BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

AUTOMATED FACTOID QUESTION ANSWERING AND FACT-CHECKING IN NATURAL LANGUAGE

AUTOMATICKÉ ODPOVÍDÁNÍ NA FAKTICKÉ OTÁZKY A OVĚŘOVÁNÍ FAKTŮ

V PŘIROZENÉM JAZYCE

PHD THESIS
DISERTAČNÍ PRÁCE

AUTHOR **Ing. MARTIN FAJČÍK**
AUTOR PRÁCE

SUPERVISOR **doc. RNDr. PAVEL SMRŽ, Ph.D.**
ŠKOLITEL

BRNO 2023

Abstract

This thesis examines two problems, that rely on a precise understanding of factual information. *In factoid question answering (QA)*, it addresses three topics, Firstly, it shows a novel probability formulation and training objective for systems that extract answer as a span of text. The experiments show that the proposed compound objective with joint probability space is Pareto optimal to other used objectives. Secondly, the thesis studies the problem of open-domain QA. It shows that extractive approaches and abstractive approaches have complementary strengths and proposes a pipelined state-of-the-art system R2-D2 that serves as a strong baseline for the community. Thirdly, it studies the effect of pruning down the retrieval corpus under R2-D2. The experiments demonstrate that for two popular datasets, NaturalQuestions and TriviaQA, two-thirds of the retrieval corpus can be removed without the loss of performance, and 92 % can be removed with a loss of performance up to -3 exact match score. Findings also indicate that the same pruning mechanism is implicitly present in modern supervised retrieval mechanisms, such as DPR. *In fact-checking*, the thesis studies two topics. Firstly, it shows that pretrained model approaches can reach competitive performance in rumor stance detection without using of any handcrafted features or metadata. Specifically, our system targets rumor stance detection in social media threads and selects whether each post supports, denies, queries, or comments on the rumor present in the discussion thread. Experiments demonstrate that using just the first thread post and the previous thread post is sufficient in obtaining strong performance of determining the current post stance. Secondly, the thesis studies evidence-grounded fact-checking. Claim-Dissector—a system that jointly identifies the relevant evidence and produces a veracity verdict—is proposed. The proposed system can find supporting and refuting evidence for a claim at any language granularity, including tokens, sentences, or paragraphs, and link them in an interpretable way with the verdict. It is demonstrated that the model allows successful transfer learning from the coarse granularity of supervision to the fine granularity of predictions. In particular, it is shown that training on sentence level of relevance is sufficient to obtain relevant token-level rationales, and training on block level indeed provides competitive sentence-level cues. The strong performance of Claim-Dissector is demonstrated across 5 datasets and 2 underlying pretrained models, including a newly collected dataset TLR-FEVER. The code for all experiments is available online.

Abstrakt

Tato práce se zabývá dvěma problémy, které spoléhají na přesné pochopení faktických informací. *Ve faktoidním zodpovídání otázek (QA)* se práce zabýva třemi tématy. Nejprve je představena nová objektivní funkce a formulace složené pravděpodobnosti pro systémy, které extrahují odpověď jako textový úsek. Experimenty ukazují, že navrhovaná objektivní funkce se složeným pravděpodobnostním prostorem je Pareto optimální vůči jiným, běžně používaným objektivním funkcím. V druhé části se práce zabývá problematikou QA nad otevřenou doménou. Ukazuje vzájemně doplňující se vlastnosti extraktivních a abstraktivních přístupů a navrhuje nový modulární systém R2-D2, který slouží jako silný systém pro srovnání (baseline) v komunitě. V třetí části práce studuje vliv zmenšovaní korpusu pro vyhledávání pomocí mechanismu prořezávání při použití R2-D2. Experimenty ukazují, že u dvou populárních datových sad — NaturalQuestions a TriviaQA — lze odstranit dvě třetiny korpusu pro vyhledávání, aniž by došlo ke zhoršení výsledných odpovědí systému a 92 % lze odstranit se zhoršením pouze do -3 skóre přesné shody (exact match). Zjištěné poznatky naznačují, že stejný mechanismus prořezávání je implicitně přítomen v moderních metodách učeného vyhledávání, jako je DPR. Dále *v oblasti ověřování faktů* se práce dotýká dvou témat. Jednak ukazuje, že předtrénované modely, které nepoužívají žádné ručně vytvořené příznaky nebo metadata, mohou dosáhnout konkurenceschopných výsledků v detekci postoje lidí k fámám. Vytvořený systém se konkrétně zaměřuje na zjišťování postojů k fámám ve vláknech sociálních sítí a určuje, jestli daný příspěvek ve vlákně podporuje, odmítá, zpochybňuje nebo komentuje fámu přítomnou v diskusním vláknu. Provedené experimenty ukazují, že použití pouze prvního příspevku vlákna a předchozího příspevku vlákna stačí k tomu, aby model určil aktuální postoj příspevku. Posledním tématem, kterým se práce zabývá, je ověřování faktů založené na vyhledávání podporující evidence. Je navržen systém Claim-Dissector, který společně identifikuje relevantní evidenci a určuje věrohodnost diskutabilního tvrzení. Navržený systém dokáže najít podpůrnou a vyvracející evidenci pro tvrzení v jakékoli jazykové granularitě, na úrovni tokenů, vět nebo odstavců, a propojit je interpretovatelným způsobem s verdiktem. Dále je ukázáno, že model umožňuje úspěšný přenos učení z hrubé granularity poskytnuté během učení na jemnou granularitu predikcí. Zejména je ukázano, že učení identifikace relevance na úrovni vět je dostatečné k získání relevantních zdůvodnění na úrovni tokenu a učení na úrovni bloku je dostatečné k získání relevantních zdůvodnění na úrovni vět. Silné výsledky systému Claim-Dissector jsou demonstrovány na 5 datových sadách, včetně nově shromážděné sady TLR-FEVER, a dvou různých předtrénovaných modelech. Kód pro všechny experimenty je k dispozici online.

Keywords

question answering, fact checking, fact-checking, QA, FC, extractive question answering, R2-D2, Claim-Dissector, RumourEval, QA corpus pruning, compound objective, TriviaQA, EfficientQA, NaturalQuestions

Klíčová slova

odpovídaní na otázky, faktoidné zodpovídaní otázek, oveřovaní faktů, fact-checking, QA, FC, extraktivní zodpovídaní otázek, prořezávaní korpusu u QA, compound objective,TriviaQA, EfficientQA, NaturalQuestions, R2-D2, Claim-Dissector

Reference

FAJČÍK, Martin. *Automated Factoid Question Answering and Fact-Checking in Natural Language*. Brno, 2023. PhD thesis. Brno University of Technology, Faculty of Information Technology. Supervisor doc. RNDr. Pavel Smrž, Ph.D.

Automated Factoid Question Answering and Fact-Checking in Natural Language

Declaration

I hereby declare that this PhD thesis was prepared as an original work by the author under the supervision of doc. RNDr. Pavel Smrž, Ph.D. I have listed all the literary sources, publications and other sources, which were used during the preparation of this thesis.

. . . .
Martin Fajčík
April 9, 2024

Acknowledgements

I would like to thank to my nearest supporter Lenka, for always understanding my neccessity to work, and dig into everything we do with an extensive pool of questions. Next, I would like to thank to my supervisor Pavel, for accepting me to my PhD studies and always staying calm when I panicked due to the approaching deadline and to Marcela, who talked me into going for PhD and helped me to get in touch with the world of science for the first time. The collosal proportion of my thanks budget is allocated to my colleagues and friends: Karel, who I often speak to loud when debugging my thoughts, Martin, who always pinpoints some formal points I am missing, Bolaji who is an excellent PhD student trying to pretend he is not, Lukas, the only person who tiredlessly provided my with guidance during the early days of my PhD, or whenever I felt lost, KarelB, Katka and MartinŠ, who listened to my "NLP readings", and proof-checked my ideas, when I had an urge to visit a nearby office due to barren and desolate nature of my place in its early days, Pablo and Neha, who made me feel like home during my stays in Switzerland, Apoorv, for provding feedback when I needed it most, prof. Henderson for allowing me to visit their language meetings, and showing me his world-class leadership of his NLP group. The last sentence belongs to thanks to my mum, who somehow raised me curious as I am, and my long-time companion Benny, who didn't cared about a single word in this thesis.

