs gold rationales for correctly retrieved documents, and no rationales for incorrect retrievals. The "Zero-shot" setting reports the zero-shot generalization performance of a model trained on Fever (the results on UKP Snopes were slightly worse). VeriSci reports the performance of our best system.

Results The results are shown in Table 3.4. In the oracle abstract setting, the abstract-level F1 scores are roughly comparable to label classification accuracies, and the $\text{Abstract}_{\text{Label+Rationale}}$ score in Row 3 implies an end-to-end classification accuracy of roughly 70%, given gold abstracts.

Access to in-domain data during training clearly improves performance. Despite the small size of SciFact, training on these data leads to relative improvements of 47% on open $\text{Sentence}_{\text{Selection+Label}}$, and 28% on open $\text{Abstract}_{\text{Label+Rationale}}$ over Fever alone (Row 6 vs. Row 5). The three pipeline components make similar contributions to the overall model error. Replacing RationaleSelection with an oracle leads

Reasoning type	Example	
Science background	**Claim:**	Rapamycin slows aging in fruit flies.
	Evidence:	*. . . feeding rapamycin to adult Drosophila produces life span extension . . .*
	Gold Verdict:	SUPPORTS
	Reasoning:	Drosophila is a type of fruit fly.
Directionality	**Claim:**	Inhibiting glucose-6-phosphate dehydrogenase impairs lipogenesis
	Evidence:	*. . . suppression of 6PGD increased lipogenesis*
	Gold Verdict:	REFUTES
	Reasoning:	A decrease (not increase) in lipogenesis would indicate lipogenesis impairment.
Numerical reasoning	**Claim:**	Bariatric surgery improves resolution of diabetes.
	Evidence:	*Strong associations were found between bariatric surgery and the resolution of T2DM, with a HR of 9.29 (95% CI 6.84-12.62)...*
	Gold Verdict:	SUPPORTS
	Reasoning:	A HR (hazard ratio) that is greater than 1 with 95% confidence indicates improvement.
Cause and effect	**Claim:**	Major vault protein (MVP) functions to decrease tumor aggression.
	Evidence:	*Knockout of MVP leads to miR-193a accumulation...inhibiting tumor progression*
	Gold Verdict:	REFUTES
	Reasoning:	Knocking out (removing) MVP inhibits tumor progression → MVP increases tumor aggression.
Coreference	**Claim:**	Low saturated fat diets have adverse effects on the development of infants
	Evidence:	*Neurological development of children in the intervention group was at least as good as ... the control group*
	Gold Verdict:	REFUTES
	Reasoning:	The intervention group in this study was placed on a low saturated fat diet.

Table 3.5: Reasoning types required to verify SCIFACT claims which are classified incorrectly by our modeling baseline. Words crucial for correct verification are highlighted.

to a roughly 20-point rise in $\text{Sentence}_{\text{Selection+Label}}$ F1 (Row 6 vs. Row 4). Replacing ABSTRACTRETRIEVAL with an oracle as well leads to a gain of roughly 20 more points (Row 4 vs. Row 1).

Nearly all correctly-labeled abstracts are supported by at least one rationale. There is only a two-point difference in F1 between $\text{Abstract}_{\text{Label-Only}}$ and $\text{Abstract}_{\text{Label+Rationale}}$ in the oracle setting (Row 3), and a one-point difference in the open setting (Row 6). The differences between $\text{Sentence}_{\text{Selection-Only}}$ and $\text{Sentence}_{\text{Selection+Label}}$ are larger, caused by examples where the model finds the evidence but fails to predict its relationship to the claim. We examine these in §3.6.4.

We evaluate the statistical robustness of our results by generating 10,000 bootstrap-resampled versions of the test set [Dror et al., 2018] and computing the standard deviation of all performance metrics. Table 3.4 shows the standard deviations in F1 score. The results indicate that the observed differences in model performance are statistically robust and cannot be attributed to random variation in the dataset.

3.6.3 Verifying claims about COVID-19

We conduct exploratory experiments using our system to verify claims concerning COVID-19. We tasked a medical student to write 36 COVID-related claims. For each claim c, we used VERISCI to predict evidence abstracts $\widehat{\mathcal{E}}(c)$. The annotator examined each $(c, \widehat{\mathcal{E}}(c))$ pair. A pair was labeled *plausible* if $\widehat{\mathcal{E}}(c)$ was nonempty, and at least half of the evidence abstracts in $\widehat{\mathcal{E}}(c)$ were judged to have reasonable rationales and labels. For 23 / 36 claims, the response of VERISCI was deemed plausible by our annotator, demonstrating that VERISCI is able to successfully retrieve and classify evidence in many cases. Two examples are shown in Table 3.1. In both cases, our system identifies both supporting *and* refuting evidence.

3.6.4 Error analysis

To better understand the errors made by VERISCI, we conduct a manual analysis of test set predictions where an evidence abstract was correctly retrieved, but where the model failed to identify any relevant rationales or predicted an incorrect label. We identify five modeling capabilities required to correct these mistakes (Table 3.5 provides examples):

- **Science background** includes knowledge of domain-specific lexical relationships.
- **Directionality** requires understanding increases or decreases in scientific quantities.
- **Numerical reasoning** involves interpreting numerical or statistical findings.
- **Cause and effect** requires reasoning about counterfactuals.
- **Coreference** involves drawing conclusions using context stated outside of a rationale sentence.

3.7 Related work

Fact checking and rationalized NLP models Fact-checking datasets include PolitiFact [Vlachos and Riedel, 2014], Emergent [Ferreira and Vlachos, 2016], LIAR [Wang, 2017], SemEval 2017 Task 8 RumorEval [**?**], Snopes [Popat et al., 2017], CLEF-2018 CheckThat! [Barrón-Cedeño et al., 2018], Verify [Baly et al., 2018], Perspectrum [Chen et al., 2019], FEVER [Thorne et al., 2018], and UKP Snopes [Hanselowski et al.,

2019]. Hanselowski et al. [2019] provides a thorough review. To our knowledge, there are no existing data sets for scientific claim verification. We refer to our task as "claim verification" rather than "fact-checking" to emphasize that our focus is to help researchers make sense of scientific findings, not to counter disinformation.

Fact-checking is one of a number of tasks where a model is required to justify a prediction via *rationales* from the source document. The ERASER dataset [DeYoung et al., 2020a] provides a suite of benchmark datasets (including SCIFACT) for evaluating rationalized NLP models.

Related scientific NLP tasks The *citation contextualization* task [Cohan et al., 2015; Jaidka et al., 2017] is to identify spans in a cited document that are relevant to a particular citation in a citing document. Unlike SCIFACT, these citations are not re-written into atomic claims and are therefore more difficult to verify. Expert annotators achieved very low (21.7%) inter-annotator agreement on the BioMedSumm dataset [Cohen et al., 2014], which contains 314 citations referencing 20 papers.

Biomedical question answering datasets include BioASQ [Tsatsaronis et al., 2015] and PubMedQA [Jin et al., 2019], which contain 855 and 1,000 "*yes / no*" questions respectively [Gu et al., 2020]. Claim verification and question answering are both-knowledge intensive tasks which require an understanding of the relationship between an input query and relevant supporting text.

Automated evidence synthesis [Marshall and Wallace, 2019; Beller et al., 2018; Tsafnat et al., 2014; Marshall et al., 2017] seeks to automate the process of creating *systematic reviews* of the medical literature[7] – for instance, by extracting PICO snippets [Nye et al., 2018] and inferring the outcomes of clinical trials [Lehman et al., 2019; DeYoung et al., 2020b]. We hope that systems for claim verification will serve as components in future evidence synthesis frameworks.

3.8 Conclusion and future work

Claim verification allows us to trace the sources and measure the veracity of scientific claims. These abilities have emerged as particularly important in the context of the current pandemic, and the broader reproducibility crisis in science. In this article, we formalize the task of scientific claim verification, and release a dataset (SCIFACT) and models (VERISCI) to support work on this task. Our results indicate that it is possible to train

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

models for scientific fact-checking and deploy them with reasonable efficacy on real-world claims related to COVID-19.

Scientific claim verification presents a number of promising avenues for research on models capable of incorporating background information, reasoning about scientific processes, and assessing the strength and provenance of various evidence sources. This last challenge will be especially crucial for future work that seeks to verify scientific claims against sources other than the research literature – for instance, social media and the news. We hope that the resources presented in this chapter encourage future research on these important challenges, and help facilitate progress toward the broader goal of scientific document understanding.

Chapter 4

Full-Document Scientific Claim Verification with MULTIVERS

This chapter contains material that was originally published in Wadden et al. [2022c].

4.1 Introduction

The proliferation of scientific mis- and dis-information on the web has motivated the study of scientific claim verification (Chapter 3), the subsequent release of a number of additional datasets [Saakyan et al., 2021; Sarrouti et al., 2021; Kotonya and Toni, 2020], and the development of modeling approaches [Pradeep et al., 2021; Li et al., 2021; Zhang et al., 2021]. The goal of the task is to verify a given scientific claim by labeling scientific research abstracts which SUPPORT or REFUTE the claim, and to select evidentiary sentences (or *rationales*) reporting the findings which justify each label.

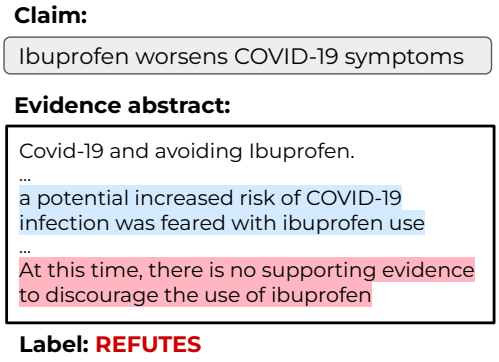


Figure 4.1: A claim from the HealthVer dataset, refuted by a research abstract. The sentence in red is a *rationale* reporting a finding that REFUTES the claim. However, this finding cannot be interpreted properly without the context in blue, which specifies that the finding applies to Ibuprofen as a treatment for COVID. MULTIVERS incorporates the full context of the evidence-containing abstract when predicting fact-checking labels.

A common approach to this task is to first extract rationales from the larger document context, and then make label predictions conditioned on the selected rationales. This "extract-then-label" approach has two important drawbacks, which we aim to address in this chapter. First, the rationales may lack information required to make a prediction when taken out-of-context; for instance, they may contain acronyms or unresolved coreferences, or lack qualifiers that specify the scope of a reported finding (Figure 4.1 provides an example). Second, the "extract-then-label" approach requires training data annotated with both sentence-level rationales and abstract-level labels. While sentence-level rationale annotations are costly and require trained experts, abstract-level labels can be created cheaply using high-precision heuristics, e.g., the titles of research papers sometimes make claims that are supported by their abstracts.

Motivated by these challenges, we introduce MULTIVERS (**Multi**task **Ver**ification for **S**cience): Given a claim and evidence-containing scientific abstract, MULTIVERS creates a shared encoding of the entire claim / abstract context, using the Longformer encoder [Beltagy et al., 2020] to accommodate long sequences. Then, it predicts an abstract-level fact-checking label and sentence-level rationales in a multitask fashion, enforcing consistency between the outputs of the two tasks during decoding. This modeling approach ensures that label predictions are made based on all available context, and enables training on instances derived via weak supervision for which abstract-level labels are available, but sentence-level rationales are not.

In experiments on three scientific claim verification datasets, we find that MULTIVERS outperforms two state-of-the-art baselines, one of which has more than 10x the parameters of our system. In addition, we show that training MULTIVERS on weakly-labeled in-domain data substantially improves performance in the zero / few-shot domain adaptation settings. The ability to achieve reasonable performance given limited labeled data is especially valuable in specialized domains, due to the high cost of collecting expert annotations.

In summary, our contributions are as follows:

1. We introduce MULTIVERS, a multitask system for full-context scientific claim verification. MULTIVERS improves fully-supervised fact-verification performance by an average of 11% on three datasets over two state-of-the-art baselines, with improvements of 14% and 26% in the few-shot and zero-shot settings.
2. We present weak supervision heuristics to assign fact-checking labels to two large scientific datasets, and show that training on these annotations more than doubles zero-shot domain adaptation performance.
3. Through ablations and analysis, we demonstrate that our multitask modeling approach achieves our goals

of incorporating full-document context into label predictions, and facilitating zero / few-shot domain adaptation.

4.2 Background

4.2.1 The scientific claim verification task

Task definition We study the scientific claim verification task as defined in Chapter 3.4 of this thesis. One detail will be important in this chapter: *rationales* for scientific claim verification may not be self-contained, and may require additional context from elsewhere in the abstract to resolve coreferential expressions or acronyms, or to determine qualifiers specifying experimental context or study population.[1] An example of this situation is provided in Figure 4.1.

Other works have cast scientific claim verification as a sentence-level natural language inference (NLI) task; in §4.4.1, we describe how we process these datasets to be compatible with the task as considered in this chapter.

Evaluation In Chapter 3.4, we defined four evaluation metrics. We subsequently found that two of these metrics are sufficient to convey the important findings for our experiments: (1) *abstract-level label-only* evaluation rewards recall in identifying relevant abstracts, and *Sentence-level selection+label* evaluation rewards precision in selecting rationales. In this chapter, we refer to these two metrics as "abstract" and "sentence" evaluation respectively.

Retrieval settings For *open* scientific claim verification, the system must retrieve candidate abstracts from a corpus of documents. In the *abstract-provided* setting, candidate abstracts for each claim are given as input. We describe the retrieval settings for all datasets in §4.4.1.

Supervision settings We consider three supervision settings. In the *zero-shot domain adaptation* setting, models may not train on any in-domain fact-checking data, though they may train on general-domain fact-checking data and other available scientific datasets. In the *few-shot domain adaptation* setting, models may

[1] This convention is consistent with related tasks in rationalized NLP for biomedical literature, such as Lehman et al. [2019] and DeYoung et al. [2020b].

train on 45 claims from the target dataset. In the *fully-supervised* setting, models may train on all claims from the target dataset.

While most existing work on scientific fact-checking has focused on the fully-supervised setting, some recent work has examined the zero-shot setting. Lee et al. [2021] use language model perplexity as a measure of claim veracity. Wright et al. [2022] generate claims based on citation sentences, and verify each generated claim against the abstracts mentioned in the claim's source citation. Given the high potential impact of fact verification systems for specialized domains, combined with the substantial cost of creating these datasets, we believe that the development of techniques for zero / few-shot domain adaptation represents an important area for continued research.

4.2.2 Scientific claim verification datasets

Between the release of SCIFACT (Chapter 3) and the work presented in this chapter, a number of additional scientific claim verification datasets were released. COVID-Fact [Saakyan et al., 2021] and HealthVer [Sarrouti et al., 2021] verify COVID-19 claims against scientific literature. PUBHEALTH [Kotonya and Toni, 2020] verifies public health claims against news and web sources. CLIMATE-FEVER [Diggelmann et al., 2020] verifies claims about climate change against Wikipedia. In this chapter, our focus is verifying claims against scientific literature. We therefore perform experiments on the COVID-Fact, HealthVer, and SCIFACT datasets. Preprocessing details and summary statistics for these datasets are included in §4.4.1.

4.2.3 Models

Motivated in part by the SCIVER shared task [Wadden and Lo, 2021] and leaderboard, a number of models have been developed for SCIFACT (the focus of the shared task). The two strongest systems on the shared task were VERT5ERINI [Pradeep et al., 2021] and PARAGRAPHJOINT [Li et al., 2021], which we adopt as baselines. More recently, ARSJOINT [Zhang et al., 2021] achieved performance competitive with these two systems.[2]

Given a claim c and candidate abstract a, these models make predictions in two steps. First, they predict rationales $\widehat{R}(c, a) = \{\widehat{r}_1(c, a), \ldots, \widehat{r}_n(c, a)\}$ likely to contain evidence. Then, they make a label

[2]Recent progress can be found on the SciFact leaderboard.

prediction $\widehat{y}(c, f_R(\widehat{R}(c, a))$ based on the claim and predicted rationales, where f_R is a function which creates a representation of the predicted rationales.

While existing models share this general approach, they use different functions f_R to construct rationale representations. For VERT5ERINI, rationale selection and label prediction are performed by two separate T5-3B models, and f_R concatenates the text of the selected rationales. As a result, the label predictor may not have access to all context needed to make a correct label prediction. PARAGRAPHJOINT and ARSJOINT attempt to address this issue by encoding the claim and full abstract (truncating to 512 tokens), and using these representations as the basis for both rationale selection and label prediction. The function f_R consists of self-attention layers over the (globally-contextualized) token representations of the predicted rationales. Thus, PARAGRAPHJOINT and ARSJOINT can incorporate abstract-level context into label decisions. However, the mechanism by which this occurs is more complex than for our proposed system and requires rationale supervision for all training instances.

4.3 The MULTIVERS model

We propose the MULTIVERS model for full-context claim verification. In §4.3.1, we describe our modeling approach. Rather than predicting rationales $\widehat{R}(c, a)$ followed by the overall fact-checking label $\widehat{y}(c, f_R(\widehat{R}(c, a)))$, we predict $\widehat{y}(c, a)$ directly based on an encoding of the entire claim and abstract, and enforce consistency of $\widehat{R}(c, a)$ with $\widehat{y}(c, a)$ during decoding. A similar idea has been shown to be effective on sentiment analysis and propaganda detection with token-level rationales [Pruthi et al., 2020]. In §4.3.2, we explain how our approach facilitates few-shot domain adaptation using weakly-labeled scientific documents.

4.3.1 Full-context claim verification

Long-document encoding Given a claim c and candidate abstract a consisting of title t and sentences $s_1, \ldots, s_n$, we concatenate the inputs separated by <s/> tokens. The <s/> token following each sentence s_i is notated as $</\text{s}>_i$:

$$<\text{s}>\ c\ <\text{s}/>\ t\ <\text{s}/>\ s_1\ </\text{s}>_1\ \ldots s_n\ </\text{s}>_n$$

The model input sometimes exceeds the 512-token limit common to transformer-based language models like BERT [Devlin et al., 2019] and RoBERTa [Liu et al., 2019]; see Table 4.1 for details on how frequently this occurs. Therefore, we use the Longformer model [Beltagy et al., 2020] as our encoder. We assign global attention to the <s> token, as well as all tokens in c and all </s> tokens.

Multitask rationale selection and label prediction Given the full-context Longformer encoding, we predict whether sentence s_i is a rationale via a binary classification head, consisting of two feedforward layers followed by a two-way softmax, on top of the globally-contextualized token </s>$_i$.

Similarly, we predict the overall fact-checking label $\widehat{y}(c, a)$ by adding a three-way classification head over the encoding of the <s> token. Since the <s> token is trained with global attention, the model makes predictions based on a representation of the entire claim and abstract.

During training, we compute the cross-entropy losses for the label and rationale predictions, and train to minimize the multitask loss:

$$L = L_{\text{label}} + \lambda_{\text{rationale}} L_{\text{rationale}} \tag{4.1}$$

where $\lambda_{\text{rationale}}$ is tuned on the dev set.

At inference time, we first predict $\widehat{y}(c, a)$ to be the label with the highest softmax score. If the predicted label is NEI, we predict no rationales. If the predicted label is either SUPPORTS or REFUTES, then we predict rationales as all sentences with an assigned softmax score of greater than 0.5. If no sentences have a rationale softmax over 0.5, then we predict the highest-scoring sentence as the sole rationale. In §4.6.2, we show that this ability to condition the rationale predictions on the label prediction (as opposed to conditioning the label on the predicted rationales) leads to substantial improvement in the zero-shot domain adaptation setting.

Candidate abstract retrieval For datasets that require retrieval of candidate abstracts, we rely on the VERT5ERINI [Pradeep et al., 2021] retrieval system, which achieved state-of-the-art performance on the SCIVER shared task (SCIVER used the SCIFACT dataset for evaluation). This model first retrieves abstracts using BM25 [Robertson and Zaragoza, 2009], then refines the predictions using a neural re-ranker based on Nogueira et al. [2020], which is trained on the MS MARCO passage dataset [Campos et al., 2016].

Dataset	Domain	Claim source	Open	Has NEI	Claim complexity	Negation method	Train claims	Eval claims	> 512 tokens
HealthVer	COVID	TREC-COVID	✗	✓	Complex	Natural	1,622	230	14.9%
COVID-Fact	COVID	Reddit	✗	✗	Complex	Automatic	903	313	12.4%
SciFact	Biomed	Citations	✓	✓	Atomic	Human	1,109	300	27.4%
Fever	Wiki	Wikipedia	-	✓	Atomic	Human	130,644	-	33.2%
PubMedQA	Biomed	Paper titles	-	✓	Complex	Automatic	58,370	-	12.1%
EvidenceInference	Biomed	ICO prompts	-	✓	Atomic	Automatic	7,395	-	42.7%

Table 4.1: Summary of datasets used in experiments. The top group of datasets are scientific claim verification datasets, and the bottom group are for pretraining. Datasets with a ✓for "Open" require that candidate abstracts be retrieved from a corpus; those with a ✗provide candidate abstracts as input. Datasets with a ✓for "Has NEI" require three-way (SUPPORTS / REFUTES / NEI) label prediction, while those with an ✗are (SUPPORTS / REFUTES) only. The "> 512 tokens" column indicates the percentage of claim / abstract contexts that exceed 512 tokens.

4.3.2 Training for domain adaptation

Three types of data are available to train scientific claim verification systems. (1) In-domain fact-checking annotations are the "gold standard", but they are expensive to create and require expert annotators. (2) General-domain fact-checking datasets like FEVER [Thorne et al., 2018] are abundantly available, but generalize poorly to scientific claims (see §4.6.1). (3) Scientific documents – either unlabeled or labeled for different tasks – are abundant, and high precision heuristics (described in §4.4.2) can be used to generate document-level fact-checking labels $y(c, a)$ for these data.

We train MULTIVERS as follows: we first pretrain on a combination of general-domain fact-checking annotations, combined with weakly-labeled in-domain data.[3] Then, we finetune on the target scientific fact-checking dataset. The multitask architecture of MULTIVERS is well-suited to this strategy, since the model can be trained on data with or without rationale annotations. When no rationales are available, we set $\lambda_{\text{rationale}} = 0$ in the loss function and train as usual. By contrast, training an "extract-then-label" model on weakly-supervised data requires creating rationale annotations $R(c, a)$, which is quite noisy (see §4.4.2).

4.4 Datasets

4.4.1 Scientific claim verification datasets

We experiment with three scientific claim verification datasets. Table 4.1 provides a summary of important dataset characteristics. HealthVer and COVID-Fact were originally released in an NLI format, pairing claims with (out-of-context) evidentiary sentences. We convert to our task format by identifying the abstracts in the CORD-19 corpus [Wang et al., 2020] containing these sentences.

We use the following terminology: an *atomic* claim makes an assertion about a single property of a single entity, while a *complex* claim may make assertions about multiple properties or entities.

SCIFACT was introduced in Chapter 3 of this thesis.

HealthVer [Sarrouti et al., 2021] consists of COVID-related claims obtained by extracting snippets from articles retrieved to answer questions from TREC-COVID [Voorhees et al., 2020], verified against abstracts from the CORD-19 corpus [Wang et al., 2020]. Claims in HealthVer may be complex. REFUTED claims occur naturally in the article snippets. HealthVer provides candidate abstracts for each claim, but some of these candidates do not contain sufficient information to justify a SUPPORTS / REFUTES verdict and are labeled NEI.

COVID-Fact [Saakyan et al., 2021] collects claims about COVID-19 scraped from a COVID-19 subreddit, and verifies them against linked scientific papers, as well as documents retrieved via Google search. Claims in COVID-Fact may be complex, and candidate abstracts for each claim are provided. All candidates either SUPPORT or REFUTE the claim. Claim negations were created automatically by replacing salient words in the original claims, and as a result the labels $y(c, a)$ are somewhat noisy.

4.4.2 Pretraining datasets

We briefly describe our pretraining datasets and the weak supervision heuristics used to construct them.

[3]We use "pretraining" as shorthand for "training on the target task with out-of-domain and/or weakly-supervised labels."

FEVER [Thorne et al., 2018] consists of claims created by re-writing Wikipedia sentences into atomic claims, verified against Wikipedia articles.

EVIDENCEINFERENCE [Lehman et al., 2019; DeYoung et al., 2020b] was released to facilitate understanding of clinical trial reports, which examine the effect of an *intervention* on an *outcome*, relative to a *comparator* ("ICO" elements). The dataset contains ICO *prompts* paired with (1) labels indicating whether the outcome *increased* or *decreased* due to the intervention, and (2) rationales justifying each label. We use rule-based heuristics to convert these prompts into claims – for instance "[intervention] increases [outcome] relative to [comparator]".

PUBMEDQA [Jin et al., 2019] was released to facilitate question-answering over biomedical research abstracts. We use the PQA-A subset, which is a large collection of abstracts with "claim-like" titles – for instance, "Vitamin B6 supplementation increases immune responses in critically ill patients." We treat the paper titles as claims and the matching abstracts as the evidence sources.

To train models requiring rationale supervision, we create weakly-supervised rationales by selecting the sentences with highest similarity to the claim as measured by cosine similarity of Sentence-BERT embeddings [Reimers and Gurevych, 2019]. These annotations are not used to train MULTIVERS. To estimate the precision of our rationale labeling heuristic, we predict rationales in the same fashion for our supervised datasets and compute the Precision@1 with which this method identifies gold rationales. The scores are relatively low: 49.4, 48.8, and 43.4 for SCIFACT, COVID-Fact, and HealthVer respectively.

4.5 Experimental setup

We describe our model training procedure, the systems against we compare MULTIVERS, and our ablation experiments.

4.5.1 Model training

Our complete training procedure consists of pretraining on the three datasets from §4.4.2, followed by finetuning on one of the target datasets from §4.4.1. We conduct experiments with three different levels of

supervision. For *zero-shot* experiments, we perform pretraining only. For *few-shot* experiments, we pretrain followed by finetuning on 45 target examples. For *fully-supervised* experiments, we pretrain and then train on all target data.

Following Li et al. [2021], we found that negative sampling was important to achieve good precision on SCIFACT, which requires document retrieval. We train with 20 negative samples per claim and retrieve 10 abstracts per claim at inference time. For the other datasets, no negative sampling was used. During model development, we experimented with training on all three target datasets combined before predicting on each one, but found that this did not improve performance.

4.5.2 Baseline systems

We use PARAGRAPHJOINT and VERT5ERINI as baselines. VERT5ERINI is the largest model, with 5.6B parameters. MULTIVERS and PARAGRAPHJOINT are comparably-sized, with 440M and 360M parameters, respectively.

In the fully-supervised setting, we compare against both baselines. For prediction on SCIFACT, we use publicly available model checkpoints as-is. For training on HealthVer and COVID-Fact, we use the code provided by the authors, starting from the available checkpoints trained on SCIFACT. Model hyperparameters (learning rate, batch size, epoch number, etc.) for all systems including MULTIVERS were tuned based solely on SCIFACT and not adjusted further.

Evaluation in the few-shot and zero-shot settings requires pretraining and finetuning as described in §4.5.1. Due to the expense of pretraining T5-3B, we do not perform these experiments for VERT5ERINI, and compare only against PARAGRAPHJOINT (which shows comparable performance in the fully-supervised setting). We pretrain PARAGRAPHJOINT on the data described in §4.4.2.

4.5.3 Ablations

Since PARAGRAPHJOINT and VERT5ERINI differ from MULTIVERS along a number of important dimensions (e.g. model architecture, number of parameters, and base encoder), we conduct ablations to characterize the performance contributions of three key components of MULTIVERS.

Pretraining data We compare the results of three different pretraining strategies. For FEVERSCI, we pretrain on all available data as described in §4.5.1. For FEVER, we pretrain on FEVER only. For *No-Pretrain*, we perform no pretraining.

Base encoder We compare the performance achieved using LongFormer as the encoder for MULTIVERS, compared to the results when we swap in RoBERTa but keep other settings identical. We use Longformer-large and RoBERTa-large.

Modeling approach We compare three modeling approaches: (1) The *Multitask* approach is the method used by MULTIVERS as described in §4.3.1. (2) The *Pipeline* approach consists of two separate Longformer modules. The first selects rationales as described in §4.3.1, but with L_{label} removed from Eq. 4.1, and the second module predicts a label given the text of the rationales selected by the first module as input. When pretraining on PUBMEDQA, we train on the rationales chosen by Sentence-BERT as described in §4.4.2. (3) The *Multitask train / Pipeline inference* (MT / PI) approach takes the model trained using the Multitask approach, and performs inference using the Pipeline approach. Specifically, MT / PI is trained to make label predictions based on full abstracts, but must make test-time label predictions based on predicted rationales only. By contrast, the Pipeline model makes label predictions based on gold and predicted rationales at train and test time, respectively.

4.6 Experimental results

We compare MULTIVERS performance relative to our baseline systems, and present ablation results.

4.6.1 Main Results

Table 4.2 compares the performance of MULTIVERS against PARAGRAPHJOINT and VERT5ERINI.[4] A few trends are apparent. First, MULTIVERS outperforms the baselines on all datasets, with relative improvements — averaged over the three datasets and two evaluation methods — of 26%, 14%, and 11% in the zero-shot, few-shot, and fully-supervised settings respectively. We examine possible causes of this improvement in

[4]We evaluate model performance in this chapter using F1 score. In Chapter 5.5 and Appendix A, we show that measuring performance using average precision leads to the same qualitative conclusions regarding model performance.

		HealthVer						COVID-Fact						SciFact					
		Abstract			Sentence			Abstract			Sentence			Abstract			Sentence		
Setting	**Model**	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1	P	R	F1
Zero	ParagraphJoint	72.3	14.4	24.0	22.9	2.7	4.9	51.3	37.9	43.6	31.5	16.0	21.3	52.9	32.4	40.2	36.4	14.9	21.1
	MultiVerS	60.6	20.5	**30.7**	25.0	4.6	**7.8**	48.8	45.7	**47.2**	32.7	18.5	**23.6**	49.0	44.6	**46.7**	39.0	21.6	**27.8**
Few	ParagraphJoint	62.7	41.6	50.0	46.0	29.3	**35.8**	73.3	60.6	66.3	44.3	30.6	36.2	44.4	51.4	47.6	33.0	35.1	34.0
	MultiVerS	63.6	47.9	**54.7**	41.9	31.0	35.7	71.3	68.1	**69.7**	39.5	35.4	**37.4**	76.4	54.1	**63.3**	51.7	40.3	**45.3**
Full	Vert5Erini	71.3	74.0	72.6	65.6	61.2	63.3	76.6	52.7	62.4	44.8	27.2	33.9	64.0	73.0	68.2	60.6	66.5	63.4
	ParagraphJoint	75.0	68.3	71.5	69.9	60.6	64.9	71.5	68.1	69.8	41.4	40.3	40.8	75.8	63.5	69.1	68.9	54.6	60.9
	MultiVerS	78.9	76.3	**77.6**	71.4	67.0	**69.1**	77.3	77.3	**77.3**	41.5	46.1	**43.7**	73.8	71.2	**72.5**	67.4	67.0	**67.2**

Table 4.2: Performance of MultiVerS and baselines. In the fully-supervised setting, we compare to ParagraphJoint and Vert5Erini, which exhibit comparable performance. In the zero and few-shot settings, we compare to ParagraphJoint only due to the high cost of pretraining Vert5Erini. We report performance using abstract-level and sentence-level evaluation as defined in §4.2.1.

§4.6.2. Second, while all models score within roughly six points of each other on HealthVer and SciFact, variability is much greater on COVID-Fact. We suspect that this is due to the automatically-generated nature of COVID-Fact negations. Third, we observe that HealthVer appears to be the most challenging dataset of the three. Few-shot abstract-level F1 scores for COVID-Fact and SciFact are generally within 10 F1 of their fully-supervised values, while the gap is roughly 20 F1 for HealthVer. This may be due to the high complexity of HealthVer claims.

4.6.2 Ablations

The results of all ablations are shown in Table 4.3.

In-domain pretraining substantially improves zero / few-shot performance In Table 4.3a, we compare the performance of models pretrained on FeverSci, Fever, and No-Pretrain. In the zero-shot setting, removing scientific data during pretraining results in a relative performance decrease of 65%, averaged over the three datasets and two evaluation methods. The decrease is driven primarily by very low recall.

In the few-shot setting, Fever pretraining scores within 4% of FeverSci, while No-Pretrain results in a 39% decrease relative to FeverSci. This suggests that training on a handful of target examples is sufficient to recalibrate a model trained for a different domain, but not to learn the task from scratch. In the fully-supervised setting, Fever pretraining is only slightly worse than FeverSci, while No-Pretrain lags by

roughly 9%. Overall, the results indicate that pretraining always helps, and pretraining on weakly-labeled in-domain data helps especially when target data are scarce.

Longformer improves performance on datasets with long documents Table 4.3b compares the performance of MULTIVERS when Longformer and RoBERTa are used as the base encoder. Using Longformer consistently helps on SCIFACT, but does not help on the other two datasets. This is unsurprising, since 27% of SCIFACT instances exceed the RoBERTa token limit, compared to less than 15% for the other two datasets (Table 4.1).

Multitask modeling improves zero / few-shot performance Results comparing our three different modeling approaches are shown in Table 4.3c. In the zero-shot setting, we find that Multitask performs best, with both MT / PI and Pipeline exhibiting performance drops greater than 50%. The Multitask approach of predicting rationales conditioned on the predicted label leads to improved recall. Similarly, in the few-shot setting, both Pipeline and MT / PI perform roughly 10% worse than Multitask. Collectively, the results indicate that Multitask makes the best use of the available data when target annotations are limited.

We also find that MT / PI outperforms Pipeline in the zero-shot setting. This supports our intuition from §4.3.2 that, while training on weakly-supervised *document-level* labels improves zero-shot performance,

	Pretraining	HealthVer	COVID-Fact	SCIFACT
Zero	FEVERSCI	**30.7 / 7.8**	**47.2 / 23.6**	**46.7 / 27.8**
	FEVER	1.3 / 0.7	25.2 / 11.2	23.9 / 11.8
Few	FEVERSCI	**54.7 / 35.7**	69.7 / 37.4	**63.3 / 45.3**
	FEVER	53.4 / 31.9	**74.4 / 42.1**	54.5 / 39.0
	No-Pretrain	39.4 / 27.0	67.8 / 22.6	24.2 / 10.8
Full	FEVERSCI	**77.6** / 69.1	77.3 / **43.7**	**72.5 / 67.2**
	FEVER	77.1 / **70.3**	**77.4** / 43.3	67.9 / 61.7
	No-Pretrain	74.5 / 69.7	69.7 / 36.6	63.3 / 58.4

(a) Effect of pretraining data. In-domain pretraining is very effective in the zero- and few-shot settings. In the zero-shot setting, "No-Pretrain" metrics are not shown since this would correspond to no training at all.

	Encoder	HealthVer	COVID-Fact	SCIFACT
Zero	Longformer	30.7 / 7.8	47.2 / 23.6	**46.7 / 27.8**
	RoBERTa	**34.2 / 9.2**	**48.3 / 26.2**	45.2 / 25.9
Few	Longformer	**54.7** / 35.7	69.7 / 37.4	**63.3 / 45.3**
	RoBERTa	51.2 / **36.9**	**72.1 / 41.0**	50.5 / 34.0
Full	Longformer	77.6 / 69.1	77.3 / **43.7**	**72.5 / 67.2**
	RoBERTa	**78.8 / 72.7**	**78.2** / 43.4	67.6 / 62.3

(b) Effect of base encoder. Longformer improves performance on SCIFACT, which has the largest fraction of instances exceeding the RoBERTa token limit.

	Approach	HealthVer	COVID-Fact	SCIFACT
Zero	Multitask	**30.7 / 7.8**	**47.2 / 23.6**	**46.7 / 27.8**
	Pipe	3.2 / 0.9	19.0 / 10.5	22.5 / 12.8
Few	Multitask	**54.7 / 35.7**	**69.7** / 37.4	**63.3 / 45.3**
	Pipe	52.8 / 29.5	68.3 / **38.2**	53.0 / 39.9
Full	Multitask	77.6 / 69.1	77.3 / 43.7	**72.5 / 67.2**
	Pipe	**78.4 / 69.2**	**77.6 / 47.7**	70.9 / 66.2

(c) Effect of model architecture. The Multitask approach performs best in the zero- and few-shot settings.

Table 4.3: Ablations examining the effects of pretraining data and modeling approach. Entries are formatted "{Abstract-level F1} / {Sentence-level F1}".

training on weakly-supervised *sentence-level* rationales

(as for Pipeline) leads to worse performance than not training on these rationales (as for MT / PI).

In the fully-supervised setting, Multitask performs best on SCIFACT, while Pipeline slightly outperforms Multitask on HealthVer and COVID-Fact. MT / PI performs substantially worse than the other approaches on all datasets. We investigate these findings further in §4.7.1; our results indicate that Pipeline may, in effect, be trained to make predictions based on insufficient evidence.

4.7 Analysis

4.7.1 Fully-supervised Pipeline performance

In §4.6.2, we found that the Pipeline approach (but not the MT / PI approach) performed on par with the Multitask approach in the fully-supervised setting. To understand this finding, we collected detailed annotations for 128 claim / evidence instances from the SCIFACT test set. For each instance, an annotator indicated whether the annotated rationales were "self-contained" — i.e. sufficient to justify the fact-checking label when taken in isolation, or "context-dependent" — i.e. only sufficient when taken in the context of the abstract. Figure 4.1 provides an example; see Choi et al. [2021] for a detailed discussion of different forms of context-dependence.[5]

	Self-contained			**Context-dependent**			
Approach	P	R	F1	P	R	F1	%**Δ**
Multitask	86.1	82.9	84.5	90.3	60.9	72.7	-14.0%
Pipeline	92.4	89.0	90.7	82.4	60.9	70.0	-22.8%
MT / PI	91.8	54.9	68.7	100.0	13.0	23.1	-66.4%
Count	82			46			

Table 4.4: Performance of the Multitask, Pipeline, and MT / PI modeling approaches on SCIFACT instances with rationales that are self-contained (can be interpreted in isolation) or context-dependent (must be interpreted in the context of the abstract). Evaluation is performed in the abstract-provided setting. We report abstract-level metrics; sentence-level results are similar. The %Δ indicates the drop in F1 score on context-dependent instances relative to self-contained instances. Multitask suffers the smallest performance loss, while MT / PI suffers the largest.

Table 4.4 compares the performance of the three modeling approaches on instances with self-contained vs. context-dependent evidence. We find that all approaches have lower performance on context-dependent instances relative to self-contained instances, but the size of the performance drop varies widely. The Multitask approach performs 14.0% worse on context-dependent instances, while the Pipeline approach

[5]Unlike Choi et al. [2021], we do not include the presence of acronyms as "context-dependent," since an acronym can be matched with its expansion based on surface-level textual features.

performs 22.8% worse. Most interestingly, MT / PI performs 66.4% worse, driven predominantly by low recall. The MT / PI model frequently (and correctly) predicts that context-dependent rationales are not sufficient to justify a SUPPORTS / REFUTES decision. These findings suggest that (1) the Multitask approach is, as expected, best at verifying claims with context-dependent evidence, and (2) the Pipeline approach has, in effect, over-fit to context-dependent rationales and learned to make predictions based on insufficient evidence.

4.7.2 Performance upper bound

To determine an "upper bound" on the achievable performance of scientific fact-checking models, we assign 151 claim-evidence pairs from SCIFACT for independent annotation by two different annotators. We estimate human-level performance by treating the first annotator's results as "gold," and the second annotator's results as predictions. For comparison, we make predictions using MULTIVERS and our two baseline models, with candidate abstracts provided as input. The results are shown in Table 4.5. Existing systems already exceed human agreement for sentence-level evaluation, but not abstract-level, indicating that experts tend to agree on the overall relationship between claim and abstract, but may disagree about exactly which sentences contain the best evidence. This constitutes another reason not to rely solely on selected rationales when predicting a fact-checking label: the choice of rationales is itself somewhat subjective.

	Abstract			Sentence		
	P	R	F1	P	R	F1
VERT5ERINI	90.7	74.3	81.7	79.6	62.2	69.8
PARAGRAPHJOINT	87.2	64.4	74.1	76.7	55.1	64.1
MULTIVERS	87.4	75.2	80.9	80.5	70.3	**75.0**
Human	94.8	84.1	**89.1**	67.4	67.4	67.4

Table 4.5: Performance on SCIFACT in the "abstract-provided" setting. Models exceed human agreement as measured by sentence-level F1, but not abstract-level.

In addition, these results suggest that one key subtask of scientific claim verification — namely, predicting whether an evidence-containing abstract SUPPORTS or REFUTES a claim — may be nearly "solved" in the setting where (1) the claims are atomic and (2) roughly 1,000 in-domain labeled claims are available for training.

4.8 Related work

Background on scientific claim verification is covered in §4.2; we discuss other relevant work here. Nye et al. [2020] have previously observed that document-level context is often required to properly interpret scientific findings.

DeYoung et al. [2020b] use an "extract-then-label" pipeline for the original EVIDENCEINFERENCE task. Multitask label prediction and rationale selection was proposed by Pruthi et al. [2020] and applied to sentiment analysis and propaganda detection. As in this chapter, the authors condition on the predicted label when predicting rationales. Another alternative to supervised rationale selection is to treat rationales as latent variables [Lei et al., 2016; Paranjape et al., 2020].

Long-document encodings for fact verification have been explored by Stammbach [2021], who use Big Bird [Zaheer et al., 2020] for full-document evidence extraction from FEVER. Domain adaptation for scientific text has been studied in a number of works, including Gururangan et al. [2020b]; Beltagy et al. [2019]; Lee et al. [2020]; Gu et al. [2021]. In those works, the primary focus is on language model pretraining. Here, we focus on training on the target task using out-of-domain and weakly-labeled data.