"Iba hlupák nemení názory, pokiaľ sa menia okolnosti." (Andrej Kiska, 2018).

Contents

List of Figures

Acronyms

AoI Assumption of Independence. 13

BCE Binary Cross-Entropy. 18, 20

CD Claim-Dissector. 16, 76, 85, 90, 139

CDQA Closed-Domain Question Answering. 4, 9–11, 24

CE Cross-Entropy. 18, 20, 21

EQA Extractive Question Answering. 4, 10–13, 16, 26, 28, 36, 37, 94, 95

FC Fact-Checking. 11, 12, 22, 74, 96

FiD Fusion-in-Decoder. 22, 39, 42, 45, 46, 50, 52, 53, 57, 64, 91

FiE Fusion-in-Encoder. 52

FS FEVER-Score. 5, 85, 88

IDK I-Don't-Know. 4, 11

IR Information Retrieval. 19, 52

LRM Language Representation Model. 4, 19–21, 24, 25, 27–31, 36, 41, 42, 68, 94, 95

MLM Masked Language Modelling. 20

MLP Multi-layer Perceptron. 5, 27, 29, 31, 77

MML Maximum Marginal Likelihood. 52

NEI Not-Enough-Information. 5, 77, 78, 80, 81, 84, 91

NLP Natural Language Processing. 73

ODQA Open-Domain Question Answering. 8–14, 16, 37, 38, 44, 52–55, 61, 63, 64, 95, 130

PMF Probability Mass Function. 4, 17, 18, 26, 31, 41

QA Question Answering. 4, 7, 9–14, 16, 20–22, 62, 84, 85, 94, 96

SSE Single Sentence Estimate. 80, 90, 119

VAT Virtual Adversarial Training. 21

Glossary

Accuracy@K Proportion of cases, where correct item is predicted in-between top-K predictions. 4, 5, 43, 47, 55, 59, 123, 126

bipolar evidence An evidence set that contains both supporting and refuting evidence. 84, 85, 90, 92

CondAcc Verification accuracy augmented evidence macro F1 documented in Appendix E.4. 92, 136

CondF1 Verification accuracy augmented evidence F1 documented in Appendix E.4. 92, 136

EfficientQA Test set for QA released in the EfficientQA competition (Min et al., 2021). 14, 47, 50, 53–57, 61, 124

EM Exact Match metric, used for relevant evidence assessment. 85

EM Exact Match metric, used for QA. 14, 19, 30, 31, 34, 35, 37, 43, 46, 50–53, 55–57, 61, 64, 95

evidence group A set of annotated evidences, e.g., a set of sentences a particular annotator annotated. 83, 85

EviF1 Binary F1 score for evidence classification. 92

F1 F$_1$ score for classification, as defined in Section 2.2. 15, 18, 66, 69–73, 85, 92, 95

F1 F$_1$ for QA metric, as defined in Section 2.2. 5, 19, 30, 31, 34–37, 84, 85, 88, 119, 139

FAVIQ-A FaVIQ: Dataset for FAct Verification from Information-seeking Questions(Park et al., 2022). 16

FEVER FEVER: Fact Extraction and VERification Dataset (Thorne et al., 2018). 15, 16

multi-hop reasoning A type of reasoning which requires fusing the information from multiple sources (e.g., multiple documents) in order to produce the output. In fact-checking, for instance, multi-hop reasoning is required to refute the claim ”*Brno is the largest city of Czechia.*” backed up with documents ”*The capital of Czech Republic is its largest city.*“, and ”*Prague is the capital of Bohemia.*“. 83, 84

NQ-Golden A dataset composed from ground-truth passages of NQ-Open and randomly sampled Wikipedia passages. The formers are to be classified from the latter.. 4, 56, 58–60

NQ-Open Filtered version of Natural Questions (Kwiatkowski et al., 2019) dataset adopted for ODQA. 4, 8, 14, 43–45, 48–51, 53–57, 63, 64, 95, 120, 121, 123–126, 130

RaI Recall at input of the model. Defined in Section 7.3.2.. 85, 86, 91

REALFC REALFC: Dataset obtained from the manual conversion of yes/no boolean questions into facts (Thorne et al., 2018). 16

TQ-Golden A dataset composed from ground-truth passages of TQ-Open and randomly sampled Wikipedia passages. The formers are to be classified from the latter.. 56, 58

TQ-Open . 8, 14, 44, 45, 50, 51, 53–57, 63, 64, 94, 95, 123–126, 131

VA Accuracy for veracity prediction. 92

VF1 Binary F1 score for veracity prediction. 92

Chapter 1

Introduction

With the development of new deep differentiable architectures (mostly the Transformers architecture (Vaswani et al., 2017) and its derivatives), natural language processing has undergone a series of developments, largely advancing tasks deemed as “too complicated to deal with” into “useful in practice” stage.

An example of such is *question answering* (QA)—the first topic of this thesis. One important goal of question answering is to complement traditional document retrieval systems when necessary. In document retrieval, the user enters a query $\mathcal{Q}$, the search engine performs retrieval and the user is shown a preview of the most relevant documents $\mathcal{D}$. In some cases, this is exactly what is needed, e.g., when the user is searching for a particular website. However, in some cases, the query is a question that can be answered directly, without forcing the user to read top returned articles manually (Etzioni, 2011). The system would do a better job if it would extract the information of interest (i.e., an answer) and provide it directly and faithfully—with a set of arguments that validate its reasoning. Despite its long history in the text retrieval community, state-of-the-art question answering only recently became competitive with humans and is a practical tool to be used in everyday life. Apparent day-to-day use cases are chatbots and voicebots, present in applications such as customer support or home assistants. According to a survey from Kinsella (2020), asking a question is the second most frequent use case of modern voice assistants among U.S. adults.

Formally, QA can be defined as follows: given a question $\mathcal{Q}$, QA focuses on (1) finding a (possibly minimal) set of documents $\mathcal{D}$ which are helpful for inferring the answer $\mathcal{A}$ and (2) inferring the answer $\mathcal{A}$ from these documents. In recent literature, this problem setting is often referred to as *open-domain question answering* (ODQA). Clearly, the task is less complicated if we skip step (1) and already provide a system with a minimal set of documents $\mathcal{D}$ necessary for answering the question—a task often referred to as *closed-domain question answering* (CDQA). Both tasks are illustrated in Figure 1.1.

In this work, we will be always referring to the particular instance of question answering named *factoid question answering*; that is a question answering where the answer is a fact. There is no broadly accepted formal definition of what a fact is (as common when formalizing something as ambiguous as natural language). In this work we will adopt the following definition: the factoid question-answer pair is such that the length of the answer in words is bounded with k, typically $k < 10$. An opposite of such is long-form question answering (Krishna et al., 2021), which focuses on long explanations that rather explain extensive processes, not merely state plain facts (Lee et al., 2019). Next, the scope of the

QA problem studied within this work covers only extractive QA (EQA); that is a problem where the answer is a text span that can be selected from (given or retrieved) text snippet.

Competitions in ODQA were organized for a long time, going back to TREC 1999 question answering (QA) track[1]. In TREC 2001, Voorhees et al. (2001) mentions that the systems submitted in the competition are composite systems. The majority of these were made of question-type classifiers, which usually differentiated between the processing of “who”, “where”, “when”, and “which” questions, followed by standard text-overlap driven document retrieval, document parsing, subsequent entity parsing, and then finally answer extraction. The extraction was done by comparing each entity’s context with the question words, picking up the best entity from the text. To tackle the ODQA, these competitions were making assumptions such as (i) the answer is always an entity, (ii) the answer string had a fixed amount of bytes, and if the answer’s entity was contained somewhere within this chunk of text, the prediction was deemed as correct, or (iii) the topics were restricted to the news domain from particular news services.