4.9 Conclusion

This work points to a number of promising future directions for scientific claim verification. These include applying the approach presented here to develop scientific claim verification models for new scientific sub-domains or other specialized fields given a handful of labeled examples, and extending the task definition to verify claims against longer contexts (e.g. full scientific papers) or larger corpora. Our task formulation also offers an opportunity to study the effects of rationale decontextualization [Choi et al., 2021], especially in cases where models may be making predictions based on insufficient evidence.

In presenting the MULTIVERS system, we addressed two challenges associated with scientific claim verification: incorporating relevant information beyond rationale boundaries by modeling full-document context, and facilitating zero / few-shot domain adaptation through weak supervision enabled by a multitask modeling approach. Our experiments show that MULTIVERS outperforms existing systems across several scientific claim verification datasets. We hope that the task, data, and modeling resources presented in this

chapter will encourage further work and progress towards the broader goals of identifying and addressing scientific mis- and disinformation.

Chapter 5

SciFact-Open: Towards Open-Domain Scientific Claim Verification

This chapter contains material that was originally published in Wadden et al. [2022b].

5.1 Introduction

The task of scientific claim verification (Chapter 3) aims to help system users assess the veracity of a scientific claim relative to a corpus of research literature. Most existing work and available datasets focus on verifying claims against a much more limited context—for instance, a single article or text snippet [Saakyan et al., 2021; Sarrouti et al., 2021; Kotonya and Toni, 2020] or a small, artificially-constructed collection of documents (Chapter 3). In Chapter 4 of this thesis we presented a state-of-the-art model for this task which achieves strong performance, in some cases approaching human agreement.

This gives rise to the question of the scalability of scientific claim verification systems to realistic, *open-domain* settings that involve verifying claims against corpora containing hundreds of thousands of documents. In these cases, claim verification systems should assist users by identifying and categorizing all available documents that contain evidence supporting or refuting each claim (Fig. 5.1). However, evaluating system performance in this setting is difficult because exhaustive evidence annotation is infeasible, an issue analogous to evaluation challenges in information retrieval (IR).

In this chapter, we construct a new test collection for open-domain scientific claim verification, called SCIFACT-OPEN, which requires models to verify claims against evidence from both the SCIFACT collection (Chapter 3), as well as additional evidence from a corpus of 500K scientific research abstracts. To avoid the burden of exhaustive annotation, we take inspiration from the *pooling* strategy [Sparck Jones and van Rijsbergen, 1975] popularized by the TREC competitions [Voorhees and Harman, 2005] and combine the predictions of several state-of-the-art scientific claim verification models—for each claim, abstracts that the models identify as likely to SUPPORT or REFUTE the claim are included as candidates for human annotation.

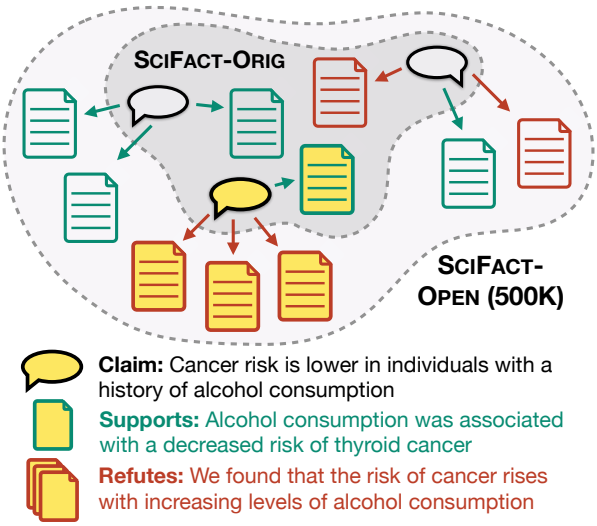


Figure 5.1: SCIFACT-OPEN, a new test collection for scientific claim verification that expands beyond the 5K abstract retrieval setting in the original SCIFACT dataset (Chapter 3) to a corpus of 500K abstracts. Each claim in SCIFACT-OPEN is annotated with evidence that SUPPORTS or REFUTES the claim. In the example shown, the majority of evidence REFUTES the claim that alcohol consumption reduces cancer risk, although one abstract indicates that alcohol consumption may reduce thyroid cancer risk specifically.

Our main contributions and findings are as follows.
(1) We introduce SCIFACT-OPEN, a new test collection for open-domain scientific claim verification, including 279 claims verified against evidence retrieved from a corpus of 500K abstracts. (2) We find that state-of-the-art models developed for SCIFACT perform substantially worse (at least 15 F1) in the open-domain setting, highlighting the need to improve upon the generalization capabilities of existing systems. (3) We identify and characterize new dataset phenomena that are likely to occur in real-world claim verification settings. These include mismatches between the specificity of a claim and a piece of evidence, and the presence of conflicting evidence (Fig. 5.1).

With SCIFACT-OPEN, we introduce a challenging new test set for scientific claim verification that more closely approximates how the task might be performed in real-word settings. This dataset will allow for further study of claim-evidence phenomena and model generalizability as encountered in open-domain scientific claim verification.

5.2 Background and Task Overview

We review the scientific claim verification task, and summarize the data collection process and modeling approaches for SCIFACT (Chapter 3), which we build upon in this chapter. We elect to use the SCIFACT dataset as our starting point because of the diversity of claims in the dataset and the availability of a number of state-of-the-art models that can be used for pooled data collection. In the following, we refer to the original SCIFACT dataset as SCIFACT-ORIG.

5.2.1 Task definition

We briefly review the scientific claim verification task definition from Chapter 3. Given a claim c and a corpus of research abstracts $\mathcal{A}$, the task is to identify all abstracts in $\mathcal{A}$ which contain evidence relevant to c, and to predict a label $y(c, a) \in \{\text{SUPPORTS}, \text{REFUTES}\}$ for each evidence abstract. All other abstracts are labeled $y(c, a) = \text{NEI}$ (Not Enough Info). We will refer to a single (c, a) pair as a *claim / abstract pair*, or CAP. Any CAP where the abstract a provides evidence for the claim c (either SUPPORTS or REFUTES) will be called an *evidentiary CAP*, or ECAP. Models are evaluated on their precision, recall, and F1 in identifying and correctly labeling the evidence abstracts associated with each claim in the dataset (or equivalently, in identifying ECAPs).[1]

5.2.2 SCIFACT-ORIG

Each claim in SCIFACT-ORIG was created by re-writing a citation sentence occurring in a scientific article, and verifying the claim against the abstracts of the cited articles (Chapter 3.3). The resulting claims are diverse both in terms of their subject matter—ranging from molecular biology to public health—as well as their level of specificity (see §5.3.3). Models are required to retrieve and label evidence from a small (roughly 5K abstract) corpus.

Models for SCIFACT-ORIG generally follow a two-stage approach to verify a given claim. First, a small collection of candidate abstracts is retrieved from the corpus using a retrieval technique like BM25 [Robertson and Zaragoza, 2009]; then, a transformer-based language model [Devlin et al., 2019; Raffel et al., 2020] is

[1]The original SCIFACT task also requires the prediction of rationales justifying each label. Due to the expense of collecting rationale annotations, in this chapter we do not require rationales; we evaluate using the *abstract-level label-only F1* metric described in Wadden et al. [2020].

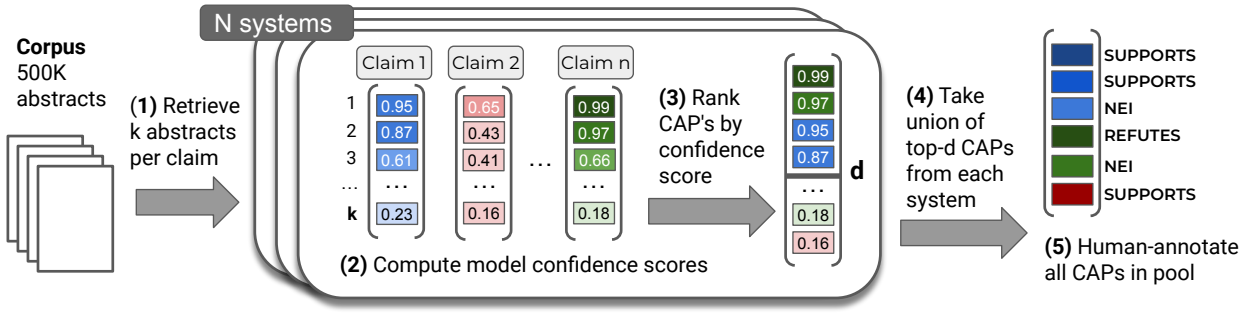


Figure 5.2: Pooling methodology used to collect evidence for SCIFACT-OPEN. We construct the pool by combining the d most-confident predictions of n different systems. A single CAP is represented as a colored box; the number in the box indicates a hypothetical confidence score. In this example, the annotation pool contains 3 CAPs from Claim 1, 2 for Claim 2, and 1 for Claim 3. Annotators found evidence for 4 / 6 of these CAPS.

trained to predict whether each retrieved document SUPPORTS, REFUTES, or contains no relevant evidence (NEI) with respect to the claim.

As we show in §5.4 and §5.5, a key determinant of system generalization is the *negative sampling ratio*. A negative sampling ratio of r indicates that the model is trained on r irrelevant CAPs for every relevant ECAP. Negative sampling has been shown to improve performance (particularly precision) on SCIFACT-ORIG [Li et al., 2021].

5.3 The SCIFACT-OPEN dataset

In this section, we describe the construction of SCIFACT-OPEN. We report the performance of claim verification models on SCIFACT-OPEN in §5.4, and perform reliability checks on the results in §5.5.

Our goal is to construct a test collection which can be used to assess the performance of claim verification systems deployed on a large corpus of scientific literature. This requires a collection of claims, a corpus of abstracts against which to verify them, and evidence annotations with which to evaluate system predictions. We use the claims from the SCIFACT-ORIG test set as our claims for SCIFACT-OPEN.[2] To obtain evidence annotations, we use all evidence from SCIFACT-ORIG as evidence in our new dataset and collect additional evidence from the SCIFACT-OPEN corpus.

For our corpus, we filter the S2ORC dataset [Lo et al., 2020b] for all articles which (1) cover topics related to medicine or biology and (2) have at least one inbound and one outbound citation. From the

[2]We remove 21 claims (out of 300 total) whose source citations lack important metadata.

roughly 6.5 million articles that pass these filters, we randomly sample 500K articles to form the corpus for SCIFACT-OPEN, making sure to include the 5K abstracts from SCIFACT-ORIG. We choose to limit the corpus to 500K abstracts to ensure that we can achieve sufficient annotation coverage of the available evidence.

Unlike SCIFACT-ORIG (which is skewed toward highly-cited articles from "high-impact" journals), we do not impose any additional quality filters on articles included in SCIFACT-OPEN; thus, our corpus captures the full diversity of information likely to be encountered when scientific fact-checking systems are deployed on real-world resources like S2ORC, arXiv,[3] or PubMed Central.[4]

5.3.1 Pooling for evidence collection

To collect evidence from the SCIFACT-OPEN corpus, we adopt a pooling approach popularized by the TREC competitions: use a collection of state-of-the-art models to select CAPs for human annotation, and assume that all un-annotated CAPs have $y(c, a) = \text{NEI}$. We will examine the degree to which this assumption holds in §5.5.

Model	Source	Negative sampling
Pooling and Eval		
VERT5ERINI	Pradeep et al. [2021]	0
PARAGRAPHJOINT	Li et al. [2021]	10
MULTIVERS	Wadden et al. [2022c]	20
MULTIVERS_{10}	Wadden et al. [2022c]	10
Eval only		
ARSJOINT	Zhang et al. [2021]	12

Table 5.1: Models used for pooled data collection and evaluation (top), and for evaluation only (bottom). "Negative sampling" indicates the negative sampling ratio. MULTIVERS_{10} shares the same architecture as MULTIVERS, but trains on fewer negative samples.

Pooling approach We annotate the d most-confident predicted CAPS from each of n claim verification systems. An overview of the process is in shown in Fig. 5.2; we number the annotation steps below to match the figure.

We select the most confident predictions for a single model as follows. **(1)** For each claim in SCIFACT-OPEN, we use an information retrieval system consisting of BM25 followed by a neural re-ranker [Pradeep et al., 2021] to retrieve k abstracts from the SCIFACT-OPEN corpus. **(2)** For each CAP, we compute the softmax scores associated with the three possible output labels, denoted $s(\text{SUPPORTS}), s(\text{REFUTES}), s(\text{NEI})$. We use $\max(s(\text{SUPPORTS}), s(\text{REFUTES}))$ as a measure of the model's confidence that the CAP contains evidence. **(3)** We rank all CAPs by model confidence, and add the d top-ranked predictions to the annotation

[3] https://arxiv.org
[4] https://www.ncbi.nlm.nih.gov/pmc

pool. The final pool **(4)** is the union of the top-d CAPs identified by each system. Since some CAPs are identified by multiple systems, the size of the final annotation pool is less than $n \times d$; we provide statistics in §5.3.2. Finally, **(5)** all CAPs in the pool are annotated for evidence and assigned a final label by an expert annotator, and the label is double-checked by a second annotator.

We choose to prioritize CAPS for annotation based on model confidence, rather than annotating a fixed number of CAPs per claim, in order to maximize the amount of evidence likely to be discovered during pooling. In §5.3.3, we confirm that our procedure identifies more evidence for claims that we would expect to be more extensively-studied.

Models and parameter settings We set $k = 50$ for abstract retrieval. In practice, we found that the great majority of evidentiary abstracts were ranked among the top 20 retrievals for their respective claims, and thus using a larger k would serve mainly to increase the number of irrelevant results. We set $d = 250$; in §5.5.1, we show that this is sufficient to ensure that our dataset can be used for reliable model evaluation.

For our models, we utilized all state-of-the-art models developed for SCIFACT-ORIG for which modeling code and checkpoints were available (to our knowledge). We used $n = 4$ systems for pooled data collection. During evaluation, we included a fifth system — ARSJOINT— which became available after the dataset had been collected. Model names, source publications, and negative sampling ratios are listed in Table 5.1.

		ECAPs		
Claims	Corpus	SCIFACT-ORIG	Pooling	Total
279	500K	209	251	460

(a) Summary of the SCIFACT-OPEN dataset, including the number of claims, abstracts, and ECAPs (evidentiary claim / evidence pairs). ECAPs come from two sources: those from SCIFACT-ORIG, and those discovered via pooling.

Num. systems	Annotated	Evidence	% Evidence
1	528	154	29.2
2	150	71	47.3
3	44	20	45.5
4	10	6	60.0
All	732	251	34.3

(b) Relevance of CAPs annotated during the pooling process. The first row indicates that 528 CAPs were identified for pooling by one system only; of those CAPs, 154 were judged by annotators as containing evidence. The more systems identified a given CAP, the more likely it is to contain evidence.

	Total	Retrieved	Annotated
ECAPs	209	187 (89%)	171 (82%)

(c) Count of how many ECAPs from SCIFACT-ORIG would have been identified during pooled data collection. "Retrieved" indicates the number of ECAPs that would have been retrieved among the top k, and "Annotated" indicates the number that would further have been included in the annotation pool.

Table 5.2: Annotation results and dataset statistics for SCIFACT-OPEN.

5.3.2 Dataset statistics

We summarize key properties of SCIFACT-OPEN. Table 5.2a provides an overview of the claims, corpus, and evidence in the dataset. Table 5.2b shows the fraction of CAPs annotated during pooling which were judged to be ECAPs (i.e. to contain evidence). Overall, roughly a third of predicted CAPs were judged as relevant; this indicates that existing systems achieve relatively low precision when used in an open-domain setting. Relevance is somewhat higher (roughly 50%) for CAPs predicted by more than one system. The majority of CAPs are selected by a single system only, indicating high diversity in model predictions. As mentioned in §5.3.1, the total number of annotated CAPs is 732 (rather than 4 models $\times$ 250 CAPs / model $=$ 1000) due to overlap in system predictions.

Table 5.2c shows how many of the ECAPs from SCIFACT-ORIG would have been annotated by our pooling procedure. The fact that the great majority of the original ECAPs would have been included in the annotation pool suggests that our approach achieves reasonable evidence coverage.

5.3.3 Evidence phenomena in SCIFACT-OPEN

We observe three properties of evidence in SCIFACT-OPEN that have received less attention in the study of scientific claim verification, and that can inform future work on this task.

Unequal allocation of evidence Fig. 5.3 shows the distribution of evidence amongst claims in SCIFACT-OPEN. We find that evidence is distributed unequally; half of all ECAPs are allocated to 34 highly-studied claims (12% of all claims in the dataset). We investigated the characteristics of highly-studied claims, and found that they tend to be short and mention a small number of common, well-studied scientific entities. For instance, entities mentioned in well-studied claims ($\geq$ 4 ECAPs) return, on average, 4 times as many documents when entered into a PubMed search, compared

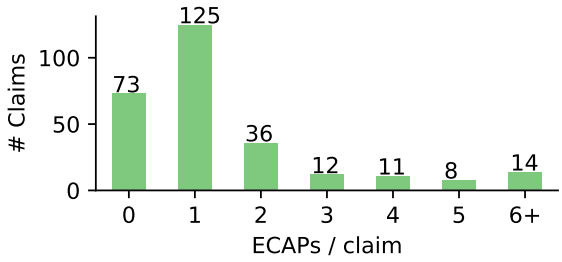


Figure 5.3: Evidence allocation among claims in SCIFACT-OPEN. The x-axis indicates the number of ECAPs (evidentiary claim / abstract pairs) associated with a given claim, and the y-axis is the number of claims with the corresponding number of ECAPS. For instance, 125 claims are associated with a single evidence-containing abstract.

to claims with no evidence. Table 5.3 shows an example.

Mismatch in claim and evidence specificity During evidence collection for SCIFACT-OPEN, annotators reported situations where a claim and abstract exhibited a relationship, but where the claim applied at a different level of *specificity* from the evidence. For instance, in Fig. 5.1, the claim and refuting evidence discuss the effects of alcohol consumption on *overall* cancer risk, while the supporting evidence indicates that alcohol consumption lowers thyroid cancer risk *in particular*; the supporting evidence is *more specific* than the claim. We also saw cases where the abstract was *more general* than the claim (e.g. claim discusses thyroid cancer, abstract discusses cancer in general), and where the abstract was *closely related* to the claim (e.g. claim discusses thyroid cancer, abstract discusses throat cancer).[5]

Claim	ECAPs
Obesity is determined in part by **genetic factors**.	19
Inhibiting **HDAC6** decreases survival of **mice** with **ARID1A mutated tumors**.	0

Table 5.3: Example of a claim with a number of ECAPs annotated during pooled data collection (top), and another with no new ECAPs (bottom). Well-studied ECAPs tend to be shorter and mention a small number of common entities.

Based on this observation, we attempted to quantify the frequency of specificity mismatches. For 206 CAPs in the SCIFACT-OPEN annotation pool, in addition to collecting a SUPPORTS / REFUTES / NEI label, annotators indicated the specificity relationship between claim and abstract, and wrote a *revision* of the claim such that the revised claim matched the specificity of the abstract. These annotations will be released as part of SCIFACT-OPEN.

Table 5.4 shows counts for different specificity relationships. We find that 91 / 206 (44%) of the examined CAPs exhibit some form of specificity mismatch. Table 5.5 provides an example of each relation type. We discuss possible implications of specificity mismatch for future work on scientific claim verification in §5.7.

Conflicting evidence Conflicting evidence occurs when a single claim is SUPPORTED by at least one ECAP in SCIFACT-OPEN, and REFUTED by another (see Fig. 5.1). Of the 81 claims in SCIFACT-OPEN with at least 2 ECAPs, 16 of them (20%) have conflict-

Category	ECAPs
Evidence matches claim	115
Evidence more specific than claim	53
Evidence more general than claim	18
Evidence closely related to claim	20

Table 5.4: Specificity relationship between claim and evidence, for 206 ECAPs. Specificity mismatches are common, comprising 44% of annotated examples.

[5] In cases of mismatching evidence, we follow the convention used in Thorne et al. [2018] to assign an overall SUPPORTS / REFUTES / NEI label.

Category	Evidence **matches** claim
Claim	Mitochondria play a major role in calcium homeostasis.
Evidence	Mitochondria . . . are essential organelles responsible for . . . calcium homeostasis.
Revision	
Explanation	No revision necessary; claim and evidence are paraphrases
Category	Evidence **more specific** than claim
Claim	Teaching hospitals provide better care than non-teaching hospitals
Evidence	Teaching centres . . . prolong survival *in women with any gynecological cancer* compared to community or general hospitals.
Revision	Teaching hospitals provide better *gynecological cancer care* than non-teaching hospitals.
Explanation	The evidence refers to *gynecological cancer care* specifically, not care care in general.
Category	Evidence **more general** than claim
Claim	*Somatic missense* mutations in NT5C2 are associated with relapse of *acute lymphoblastic leukemia*.
Evidence	T5C2 mutant proteins show . . . resistance to chemotherapy
Revision	*Mutations* in NT5C2 are associated with relapse of *cancer*.
Explanation	Evidence mentions T5C2 mutations in general, while the claim mentions *somatic missense mutations* specifically. The evidence discusses chemotherapy resistance generally, while the claim discusses relapse of *acute lymphoblastic leukemia* specifically.
Category	Evidence **closely related** to claim
Claim	Near-infrared wavelengths increase penetration depth in *fiberoptic confocal microscopy*
Evidence	Longer wavelength can . . . increase the effective penetration depth of *OCT (optical coherence-domain tomography)* imaging
Revision	Near-infrared wavelengths increase penetration depth in *optical coherence-domain tomography*.
Explanation	The claim discusses *fiberoptic confocal microscopy*. The evidence discusses a different imaging technique, *optical coherence-domain tomography*.

Table 5.5: Examples of different forms of claim-evidence specificity mismatch. In each example, information specific to claim or evidence is shown in *italics*. The revision re-writes the claim to match the specificity of the evidence.

ing evidence. In examining these conflicts, we found that they were often a result of specificity mismatches as shown in Fig. 5.1, indicating that modeling evidence specificity represents an important area for future work.

5.4 Model performance on SCIFACT-OPEN

We evaluate all models from Table 5.1 on SCIFACT-OPEN. These models represent the state-of-the-art on SCIFACT-ORIG, making them strong baselines to assess the difficulty of our new test collection.

	SciFact-Orig			SciFact-Open			
Model	P	R	F1	P	R	F1	Average Precision
Vert5Erini	64.0	73.0	68.2	$25.0_{1.9}$	$67.2_{2.9}$	$36.4_{2.2}$	$27.5_{3.2}$
ParagraphJoint	75.8	63.5	69.1	$54.7_{3.2}$	$46.5_{3.5}$	$50.3_{2.7}$	$40.5_{3.1}$
MultiVerS	73.8	71.2	72.5	$73.6_{2.9}$	$40.7_{3.3}$	$52.4_{3.0}$	$44.9_{3.7}$
MultiVerS$_{10}$	63.0	73.0	67.6	$49.6_{3.0}$	$53.0_{3.7}$	$51.3_{2.4}$	$43.4_{3.4}$
ARSJoint *	72.2	70.3	71.2	$46.1_{2.9}$	$37.6_{3.4}$	$41.4_{2.7}$	-

Table 5.6: System performance on SciFact-Open. For comparison, metrics on SciFact-Orig are also reported. Performance is substantially lower on SciFact-Open relative to SciFact-Orig. Precision, recall, and F1 vary widely by system, based on the negative sampling rate used during training. Subscripts indicate standard deviations over 1,000 bootstrap-resampled versions of the claims in SciFact-Open.
*The results for ARSJoint are not comparable with the other systems, since ARSJoint was not used for data collection. We did not compute model confidence scores for ARSJoint; therefore average precision is not reported.

SciFact-Open is challenging Table 5.6 shows the performance of all models on SciFact-Open, as well as on SciFact-Orig for comparison. Due to the wide variation in the precision and recall of different models on SciFact-Open, we also report average precision, which summarizes performance via the area under the precision / recall curve. We find that models rank similarly on F1 and average precision. Model performance drops by 15 to 30 F1 on SciFact-Open relative to SciFact-Orig, indicating that all models have trouble generalizing to large corpora unseen during training. ParagraphJoint, MultiVerS, and MultiVerS$_{10}$ all exhibit similar performance (within one standard deviation of each other), while Vert5Erini performs worse due to low precision.

Negative sampling affects generalization As mentioned in §5.2.2, all models except Vert5Erini were trained with negative sampling. We observe that negative sampling rate has a much larger impact on precision and recall in the open setting than was observed for SciFact-Orig. Vert5Erini has recall more than double its precision; for MultiVerS, the situation is reversed. The behavior of MultiVerS$_{10}$ is much more similar to ParagraphJoint than MultiVerS, indicating that negative sampling has a larger impact on model generalization behavior than does model architecture. ARSJoint is qualitatively similar to ParagraphJoint and MultiVerS$_{10}$, but with lower overall performance since its top predictions are not annotated for evidence (see §5.5.3).

Models have low agreement on SciFact-Open Fig. 5.4 shows the overlap among the ECAPs predicted by different systems, measured using Jaccard similarity. Overlap is relatively high (≥ 0.5) for predictions

involving abstracts that were found in SCIFACT-ORIG, and is much lower (≤ 0.2) on abstracts added in SCIFACT-OPEN. From a data collection standpoint, low agreement on SCIFACT-OPEN is a benefit, as it ensures that a diverse set of documents was included in the annotation pool. From a modeling standpoint, it suggests that agreement between existing models when deployed on novel corpora is lower than what has previously been observed. Understanding the differences in the information being identified by each model represents an important direction for future work.

5.5 Dataset reliability

The total number of annotations collected during pooling (§5.3.1) is determined by two parameters: the number of annotations per system d, and the number of systems n for which we collect annotations. These parameters must be large enough that increasing them further is unlikely to (1) lead to the discovery of a large number of additional ECAPs or (2) alter the performance metrics of models evaluated on the dataset. Following Zobel [1998], we conduct checks to ensure that conditions (1) and (2) hold for our choices of d and n.

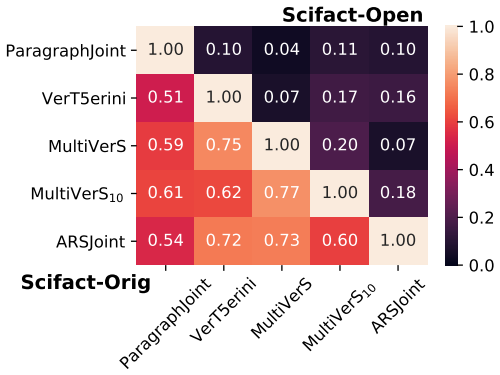


Figure 5.4: Overlap between the ECAPs predicted by different systems, as measured by Jaccard similarity. Cells below the diagonal show the similarity for abstracts contained in SCIFACT-ORIG, while cells above the diagonal show similarity for abstracts that were added in SCIFACT-OPEN. Overlap is high on abstracts from SCIFACT-ORIG, but much lower when models generalize to documents not seen during training.

5.5.1 Annotations per system

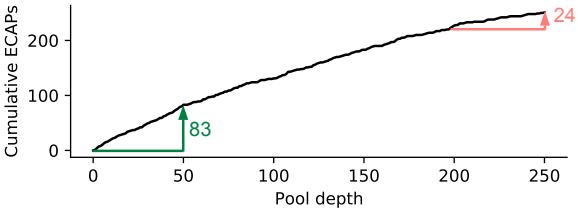
To ensure that the number of annotations per system $d = 250$ (also called the *pool depth*[6]) is large enough to ensure reliable evaluation, we examine how much additional evidence is discovered, and how our evaluation metrics change, as d increases from 0 to its final value.

Fig. 5.5a shows the total number of ECAPS discovered as a function of pool depth. Annotating the 50

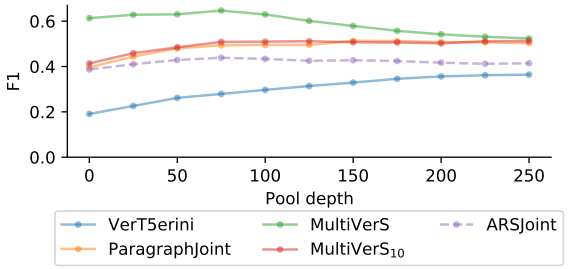
[6]In TREC, the pool depth refers to the number of annotations collected per system for a single query. We use it to refer to the number of annotations collected per system.

most-confident CAPs per system leads to the discovery of 83 ECAPs, while increasing pool depth from 200 to 250 yields 24 new ECAPs—a more than three-fold decrease. This indicates that condition (1) approximately holds; the majority of the evidence in the corpus has been annotated by $d = 250$.

Fig. 5.5b shows the F1 score of each model as a function of pool depth. While F1 scores change initially, increasing the pool depth from $d = 225$ to $d = 250$ changes the F1 score of each model by less than 2%. This indicates that condition (2) also holds: further increases to pool depth are unlikely to affect performance metrics. We also find that generalization behavior is influenced more by negative sampling rate than by model architecture. Performance of MULTIVERS decreases with depth, indicating that it was over-fit to the documents in SCIFACT-ORIG, while VERT5ERINI improves with depth. These observations hold if we use average precision rather than F1 to measure performance (see Appendix A).



(a) Total number of ECAPs discovered as a function of pool depth. For instance, annotating to a depth $d = 100$ would have resulted in the discovery of roughly 120 ECAPs.



(b) F1 score as a function of pool depth. The blue dot at pool depth 100 indicates that VERT5ERINI would have achieved an F1 score of roughly 30, if annotation had stopped at a depth of 100. Results are ARSJOINT are shown as a dashed line to indicate that this system was not used for data collection.

Figure 5.5: Effect of pool depth on evidence discovery and evaluation metrics. As pool depth increases, fewer new ECAPs are discovered and F1 score stabilizes.

5.5.2 System count

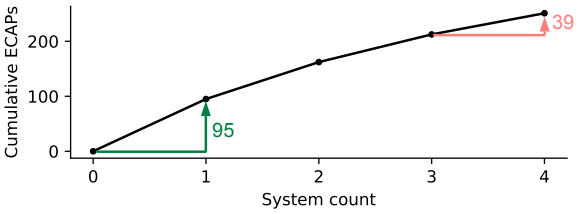
We repeat the analysis from §5.5.1, but this time varying the number of systems used for data collection (the *system count*).[7] As was the case for pool depth, Fig. 5.6a shows that fewer new ECAPs are discovered as more systems' predictions are annotated. Fig. 5.6b shows that F1 scores stabilize as system count increases, but not as completely as for pool depth; adding a fourth system still leads a 10% change in F1 score for VERT5ERINI and MULTIVERS. Thus, while conditions (1) and (2) are increasingly satisfied as the system count increases, SCIFACT-OPEN would likely benefit from the collection of additional data identified by new models. Unfortunately, unlike pool depth, the system count that we can achieve is limited by the number of

[7]There are 4! possible system orderings. We compute metrics using all orderings, and display the mean and standard deviation (as error bars) in Fig. 5.6.

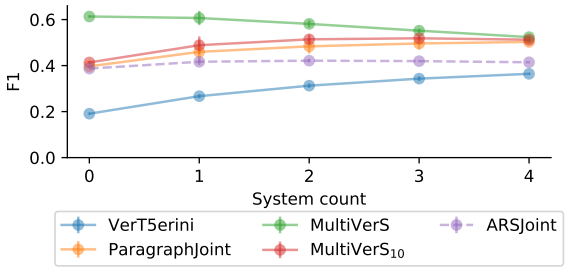
available systems for this task.

5.5.3 System inclusion

To measure the effect on measured performance of including a given system in the annotation pool, we evaluate each system on the evidence that would have been collected if that system's predictions had not been included. Results are shown in Table 5.7. All systems except MULTIVERS suffer a roughly 15% drop. When excluded from data collection, PARAGRAPHJOINT and MULTIVERS$_{10}$ both have performance comparable to ARSJOINT. MULTIVERS does not benefit from having its own predictions included, since it was over-fit to SCIFACT-ORIG and struggles to identify new evidence not seen during training. Overall, for fair model comparisons, the performance of new models should be compared against the "Excluded" performance of models used for data collection.



(a) Total ECAPs discovered as a function of system count.



(b) F1 score as a function of system count.

Figure 5.6: Effect of system count (i.e. number of systems used during pooling) on evidence discovery and evaluation metrics. As in Fig. 5.5, we see diminishing returns to increasing system count.

5.6 Related work

TREC and pooled data collection Pooling for IR evaluation was popularized by the TREC information retrieval competitions [Voorhees and Harman, 2005], with a number of recent competitions focusing on retrieval in the biomedical domain [Roberts et al., 2020b,c, 2016]. Relative to this work, TREC datasets are characterized by a smaller number of queries (or "topics"), a larger number of models available for an-

Model	Included	Excluded	% Change
VERT5ERINI	36.4	30.5	-16.3
PARAGRAPHJOINT	50.3	42.3	-15.9
MULTIVERS	52.4	51.6	-1.5
MULTIVERS$_{10}$	51.3	43.7	-14.7
ARSJoint	-	41.4	-

Table 5.7: Change in F1 score when each model is included in the annotation pool, vs. excluded. Omission leads to a performance decrease of roughly 15% for all models except MULTIVERS.

notation, and a fixed number of annotations per topic (often around 50)—although previous works have proposed strategies to prioritize topics or models for annotation [Zobel, 1998; Cormack et al., 1998]. In contrast, to maximize our annotation yield, we collect a variable number of annotations per claim based on model confidence.

Claim verification and revision Stammbach et al. [2021] studied scientific claim verification against a large research corpus, but simplified the task by evaluating accuracy at predicting a single global truth label per claim, rather than identifying all relevant documents. The Climate-FEVER dataset [Diggelmann et al., 2020] is also open-domain, but assumes a global truth label and verifies claims against Wikipedia, not research papers.

In §5.3.3, we proposed claim revisions as a solution to claim / evidence specificity mismatch. Claim revision has previously been studied for fact verification over Wikipedia, with the goal of changing the claim from REFUTED to SUPPORTED or vice versa [Thorne and Vlachos, 2021; Schuster et al., 2021; Shah et al., 2020]. Previous work has also examined the related task of generating claims based on citation contexts [Wright et al., 2022] and revising questions to match the specificity of answers found in Wikipedia [Min et al., 2020].

5.7 Discussion & Conclusion

In this chapter, we introduced a new test collection, SCIFACT-OPEN, to support performance evaluation for open-domain scientific claim verification. The construction of SCIFACT-OPEN was enabled by our adaptation of the pooling strategy from IR for identification and annotation of evidence from a corpus of 500K documents. We hope such methodology can see further usage on other NLP tasks for which exhaustive annotation is infeasible.

In analyzing the evidence in SCIFACT-OPEN (§5.3.3), we found that some claims possess a large amount of conflicting evidence, and that evidence may not always match the specificity of the claims as written. We consider two future directions to improve the expressiveness of scientific claim verification. (1) As discussed in §5.3.3, one could still require systems to label each ECAP, but also to generate a revised claim matching the specificity of each evidence abstract. This output would provide users with fine-grained information

indicating the conditions under which an input claim is likely to hold. We release 91 claim revisions, which can be used to facilitate exploratory research in this direction. (2) One could use the evidence identified by a claim verification system as input into a summarization system [DeYoung et al., 2021; Wallace et al., 2021] — potentially using additional quality criteria (e.g. citation count, publication venue) to filter or re-weight the articles included in the summary. This approach has the benefit of providing a concise summary to the user, but there is a greater risk of hallucination [Maynez et al., 2020].

Overall, our analysis indicates that evaluations using SCIFACT-OPEN can provide key insights into modeling challenges associated with scientific claim verification. In particular, the performance of existing models declines substantially when evaluated on SCIFACT-OPEN, suggesting that current claim verification systems are not yet ready for deployment at scale. It is our hope that the dataset and analyses presented in this chapter will facilitate future modeling improvements, and lead to substantial new understanding of the scientific claim verification task.

Chapter 6

Entity-Centric Query Refinement

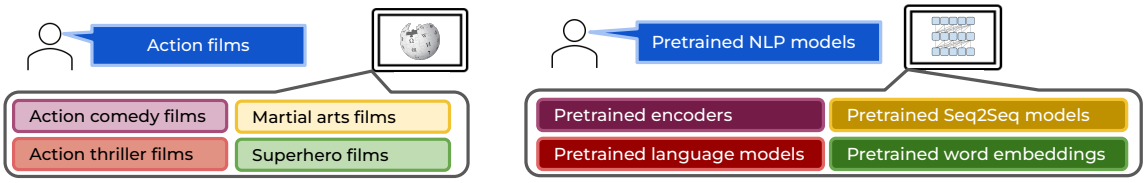
This chapter contains material that was originally published in Wadden et al. [2022a].

6.1 Introduction

During interactive search, the system user may issue queries that are under-specified, ambiguous, or open-ended. For instance, a user interested in finding a new movie might search for "Action films", or a computer vision researcher interested in learning more about NLP might search for "Pretrained NLP models". These forms of interaction are examples of *exploratory search* [Marchionini, 2006; White and Roth, 2009].

We focus specifically on queries whose answer is a list of entities, known as *list-intent* queries. For example, "Rush Hour" is one of the thousands of answers to the query "Action films". List-intent queries are common, comprising 10% of all web searches [Chakrabarti et al., 2020]. However, simply displaying the answer to such a query (e.g. a list of all action films) is more likely to cause confusion than to satisfy the user's information needs. Instead, systems for *query refinement* – also known as *query recommendation* or *query suggestion* [Sordoni et al., 2015; Baeza-Yates et al., 2004] – can assist users by offering a set of followup queries that clarify and focus the user's search, progressively drilling down on the topics and entities of most interest (Fig. 6.1).

With this motivation in mind, we propose the task of *entity-centric query refinement*. The task input is a list-intent query. The task output is a collection of k *query refinements*, referred to as a *refinement set*. The refinement set should provide a reasonably comprehensive overview of the set of entities answering the input



(a) A query paired with a refinement set consisting of $k = 4$ query subtypes available in an existing knowledge base.

(b) Refinements for a query unlikely to be covered by existing taxonomies, generated by a text-to-text model.

Figure 6.1: Examples of the entity-centric query refinement task. We propose a method to select high-quality refinement sets for queries covered by an existing taxonomy (Fig. 6.1a), and use these refinement sets to train a model which can generate refinements for queries unlikely to be covered by any taxonomy (Fig. 6.1b).

query. For instance, Fig. 6.1b shows an example of $k = 4$ refinements that could familiarize the system user with some common types of pretrained NLP models, and point the user in interesting new search directions.

To our knowledge, no datasets are available which provide instances of list-intent queries paired with refinement sets satisfying our task goals. Therefore, we propose a method to create query / refinement set pairs which can be used to train a model for this task. We leverage the YAGO3 [Mahdisoltani et al., 2015] knowledge base, using YAGO entity types as training queries. YAGO types are based on the Wikipedia category system, which provides a rich, crowdsourced taxonomy of real-world entity types. Given a YAGO type, we consider all subtypes in the taxonomy as potential refinements (Fig. 6.1a shows an example). We propose a method **Q**uery **R**efinement via **E**ntity **S**pace **P**artitioning (`QRESP`), which selects as refinements the k subtypes which provide the most comprehensive and non-redundant summary of the entities answering the input query. In head-to-head comparisons, we find that human annotators prefer refinement sets chosen using our proposed method over refinement sets consisting of k randomly-chosen subtypes of the input query.

We use the resulting dataset to train a T5 model [Raffel et al., 2020] capable of generating a refinement set for any input query. Since no evaluation dataset is available, we perform comparisons on both in-domain queries (held-out categories from the YAGO taxonomy) as well as out-of domain queries selected from Natural Questions [Kwiatkowski et al., 2019] and the TREC 2009 Million Query Track [Carterette et al., 2009]. We find that the outputs of a model trained on `QRESP` refinement sets are preferred over the outputs of a model trained on random query subtypes, suggesting that the properties captured by `QRESP` are generalizable to new queries. However, we also find that our models sometimes predict off-topic or irrelevant refinements, particularly on queries from Natural Questions and TREC. This points toward the need for future work

to improve the reliability of refinement systems under domain shift, and to develop automated metrics of refinement quality which can be used to speed up the model development process.

In summary, our contributions are threefold: (1) We introduce the task of entity-centric query refinement, and outline key desiderata and evaluation criteria for this task. (2) We propose QRESP, which optimizes an entity-centric cost function to select refinement sets for queries covered by an existing knowledge base. The resulting refinement sets can be used both for entity exploration within the knowledge base, and as instances to train a refinement set generation model. (3) We develop a baseline model trained on instances selected by QRESP to generate refinement sets for queries unseen during training, and identify important areas for future work on this task based on analysis of our system outputs.

6.2 Task definition

6.2.1 Entity-centric query refinement

We aim to generate refinements which facilitate exploration and discovery for list-intent queries, helping the user drill down on entities of interest. Formally, the task input is a query q whose answer is a list of entities. Equivalently, the query q specifies an entity type. We refer to the list of entities answering q as the *answers* to q, or $\mathcal{A}(q)$. The task output is a collection of k *refinements*[1] $\mathcal{R}(q) = \{q'_1, \ldots, q'_k\}$. We refer to $\mathcal{R}(q)$ as a *refinement set*, or "RS". Each refinement q'_i should itself be a list-intent query. In addition, each answer to q'_i should be among the answers to q: $\mathcal{A}(q'_i) \subseteq \mathcal{A}(q)$. In other words, each q'_i should specify a subtype of q. For instance, every movie that is an answer to the refinement "martial arts films" is also an answer to the input query "action films".

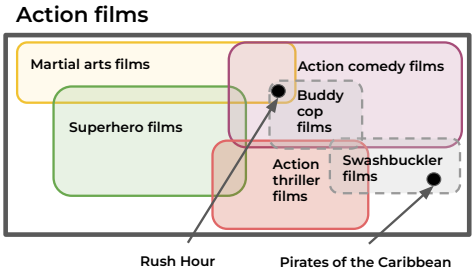


Figure 6.2: An entity-space view of a refinement set for the query "Action films". Rectangles indicate refinements, and black circles indicate entities. The four rectangles with solid borders correspond to the refinements in Fig. 6.1. The rectangles with dashed borders show two subgenres that were not included in the refinement set, since they are redundant / do not cover many films. The four selected refinements approximately partition the answer space.