In contrast, only in 2016, (Wang and Jiang, 2017) it was first demonstrated that the freshly adopted neural systems can extract precise textual spans containing an answer in CDQA dataset SQuAD (Rajpurkar et al., 2016). In 2017, Chen et al. (2017a) showed that SQuAD can be also approached in the open domain, considering the whole of Wikipedia as an unstructured textual corpus used for document retrieval. In 2018, the human performance on popular SQuAD was exceeded in CDQA. In 2019, it was shown (Lee et al., 2019) that neural document retrieval can outperform classical text-overlap-driven approaches for retrieval in ODQA. Finally, in 2021, ODQA systems outmatched human teams made up of trivia experts in trivia quiz EfficientQA[2](Min et al., 2021) on questions where the answers were not guessed by humans instantly. Answer to these questions could then be discussed within the teams of up to 8 people (more details in Subsection 5.4), but nevertheless, systems correctly answered these questions significantly more frequently than human teams. Apart from research work, another proof of advancements is that since 2020, the popular search engine Google provides a question answering service[3], highlighting sentences which

[1] https://trec.nist.gov/data/qa.html. Accessed 15.3.2023.

[2] One of these systems was our R2-D2 system, further described in the thesis.

[3] https://www.theverge.com/2020/6/4/21280115/google-search-engine-yellow-highlight-featured-snippet-anchor-text. Accessed 15.3.2023.

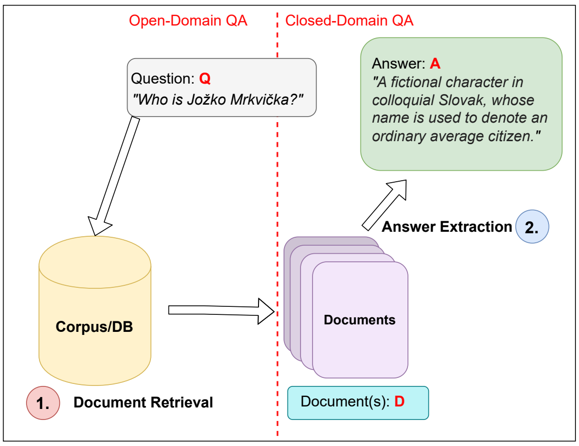


Figure 1.1: Open-Domain vs Closed-Domain Question Answering.

possibly contain an answer. This evidence demonstrates the advances in QA. However, there is a great deal of not well-understood processes underlying these rapid advancements, such as the complementarity of different answer inference approaches (Fajcik et al., 2021b) or impact of different optimization objectives on answer inference (Fajcik et al., 2021c). These problems are studied within the scope of this thesis.

A different example of a task which undergone a shift from research-only interest to being practically useful, and the second topic of this thesis, is *fact-checking* (FC). Given a factual claim $\mathcal{C}$, decide upon veracity $\mathcal{V}$ of this claim (e.g., true, false, unverifiable, . . .).

The advantages of free speech and direct access to large audiences through social media have led to the appearance of a colossal amount of unverified information. This is uncontrolled by official cables or traditional media. This has given rise to many fact-checking organizations including *FactCheck.org*[4], *Snopes*[5], *Politifact*[6], and *EUfactcheck*[7], and tools such as *Google Factcheck*[8]. Furthermore, the spread of unverified information has led towards the modern incarnation of the *fact-checker*—a professional who assesses the veracity of popular claims based on trustworthy evidence. Development of automated fact-checkers useful in assisting these professionals is a challenging task with significant real-word impact (Nakov et al., 2021).

From the technical perspective, fact-checking draws a lot of parallels with question answering. One such parallel is that every question, which can be answered with *”yes“*, *”no“*, or *“i-don’t-know”*(IDK) answer, can be converted to *“true”*, *false”* or *“unverifiable”* fact via 1-to-1 mapping (Sulem et al., 2022). Another parallel is that an instance of CDQA can be also converted to a fact. However, here the label conversion can be non-trivial, especially if the instance is unanswerable. An example of these parallels is illustrated with Figure 1.2. Analogically to the ODQA, fact-checking can be approached with an open-domain retrieval of documents.

Until recently, the majority of FC approaches relied on superficial cues of credibility, such as the way the claim is written, the statistics captured in the claim author’s profile, or the stances of its respondents on social networks (Zubiaga et al., 2016; Derczynski et al.,

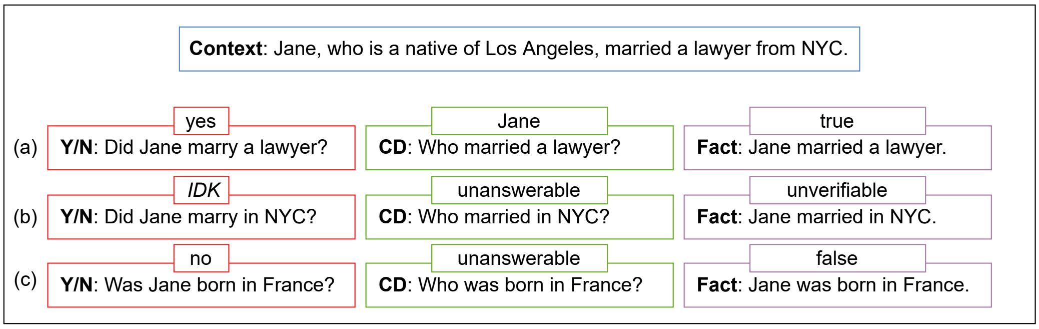
[4] https://www.factcheck.org/. Accessed 15.3.2023.

[5] https://www.snopes.com/. Accessed 15.3.2023.

[6] https://www.politifact.com/. Accessed 15.3.2023.

[7] https://eufactcheck.eu/. Accessed 15.3.2023.

[8] https://toolbox.google.com/factcheck/explorer. Accessed 15.3.2023.



Context: Jane, who is a native of Los Angeles, married a lawyer from NYC.

(a) yes — **Y/N**: Did Jane marry a lawyer? | Jane — **CD**: Who married a lawyer? | true — **Fact**: Jane married a lawyer.

(b) *IDK* — **Y/N**: Did Jane marry in NYC? | unanswerable — **CD**: Who married in NYC? | unverifiable — **Fact**: Jane married in NYC.

(c) no — **Y/N**: Was Jane born in France? | unanswerable — **CD**: Who was born in France? | false — **Fact**: Jane was born in France.

Figure 1.2: Examples of Yes/No/IDK QA(left), CDQA (center) and Facts (right). The non-trivial conversion of CDQA instances into Yes/No and Facts is shown in (b). While EQA (b) is *unanswerable*, both conversions correspond to *no/false* label. Inspired by Sulem et al. (2022).

2017; Gorrell et al., 2019; Fajcik et al., 2019; Li et al., 2019). From 2018, a significant effort towards evidence-grounded systems has been made with the introduction of FEVER dataset (Thorne et al., 2018) and subsequent FEVER workshops[9]. In contrast to previous superficial methods, evidence-grounded systems rely on a set of trustworthy documents (e.g., Wikipedia, newspaper from trusted sources, governmental reports & laws, company guidelines, verified responses on public forums, etc.), and verify a claim if and only if it is supported by facts found in this trustworthy set. The retrieval of these documents grounding the verdict demonstrates another parallel with ODQA.

However, the difference to question answering appears when considering a practical perspective. When considering journalistic fact-checking, the value of the final verdict is just as important as the production of justification for the verdict. Disproving a claim without linking it to factual evidence often fails to be persuasive and can even cause a "backfire" effect—refreshing and strengthening the belief into an erroneous claim (Lewandowsky et al., 2012). Therefore the interpretability requirement is a must for FC systems. In contrast, for QA, it is often sufficient to show the answer within the found context. For FC, it is necessary to have a set of non-trivial arguments that justify the verdict, such as the set of supporting/refuting documents. In this thesis, we focus on both, superficial and evidence-grounded approaches. Firstly, determining rumor stance from tweets and Reddit posts is examined (Fajcik et al., 2019). Finally, a novel framework for interpretable evidence-grounded fact-checking is presented (Fajcik et al., 2023).

1.1 Goals

The goals of this thesis are two-fold. The first goal is to extend the understanding of question answering models; in particular their objectives, and the complementarity of popular architectures, and leverage newfound properties to improve the state-of-the-art. The second goal studies different approaches to stance detection in fact-checking, and the link between stances of individual evidences[10] and their aggregation into final veracity verdict.