[1] In this chapter, we provide k as a model input; dynamically choosing k based on the query represents an important future research direction.

6.2.2 Desiderata for refinement sets

The query refinement task is inherently open-ended, and there may be many reasonable RSs (refinement sets) for any given query. Inspired by prior work on faceted search interfaces for the Wikipedia category taxonomy [Li et al., 2010], we conceptualize refinement generation from the standpoint of *entity discovery*: given an input query q, can we design a refinement system such that any entity $e^* \in \mathcal{A}(q)$ is discoverable after a few rounds of system interaction? From this standpoint, the best-possible refinement set would partition the entities $\mathcal{A}(q)$ into k disjoint, equally-sized subsets, such that each answer $e_j \in \mathcal{A}(q)$ is an answer to exactly one refinement $q'_i \in \mathcal{R}(q)$. This would ensure that any entity in $\mathcal{A}(q)$ is discoverable after at most $\log_k(|\mathcal{A}(q)|)$ refinements, since the refinements would induce a k-ary search tree over the entities answering q (see Appendix B.1). We refer to such a refinement set as *ideal*.

Query: Action films		
Refinement	Fluent	Relevant
Martial arts films	✓	✓
Romance films	✓	✗
Martially flims arts of	✗	✗

Table 6.1: Stage 1 evaluation screens out individual refinements which are not fluent and relevant.

In practice, it will almost never be possible to generate an ideal RS, since each refinement must specify a semantic category expressed in natural language. Fig. 6.2 provides an example showing how the refinements for the query "action films" from Fig. 6.1 provide a good approximation to an ideal RS. We formalize this notion in §6.3.

6.2.3 Evaluation criteria

Efficient entity discovery provides motivation for our task formulation, but does not admit a simple evaluation strategy. Therefore, to assess the usefulness of proposed refinement sets, we conduct A / B tests where annotators compare two competing refinement sets $\mathcal{R}_{\text{A}}(q)$ and $\mathcal{R}_{\text{B}}(q)$ on a number of attributes. The evaluation includes two stages.

Stage 1: Validity of individual refinements First, the annotator confirms that the individual refinements making up $\mathcal{R}_{\text{A}}(q)$ and $\mathcal{R}_{\text{B}}(q)$ conform to the task definition, requiring:

1. **Fluency**: Each refinement must be fluent and grammatical.

Query: Action films				
	Refinement set	Comprehensive	Interesting	Non-redundant
1	Action comedies, Action thrillers, Martial arts films, Spy films	✓	✓	✓
2	Action films set in {Asia, North America, Africa, Europe}	✓	✗	✓
3	Karate films, Kung Fu films, Boxing films, Films with boxing	✗	✓	✗

Table 6.2: Stage 2 evaluation assesses the overall quality of refinement sets. The notation "{Asia, North America, ...}" means "Action films set in Asia, Action films set in North America, ...". Row (2) is comprehensive since many action films take place on one of the listed continents, but is not interesting since many different kinds of queries can be categorized by continent. Row (3) is redundant and not comprehensive since it only covers martial arts movies, and repeats "boxing films". Human evaluation makes comparisons between two refinement sets, rather than binary ✓/✗ decisions for a single set; we show binary decisions for illustration.

2. **Relevance**: Each refinement must specify a subtype of q. Non-fluent refinements are automatically judged as not relevant.

Table 6.1 provides some examples of queries that pass and fail these requirements. If fewer than half of the refinements from either RS satisfy the criteria, annotation stops here. In our experiments (§6.5), we find that virtually all refinements are fluent, and the Stage 1 screen serves in practice to filter out irrelevant refinements.

Stage 2: Overall refinement set quality If the majority of refinements in both $\mathcal{R}_{\text{A}}(q)$ and $\mathcal{R}_{\text{B}}(q)$ are judged valid, the annotator compares the two RSs, based on four attributes:

1. **Comprehensiveness**: Does $\mathcal{R}(q)$ provide a good overview of the entities answering the query q?
2. **Interestingness**: Do the refinements in $\mathcal{R}(q)$ provide new information about the different kinds of entities answering q, or are they generic and uninteresting?
3. **Non-redundancy**: Does each refinement in $\mathcal{R}(q)$ specify a unique entity type, or are some of them redundant?
4. **Overall usefulness**: Overall, how useful are the refinements $\mathcal{R}(q)$ for learning more about the entities answering q?

Table 6.2 provides examples. For each attribute, the annotator selects whether $\mathcal{R}_{\text{A}}$ is better, $\mathcal{R}_{\text{B}}$ is better, or whether the two RSs are equally good. Details of the annotation process are provided in §6.4.2.

6.3 Refinement set selection

To build a baseline system for RS generation, we proceed as follows: (1) Leverage an existing knowledge base to create a collection of query / RS pairs that are close to ideal, and (2) Train a text-to-text model on this collection, with the goal of generalizing to queries not covered by a taxonomy. We describe (1) in this section and §6.4, and describe (2) in §6.5.

Source taxonomy We use the YAGO3 taxonomy [Suchanek et al., 2007; Mahdisoltani et al., 2015] (referred to simply as YAGO) as our source to construct a training dataset. The YAGO entity type system is adopted from the Wikipedia category system, a crowdsourced polyhierarchy used to organize Wikipedia pages. We use types in the YAGO taxonomy as input queries q. Given a type q, we consider all sub-types of q in the taxonomy as refinement *candidates*, denoted $\mathcal{C}(q)$. From this collection of K candidates, we aim to select k sub-types to form our refinement set $\mathcal{R}(q)$. We use the entities in the YAGO knowledge base that are instances of type q as the answers to q, or $\mathcal{A}(q)$. As a concrete example, in Fig. 6.2, "Action films" is the query q, the shaded rectangles correspond to members of the candidate set $\mathcal{C}(q)$, and the black points are two answers in $\mathcal{A}(q)$.

Even when a list of K candidate subtypes is available, selecting the best k refinements is challenging because (1) K may be large; for example, there are 68 subtypes of the type "Action films", (2) Many of the candidate subtypes may be too specific to be part of a useful summary (e.g. "Tomb Raider films" is a subtype of "Action films"), and (3) subtypes may be redundant, overlapping, or generic.

`QRESP` for refinement selection To select an RS of k refinements from among the K candidate subtypes associated with a given input query, we propose `QRESP`: **Q**uery **R**efinement via **E**ntity **S**pace **P**artitioning. We define a cost function which selects RSs according to the desiderata described in §6.2.2, rewarding RSs whose answers approximately partition the answers to the original query. Then, we minimize this cost function over all refinement sets.

Let e_j denote a particular entity in $\mathcal{A}(q)$, and let $n = |\mathcal{A}(q)|$. Given a pool of candidate refinements $\mathcal{C}(q)$, let $\mathfrak{R}(q)$ indicate the collection of all size-k subsets of $\mathcal{C}(q)$. Given a possible refinement set $\mathcal{R}(q) \in \mathfrak{R}(q)$, and a refinement $q'_i \in \mathcal{R}(q)$, let $a_{ij} = \mathbb{1}\left[e_j \in \mathcal{A}(q'_i)\right]$. Define $c_j = \sum_{i=1}^{k}(a_{ij})$, the number of refinements

in $\mathcal{R}(q)$ for which entity j is an answer. Let $n_i = \sum_{j=1}^{n}(a_{ij}) = |\mathcal{A}(q'_i)|$, the number of entities that answer refinement i. Then we define a cost function $\mathcal{S}$ measuring the quality of a given $\mathcal{R}(q)$, and minimize over $\mathfrak{R}(q)$:

$$\mathcal{S}(\mathcal{R}(q)) = \left[\sum_{j=1}^{n} |c_j - 1|\right] - \min_{i \in \{1,\ldots,k\}} n_i \tag{6.1}$$

$$\mathcal{R}^*(q) = \arg\min_{\mathfrak{R}(q)} \mathcal{S}(\mathcal{R}(q)). \tag{6.2}$$

The first term in Eq. 6.1 is minimized when each answer to q is an answer to exactly one of the q'_i, encouraging comprehensiveness and non-redundancy. The second term encourages the smallest refinement to have as many answers as possible. Combined with the first term, this rewards the selection of refinements which all have a similar number of answers.

Eq. 6.2 selects the refinement set $R^*(q)$ minimizing Eq. 6.1. In Appendix B.2, we provide a proof that the RS selected by QRESP is the *best achievable* in the following sense: the scoring function $\mathcal{S}(\mathcal{R}(q))$ achieves its global minimum if and only if $\mathcal{R}(q)$ is *ideal* as defined in §6.2.2. Thus, selecting refinements according to QRESP yields the RS that is closest to ideal, given the subcategories $\mathcal{C}(q)$ available in YAGO. We use integer linear programming to perform the combinatorial optimization over $\mathfrak{R}(q)$ in Eq. 6.2; see Appendix B.3 for details.

6.4 A dataset for entity-centric query refinement

We apply QRESP to YAGO to create a dataset for entity-centric query refinement, and conduct A / B tests to assess whether our approach aligns with human judgments. We set the number of refinements to $k = 5$ for all experiments; this is large enough to enable diverse refinement sets, but not so large as to be overwhelming for users.

6.4.1 Dataset creation

We apply QRESP to select RSs from the YAGO taxonomy. We use YAGO types with at least 50 answer entities as our training queries, as it may be difficult to select non-trivial refinements for types with only a few

Query: Action films	
$\mathcal{D}_{\texttt{QRESP}}$	{Action comedy, Action thriller, Martial arts, Science fiction action, Spy} films
$\mathcal{D}_{\text{Random}}$	{1910s, 1920s, Australian, Brazilian, Nigerian} action films
$\mathcal{D}_{\text{Random-F}}$	{Action comedy, Action thriller, Animated action, The purge} films, Action films based on actual events.

Table 6.3: Refinement sets from $\mathcal{D}_{\texttt{QRESP}}$, $\mathcal{D}_{\text{Random}}$, and $\mathcal{D}_{\text{Random-F}}$. $\mathcal{D}_{\text{Random}}$ includes many refinements based on a time period or a country of origin, which does not provide interesting new information about the topic. $\mathcal{D}_{\text{Random-F}}$ is overly specific (e.g. "The Purge films"), and thus does not provide as good an overview as $\mathcal{D}_{\texttt{QRESP}}$.

answers. Given a type q with candidate subtypes $\mathcal{C}(q)$, we use rule-based filters to remove candidates that differ from q only by the addition of a date, location, or a gender (e.g. "singers" $\rightarrow$ "female singers"), as these tend to lead to generic refinements. We refer to the filtered collection as $\mathcal{C}_{\text{Filter}}(q)$. Queries with fewer than k subtypes post-filtering are removed. For the remaining queries, we apply QRESP to the candidates $\mathcal{C}_{\text{Filter}}(q)$ to select refinements $\mathcal{R}_{\texttt{QRESP}}(q)$. We call the resulting dataset $\mathcal{D}_{\texttt{QRESP}} = (q_i, \mathcal{R}_{\texttt{QRESP}}(q_i))_{i=1}^{N}$. We also construct two ablation datasets for comparison. For $\mathcal{D}_{\text{Random}}$, we select refinements by randomly choosing k subtypes from $\mathcal{C}(q)$, without filtering. For $\mathcal{D}_{\text{Random-F}}$, we choose k random subtypes from $\mathcal{C}_{\text{Filter}}(q)$. Table 6.3 provides examples. Our training dataset consists of $8,958$ query / RS instances for $\mathcal{D}_{\texttt{QRESP}}$ and $\mathcal{D}_{\text{Random-F}}$, and $17,598$ instances for $\mathcal{D}_{\text{Random}}$.

We hold out 282 query / RS instances to be used for model development in §6.5. Development queries and their subtypes are excluded from the train set. We select dev set queries by randomly sampling YAGO types with at least 15 subtypes, each of which must have at least 200 answer entities. We use categories with many subtypes and answers because these are the categories for which query refinement has the most potential to facilitate user exploration and discovery. From our development set, we manually select a collection of 105 examples to be used for human evaluation in A / B tests. We select a diverse set of interesting, open-ended queries across a variety of domains, intended to mirror the types of queries likely to be seen in real-world use.

6.4.2 Human evaluation

We compare refinements from $\mathcal{R}_{\texttt{QRESP}}$ with refinements chosen randomly, $\mathcal{R}_{\text{Random}}$. In addition, we compare $\mathcal{R}_{\texttt{QRESP}}$ to $\mathcal{R}_{\text{Random-F}}$, to confirm that the improved ratings are due to selection using QRESP, and not simply the filtering out of uninteresting candidates.

A / B test results are shown in Table 6.4. We perform statistical significance tests for all human

		A = QRESP vs. **B** = Random			**A** = QRESP vs. **B** = Random-F		
	$N = 105$	**A**	Neutral	**B**	**A**	Neutral	**B**
Stage 1	Fluent + Relevant	**98%**	-	97%	100%	-	100%
Stage 2	Comprehensive	**88%****	10%	2%	**75%****	15%	10%
	Interesting	**71%****	26%	3%	**64%****	24%	12%
	Non-redundant	-	-	-	-	-	-
	Overall	**86%****	10%	4%	**73%****	17%	10%

Table 6.4: A / B tests comparing refinement sets from $\mathcal{D}_{\texttt{QRESP}}$ against $\mathcal{D}_{\text{Random}}$ (left) and $\mathcal{D}_{\text{Random-F}}$ (right). N indicates the number of annotated instances. For Stage 1 evaluation, we report the percentage of refinement sets from each system that passed the Stage 1 screen. For Stage 2 evaluation, we report the percentage of queries for which annotators preferred refinements from System A vs. System B. The "non-redundant" evaluation was added after these tests were performed, and is left blank. ** indicates significance at $p < 0.001$.

comparisons. More than 97% of RSs from all approaches pass the Stage 1 screen for relevance and fluency. This is expected, since the refinements for training queries come from the YAGO taxonomy. In Stage 2, RSs selected by QRESP are preferred by annotators 86% of the time over random RSs, and 73% of the time over random RSs post-filtering. $\mathcal{R}_{\texttt{QRESP}}$ is preferred more frequently for comprehensiveness than for interestingness, likely because Eq. 6.1 explicitly rewards comprehensiveness. Overall, the results indicate that QRESP captures human intuitions about refinement quality.

6.5 Refinement generation

Having established that QRESP selects high-quality refinement sets for queries covered by a knowledge base, we finetune a pretrained language model to generate refinements for queries not found in the KB. We experiment with two evaluation datasets: held-out categories from the YAGO taxonomy, and a collection of list-intent queries selected from Natural Questions [Kwiatkowski et al., 2019] and the TREC 2009 Million Query Track [Carterette et al., 2009].

6.5.1 Model

We use T5-3B [Raffel et al., 2020] as our base model. Given a dataset $\mathcal{D}$ consisting of training pairs $(q_i, \mathcal{R}(q_i))_{i=1}^{N}$, we provide q as the input for T5 and train it to generate $\mathcal{R}(q)$. We format $\mathcal{R}(q)$ by concatenating the individual refinements in alphabetical order, separated by a sentinel token. At prediction time, we provide q as input and greedily decode (greedy outperformed sampling and beam search on automated

Model	Sequence		Set			Perplexity
	BLEU	ROUGE-L	P	R	F1	
$\mathcal{M}_{\texttt{QRESP}}$	**67.1**	**69.4**	**24.8**	**24.3**	**24.6**	**2.01**
$\mathcal{M}_{\text{Separate}}$	66.9	68.2	19.6	19.2	19.4	2.35
$\mathcal{M}_{\text{Random-F}}$	61.5	66.0	15.1	14.3	14.7	2.11
$\mathcal{M}_{\text{Random}}$	57.6	63.8	6.8	6.7	6.8	2.19

Table 6.5: Automated evaluation of generation models, using $\mathcal{D}_{\texttt{QRESP}}$ as "silver" evaluation targets. Evaluations are categorized into Sequence-based, Set-based, and Perplexity-based. $\mathcal{M}_{\texttt{QRESP}}$ outperforms models trained on randomly-chosen refinements, or trained to generate refinements separately rather than as a single sequence.

metrics). We train models on $\mathcal{D}_{\texttt{QRESP}}$, $\mathcal{D}_{\text{Random-F}}$, and $\mathcal{D}_{\text{Random}}$; we call these $\mathcal{M}_{\texttt{QRESP}}$, $\mathcal{M}_{\text{Random-F}}$, and $\mathcal{M}_{\text{Random}}$, respectively. We train with a batch size of 32 for 8000 steps, using the default T5 optimizer.

As an additional ablation, we train a model $\mathcal{M}_{\text{Separate}}$ on $\mathcal{D}_{\texttt{QRESP}}$, which predicts a single refinement q'_i at a time, rather than predicting a full refinement set $\mathcal{R}(q)$. At prediction time, given an input q, we sample 5 separate predictions from $\mathcal{M}_{\text{Separate}}$ and concatenate to form $\mathcal{R}(q)$; a similar approach is used by MacAvaney et al. [2021] to generate possible query intents for search result diversification.

6.5.2 Automated evaluations

The results of §6.4 indicate that human annotators prefer refinements from $\mathcal{D}_{\texttt{QRESP}}$ over those from $\mathcal{D}_{\text{Random}}$ and $\mathcal{D}_{\text{Random-F}}$. Therefore, for our automated evaluations, we treat RSs from the $\mathcal{D}_{\texttt{QRESP}}$ dev set as "silver" data against which to evaluate the predictions of our trained models. We evaluate using the following metrics:

1. **Sequence-based metrics**: We treat a generated RS as a text sequence with no additional structure, and compare it to the corresponding silver RS using BLEU and ROUGE-L (ROUGE-1 and -2 showed the same trend).
2. **Set-based metrics**: We treat the generated RS as a set of k refinements, and evaluate the precision, recall, and F1 relative to the set of silver refinements.
3. **Perplexity**: We evaluate the perplexity of the silver RSs, formatted as a sequence, under each generation model.

The results are shown in Table 6.5. $\mathcal{M}_{\texttt{QRESP}}$ performs best on all metrics. Training on randomly-chosen candidate subtypes ($\mathcal{M}_{\text{Random-F}}$ and $\mathcal{M}_{\text{Random}}$) decreases performance, particularly as measured by

	Query: Physicians
$\mathcal{M}_{\texttt{QRESP}}$	Alternative medicine physicians, cardiologists, dermatologists, ophthalmologists, psychiatrists
$\mathcal{M}_{\text{Random}}$	{Canadian, German, Norwegian} dermatologists, Pediatricians, Women physicians
$\mathcal{M}_{\text{Random-F}}$	Fictional physicians, Oncologists, Psychiatric physicians, Radiologists, surgeons
$\mathcal{M}_{\text{Separate}}$	{Atheist, Baritone, Fictional, Military} physicians, Neurologists

Table 6.6: Refinements of $\mathcal{M}_{\texttt{QRESP}}$ and three ablations on a YAGO evaluation query. The $\mathcal{M}_{\texttt{QRESP}}$ suggestions cover 5 common types of physicians. In contrast, some ablation refinements are generic (Canadian dermatologists) or idiosyncratic (Fictional physicians).

		A = QRESP vs. **B** = Random			**A** = QRESP vs. **B** = Separate		
	$N = 105$	**A**	Neutral	**B**	**A**	Neutral	**B**
Stage 1	Fluent + Relevant	93%	-	**100%***	93%	-	**95%**
Stage 2	Comprehensive	**77%****	12%	11%	**47%**	20%	33%
	Interesting	**70%****	22%	8%	**28%**	45%	27%
	Non-redundant	**22%**	66%	12%	**42%****	45%	14%
	Overall	**76%****	17%	7%	**46%***	26%	28%

Table 6.7: Results of A / B tests on the YAGO human evaluation set. $\mathcal{M}_{\texttt{QRESP}}$ is preferred over both ablations. * and ** indicate significance at $p < 0.05$ and $p < 0.001$, respectively.

F1. We also observe that sequential refinement set generation by $\mathcal{M}_{\texttt{QRESP}}$ outperforms separate prediction of individual refinements by $\mathcal{M}_{\text{Separate}}$. We examine the reasons for this improvement in §6.5.3.

6.5.3 Human evaluation

We conduct A / B tests on two datasets. First, we evaluate on the (in-domain) human evaluation set of YAGO types described in §6.4.1. Second, to measure the ability to generalize to *out-of-domain* queries, we evaluate on 93 real-world list-intent queries selected by one of the paper authors from the Natural Questions dataset and the TREC 2009 Million Query Track (referred to as NQ+TREC). As with YAGO, we aimed to select exploratory, list-intent queries on a variety of topics.

Results on YAGO We compare refinements from $\mathcal{M}_{\texttt{QRESP}}$ against $\mathcal{M}_{\text{Random}}$, $\mathcal{M}_{\text{Random-F}}$ and $\mathcal{M}_{\text{Separate}}$. Table 6.6 shows the predictions of each system on a single query. Table 6.7 shows the results of A / B tests comparing $\mathcal{M}_{\texttt{QRESP}}$ against $\mathcal{M}_{\text{Random}}$ and $\mathcal{M}_{\text{Separate}}$; results for $\mathcal{M}_{\text{Random-F}}$ are similar to $\mathcal{M}_{\text{Random}}$ and are included in Appendix **??**.

Compared to $\mathcal{M}_{\text{Random}}$, refinements from $\mathcal{M}_{\texttt{QRESP}}$ are much more likely to be comprehensive and

Query: Functions of government	
$\mathcal{M}_{\texttt{QRESP}}$	{Agricultural, Defense, Education, Finance, Foreign} functions of the government
$\mathcal{M}_{\text{Random}}$	Functions of the {Canadian, Yukon, Quebec, Northern Mariana Islands, United States} government

(a) $\mathcal{M}_{\texttt{QRESP}}$ offers a list of five interesting, diverse government functions. $\mathcal{M}_{\text{Random}}$ provides a generic list of five locations, including two Canadian provinces.

Query: Popular YouTube Channels	
$\mathcal{M}_{\texttt{QRESP}}$	{Celebrity YouTube, Music video, Religious YouTube, Religious television, Religious video} channels
$\mathcal{M}_{\text{Random}}$	{American, Australian, British, Japanese, Pakistani} popular YouTube channels

(b) Three $\mathcal{M}_{\texttt{QRESP}}$ refinements were judged irrelevant since they don't mention YouTube specifically. $\mathcal{M}_{\text{Random}}$ makes five generic, but relevant, refinements by once against listing five locations.

Table 6.9: Predictions on two queries from NQ+TREC. $\mathcal{M}_{\texttt{QRESP}}$ provides high-quality refinements for the first query, but is slightly off-topic for the second one.

interesting. On the other hand, the models are similar in terms of non-redundancy, since $\mathcal{M}_{\text{Random}}$ tends to offer overly-specific refinements which provide a poor summary but do not overlap. $\mathcal{M}_{\texttt{QRESP}}$ and $\mathcal{M}_{\text{Separate}}$ have nearly identical interestingness. However, $\mathcal{M}_{\texttt{QRESP}}$ enjoys a large advantage in non-redundancy, since it can condition each new refinement on earlier refinements in the RS.

Results on NQ+TREC We compare predictions from $\mathcal{M}_{\texttt{QRESP}}$ vs. $\mathcal{M}_{\text{Random}}$. Interestingly, only 56% of RSs generated by $\mathcal{M}_{\texttt{QRESP}}$ pass the Stage 1 screen, compared to 81% for $\mathcal{M}_{\text{Random}}$. However, the RSs from $\mathcal{M}_{\texttt{QRESP}}$ that pass the screen are preferred over RSs from $\mathcal{M}_{\text{Random}}$ in terms of comprehensiveness, interestingness, and overall quality (Table 6.8). While the differences do not reach the threshold of statistical significance, this trend suggests that the two models behave differently on out-of-domain queries. $\mathcal{M}_{\texttt{QRESP}}$ refinements can be interesting and informative (Table 6.9a), but sometimes go off-topic (Table 6.9b). $\mathcal{M}_{\text{Random}}$ tends to make "safe" refinements that are generic but relevant.

	A = QRESP vs. **B** = Random		
$N = 93$	**A**	Neutral	**B**
Fluent + Relevant	56%	-	**81%****
Comprehensive	**45%**	30%	26%
Interesting	**40%**	34%	26%
Non-redundant	23%	51%	**26%**
Overall	**43%**	28%	30%

Table 6.8: Human evaluations on NQ+TREC. $\mathcal{M}_{\texttt{QRESP}}$ refinements tend to be more interesting and comprehensive, provided that they are relevant. ** indicates $p < 0.001$.

6.6 Related work

Query refinement *Search results clustering* [Carpineto et al., 2009] offers refinements by retrieving a collection of documents in response to an input query, clustering them, and assigning an informative name to each cluster. Names can be assigned based on word count statistics [Zamir and Etzioni, 1999; Osinski and Weiss, 2005], or generated using Seq2Seq models [Medlar et al., 2021]. *Faceted search* [Tunkelang, 2009; Hearst, 2006] also retrieves documents in response to the user's search, but organizes them according to a faceted concept hierarchy, which can be used as a source of refinements. The hierarchy can be created by the system designers [Yee et al., 2003], adapted from an existing taxonomy [Li et al., 2010; Arenas et al., 2016], or constructed automatically [Stoica et al., 2007].

Like our work, these approaches output a collection of query refinements, aided by an existing taxonomy in the case of faceted search. Unlike these approaches, we optimize an entity-centric objective function to select a set of refinements which provides a comprehensive summary of the input query, rather than organizing a collection of retrieved documents. In addition, unlike faceted search, our proposed baseline can provide refinements for an arbitrary input query which may not be covered by an existing taxonomy.

Systems have also been trained on *search query logs* to predict likely followup queries given a user search history [Sordoni et al., 2015; Dehghani et al., 2017; Boldi et al., 2008]. Our modeling goals differs from this setting in that (1) we do not assume access to query logs, which are often proprietary, and (2) we aim to output a set of refinements with a specific entity structure, rather than reproducing the search behavior of users.

Under-specified queries Researchers in NLP and IR have developed systems to resolve ambiguity [Min et al., 2020], incorporate dialogue context [Elgohary et al., 2019; Anantha et al., 2021], and ask questions [Aliannejadi et al., 2019; Sekulic et al., 2021; Zamani et al., 2020] to clarify user intent in response to ambiguous or multi-faceted input queries. As an alternative to clarifying user intent directly, *search results diversification* [Santos et al., 2015] predicts possible user intents with the aid of taxonomies [Agrawal et al., 2009], search logs [Santos et al., 2010], or recently transformer language models [MacAvaney et al., 2021], and then selects a collection of documents which comprehensively and non-redundantly addresses all predicted user intents [Carbonell and Goldstein-Stewart, 1998].

In general, these approaches were developed to distinguish between a handful of possible query intents, while our goal is to facilitate exploration and entity discovery for open-ended list-intent queries. Like search results diversification, we also aim for comprehensiveness and non-redundancy. However, we measure these quantities directly over sets of entities, rather than over collections of documents, whose similarity to each other and to the input query must be approximated using a metric like cosine distance.

6.7 Conclusion and future work

In this chapter, we proposed the task of entity-centric query refinement. We developed QRESP to select high-quality refinement sets which partition the set of entities answering the input query, and showed that our methodology has good agreement with human judgments. We then demonstrated that a text-to-text model trained on QRESP-selected data was able to generate refinement sets for queries not found in an existing knowledge base.

Our findings point toward two key open challenges for entity-centric query refinement. First, in §6.5.3, we found that $\mathcal{M}_{\texttt{QRESP}}$ can sometimes generate off-topic or irrelevant refinements under domain shift. This points to a need for domain adaptation techniques which do not require supervised query / refinement set pairs. A second challenge is the development of high-quality automated evaluation metrics to speed the model development process. While we experimented with a number of evaluation metrics and found that they correlated with human judgments of refinement quality (§6.5.2), the metrics we examined perform comparisons to a single reference refinement set. Given the open-endedness of the task, these approaches may not adequately reward refinements which a human would judge as high-quality, but which do not closely match the reference.

The QRESP framework has the potential to facilitate progress on both challenges. In this chapter, we used the QRESP scoring function (Eq. 6.1) to select refinement sets from a pool of candidates for which gold answer entities could be found in a knowledge base. In the future, this same scoring function could be used to evaluate refinement sets proposed by a generation model, using an entity-centric QA model [Ling et al., 2020; Févry et al., 2020] to predict the list of entities answering each refinement. This QRESP-QA score would serve as a flexible automated metric to compare the performance of refinement generation models.

Further, the availability of a flexible automated metric for refinement set quality could be leveraged to

improve out-of-domain generalization. For instance, the QRESP-QA score could be used as a reward signal to be optimized via reinforcement learning; this approach could steer the model away from generating irrelevant refinements. Similarly, QRESP-QA could be used at inference time to re-rank a list of generated candidate refinements, filtering out irrelevant or off-topic candidates.

In summary, we believe that the entity-centric query refinement task presents a number of exciting research avenues for researchers. We hope that our dataset, modeling baselines, and analysis will motivate future work on this challenging and relevant task.

Chapter 7

Conclusion

7.1 Summary

In this thesis, we reported progress on three real-world knowledge-intensive NLP tasks, motivated by the goal of scientific understanding and discovery, and sharing a common set of research challenges (Chapter 1). We summarize our contributions in terms of these challenges.

1. **Non-standard task formulations**: We proposed the task of scientific claim verification (Chapter 3), and updated the conventional fact-checking formulation for our setting by requiring systems to retrieve *all* relevant evidence. Further, we proposed the task of entity-centric query refinement (Chapter 6), and proposed human evaluation procedures to assess the quality of generated refinement sets.
2. **Scarce labeled data**: For scientific claim verification, we proposed to leverage citation sentences as a scalable source of claims linked to evidence (Chapter 3), and proposed a pooled data collection strategy to obtain good annotation coverage over a large scientific corpus (Chapter 5). We also proposed heuristics enabling us to perform weakly-supervised domain adaptation (Chapter 4). For entity-centric query refinement, we proposed `QRESP`, a programmatic weak supervision approach to obtain training data from an existing concept taxonomy without collecting any labeled examples (Chapter 6).
3. **Novel linguistic phenomena** For scientific information extraction, we proposed the DyGIE++ architecture to capture cross-sentence linguistic context and accommodate overlapping and nested entity mentions (Chapter 2). For scientific fact-checking, we developed MultiVerS for full-document scientific claim

verification (Chapter 4).

We expect that the challenges encountered in this thesis will be common to many knowledge-intensive tasks in realistic settings. and we hope that the approaches developed herein can serve as useful examples for other researchers pursuing similar goals.

7.2 Future directions

We discuss some possible extensions of the work presented in this thesis, and then briefly offer some thoughts on this work with respect to larger developments in NLP, particularly large language models.

7.2.1 Extensions

For scientific fact-checking, one important future direction relates to claim *ambiguity*, which we refer to as *claim / evidence specificity mismatch* in Chapter 5. As a concrete example, a claim like "alcohol consumption increases cancer risk" admits many possible sub-claims specifying the type of alcohol being consumed, the form of cancer, and the population being studied. Instead of simply returning a list of documents supporting and refuting the original (overly general) claim, a better approach might be to interactively *refine* the user's claim — not unlike the approach taken for entity-centric query refinement in Chapter 6. Another important direction would be to *explain* the relationship between a claim and a piece of evidence, or between two pieces of evidence — for instance "Document A studies the effect of drinking on lung cancer, while Document B studies its effects on liver cancer" [Luu et al., 2021]. This could be further extended to explaining relationships between clusters of related papers. The biggest challenge presented by these directions is arguably the evaluation: do humans agree on the aspects of different claims or papers that are worthy of comparison? Can these intuitions be operationalized into automated metrics, or via learned models of human preferences?

For automated fact-checking more generally, another key research challenge relates to the modeling of *deception by omission*. In particular, media outlets generally will not report claims that are patently false, but they may omit key facts that bias the reader's interpretation of the reported information. As an example,

consider the newspaper headline "University creates a new COVID strain that has an 80% kill rate"[1]. This claim sounds quite alarming, because it omits two key pieces of information: (1) The reported kill rate is in lab mice, and (2) Wild-type (ordinary) COVID killed 100% of lab mice. Detecting this form of deception raises interesting challenges related to commonsense knowledge and reasoning [Bhagavatula et al., 2020], because it requires understanding the likely assumptions that a human reader would bring to bear when interpreting the headline: namely, that the reported kill rate refers to death among humans.

One other major research direction for scientific NLP centers around multimodality. A source of motivation for the work in this thesis was that we might be able to leverage the knowledge found in scientific literature to accelerate scientific discovery. As a concrete example, we might search for molecules capable of treating a disease by formulating the problem as a knowledge graph completion task: the molecule and disease are represented as graph nodes, and an edge indicates that the molecule treats the disease [Bonner et al., 2021; Nadkarni et al., 2021; Zitnik et al., 2018]. The molecule in this example can be represented by (1) Its IUPAC name (2) Its SMILES string, (3) A graph of its 3-d structure, or (4) A textual description of its properties [Edwards et al., 2022]. While there has been promising recent work on translating between these modalities — most prominently the ill-fated Galactica [Taylor et al., 2022] — it remains an open question how best to represent structured scientific objects in a way that leverages knowledge expressed in the literature.

7.2.2 Broader developments in NLP

In the last two years, very large language models [Brown et al., 2020; Chowdhery et al., 2022; Zhang et al., 2022; Scao et al., 2022] have changed the landscape of NLP. These models exhibit an ability to accomplish many tasks after being shown a small handful of demonstrations, and to perform multi-step reasoning in-context (referred to as *chain-of-thought* [Wei et al., 2022]). Within a week of the defense date for this thesis, major tech companies have announced that these systems will be integrated into web search engines [Roose]. Despite this extremely rapid progress, LLMs often generate text that is not factually accurate (particularly when discussing long-tail entities), and can exhibit bias against protected classes of individuals.

In the long term, we can hope for the development of new classes of NLP models which possess innate

[1] `https://www.politifact.com/factchecks/2022/oct/20/instagram-posts/claim-boston-university-created-covid-19-strain-80/`

notions of truthfulness, factuality, and ethics. In the shorter term, it is possible that the sorts of tasks and systems presented in this work could serve as "guard rails" to perform automatic checks on the outputs of LLMs. For instance, suppose an LLM is used to generate a summary of a scientific paper; one can run an IE system on the generated summary, extract entity mentions, and confirm that each entity mentioned in the summary was mentioned in the source document [DeYoung et al., 2021]. Similarly, fact-checkers can be used to attribute the text generated by a language model to sources found on the web [Bohnet et al., 2022]. Thus, progress on tasks like IE and fact checking may translate into more robust and reliable LLM-based interactive systems. We expect that these sorts of guard rails will be important in practice as LLMs begin to see widespread adoption.

Bibliography

Rakesh Agrawal, Sreenivas Gollapudi, Alan Halverson, and Samuel Ieong. 2009. Diversifying search results. In *WSDM*.

Mohammad Aliannejadi, Hamed Zamani, Fabio A. Crestani, and W. Bruce Croft. 2019. Asking Clarifying Questions in Open-Domain Information-Seeking Conversations. In *SIGIR*.

R. Anantha, Svitlana Vakulenko, Zhucheng Tu, S. Longpre, Stephen G. Pulman, and Srinivas Chappidi. 2021. Open-Domain Question Answering Goes Conversational via Question Rewriting. In *NAACL*.

Marcelo Arenas, Bernardo Cuenca Grau, Evgeny Kharlamov, Sarunas Marciuska, and Dmitriy Zheleznyakov. 2016. Faceted search over RDF-based knowledge graphs. *Journal of Web Semantics*.

Isabelle Augenstein, C. Lioma, Dongsheng Wang, Lucas Chaves Lima, Casper Hansen, Christian Hansen, and J. Simonsen. 2019. MultiFC: A Real-World Multi-Domain Dataset for Evidence-Based Fact Checking of Claims. In *EMNLP*.

Ricardo Baeza-Yates, Carlos A. Hurtado, and Marcelo Mendoza. 2004. Query Recommendation Using Query Logs in Search Engines. In *EDBT Workshops*.

Ramy Baly, Mitra Mohtarami, James Glass, Lluís Màrquez, Alessandro Moschitti, and Preslav Nakov. 2018. Integrating stance detection and fact checking in a unified corpus. In *NAACL*.

Alberto Barrón-Cedeño, Tamer Elsayed, Reem Suwaileh, Lluís Màrquez i Villodre, Pepa Atanasova, Wajdi Zaghouani, Spas Kyuchukov, Giovanni Da San Martino, and Preslav Nakov. 2018. Overview of the clef-2018 checkthat! lab on automatic identification and verification of political claims. task 2: Factuality. In *CLEF*.

Elaine Beller, Justin Clark, Guy Tsafnat, Clive Elliott Adams, Heinz Diehl, Hans Lund, Mourad Ouzzani, Kristina Thayer, James Thomas, Tari Turner, J. S. Xia, Karen A. Robinson, and Paul P Glasziou. 2018. Making progress with the automation of systematic reviews: principles of the international collaboration for the automation of systematic reviews (icasr). *Systematic Reviews*, 7.

Iz Beltagy, Kyle Lo, and Arman Cohan. 2019. SciBERT: A Pretrained Language Model for Scientific Text. In *EMNLP*.

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv*, abs/2004.05150.

Ksenia Bestuzheva, Mathieu Besançon, Wei-Kun Chen, Antonia Chmiela, Tim Donkiewicz, Jasper van Doornmalen, Leon Eifler, Oliver Gaul, Gerald Gamrath, Ambros Gleixner, Leona Gottwald, Christoph Graczyk, Katrin Halbig, Alexander Hoen, Christopher Hojny, Rolf van der Hulst, Thorsten Koch, Marco Lübbecke, Stephen J. Maher, Frederic Matter, Erik Mühmer, Benjamin Müller, Marc E. Pfetsch, Daniel Rehfeldt, Steffan Schlein, Franziska Schlösser, Felipe Serrano, Yuji Shinano, Boro Sofranac, Mark Turner, Stefan Vigerske, Fabian Wegscheider, Philipp Wellner, Dieter Weninger, and Jakob Witzig. 2021. The SCIP Optimization Suite 8.0. Technical report, Optimization Online.

Chandra Bhagavatula, Ronan Le Bras, Chaitanya Malaviya, Keisuke Sakaguchi, Ari Holtzman, Hannah Rashkin, Doug Downey, Scott Yih, and Yejin Choi. 2020. Abductive commonsense reasoning.

Bernd Bohnet, Vinh Quang Tran, Pat Verga, Roee Aharoni, Daniel Andor, Livio Baldini Soares, Jacob Eisenstein, Kuzman Ganchev, Jonathan Herzig, Kai Hui, Tom Kwiatkowski, Ji Ma, Jianmo Ni, Tal Schuster, William W. Cohen, Michael Collins, Dipanjan Das, Donald Metzler, Slav Petrov, and Kellie Webster. 2022. Attributed question answering: Evaluation and modeling for attributed large language models. *ArXiv*, abs/2212.08037.

Paolo Boldi, Francesco Bonchi, Carlos Castillo, Debora Donato, A. Gionis, and Sebastiano Vigna. 2008. The query-flow graph: model and applications. In *CIKM*.

Stephen Bonner, Ian P. Barrett, Cheng Ye, Rowan Swiers, Ola Engkvist, Andreas Bender, and William

Hamilton. 2021. A review of biomedical datasets relating to drug discovery: A knowledge graph perspective. *ArXiv*, abs/2102.10062.

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, T. J. Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeff Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language Models are Few-Shot Learners. In *NeurIPS*.

Daniel Fernando Campos, T. Nguyen, M. Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, L. Deng, and Bhaskar Mitra. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *NeurIPS*.

Jaime G. Carbonell and Jade Goldstein-Stewart. 1998. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *SIGIR*.

Claudio Carpineto, Stanislaw Osinski, Giovanni Romano, and Dawid Weiss. 2009. A survey of Web clustering engines. *ACM Computing Surveys*.

Ben Carterette, Virgiliu Pavlu, Hui Fang, and Evangelos Kanoulas. 2009. Million Query Track 2009 Overview. In *TREC*.

Kaushik Chakrabarti, Zhimin Chen, Siamak Shakeri, Guihong Cao, and Surajit Chaudhuri. 2020. TableQnA: Answering List Intent Queries With Web Tables. *ArXiv*, abs/2001.04828.

Sihao Chen, Daniel Khashabi, Wenpeng Yin, Chris Callison-Burch, and Dan Roth. 2019. Seeing things from a different angle: Discovering diverse perspectives about claims. In *NAACL*.

Eunsol Choi, Jennimaria Palomaki, Matthew Lamm, Tom Kwiatkowski, Dipanjan Das, and Michael Collins. 2021. Decontextualization: Making Sentences Stand-Alone. *TACL*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, Parker Schuh, Kensen Shi, Sasha Tsvyashchenko, Joshua Maynez, Abhishek Rao, Parker Barnes, Yi Tay, Noam M. Shazeer, Vinodkumar Prabhakaran, Emily Reif, Nan Du, Benton C. Hutchinson, Reiner Pope, James Bradbury, Jacob Austin, Michael Isard, Guy Gur-Ari, Pengcheng Yin, Toju Duke, Anselm Levskaya, Sanjay Ghemawat, Sunipa Dev, Henryk Michalewski, Xavier García, Vedant Misra, Kevin Robinson, Liam Fedus, Denny Zhou, Daphne Ippolito, David Luan, Hyeontaek Lim, Barret Zoph, Alexander Spiridonov, Ryan Sepassi, David Dohan, Shivani Agrawal, Mark Omernick, Andrew M. Dai, Thanumalayan Sankaranarayana Pillai, Marie Pellat, Aitor Lewkowycz, Erica Moreira, Rewon Child, Oleksandr Polozov, Katherine Lee, Zongwei Zhou, Xuezhi Wang, Brennan Saeta, Mark Díaz, Orhan Firat, Michele Catasta, Jason Wei, Kathleen S. Meier-Hellstern, Douglas Eck, Jeff Dean, Slav Petrov, and Noah Fiedel. 2022. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311.