1.2 Contributions

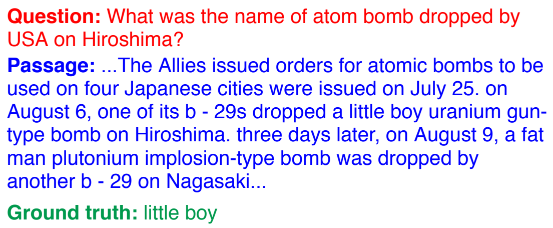
In this section, the summary of the thesis contributions is presented. Following the stated goals, the thesis targets these with 5 research projects, each having its own contributions. These can be divided into factoid question answering, and fact-checking parts (more in Section 1.3).

Rethinking the Objectives of Extractive QA. (Fajcik et al., 2021c)
Motivation: First, the commonly taken assumption of independence in extractive QA is challenged. Considering a question $\mathcal{Q}$ and a document or a set of documents $\mathcal{D}$, the EQA commonly estimates the categorical probability mass function $\mathbf{P}(\mathcal{A}|\mathcal{D},\mathcal{Q})$. In practice, the domain of $\mathbf{P}(\mathcal{A}|\cdot)$ lies in all possible span start and span end positions $(a_s, a_e) \sim \mathcal{A}$ in each document. For instance, if only possible answer in single document lies at the position $(2,4)$, the ideal estimated distribution should assign all probability mass to this particular position, formally $\mathrm{P}(2,4|\cdot) = 1$. Following the earliest work on span extraction (Wang and

[9] https://fever.ai/. Accessed 15.3.2023.

[10] Following practices from archaeology, the thesis uses the plural term "evidences" instead of "pieces of evidence" for brevity.

Jiang, 2017), majority of the influential work in the field (Xiong et al., 2017; Seo et al., 2017; Chen et al., 2017a; Yu et al., 2018; Devlin et al., 2019; Cheng et al., 2020) computed the joint probability through assumption of independence (AoI) $\mathrm{P}(a_s, a_e|\cdot) = \mathrm{P}(a_s|\cdot)\,\mathrm{P}(a_e|\cdot)$. However, the AoI may lead to obviously wrong predictions, especially in high-entropy scenarios where model considers several answer candidates, as shown in Figure 1.3.



Question: What was the name of atom bomb dropped by USA on Hiroshima?

Passage: ...The Allies issued orders for atomic bombs to be used on four Japanese cities were issued on July 25. on August 6, one of its b - 29s dropped a little boy uranium gun-type bomb on Hiroshima. three days later, on August 9, a fat man plutonium implosion-type bomb was dropped by another b - 29 on Nagasaki...

Ground truth: little boy

$\mathrm{P}(a_s, a_e\|\cdot)$	**Top Predictions from QA model**
33.3	little boy uranium gun-type bomb on Hiroshima. three days later, on August 9, a fat man
32.15	little boy
23.51	fat man
3.60	a fat man
2.08	a little boy uranium gun - type bomb on hiroshima. three days later, on august 9, a fat man
1.03	a little boy

Figure 1.3: An example of an error which comes with an independence assumption. The model assigns high probability mass to boundaries around "little boy", and "fat man" answers. However, the maximum probability is assigned to start of one and the end of another answer. Hence the model produces obviously wrong answer.

Contributions: Motivated by this observation the thesis contributions are: (i) introduction of new ways of efficient joint probability computation, adding negligible theoretical computation/memory complexity overhead and zero practical overhead over the assumption of independence (Appendix F.1); (ii) introduction of compound objective (composed of both, joint probability term and AoI term) and its comparison with objectives using only joint probability formulation $\mathrm{P}(a_s, a_e|\cdot)$, AoI term $\mathrm{P}(a_s|\cdot)\,\mathrm{P}(a_e|\cdot)$ or conditional probability factorization $\mathrm{P}(a_s|\cdot)\,\mathrm{P}(a_e|a_s, \cdot)$; (iii) a large scale evaluation on 3 QA model architectures, trained with 10 random seeds, across 6 different EQA datasets supported with statistical sets and (iv) a manual analysis which provides a closer look on the different impacts of independent and compound objectives. Conclusively, we show that optimizing the compound objective and decoding the direct joint probability distribution $\mathbf{P}(a_s, a_e|\cdot)$ is consistently superior or equal to different objectives in exact match metric across all considered datasets.

Fusing the Retrieval, Abstractive QA and Extractive QA (Fajcik et al., 2021b)
Motivation: ODQA is commonly approached with the traditional *retriever-reader* architecture. Such ODQA systems (Chen et al., 2017a) seek evidence for answering the questions inside the knowledge source using the *retriever* and then extract the answer from the retrieved knowledge using the *reader*. The knowledge source is often a large corpus of short snippets of natural language (e.g., taken from an encyclopedia). Some work (Nogueira and Cho, 2019; Luan et al., 2021) further adopts computationally expensive *reranking* step on top-K_1 retrieved documents to further increase the accuracy of relevant documents selected at top-K_2 places ($K_2 < K_1$), which are then fed to even more computationally expensive

models. Moreover, other work uses *abstractive reader* (sometimes referred to as generative reader), fusing the information from documents and generating the answer in an autoregressive fashion, word-by-word[11](Lewis et al., 2020; Min et al., 2020; Izacard and Grave, 2021b). A different line of work uses *extractive reader*, extracting the string answer from the grounding documents (Chen et al., 2017a; Lee et al., 2022; Cheng et al., 2021b).

Contributions: Here, the thesis contributes in: (i) the proposal of a simple novel approach to aggregate scores from all system components that demonstrates that combining extractive and generative approaches is superior to a posterior averaging ensemble of homogeneous models; (ii) the state-of-the-art performance for two large and popular datasets NQ-Open (Kwiatkowski et al., 2019; Lee et al., 2022) and TQ-Open (Joshi et al., 2017), while having the same knowledge source and the retriever as in the previous works[12] (Karpukhin et al., 2020; Izacard and Grave, 2021b); (iii) it is shown that the extractive reader can sometimes match the performance of the generative approaches without taking the advantage of the fusion between retrieved passages. This indicates that the evidence aggregation from multiple passages in the generative approaches is either not learned or not necessary to perform well on these datasets; (iv) while using rerankers is a common practice in information retrieval (Nogueira et al., 2019, 2020; Luan et al., 2021), our work (Fajcik et al., 2021b) was the first to study the interaction of novel dense retrieval methods and reranking in the context of ODQA; (v) we verified that the previously proposed compound objective improves performance also in the ODQA setting.

Published in 2021, our system *R2-D2* became a standard baseline to compare to in ODQA (Lee et al., 2022; Asai et al., 2022; Kedia et al., 2022; Jiang et al., 2022; Izacard et al., 2022, *inter alia*).

Pruning the Index Contents for Memory Efficient Open-Domain QA (Fajcik et al., 2021a; Min et al., 2021)
Motivation: The unstructured knowledge source of ODQA is a large corpus of short snippets of natural language. For instance, Karpukhin et al. (2020) uses a corpus of 21M passages created from articles on Wikipedia split into 100-word segments. Such a massive index of external documents—only from Wikipedia—scales in the order of tens of GiB. Motivated by this observation, we pose a research question: *How much of this set can we prune out, without damaging the system's performance?*.

Contributions: As our contribution, we gain evidence towards answering this question with our content-based pruning approach—a strong binary classifier that selects whether the passage is irrelevant or not apriori—without seeing any question—on popular ODQA datasets NQ-Open (Lee et al., 2022; Kwiatkowski et al., 2019), TQ-Open (Joshi et al., 2017) and EfficientQA (Min et al., 2021). Surprisingly, we find that most (about 92 %) of the information content can be pruned away with only minor (-3 EM) performance degradation to be seen in the current open-domain pipelined QA systems across datasets.

Furthermore, the proposed approach was tested within the NeurIPS competition EfficientQA[13] (Min et al., 2021), placing our system at 3rd place in systems having less than 6GiB in total. We were only attendants in the published leaderboard with university-only affiliations.

Baseline for Rumour Stance Detection (Fajcik et al., 2019)
Motivation: Fighting false rumours at the internet is a tedious task. Sometimes, even

[11] Such models can generate string which never occurred in any grounding document.
[12] Works with better retrieval were matched and sometimes outperformed too. More in Chapter 4.
[13] https://efficientqa.github.io/. Accessed 15.3.2023.

understanding what an actual rumor is about may prove challenging. And only then one can actually judge its veracity with an appropriate evidence. The works of Ferreira and Vlachos (2016) and Enayet and El-Beltagy (2017) focused on predictions of rumor veracity in thread discussions. These works indicated that veracity is correlated with discussion participants' stances towards the rumor.