Fenia Christopoulou, Makoto Miwa, and Sophia Ananiadou. 2018. A walk-based model on entity graphs for relation extraction. In *ACL*.

Arman Cohan, Luca Soldaini, and Nazli Goharian. 2015. Matching citation text and cited spans in biomedical literature: a search-oriented approach. In *NAACL*.

Kevin Bretonnel Cohen, Hoa Trang Dang, Anita de Waard, Prabha Yadav, and Lucy Vanderwende. 2014. Tac 2014 biomedical summarization track. `https://tac.nist.gov/2014/BiomedSumm/`.

Gordon V. Cormack, Christopher R. Palmer, and Charles L. A. Clarke. 1998. Efficient construction of large test collections. In *SIGIR*.

Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest, and Clifford Stein. 2009. *Introduction to Algorithms, Third Edition*.

Andrew M Dai and Quoc V Le. 2015. Semi-supervised sequence learning. In *NeurIPs*.

Mostafa Dehghani, Sascha Rothe, Enrique Alfonseca, and Pascal Fleury. 2017. Learning to Attend, Copy, and Generate for Session-Based Query Suggestion. In *CIKM*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Jay DeYoung, Iz Beltagy, Madeleine van Zuylen, Bailey Kuehl, and Lucy Lu Wang. 2021. MS^2: Multi-Document Summarization of Medical Studies. In *EMNLP*.

Jay DeYoung, Sarthak Jain, Nazneen Fatema Rajani, Eric Lehman, Caiming Xiong, Richard Socher, and Byron C. Wallace. 2020a. Eraser: A benchmark to evaluate rationalized nlp models. In *ACL*.

Jay DeYoung, Eric Lehman, Ben Nye, Iain James Marshall, and Byron C. Wallace. 2020b. Evidence inference 2.0: More data, better models. In *BioNLP@ACL*.

T. Diggelmann, Jordan L. Boyd-Graber, Jannis Bulian, Massimiliano Ciaramita, and Markus Leippold. 2020. CLIMATE-FEVER: A Dataset for Verification of Real-World Climate Claims. In *Tackling Climate Change with ML workshop @ NeurIPS*.

George R. Doddington, Alexis Mitchell, Mark A. Przybocki, Lance A. Ramshaw, Stephanie Strassel, and Ralph M. Weischedel. 2004. The automatic content extraction (ace) program - tasks, data, and evaluation. In *International Conference on Language Resources and Evaluation*.

Rotem Dror, Gili Baumer, Segev Shlomov, and Roi Reichart. 2018. The hitchhiker's guide to testing statistical significance in natural language processing. In *ACL*.

Carl N. Edwards, T. Lai, Kevin Ros, Garrett Honke, and Heng Ji. 2022. Translation between molecules and natural language. In *EMNLP*.

Ahmed Elgohary, Denis Peskov, and Jordan L. Boyd-Graber. 2019. Can You Unpack That? Learning to Rewrite Questions-in-Context. In *EMNLP*.

William Ferreira and Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *NAACL*.

Thibault Févry, Livio Baldini Soares, Nicholas FitzGerald, Eunsol Choi, and Tom Kwiatkowski. 2020. Entities as Experts: Sparse Memory Access with Entity Supervision. In *EMNLP*.

Matt Gardner, Joel Grus, Mark Neumann, Oyvind Tafjord, Pradeep Dasigi, Nelson F. Liu, Matthew Peters, Michael Schmitz, and Luke S. Zettlemoyer. 2017. Allennlp: A deep semantic natural language processing platform.

Ralph Grishman. 2019. Twenty-five years of information extraction. *Natural Language Engineering*, 25:677 – 692.

Yu Gu, Robert Tinn, Hao Cheng, M. Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing. *ArXiv*, abs/2007.15779.

Yuxian Gu, Robert Tinn, Hao Cheng, Michael R. Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2021. Domain-Specific Language Model Pretraining for Biomedical Natural Language Processing. *ACM Transactions on Computing for Healthcare (HEALTH)*.

Zhijiang Guo, Michael Schlichtkrull, and Andreas Vlachos. 2022. A survey on automated fact-checking. *TACL*.

Suchin Gururangan, Ana Marasovic, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020a. Don't stop pretraining: Adapt language models to domains and tasks. In *ACL*.

Suchin Gururangan, Ana Marasović, Swabha Swayamdipta, Kyle Lo, Iz Beltagy, Doug Downey, and Noah A. Smith. 2020b. Don't Stop Pretraining: Adapt Language Models to Domains and Tasks. In *ACL*.

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. 2018. Annotation artifacts in natural language inference data. In *NAACL*.

Andreas Hanselowski, Christian Stab, Claudia Schulz, Zile Li, and Iryna Gurevych. 2019. A richly annotated corpus for different tasks in automated fact-checking. In *CoNLL*.

Marti A. Hearst. 2006. Clustering versus faceted categories for information exploration. *Communications of the ACM*.

Jerry R. Hobbs. 1993. The generic information extraction system. In *Message Understanding Conference*.

Kokil Jaidka, Muthu Kumar Chandrasekaran, Devanshu Jain, and Min-Yen Kan. 2017. The cl-scisumm shared task 2017: Results and key insights. In *BIRNDL@JCDL*.

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W. Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. In *EMNLP*.

V. Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Yu Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *EMNLP*.

Jin-Dong Kim, Tomoko Ohta, Yuka Tateisi, and Jun'ichi Tsujii. 2003. Genia corpus - a semantically annotated corpus for bio-textmining. *Bioinformatics*, 19 Suppl 1:i180–2.

Neema Kotonya and F. Toni. 2020. Explainable Automated Fact-Checking for Public Health Claims. In *EMNLP*.

Martin Krallinger, Obdulia Rabal, Saber Ahmad Akhondi, Martín Pérez Pérez, Jésús López Santamaría, Gael Pérez Rodríguez, Georgios Tsatsaronis, Ander Intxaurrondo, José Antonio Baso López, Umesh Nandal, Erin M. van Buel, A. Poorna Chandrasekhar, Marleen Rodenburg, Astrid Lægreid, Marius A. Doornenbal, Julen Oyarzábal, Anália Lourenço, and Alfonso Valencia. 2017. Overview of the biocreative vi chemical-protein interaction track.

Chaitanya Kulkarni, Wei Xu, Alan Ritter, and Raghu Machiraju. 2018. An annotated corpus for machine reading of instructions in wet lab protocols. In *NAACL-HLT*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural Questions: A Benchmark for Question Answering Research. *TACL*.

Jinhyuk Lee, Wonjin Yoon, Sungdong Kim, Donghyeon Kim, Sunkyu Kim, Chan Ho So, and Jaewoo Kang. 2020. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*.

Kenton Lee, Luheng He, and Luke S. Zettlemoyer. 2018. Higher-order coreference resolution with coarse-to-fine inference. In *NAACL-HLT*.

Nayeon Lee, Yejin Bang, Andrea Madotto, Madian Khabsa, and Pascale Fung. 2021. Towards Few-shot Fact-Checking via Perplexity. In *NAACL*.

Eric Lehman, Jay DeYoung, Regina Barzilay, and Byron C. Wallace. 2019. Inferring which medical treatments work from reports of clinical trials. In *NAACL*.

Tao Lei, Regina Barzilay, and Tommi S. Jaakkola. 2016. Rationalizing neural predictions. In *ACL*.

Chengkai Li, Ning Yan, Senjuti Basu Roy, Lekhendro Lisham, and Gautam Das. 2010. Facetedpedia: dynamic generation of query-dependent faceted interfaces for wikipedia. In *WWW*.

Qi Li and Heng Ji. 2014. Incremental joint extraction of entity mentions and relations. In *ACL*.

Qi Li, Heng Ji, and Liang Huang. 2013. Joint event extraction via structured prediction with global features. In *ACL*.

Xiangci Li, G. Burns, and Nanyun Peng. 2021. A Paragraph-level Multi-task Learning Model for Scientific Fact-Verification. In *Workshop on Scientific Document Understanding @ AAAI*.

Jeffrey Ling, Nicholas FitzGerald, Zifei Shan, Livio Baldini Soares, Thibault Févry, David Weiss, and T. Kwiatkowski. 2020. Learning Cross-Context Entity Representations from Text. *ArXiv*, abs/2001.03765.

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

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Kinney, and Daniel S. Weld. 2020a. S2ORC: The Semantic Scholar Open Research Corpus. In *ACL*.

Kyle Lo, Lucy Lu Wang, Mark Neumann, Rodney Michael Kinney, and Daniel S. Weld. 2020b. S2ORC: The Semantic Scholar Open Research Corpus. In *ACL*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018a. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *EMNLP*.

Yi Luan, Luheng He, Mari Ostendorf, and Hannaneh Hajishirzi. 2018b. Multi-task identification of entities, relations, and coreference for scientific knowledge graph construction. In *EMNLP*.

Yi Luan, Dave Wadden, Luheng He, Amy Shah, Mari Ostendorf, and Hannaneh Hajishirzi. 2019. A general framework for information extraction using dynamic span graphs. In *NAACL-HLT*.

Kelvin Luu, Xinyi Wu, Rik Koncel-Kedziorski, Kyle Lo, Isabel Cachola, and Noah A. Smith. 2021. Explaining relationships between scientific documents. In *ACL*.

Sean MacAvaney, Craig MacDonald, Roderick Murray-Smith, and Iadh Ounis. 2021. IntenT5: Search Result Diversification using Causal Language Models. *ArXiv*, abs/2108.04026.

Farzaneh Mahdisoltani, Joanna Asia Biega, and Fabian M. Suchanek. 2015. YAGO3: A Knowledge Base from Multilingual Wikipedias. In *CIDR*.

Gary Marchionini. 2006. Exploratory Search: From Finding to Understanding. *Communications of the ACM*.

Iain James Marshall, Joël Kuiper, Edward Banner, and Byron C. Wallace. 2017. Automating biomedical evidence synthesis: Robotreviewer. *ACL*.

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

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan T. McDonald. 2020. On Faithfulness and Factuality in Abstractive Summarization. In *ACL*.

Alan Medlar, Jing Li, and Dorota Glowacka. 2021. Query Suggestions as Summarization in Exploratory Search. In *CHIIR*.

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. Ambigqa: Answering ambiguous open-domain questions. *ArXiv*, abs/2004.10645.

Makoto Miwa and Mohit Bansal. 2016. End-to-end relation extraction using lstms on sequences and tree structures. In *ACL*.

R.K. Nadkarni, David Wadden, Iz Beltagy, Noah A. Smith, Hannaneh Hajishirzi, and Tom Hope. 2021. Scientific Language Models for Biomedical Knowledge Base Completion: An Empirical Study. *ArXiv*, abs/2106.09700.

Preslav I Nakov, Ariel S Schwartz, and Marti Hearst. 2004. Citances: Citation sentences for semantic analysis of bioscience text. In *SIGIR workshop on Search and Discovery in Bioinformatics*.

Trung Minh Nguyen and Thien Huu Nguyen. 2019. One for all: Neural joint modeling of entities and events. In *AAAI*.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. 2020. Document Ranking with a Pretrained Sequence-to-Sequence Model. In *EMNLP*.

Benjamin Nye, Junyi Jessy Li, Roma Patel, Yinfei Yang, Iain James Marshall, Ani Nenkova, and Byron C. Wallace. 2018. A corpus with multi-level annotations of patients, interventions and outcomes to support language processing for medical literature. In *ACL*.

Benjamin E. Nye, Jay DeYoung, E. Lehman, A. Nenkova, I. Marshall, and Byron C. Wallace. 2020. Understanding Clinical Trial Reports: Extracting Medical Entities and Their Relations. *ArXiv*, abs/2010.03550.

Stanislaw Osinski and Dawid Weiss. 2005. A concept-driven algorithm for clustering search results. *IEEE Intelligent Systems*.

Bhargavi Paranjape, Mandar Joshi, John Thickstun, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. An information bottleneck approach for controlling conciseness in rationale extraction. *ArXiv*, abs/2005.00652.

Nanyun Peng, Hoifung Poon, Chris Quirk, Kristina Toutanova, and Wen tau Yih. 2017. Cross-sentence n-ary relation extraction with graph lstms. *Transactions of the Association for Computational Linguistics*, 5:101–115.

Gordon Pennycook, Jonathon McPhetres, Yunhao Zhang, Jackson G. Lu, and David G. Rand. 2020. Fighting COVID-19 misinformation on social media: Experimental evidence for a scalable accuracy nudge intervention. *Psychological Science*.

Matthew E. Peters, Waleed Ammar, Chandra Bhagavatula, and Russell Power. 2017. Semi-supervised sequence tagging with bidirectional language models. In *ACL*.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *NAACL*.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vassilis Plachouras, Tim Rocktaschel, and Sebastian Riedel. 2021. KILT: a Benchmark for Knowledge Intensive Language Tasks. In *NAACL*.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel. 2019. Language Models as Knowledge Bases? In *EMNLP*.

Kashyap Popat, Subhabrata Mukherjee, Jannik Strötgen, and Gerhard Weikum. 2017. Where the truth lies: Explaining the credibility of emerging claims on the web and social media. In *WWW*.

Ronak Pradeep, Xueguang Ma, Rodrigo Nogueira, and Jimmy Lin. 2021. Scientific Claim Verification with VerT5erini. In *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis @EACL*.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. 2012. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, pages 1–40. Association for Computational Linguistics.

Danish Pruthi, Bhuwan Dhingra, Graham Neubig, and Zachary Chase Lipton. 2020. Weakly- and Semi-supervised Evidence Extraction. In *EMNLP Findings*.

Yujie Qian, Enrico Santus, Zhijing Jin, Jiang Guo, and Regina Barzilay. 2018. Graphie: A graph-based framework for information extraction. In *NAACL-HLT*.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, W. Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *JMLR*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *EMNLP*.

Adam Roberts, Colin Raffel, and Noam M. Shazeer. 2020a. How Much Knowledge Can You Pack into the Parameters of a Language Model? In *EMNLP*.

Kirk Roberts, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, Kyle Lo, Ian Soboroff, Ellen M. Voorhees, Lucy Lu Wang, and William R. Hersh. 2020b. TREC-COVID: rationale and structure of an information retrieval shared task for COVID-19. *Journal of the American Medical Informatics Association*.

Kirk Roberts, Dina Demner-Fushman, Ellen M. Voorhees, and William R. Hersh. 2016. Overview of the TREC 2016 clinical decision support track. *TREC*.

Kirk Roberts, Dina Demner-Fushman, Ellen M. Voorhees, William R. Hersh, Steven Bedrick, Alexander J. Lazar, and Shubham Pant. 2020c. Overview of the TREC 2020 Precision Medicine Track. *TREC*.

S. Robertson and H. Zaragoza. 2009. The Probabilistic Relevance Framework: BM25 and Beyond. *Foundations and Trends in Information Retrieval*.

Kevin Roose. Bing (yes, bing) just made search interesting again. *The New York Times*.

Arkadiy Saakyan, Tuhin Chakrabarty, and Smaranda Muresan. 2021. COVID-Fact: Fact Extraction and Verification of Real-World Claims on COVID-19 Pandemic. In *ACL*.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. In *NAACL*.

Rodrygo L. T. Santos, Craig MacDonald, and Iadh Ounis. 2010. Exploiting query reformulations for web search result diversification. In *WWW*.

Rodrygo L. T. Santos, Craig MacDonald, and Iadh Ounis. 2015. Search Result Diversification. *Foundations and Trends in Information Retrieval*.

Mourad Sarrouti, Asma Ben Abacha, Yassine Mrabet, and Dina Demner-Fushman. 2021. Evidence-based Fact-Checking of Health-related Claims. In *EMNLP*.

Teven Le Scao, Angela Fan, Christopher Akiki, Elizabeth-Jane Pavlick, Suzana Ili'c, Daniel Hesslow, Roman Castagn'e, Alexandra Sasha Luccioni, Franccois Yvon, Matthias Gallé, Jonathan Tow, Alexander M. Rush, Stella Rose Biderman, Albert Webson, Pawan Sasanka Ammanamanchi, Thomas Wang, Benoît Sagot, Niklas Muennighoff, Albert Villanova del Moral, Olatunji Ruwase, Rachel Bawden, Stas Bekman, Angelina McMillan-Major, Iz Beltagy, Huu Nguyen, Lucile Saulnier, Samson Tan, Pedro Ortiz Suarez, Victor Sanh, Hugo Laurenccon, Yacine Jernite, Julien Launay, Margaret Mitchell, Colin Raffel, Aaron Gokaslan, Adi Simhi, Aitor Soroa Etxabe, Alham Fikri Aji, Amit Alfassy, Anna Rogers, Ariel Kreisberg Nitzav, Canwen Xu, Chenghao Mou, Chris C. Emezue, Christopher Klamm, Colin Leong, Daniel Alexander van Strien, David Ifeoluwa Adelani, Dragomir R. Radev, Eduardo G. Ponferrada, Efrat Levkovizh, Ethan Kim, Eyal Bar Natan, Francesco De Toni, Gérard Dupont, Germán Kruszewski, Giada Pistilli, Hady ElSahar, Hamza Benyamina, Hieu Tran, Ian Yu, Idris Abdulmumin, Isaac Johnson, Itziar Gonzalez-Dios, Javier de la Rosa, Jenny Chim, Jesse Dodge, Jian Zhu, Jonathan Chang, Jorg Frohberg, Josephine L. Tobing, Joydeep Bhattacharjee, Khalid Almubarak, Kimbo Chen, Kyle Lo, Leandro von Werra, Leon Weber, Long Phan, Loubna Ben Allal, Ludovic Tanguy, Manan Dey, Manuel Romero Muñoz, Maraim Masoud, Mar'ia Grandury, Mario vSavsko, Max Huang, Maximin Coavoux, Mayank Singh, Mike Tian-Jian Jiang, Minh Chien Vu, Mohammad Ali Jauhar, Mustafa Ghaleb, Nishant Subramani, Nora Kassner, Nurulaqilla Khamis, Olivier Nguyen, Omar Espejel, Ona de Gibert, Paulo Villegas, Peter Henderson, Pierre Colombo, Priscilla Amuok, Quentin Lhoest, Rheza Harliman, Rishi Bommasani, Roberto L'opez, Rui Ribeiro, Salomey Osei, Sampo Pyysalo, Sebastian Nagel, Shamik Bose, Shamsuddeen Hassan Muhammad, Shanya Sharma, S. Longpre, Somaieh Nikpoor, Stanislav Silberberg, Suhas Pai, Sydney Zink, Tiago Timponi Torrent, Timo Schick, Tristan Thrush, Valentin Danchev, Vassilina Nikoulina, Veronika Laippala, Violette Lepercq, Vrinda Prabhu, Zaid Alyafeai, Zeerak Talat, Arun Raja, Benjamin Heinzerling, Chenglei Si, Elizabeth Salesky, Sabrina J. Mielke, Wilson Y. Lee, Abheesht Sharma, Andrea Santilli, Antoine Chaffin, Arnaud Stiegler, Debajyoti Datta, Eliza Szczechla, Gunjan Chhablani, Han Wang, Harshit