Contributions: Taking part in the RumourEval2019 (Gorrell et al., 2019)—a challenge focused on classifying whether posts from Twitter and Reddit *supports*, *denies*, *queries*, or *comments* a hidden rumor, truthfulness of which is the topic of an underlying discussion thread—the thesis contributes in designing a simple model that automatically determines the stance of each participant in the discussion thread on Twitter and Reddit. It employs an end-to-end system that has reached the F1 score of 61.67 % on the provided test dataset. Without any hand-crafted feature, the system finished at the 2nd place in the competition, only 0.2 % F1 behind the winner. In particular, our system follows the assumption that the stance of the discussion's post depends only on itself, on the source thread post (the one discussion originates from), and on the previous thread post.

Considering its simplicity, and an advantage of then novel pretrained BERT model (Devlin et al., 2019), our stance classifier became a standard baseline in the rumor detection community (Scarton et al., 2020; Khandelwal, 2021; Wang et al., 2021; Rao et al., 2021, *inter alia*).

Interpretable Relevance-Grounded Veracity Prediction (Fajcik et al., 2023)
Motivation: Today's automated fact-checking systems are moving from predicting the claim's veracity by capturing the superficial cues of credibility, such as the way the claim is written, the statistics captured in the claim author's profile, or the stances of its respondents on social networks (Zubiaga et al., 2016; Derczynski et al., 2017; Gorrell et al., 2019; Fajcik et al., 2019; Li et al., 2019) towards evidence-grounded systems which, given a claim, identify relevant sources and then use these to predict the claim's veracity (Thorne et al., 2018; Jiang et al., 2020; Park et al., 2022). In practice, providing precise evidence turns out to be at least as important as predicting the veracity itself. Disproving a claim without linking it to factual evidence often fails to be persuasive and can even cause a "backfire" effect—refreshing and strengthening the belief into an erroneous claim (Lewandowsky et al., 2012) (more in Appendix F.2).

For evidence-grounded fact-checking, most of the existing state-of-the-art systems (Jiang et al., 2021; Stammbach, 2021; Khattab et al., 2021a) employ a 3-stage cascade approach; given a claim, (1) they retrieve relevant documents, (2) rerank relevant evidences (sentences, paragraphs or larger text blocks) within these documents, and (3) predict the claim's veracity from the top-k (usually k=5) relevant evidences.

This comes with several drawbacks; firstly, *the multiple steps of the system lead to error propagation*, i.e. the input to the last system might often be too noisy to contain any information. Some previous work focused on merging evidence reranking and veracity prediction into a single step (Ma et al., 2019; Schlichtkrull et al., 2021). Secondly, in open-domain setting, *number of relevant evidences can be significantly larger than k*[14], especially when there is a lot of repeated evidence. Thirdly, in open-domain setting, *sometimes there is both supporting and refuting evidence.* The re-ranking systems often do not distinguish whether evidence is relevant because it supports or refutes the claim, and thus may select the evidence from one group based on the in-built biases.

[14] e.g.,~3.7 % of FEVER dataset's (Thorne et al., 2018) non-exhaustive annotations.

To further strengthen the persuasive effect of the evidences and understand the model's reasoning process, some of these systems provide cues of interpretability (Popat et al., 2018; Liu et al., 2020). However, the interpretability in the mentioned work was often considered a useful trait, which was evaluated only qualitatively, as the labor-intensive human evaluation was out of the scope of their focus.

Contributions: To this extent, we contribute with *Claim-Dissector* (CD), a latent variable model which: (i) jointly ranks top-relevant, top-supporting, and top-refuting evidences, and predicts veracity of the claim in an interpretable way, where the probability of the claim's veracity is estimated using the linear combination of per-evidence probabilities, (ii) can provide fine-grained (sentence-level or token-level) evidence, while using only coarse-grained supervision (on block-level or sentence-level respectively), (iii) can be parametrized from a spectrum of language representation models (such as RoBERTa or DeBERTaV3 (Liu et al., 2019; He et al., 2021), (iv) achieves competitive performance on FEVER dataset (Thorne et al., 2018) and state-of-the-art on FAVIQ-A and REALFC datasets (Park et al., 2022; Thorne et al., 2021).

1.3 Thesis Plan

In Chapter 2, the thesis introduces basic notation, clarifies terminology, refreshes the inner workings of pretrained models and training techniques, and defines common metrics used across the work. Next, in the first part of the thesis focusing on QA, Chapter 3 introduces work on compound objective and its comparison with other objective types common in EQA. Then, Chapter 4 builds upon the common techniques used in ODQA and introduces R2-D2, a pipelined system that leverages the strengths of extractive and generative QA models, fuses the outputs with retrieval and reranking scores, achieving strong results across ODQA datasets. Furthermore, Chapter 5 introduces the index pruning approach and its application in the R2-D2 system. The second part of the thesis, oriented around fact-checking, follows. Chapter 6 introduces our system, which determines the stances of respondents in social media discussions towards the hidden rumor without any handcrafted features and achieved strong performance in the RumourEval2019 competition. In contrast, the subsequent Chapter 7 introduces a novel framework, that establishes an interpretable link between the computation of the grounding evidence relevance on different levels of language granularity, and final veracity assessment. The final takeaways, including limitations of systems introduced within the thesis, and explicit dicussion of author's publications related to this thesis are summarized in Chapter 8. Code for all experiments is released online (see Appendix B).

Chapter 2

Preliminaries

This section presents unified terminology and introduces basic concepts of language processing, used within this thesis. The list of concepts described here is non-exhaustive and is written only for the purpose of notational clarification and refreshing the knowledge of the reader.

2.1 Basic Functions & Notation

Random Event is an event, the outcome (e.g., a measurement) of which is subject to uncertainty. It is assumed that all possible outcomes $O = \{o_1, o_2, ...\}$ of such an event are known. Obtaining an outcome from the random event is also referred to as *sampling* (denoted with binary operator $\sim$).

Probability Mass Function / Probability Distribution Probability Mass Function (PMF) is a function $\mathbf{P}(X) : O \to P$, that assigns a probability $\mathrm{P}(X = o) = p \in P$ (defined below) to each individual outcome $o \sim X$ of the random event X. The thesis uses terms PMF and Probability Distribution (sometimes abbreviated to just "distribution") interchangeably.

Probability Probability of a random event X, written as $\mathrm{P}(X = a_1)$, quantifies the likelihood that random event X will yield value $a_1 \sim X$ upon sampling. In this work, we only assume probabilities for discrete events X (i.e., they have a finite set of outcomes). Thus each probability has the property of (i) being non-negative and (ii) being lesser or equal to 1.

Notational Shortcuts In some cases, probability assignment $\mathrm{P}(X = a)$ is not written explicitly to avoid long formulas. Instead only $\mathrm{P}(a)$ notation is used. It can always be told, whether a term is a distribution function or a single value—a probability of an outcome—according to notation $\mathbf{P}$ for a distribution function vs. P for a probability.

Dense Dense function, or dense layer function refers to a affine projection, with an optional additive shift. Formally, given a d-dimensional vector $\boldsymbol{x} \in \mathbb{R}^d$, a dense function dense : $\mathbb{R}^d \to \mathbb{R}^{d_2}$ is defined as $\mathrm{dense}(\boldsymbol{x}) = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$, where $\boldsymbol{W} \in \mathbb{R}^{d_2 \times d}$ and $\boldsymbol{b} \in \mathbb{R}^{d_2}$.

Softmax Softmax function, applied to a vector or a set, exponentiates and normalizes the items within. Formally, for $\boldsymbol{x} \in \mathbb{R}^d$ softmax : $\mathbb{R}^d \to \mathbb{R}^d$, i-th softmax output is defined as $\mathrm{softmax}(\boldsymbol{x})_i = \frac{\exp(\boldsymbol{x}_i)}{\sum_{j=1}^{d} \exp(\boldsymbol{x}_j)}$; and similarly for a set $A \subset \mathbb{R}$ and its i-th element $a_i \in A$, according to index set $i \in I_A$, it is defined as $\mathrm{softmax}(A)_i = \frac{\exp(a_i)}{\sum_{x \in A} \exp(x)}$. The coefficients obtained with softmax normalization can be directly used to parametrize Categorical PMF. Hence, some formulas directly assign its outputs as distribution.

Tanh $\tanh : \mathbb{R} \rightarrow [-1, 1]$ function refers to function $\tanh(x) = \frac{sinh(x)}{cosh(x)} = \frac{e^x - e^{-x}}{e^x + e^{-x}}$.

ReLU $\mathrm{ReLU} : \mathbb{R} \rightarrow [0, \infty]$ function refers to function $\mathrm{ReLU}(x) = \max(x, 0)$.

Sigmoid $\mathrm{sigmoid} : \mathbb{R} \rightarrow [0, 1]$ function is a non-linear function $(1 + \exp(-x))^{-1}$. When *Tanh*, *ReLU* or *Sigmoid* are applied on a vector $\boldsymbol{x}$, they are meant to apply as elementwise operator $\mathrm{f}(\boldsymbol{x})_\mathrm{i} = \mathrm{f}(\boldsymbol{x}_\mathrm{i})$.

Cross-Entropy Cross-Entropy (CE) measures the (non-negative) error attained from the usage of the (model) function $\mathbf{P}(Y)$ over data generated from random event X, that follows the empirical distribution $\mathbf{P}(X)$. Formally, empirical cross-entropy can be estimated as $\mathrm{H}(X, Y) = -\sum_{x \sim X} \log \mathrm{P}(Y = x)$.

Binary Cross-Entropy Binary Cross-Entropy (BCE) is a special case of Cross-Entropy, computed for a random event with just two outcomes. It is often implemented with Bernoulli distribution, which requires just 1 (usually sigmoid-normalized) parameter.

The reader is referred to Bishop (2006) for a comprehensive explanation of the basics defined in this section.

2.2 Metrics

This section describes statistics commonly used to measure performance across chapters of this thesis. The list is non-exhaustive; metrics used just for a specific dataset or a specific method are described in the respective chapter. As usual within the machine learning literature, for binary classification, we define True Positives (TPs), False Positives (FPs), True Negatives (TNs), and False Negatives (FNs). TPs denote the number of times, the *true class* was *predicted* on the dataset. FPs denote the number of times, the *false class* was *predicted* on the dataset. FNs denote the number of times, the *true class* was *not predicted* on the dataset. Lastly, TNs denote the number of times, the *false class* was *not predicted* on the dataset. Next, we define *precision* as $\frac{TP}{TP+FP}$ and *recall* as $\frac{TP}{TP+FN}$. We refer the reader to Wikipedia[1] for exact details for computation of accuracy/binary F1. Unless said otherwise, the less frequent class is considered a true class, and the more frequent class is a false class (Chicco and Jurman, 2020).

Accuracy describes the relative proportion of correctly classified examples. It is always reported as percentage.

F1 score for Classification is a harmonic mean of *precision* and *recall*—can be binary, or multi-class, in case of which it can be micro-averaged or macro-averaged. For the binary case, we use the standard definition. For **macro**-averaged version, F1 is computed for every class independently, using the 1–vs–all strategy (treating the selected class as true, and all other classes as false). The F1s computed for each class are then averaged. For **micro**-averaged version, the TPs, FPs, TNs, and FN counts are also computed using the 1–vs–all strategy. Next, all TPs are summed into microTPs, FPs are summed into microFPs, etc. Finally, F1 is computed from microTPs, microFPs, microTNs, and microFNs. In all cases, it is reported as percentage.

[1] https://en.wikipedia.org/wiki/F-score. Accessed 6.3.2023.

F1 for Span Classification is a special case of F1. Consider a predicted sequence $P_s = \{p_1, p_2, p_3, ..., p_n\}$ and a ground-truth answer sequence $G = \{g_1, g_2, g_3, ..., g_m\}$, tokenized with the same tokenization method. TPs are equal to the number of overlapping tokens $|P \cap G|$. The total number of predicted positives TPs+FPs is given by P_s's cardinality $|P_s|$. The total number of true items TPs+FNs is given by the cardinality of $|G|$. Therefore precision is defined as $\frac{|P_s \cap G|}{|P_s|}$ and recall as $\frac{|P_s \cap G|}{|G|}$. Per-sample F1 for each example is computed as a harmonic mean of such precision and recall. Most of the datasets contain more than one ground truth G. In such case, the maximum F1 across all ground truths is considered per sample. Total F1 then makes an average over each sample's F1. In practice, both predicted, and ground-truth sequences are often unicode normalized, and their stopwords are dropped[2]. It is always reported as percentage.

Exact Match (EM) is a considering whether predicted sequence P exactly matches at least one of the sample-annotated ground-truth sequences $G_1, G_2, ...$ (similar to F1 for span classification). In practice, the same preprocessing as for F1 usually applies.[3]

Accuracy@K (also referred to as Top-K Accuracy, HIT@K, Recall@K[4] in literature) computes the proportion of examples where the correct item (class, document, answer,...) is in the top-K most probable elements according to model under evaluation. In this thesis, it is used to evaluate the document-with-answer retrieval. For instance, Accuracy@10=25 % means that in 25 % of cases, at least one document with an answer was located in top-10 retrieved documents. In practice, both document $d = d_1, d_2, d_3, ...$ and answer $a = a_1, a_2, a_3, ...$ are unicode normalized and tokenized. Then it is checked whether an answer sequence is present within the document sequence[5].

2.3 Language Representation Models

This section reviews the details for pretraining, and architectural properties of language representation models (LRMs) applied in the thesis. We define LRM as any parametric model that is first pretrained with a different objective, then one being used in a setup described within the scope of this thesis. All presented parametric models are based on *transformer* model architecture (Vaswani et al., 2017), with a few modifications as discussed further.

BERT (Devlin et al., 2019) follows architecture of the transformer encoder. The architectural differences are: (i) the GeLU (Hendrycks and Gimpel, 2016) activation function is used instead of ReLU and (ii) the inputs embeddings constructed from the summation of learned token embeddings, segment embeddings, and positional embeddings. The 30,000 token vocabulary of lexemes is found by the WordPiece method (Wu et al., 2016).

During pretraining, the corpus is split into sentences. BERT's input is composed of 2 tokenized sentences, each tokenized as $s_{11}, s_{12}, s_{13}, s_{14}, ..., s_{1n}$ and $s_{21}, s_{22}, s_{23}, ..., s_{2m}$.

$$\texttt{[CLS]}\, s_{11}, s_{12}, s_{13}, s_{14}, ..., s_{1n}\, \texttt{[SEP]}\, s_{21}, s_{22}, s_{23}, ..., s_{2m}\, \texttt{[SEP]}$$

[2] F1 score as computed in Rajpurkar et al. (2016) available at https://cutt.ly/R8AiyeR (Accessed 6.3.2023) is often used as a point of reference.

[3] EM score as computed in Rajpurkar et al. (2016) available at https://cutt.ly/e8SZA8U. (Accessed 7.3.2023) is often used as a point of reference.

[4] In IR literature (differently from this work), recall@K also often measures how many relevant documents from the total number of relevant documents were in top-K retrieved documents.

[5] Answer matching as computed in Karpukhin et al. (2020) available at https://cutt.ly/n8SC0wG (Accessed 7.3.2023) is often used as a point of reference.

The 2 sentences are either consecutive or randomly picked (uniformly). [CLS], and [SEP] are so-called special tokens. Next, 15 % of tokens at the input (except the special tokens) are *selected* and of those (i) 80 % are replaced with special [MASK] token, 10 % are replaced with a random token from the vocabulary, and last 10 % are left unchanged. The input representations are summed from a learned token, segment, and positional representations for each token, including special tokens. There are just two segment representations, first for tokens of the first sentence, and second for tokens of the second sentence. Next, the summed representations are encoded by N layers of the transformer encoder.