Pandey, Hendrik Strobelt, Jason Alan Fries, Jos Rozen, Leo Gao, Lintang Sutawika, M Saiful Bari, Maged S. Al-shaibani, Matteo Manica, Nihal V. Nayak, Ryan Teehan, Samuel Albanie, Sheng Shen, Srulik Ben-David, Stephen H. Bach, Taewoon Kim, Tali Bers, Thibault Févry, Trishala Neeraj, Urmish Thakker, Vikas Raunak, Xiang Tang, Zheng Xin Yong, Zhiqing Sun, Shaked Brody, Y Uri, Hadar Tojarieh, Adam Roberts, Hyung Won Chung, Jaesung Tae, Jason Phang, Ofir Press, Conglong Li, Deepak Narayanan, Hatim Bourfoune, Jared Casper, Jeff Rasley, Max Ryabinin, Mayank Mishra, Minjia Zhang, Mohammad Shoeybi, Myriam Peyrounette, Nicolas Patry, Nouamane Tazi, Omar Sanseviero, Patrick von Platen, Pierre Cornette, Pierre Franccois Lavall'ee, Rémi Lacroix, Samyam Rajbhandari, Sanchit Gandhi, Shaden Smith, Stéphane Requena, Suraj Patil, Tim Dettmers, Ahmed Baruwa, Amanpreet Singh, Anastasia Cheveleva, Anne-Laure Ligozat, Arjun Subramonian, Aur'elie N'ev'eol, Charles Lovering, Daniel H Garrette, Deepak R. Tunuguntla, Ehud Reiter, Ekaterina Taktasheva, Ekaterina Voloshina, Eli Bogdanov, Genta Indra Winata, Hailey Schoelkopf, Jan-Christoph Kalo, Jekaterina Novikova, Jessica Zosa Forde, Jordan Clive, Jungo Kasai, Ken Kawamura, Liam Hazan, Marine Carpuat, Miruna Clinciu, Najoung Kim, Newton Cheng, Oleg Serikov, Omer Antverg, Oskar van der Wal, Rui Zhang, Ruochen Zhang, Sebastian Gehrmann, S. Osher Pais, Tatiana Shavrina, Thomas Scialom, Tian Yun, Tomasz Limisiewicz, Verena Rieser, Vitaly Protasov, Vladislav Mikhailov, Yada Pruksachatkun, Yonatan Belinkov, Zachary Bamberger, Zdenvek Kasner, Alice Rueda, Amanda Pestana, Amir Feizpour, Ammar Khan, Amy Faranak, Ananda Santa Rosa Santos, Anthony Hevia, Antigona Unldreaj, Arash Aghagol, Arezoo Abdollahi, Aycha Tammour, Azadeh HajiHosseini, Bahareh Behroozi, Benjamin Olusola Ajibade, Bharat Kumar Saxena, Carlos Muñoz Ferrandis, Danish Contractor, David M. Lansky, Davis David, Douwe Kiela, Duong Anh Nguyen, Edward Tan, Emily Baylor, Ezinwanne Ozoani, Fatim T Mirza, Frankline Ononiwu, Habib Rezanejad, H.A. Jones, Indrani Bhattacharya, Irene Solaiman, Irina Sedenko, Isar Nejadgholi, Jan Passmore, Joshua Seltzer, Julio Bonis Sanz, Karen Fort, Lívia Macedo Dutra, Mairon Samagaio, Maraim Elbadri, Margot Mieskes, Marissa Gerchick, Martha Akinlolu, Michael McKenna, Mike Qiu, M. K. K. Ghauri, Mykola Burynok, Nafis Abrar, Nazneen Rajani, Nour Elkott, Nourhan Fahmy, Olanrewaju Modupe Samuel, Ran An, R. P. Kromann, Ryan Hao, Samira Alizadeh, Sarmad Shubber, Silas L. Wang, Sourav Roy, Sylvain Viguier, Thanh-Cong Le, Tobi Oyebade, Trieu Nguyen Hai Le, Yoyo Yang, Zachary Kyle Nguyen, Abhinav Ramesh Kashyap, Alfredo Palasciano, Alison Callahan, Anima Shukla, Antonio Miranda-Escalada, Ayush Kumar Singh, Benjamin

Beilharz, Bo Wang, Caio Matheus Fonseca de Brito, Chenxi Zhou, Chirag Jain, Chuxin Xu, Clémentine Fourrier, Daniel Le'on Perin'an, Daniel Molano, Dian Yu, Enrique Manjavacas, Fabio Barth, Florian Fuhrimann, Gabriel Altay, Giyaseddin Bayrak, Gully A. Burns, Helena U. Vrabec, Iman I.B. Bello, Isha Dash, Ji Soo Kang, John Giorgi, Jonas Golde, Jose David Posada, Karthi Sivaraman, Lokesh Bulchandani, Lu Liu, Luisa Shinzato, Madeleine Hahn de Bykhovetz, Maiko Takeuchi, Marc Pàmies, María Andrea Castillo, Marianna Nezhurina, Mario Sanger, Matthias Samwald, Michael Cullan, Michael Weinberg, M Wolf, Mina Mihaljcic, Minna Liu, Moritz Freidank, Myungsun Kang, Natasha Seelam, Nathan Dahlberg, Nicholas Michio Broad, Nikolaus Muellner, Pascale Fung, Patricia Haller, R. Chandrasekhar, R. Eisenberg, Robert Martin, Rodrigo L. Canalli, Rosaline Su, Ruisi Su, Samuel Cahyawijaya, Samuele Garda, Shlok S Deshmukh, Shubhanshu Mishra, Sid Kiblawi, Simon Ott, Sinee Sang-aroonsiri, Srishti Kumar, Stefan Schweter, Sushil Pratap Bharati, T. A. Laud, Th'eo Gigant, Tomoya Kainuma, Wojciech Kusa, Yanis Labrak, Yashasvi Bajaj, Y. Venkatraman, Yifan Xu, Ying Xu, Yun chao Xu, Zhee Xao Tan, Zhongli Xie, Zifan Ye, Mathilde Bras, Younes Belkada, and Thomas Wolf. 2022. Bloom: A 176b-parameter open-access multilingual language model. *ArXiv*, abs/2211.05100.

Tal Schuster, Adam Fisch, and R. Barzilay. 2021. Get Your Vitamin C! Robust Fact Verification with Contrastive Evidence. *ArXiv*, abs/2103.08541.

Tal Schuster, Darsh J. Shah, Yun Jie Serene Yeo, Daniel Filizzola, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *EMNLP*.

Ivan Sekulic, Mohammad Aliannejadi, and Fabio A. Crestani. 2021. Towards Facet-Driven Generation of Clarifying Questions for Conversational Search. In *SIGIR*.

Lei Sha, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. Jointly extracting event triggers and arguments by dependency-bridge rnn and tensor-based argument interaction. In *AAAI*.

Darsh J. Shah, Tal Schuster, and R. Barzilay. 2020. Automatic Fact-guided Sentence Modification. In *AAAI*.

Amir Soleimani, Christof Monz, and Marcel Worring. 2019. Bert for evidence retrieval and claim verification. In *European Conference on Information Retrieval*.

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jianyun Nie. 2015. A Hierarchical Recurrent Encoder-Decoder for Generative Context-Aware Query Suggestion. In *CIKM*.

Karen Sparck Jones and C. J. van Rijsbergen. 1975. Report on the need for and provision of an "ideal" information retrieval test collection. In *British Library Research and Development Report 5266, Computer Laboratory, University of Cambridge.*

Dominik Stammbach. 2021. Evidence Selection as a Token-Level Prediction Task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER) @ EMNLP*.

Dominik Stammbach, Boyang Zhang, and Elliott Ash. 2021. The Choice of Knowledge Base in Automated Claim Checking. *ArXiv*, abs/2111.07795.

Emilia Stoica, Marti A. Hearst, and Megan Richardson. 2007. Automating Creation of Hierarchical Faceted Metadata Structures. In *NAACL*.

Fabian M. Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *WWW*.

Beth M. Sundheim. 1993. The message understanding conferences. In *NIST's TIPSTER Text Program*.

Ross Taylor, Marcin Kardas, Guillem Cucurull, Thomas Scialom, Anthony S. Hartshorn, Elvis Saravia, Andrew Poulton, Viktor Kerkez, and Robert Stojnic. 2022. Galactica: A large language model for science. *ArXiv*, abs/2211.09085.

James Thorne, Max Glockner, Gisela Vallejo, Andreas Vlachos, and Iryna Gurevych. 2021. Evidence-based Verification for Real World Information Needs. *ArXiv*, abs/2104.00640.

James Thorne and Andreas Vlachos. 2021. Evidence-based factual error correction. In *ACL*.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. Fever: a large-scale dataset for fact extraction and verification. In *NAACL*.

Guy Tsafnat, Paul P Glasziou, Miew Keen Choong, Adam G. Dunn, Filippo Galgani, and Enrico W. Coiera. 2014. Systematic review automation technologies. *Systematic Reviews*, 3:74–74.

George Tsatsaronis, Georgios Balikas, Prodromos Malakasiotis, Ioannis Partalas, Matthias Zschunke, Michael R. Alvers, Dirk Weissenborn, Anastasia Krithara, Sergios Petridis, Dimitris Polychronopoulos, Yannis Almirantis, John Pavlopoulos, Nicolas Baskiotis, Patrick Gallinari, Thierry Artières, Axel-Cyrille Ngonga Ngomo, Norman Heino, Éric Gaussier, Liliana Barrio-Alvers, Michael Schroeder, Ion Androutsopoulos, and Georgios Paliouras. 2015. An overview of the bioasq large-scale biomedical semantic indexing and question answering competition. In *BMC Bioinformatics*.

Daniel Tunkelang. 2009. *Faceted Search*.

Andreas Vlachos and Sebastian Riedel. 2014. Fact checking: Task definition and dataset construction. In *ACL Workshop on Language Technologies and Computational Social Science*.

Ellen M. Voorhees, Tasmeer Alam, Steven Bedrick, Dina Demner-Fushman, William R. Hersh, Kyle Lo, Kirk Roberts, Ian Soboroff, and Lucy Lu Wang. 2020. TREC-COVID: Constructing a Pandemic Information Retrieval Test Collection. In *SIGIR*.

Ellen M. Voorhees and Donna Harman. 2005. *TREC: Experiment and Evaluation in Information Retrieval*. MIT press.

Denny Vrandecic and Markus Krotzsch. 2014. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57:78–85.

David Wadden, Nikita Gupta, Kenton Lee, and Kristina Toutanova. 2022a. Entity-centric query refinement.

David Wadden and Kyle Lo. 2021. Overview and Insights from the SciVer Shared Task on Scientific Claim Verification. In *Proceedings of the Second Workshop on Scholarly Document Processing @ NAACL*.

David Wadden, Kyle Lo, Bailey Kuehl, Arman Cohan, Iz Beltagy, Lucy Lu Wang, and Hannaneh Hajishirzi. 2022b. Scifact-open: Towards open-domain scientific claim verification. In *EMNLP*.

David Wadden, Kyle Lo, Lucy Lu Wang, Arman Cohan, Iz Beltagy, and Hannaneh Hajishirzi. 2022c. MultiVerS: Improving scientific claim verification with weak supervision and full-document context. In *NAACL Findings*.

David Wadden, Kyle Lo, Lucy Lu Wang, Shanchuan Lin, Madeleine van Zuylen, Arman Cohan, and Hannaneh Hajishirzi. 2020. Fact or Fiction: Verifying Scientific Claims. In *EMNLP*.

David Wadden, Ulme Wennberg, Yi Luan, and Hannaneh Hajishirzi. 2019. Entity, relation, and event extraction with contextualized span representations. In *EMNLP*.

Byron C. Wallace, Sayantani Saha, Frank Soboczenski, and Iain James Marshall. 2021. Generating (Factual?) Narrative Summaries of RCTs: Experiments with Neural Multi-Document Summarization. In *AMIA Joint Summits on Translational Science proceedings*.

Lucy Lu Wang, Kyle Lo, Yoganand Chandrasekhar, Russell Reas, Jiangjiang Yang, Doug Burdick, Darrin Eide, Kathryn Funk, Yannis Katsis, Rodney Michael Kinney, Yunyao Li, Ziyang Liu, William Merrill, Paul Mooney, Dewey A. Murdick, Devvret Rishi, Jerry Sheehan, Zhihong Shen, Brandon Stilson, Alex D. Wade, Kuansan Wang, Nancy Xin Ru Wang, Christopher Wilhelm, Boya Xie, Douglas M. Raymond, Daniel S. Weld, Oren Etzioni, and Sebastian Kohlmeier. 2020. CORD-19: The COVID-19 open research dataset. In *Proceedings of the 1st Workshop on NLP for COVID-19 @ ACL*.

William Yang Wang. 2017. "Liar, Liar Pants on Fire": A New Benchmark Dataset for Fake News Detection. In *ACL*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Ed Huai hsin Chi, Quoc Le, and Denny Zhou. 2022. Chain of thought prompting elicits reasoning in large language models.

Ryen W. White and Resa A. Roth. 2009. Exploratory Search: Beyond the Query-Response Paradigm. *Synthesis Lectures on Information Concepts, Retrieval, and Services*.

Sarah Wiegreffe and Ana Marasović. 2021. Teach me to explain: A review of datasets for explainable natural language processing. In *NeurIPS Datasets and Benchmarks*.

Dustin Wright, David Wadden, Kyle Lo, Bailey Kuehl, Arman Cohan, Isabelle Augenstein, and Lucy Lu Wang. 2022. Generating Scientific Claims for Zero-Shot Scientific Fact Checking. In *ACL*.

Bishan Yang and Tom M. Mitchell. 2016. Joint extraction of events and entities within a document context. In *HLT-NAACL*.

Ka-Ping Yee, Kirsten Swearingen, Kevin Li, and Marti A. Hearst. 2003. Faceted metadata for image search and browsing. In *CHI*.

Wenpeng Yin and Dan Roth. 2018. Twowingos: A two-wing optimization strategy for evidential claim verification. In *EMNLP*.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontañón, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, and Amr Ahmed. 2020. Big Bird: Transformers for Longer Sequences. In *NeurIPS*.

Hamed Zamani, Susan T. Dumais, Nick Craswell, Paul N. Bennett, and Gord Lueck. 2020. Generating Clarifying Questions for Information Retrieval. In *WWW*.

Oren Zamir and Oren Etzioni. 1999. Grouper: A Dynamic Clustering Interface to Web Search Results. *Computer Networks*.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

Tongtao Zhang, Heng Ji, and Avirup Sil. 2019. Joint entity and event extraction with generative adversarial imitation learning. *Data Intelligence*, 1:99–120.

Yuhao Zhang, Peng Qi, and Christopher D. Manning. 2018. Graph convolution over pruned dependency trees improves relation extraction. In *EMNLP*.

Zhiwei Zhang, Jiyi Li, Fumiyo Fukumoto, and Yanming Ye. 2021. Abstract, Rationale, Stance: A Joint Model for Scientific Claim Verification. In *EMNLP*.

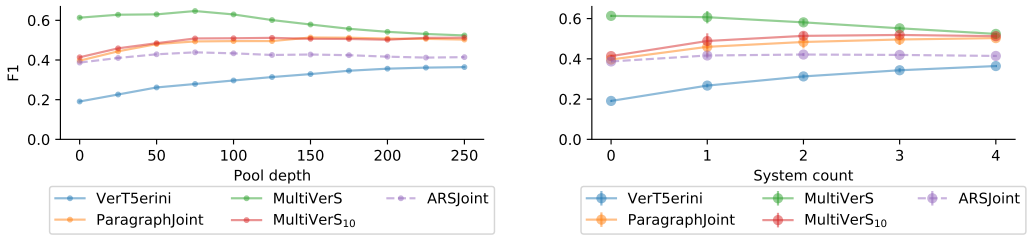
Marinka Zitnik, Monica Agrawal, and Jure Leskovec. 2018. Modeling polypharmacy side effects with graph convolutional networks. *Bioinformatics*.

Justin Zobel. 1998. How reliable are the results of large-scale information retrieval experiments? In *SIGIR*.

Appendix A

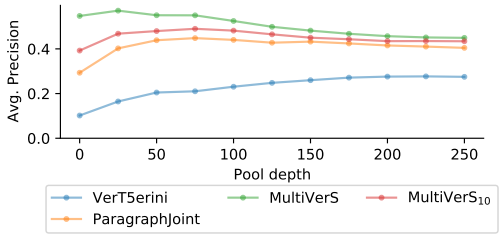
Additional reliability checks for SciFact-Open

In Chapter 5.5, we examined the effect of pool depth and system count on model performance. We mentioned that our findings were robust to our choice of evaluation metric (F1 score vs. average precision). Figure A.1 provides plots confirming this. The fact that F1 and average precision exhibit the same trends indicates that simply re-calibrating models' classification thresholds would not change the qualitative conclusions drawn in Chapter 5.5.

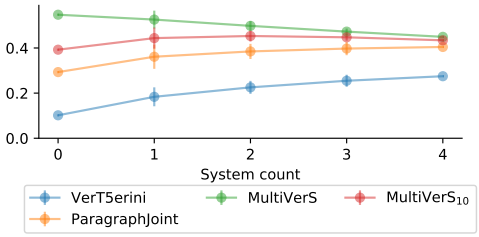


(a) F1 score as a function of pool depth.

(b) F1 score as a function of system count.



(c) Average precision as a function of pool depth.



(d) Average precision as a function of system count.

Figure A.1: Effect of pool depth (left column) and system count (right column) on model performance, as measured by F1 score (top row) and average precision (bottom row).

Appendix B

Proofs and algorithms for QRESP

B.1 QRESP and entity discovery

In §6.2.2 we mentioned that, when interacting with a (hypothetical) system which outputs k refinements whose answers evenly partition the answers to the input query q, any entity answering q is discoverable after at most $\log_k(|\mathcal{A}(q)|)$ rounds of system interaction. To see why, suppose we aim to discover a target entity $e^* \in \mathcal{A}(q)$. When given q as input, this ideal system would output k refinements, each with $|\mathcal{A}(q)|/k$ answers, and exactly one of which has e^* as an answer. We would then input this refinement q' as our new query, and the system would output k new refinements, each with $|\mathcal{A}(q)|/k^2$ answers, exactly one of which has e^* as an answer. We would use this refinement q'' as our new input query, and repeat the process until we arrive at a refinement that includes only e^*. This process induces a $k-ary$ tree over the set of all entities answering the original query q, with depth $\log_k(|\mathcal{A}(q)|)$. In other words, any entity answering q can be discovered after $\log_k(|\mathcal{A}(q)|)$ system interactions.

This type of system interaction can be viewed as a form of Huffman coding, where each entity $e_j \in \mathcal{A}(q)$ is represented by a k-ary prefix code indicating the sequence of refinements used to discover the entity. Assuming that all entities in $\mathcal{A}(q)$ are equally likely to be the target entity e^*, choosing refinements which partition the entity space into equal-sized subsets induces an optimal prefix code [Cormen et al., 2009, Chapter 16.3]. Future work could extend this framework to model entity popularity, assigning shorter codes (i.e. shorter refinement sequences) to more frequently-searched entities.

B.2 Best achievable refinements

We provide a proof that the scoring function $\mathcal{S}(\mathcal{R}(q))$ defined in Eq. 6.1 achieves its global minimum if and only if $\mathcal{R}(q)$ is ideal – i.e. all refinements in $\mathcal{R}(q)$ have the same number of answers, and every entity answering q answers exactly one $q'_i \in \mathcal{R}(q)$.

For brevity, we denote an RS $\mathcal{R}(q)$ simply as $\mathcal{R}$. Assume $n \triangleq |\mathcal{A}(q)|$ is divisible by k; this eliminates some edge cases but does not change the main idea. Re-write Eq. 6.1 by defining $t_1 \triangleq \sum_j |c_j - 1|$ and $t_2 \triangleq \min_i n_i$, so $\mathcal{S}(\mathcal{R}) = t1 - t2$. We claim that $\mathcal{R}$ is ideal as defined in §6.2.2 if and only if $t_1 = 0$ *and* $t_2 = n/k$. As proof, the forward direction follows from the definition. For the reverse direction, $t_1 = 0$ means that each e_j answers exactly one q'_i. Combined with the fact that the smallest $\mathcal{A}(q'_i)$ has n/k answers, this implies that *all* q'_i have n/k answers, thus $\mathcal{R}$ is ideal. It follows that $\mathcal{S}(\mathcal{R}) = -n/k$ if $\mathcal{R}$ is ideal.

Now assume that $\mathcal{R}$ is some RS for which $\mathcal{S}(\mathcal{R}) \leq -n/k$. Since t_1 is lower-bounded by 0, it must be that $t_2 \geq n/k$; equivalently, $t_2 = n/k + \delta$ for some integer $\delta \geq 0$. This necessitates that $|\mathcal{A}(q'_i)| \geq n/k + \delta$ for all i, which implies $t_1 \geq \delta k$. Then $\mathcal{S}(\mathcal{R}) = t_1 + t_2 \geq \delta k - (n/k + \delta) = (k-1)\delta - n/k$. If $\delta > 0$, then $\mathcal{S}(\mathcal{R}) > -n/k$, contradicting our assumption. On the other hand, if $\delta = 0$, then we have $t_1 = 0$ and $t_2 = n/k$, which (from our earlier claim) is only possible if $\mathcal{R}$ is ideal. Thus, $\mathcal{S}(\mathcal{R})$ achieves its minimum if and only if $\mathcal{R}$ is ideal.

B.3 Optimization

We describe how to convert the optimization problem Eq. 6.2 into an integer linear program. We use the same notation as in §6.3, with one change: instead of using i to index refinements in $\mathcal{R}(q)$, we use it to index refinement *candidates* $q'_i \in \mathcal{C}(q)$. Let $x_i = \mathbb{1}\left[q'_i \in \mathcal{R}(q)\right]$, $a_{ij} = \mathbb{1}\left[e_j \in \mathcal{A}(q'_i)\right]$, and $c_j = \sum_i x_i a_{ij}$. Let $n_i = |A(q'_i)|$ and $n_{max} = \max_i |A(q'_i)|$. Then Eq. 6.2 can be re-written as:

$$\min_{x_i, c_j, y_j, \xi} \quad \left[\sum_{j=1}^{n} y_j \right] - \xi$$

$$\begin{aligned}
\text{s.t.} \qquad & c_j - 1 \leq y_j \text{ and } 1 - c_j \leq y_j && \forall j \\
& c_j = \sum_i x_i a_{ij} && \forall j \\
& k = \sum_i x_i \\
& \xi \leq (1 - x_i) n_{max} + x_i n_i && \forall i
\end{aligned}$$

We use SCIP to perform the optimization [Bestuzheva et al., 2021].