The BERT's output contains exactly the same amount of output representations as its input, therefore it is possible to align every input token with its position-aligned output representation. The masked language modeling MLM loss is computed only from representations aligned with *selected* tokens. Such representations are projected to vocabulary size and normalized via softmax to obtain per-token probabilities. The CE loss is computed for each of these, maximizing the probability of the selected-and-possibly-masked token. The training with a minibatch size of 256 took approximately 1M steps to converge on a corpus with 3.3B words. Two sentence format was used for Next Sentence Prediction objective, which was found not-useful by later work (Liu et al., 2019), so its description is omitted.

During fine-tuning for QA, the model is presented with a question in the first segment and a context in the second segment. Therefore the representations of context tokens $\boldsymbol{H}$ can be isolated from the output matrix by aligning these representations with context-token representations (more in Section 3.2.2).

RoBERTa (Liu et al., 2019) is a robustly optimized BERT, trained with more data, and minor pretraining differences. These include (i) training with 8x larger minibatch size, about 10x more training data, with half as many steps as BERT, (ii) dynamic masking—potentially masked tokens were selected during training, achieving more masking patterns for the same samples—as opposed to static masking where these tokens were selected in the preprocessing for BERT, and (iii) tokenization with byte-pair-encoding (Sennrich et al., 2016).

ALBERT (Lan et al., 2020) introduced a BERT-like LRM with (i) parameter sharing across all transformer layers, (ii) low-dimensional token embeddings with upscaling, (iii) additional sentence-order prediction loss and (iv) n-gram masking. Thanks to parameter sharing, the authors could increase the model's dimensionality, without a dramatic increase in parameter size. When comparing with a 110M BERT, input embeddings were 6x smaller, and upscaled with $W \in \mathbb{R}^{d \times d_m}$ projection to a model dimensionality d_m. Sentence order prediction loss, computed from [CLS]-token representation through projection and BCE, computes whether consecutive sentences at the model's input switched their original order or not. Lastly, following SpanBERT's n-gram masking approach (Joshi et al., 2020), authors do not mask 15 % of the input tokens at random but mask sub-sequences of the random length instead. RoBERTa's training corpus is used for pretraining.

ELECTRA (Clark et al., 2020) used BERT as a source of corrupted data (a generator), and trained a discriminator model with BERT-like architecture, that predicts for every token whether it was corrupted or not (so-called *replaced token detection*). The generator and the discriminator are trained jointly. The smaller generator model uses BERT's MLM objective. Then, based on the token probability computed for MLM loss, the replacement token is sampled for all selected tokens. Lastly, the discriminator encodes inputs with replacements and computes BCE for every token position, computing the probability whether the token was “real” or “fake”. In fine-tuning for downstream applications, only the discriminator is used. RoBERTa’s training corpus is used for pretraining.

DeBERTaV3 (He et al., 2021) uses an ELECTRA-based training setup, virtual adversarial training (VAT) (Miyato et al., 2018), introduces several architectural changes, and non-shared token representations. Unlike ELECTRA (i), the token embeddings are not shared, but disentangled; the generator uses embeddings $\boldsymbol{E}$, whereas the discriminator uses token embeddings $\boldsymbol{E} + \boldsymbol{E}_{\boldsymbol{\Delta}}$. Furthermore, discriminator’s loss gradients are never propagated towards $\boldsymbol{E}$. Next (ii), DeBERTa uses disentangled attention, averaging 3 types of attention scores on each layer—attention of positional embeddings to token-level embeddings, token-to-positional, and token-to-token embeddings. The positional embeddings are selected in such a way, that they effectively fulfill the role of relative positional encoding[6]. And finally (iii), it adds the 1D convolutional layer after 1st layer of the transformer. RoBERTa’s training corpus is used.

T5 (Raffel et al., 2020) introduced a pretrained encoder-decoder LRM model (unlike encoder-only models above) based on a modified transformer architecture, that was pretrained on a mixture of 22 supervised and unsupervised tasks. Authors have used textual cues at the encoder’s input to encode each supervised task e.g., encoder input ”*translate English to German: That is good.*“ and decoder input ”*Das ist gut.*“. The unsupervised task was the generation of the missing subsequences (15 % of the encoder’s input, in a similar way as SpanBERT (Joshi et al., 2020)), infilled with special tokens. T5’s vocabulary contains special tokens <1>, <2>, <3>, . . . that are used to represent the missing subsequences. For instance, the text ”*Thank you for inviting me to your party last week*“, is masked as ”*Thank you <1> me to your party <2> week.*“, which is used as the encoder’s input. The target sequence the decoder is asked to generate is then ”*<1> for inviting <2> last <3>*“. The objective is the standard CE for autoregressive language models. The unsupervised pretraining is done on the newly collected C4 corpus[7] of ~156B tokens (about 47x more than BERT).

The major architectural difference are relative positional encodings, added as a scalar to every self-attention score, instead of absolute encoding at the input. Other minor architectural changes include (i) no additive biases included in layer normalization, and (ii) residual skip connection is computed after layer normalization (not before), dropouts are applied differently.

[6] Relative positional encodings do not encode the absolute position i, but a distance between two positions $i - j$ at the model’s input.

[7] Available at https://www.tensorflow.org/datasets/catalog/c4. Accessed 8.3.2023.

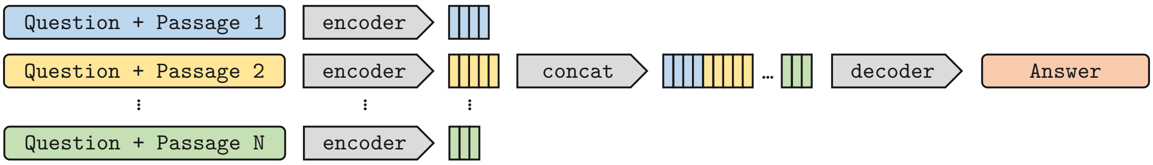


Figure 2.1: Schema of Fusion-in-Decoder architecture for QA. Each encoder input is a different passage, concatenated with a question. The figure is taken from Izacard and Grave (2021b).

2.3.1 Fusion-in-Decoder Architecture

This thesis builds or compares to Fusion-in-Decoder (FiD) architecture (Izacard and Grave, 2021b) in several chapters. For instance, R2-D2's generative reader (Subsection 4.2.3) and an EGG method (Section 7.5) are based on T5 (Raffel et al., 2020) connected in FiD setup. Fusion-in-Decoder is a method for processing very long inputs by encoder-decoder models pretrained on short inputs. Specifically, the input I is chunked (often naturally, e.g. different documents) into $I_1, I_2, I_3...$ and each is independently encoded through the model's encoder into representations $\boldsymbol{H}_1, \boldsymbol{H}_2, \boldsymbol{H}_3, ..., , \boldsymbol{H}_i, ...$, where $\boldsymbol{H}_i \in \mathbb{R}^{L_i \times d}$, d is the model's hidden dimensionality and L_i is the length of the i-th chunk in tokens. All n chunk representations are then concatenated into $\boldsymbol{H} = [\boldsymbol{H}_1, \boldsymbol{H}_2, ..., \boldsymbol{H}_n]$, $\boldsymbol{H} \in \mathbb{R}^{L \times d}$, and $L = \sum_i L_i$. Single representation $\boldsymbol{H}$ is co-attended in the transformer's decoder (Figure 2.1).

2.4 Techniques for Training Memory-Expensive Models

This section discusses technical details crucial for training large context-grounded models typical for document-grounded reasoning (common in QA and FC).

Gradient Checkpointing (Chen et al., 2016a) is a technique for controlling the memory-computation trade-off. In a typical deep learning model, the forward pass involves computing intermediate activations, which are stored for the backward pass to efficiently compute gradients during backpropagation. Instead of storing all intermediate activations during the forward pass, gradient checking stores only a subset of them, known as ”checkpoints“, which are used to recompute the remaining activations during backpropagation. This allows for a smaller memory footprint during training, at the cost of some additional computation during the backward pass. The gradient checkpointing in Transformers library (Wolf et al., 2020), used for implementing Transformer networks in this thesis, does not save intermediate variables for any pretrained model layers, for maximu memory efficiency.

Gradient Accumulation is a memory-computation trade-off technique, which allows training model with a larger minibatch size, than the number of samples that fit into memory. Considering a dataset $X = \{x_1, x_2, ...\}$ and a per-sample loss function ℓ_θ, the loss for minibatch of examples $X_\mathcal{B} \subset X$ is $\mathcal{L}_\theta(X_\mathcal{B}) = \frac{1}{|X_\mathcal{B}|} \sum_{x \in X_\mathcal{B}} \ell_\theta(x)$. Then, the gradient ∇_θ w.r.t. parameters θ is $\nabla_\theta \mathcal{L}_\theta(X_\mathcal{B}) = \sum_{x \in X_\mathcal{B}} \nabla_\theta \frac{1}{|X_\mathcal{B}|} \ell_\theta(x)$. This means, if at least 1 sample fits the memory of our hardware, any minibatch size can be simulated, by computing the result $\nabla_\theta \frac{1}{|X_\mathcal{B}|} \ell_\theta(x)$ in each iteration, and summing to results for n iterations to achieve the minibatch size n. The optimization step (corresponding to the change of the model's parameters θ) is performed only every n iterations. Note that this is especially advantageous for multi-GPU implementations (explained below).

Multi-GPU Training is a technique, which allows to optimize the same model at the different GPUs (and possibly different machines) concurrently. Each GPU g from the pool of GPU's G obtains a piece of minibatch $X_{\mathcal{B}g}$, $X_\mathcal{B} = \bigcup_{g \in G} X_{\mathcal{B}g}$, and computes its gradient $\nabla_\theta \mathcal{L}(X_{\mathcal{B}g}) = \nabla_\theta \sum_{x \in X_\mathcal{B}} \frac{1}{|X_{\mathcal{B}g}|} \ell_\theta(x)$. When $\nabla_\theta \mathcal{L}(X_{\mathcal{B}g})$ is computed on each GPU, the machines will perform synchronization, and average their gradients to obtain gradient for minibatch $\nabla_\theta \mathcal{L}(X_\mathcal{B}) = \frac{1}{|G|} \sum_{g \in G} \nabla_\theta \mathcal{L}(X_{\mathcal{B}g})$. The usage of multi-GPU training is especially beneficial for maximum GPU utilization when combined with gradient accumulation. The synchronization then needs to be done only once the accumulation is done, resulting in significantly less frequent synchronization and thus much less overhead.

Part I

Factoid Question Answering

Chapter 3

Rethinking the Objectives of Extractive QA

The goal of extractive question answering (EQA) is to find the span boundaries—the start and the end of the span from text evidence, which answers a given question in the provided evidence[1]. Therefore, a natural choice of the objective for this problem is to model the probabilities of the span boundaries. In the last years, there was a lot of effort put into building better neural models underlying the desired probability distributions. However, there has been little progress seen toward the change of the objective itself. For instance, the "default" choice of objective for modeling the probability over spans in SQuADv1.1 (Rajpurkar et al., 2016)—maximization of independent span boundary probabilities $\mathrm{P}(a_s|\cdot)\,\mathrm{P}(a_e|\cdot)$ for the answer at position $\langle a_s,a_e\rangle$—has stayed the same over the course of years in many influential works (Xiong et al., 2017; Seo et al., 2017; Chen et al., 2017a; Yu et al., 2018; Devlin et al., 2019; Cheng et al., 2020) since the earliest work on this dataset—the submission of Wang and Jiang (2017). Based on the myths of worse performance of different objectives, these works adopt the deeply rooted assumption of independence. However, this assumption may lead to obviously wrong predictions, as shown in Figure 3.1. In addition, this assumption leads to degenerate distribution $P(a_s, a_e|\cdot)$, as high probability mass is assigned to many trivially wrong[2] answers.

Some of the earlier work (Wang and Jiang, 2017; Weissenborn et al., 2017) and recent approaches including large language representation models (LRMs) like XLNet (Yang et al., 2019c), ALBERT (Lan et al., 2020) or ELECTRA (Clark et al., 2020) started modeling the span probability via conditional probability factorization $P(a_e|a_s,\cdot)P(a_s|\cdot)$. However, is it unknown whether this objective improves any performance at all, as almost none of the recent works reported results on its effect, not even described its existence (except ELECTRA paper). Additionally, this objective requires beam search which slows down inference in test time. Exceptionally, Lee et al. (2016) proposed one way for modelling $\mathbf{P}(a_s, a_e)$ directly, but the approach was only sparsely adopted (Lee et al., 2019; Khattab et al., 2021b). This may be caused by the belief, that enumerating all possible spans has a large complexity (Cheng et al., 2021a). However, in practice we find the complexity to be

[1] In this chapter, all definitions and derivations are made for CDQA problem, where the answer is extracted from a single piece of evidence. The presented approaches can be easily extended to the multi-document scenario, as shown in the subsequent Chapter 4.

[2] We roughly define "trivially wrong" as not resembling any string form human would answer, e.g., the first or the second last answer of Figure 1.3.

Question: What was the name of atom bomb dropped by USA on Hiroshima?

Passage: ...The Allies issued orders for atomic bombs to be used on four Japanese cities were issued on July 25. on August 6, one of its b - 29s dropped a little boy uranium gun-type bomb on Hiroshima. three days later, on August 9, a fat man plutonium implosion-type bomb was dropped by another b - 29 on Nagasaki...

Ground truth: little boy

P$(a_s, a_e\|\cdot)$	**Top Predictions from QA model**
33.3	little boy uranium gun-type bomb on Hiroshima. three days later, on August 9, a fat man
32.15	little boy
23.51	fat man
3.60	a fat man
2.08	a little boy uranium gun - type bomb on hiroshima. three days later, on august 9, a fat man
1.03	a little boy

Figure 3.1: An example of an error that comes with an independence assumption. The model assigns high probability mass to boundaries around "little boy", and "fat man" answers. However, the maximum probability is assigned to the start of one and the end of another answer. Hence the model produces obviously wrong answer.

often similar to the assumption of independence when implementing the objective efficiently. We continue with the analysis and the in-depth discussion on complexity in Appendix F.1.

In this work, we try to break the myths about the objectives that have been widely used previously. We experiment with the joint objective and we also introduce a new *compound* objective, that deals with modeling joint probability $P(a_s, a_e|\cdot)$ directly while keeping the traditional independent objective as an auxiliary objective. We experiment with 5 different joint probability function realizations and find that with current LRMs, simple dot product works the best. However, we show that this is not a rule, and for some models, other function realizations might be better. The conducted experiments demonstrate that using compound objective is superior or equal to (i.e., Pareto optimal to) previously used objectives across the various choices of models or datasets.

In summary, the contributions of this chapter are:

1. introduction of new ways of efficient joint probability computation, adding negligible theoretical computation/memory complexity overhead and no practical overhead over the assumption of independence,

2. introduction of the compound objective and its comparison with the traditional objectives based on the assumption of independence, conditional probability factorization, or direct joint probability,

3. a thorough evaluation on the wide spectrum of models and datasets comparing different objectives supported by statistical tests,

4. a manual analysis that provides a closer look at the different impacts of independent and compound objectives.

3.1 Probabilistic Assumptions for the Answer Span

This section describes the common approach to the EQA, with its independent modeling of the answer span start and end positions. Secondly, it defines an assumption based on conditional factorization of span probability. Finally a function family for computing joint span probability and a combination of independent and joint assumption we call the *compound* objective are proposed.

The EQA can be defined as follows: Given a question q and a passage or a set of passages D, find a string a from D such that a answers the question q. This can be expressed by modeling a categorical probability mass function (PMF) that has its maximum in the answer start and end indices $a = \langle a_s, a_e \rangle$ from the passage D as $\mathbf{P}(a_s, a_e|q, D)$ for each question-passage-answer triplet (q, D, a) from the dataset $\mathcal{D}$. The parameters θ of such a model can be estimated by minimizing the negative maximum likelihood objective

$$-\sum_{(q,D,a)\in\mathcal{D}} \log \mathrm{P}_\theta(a_s, a_e|q, D). \tag{3.1}$$

During inference, the most probable answer span $\langle a_s, a_e \rangle$ is predicted. Although there are works that were able to model the joint probability explicitly (Lee et al., 2016), modeling it directly results in a number of categories quadratic to the passage's length. Optimizing such models may be seen as challenging, as there are often more classes than the amount of data points within the current datasets. For illustration, considering a typical length of BERT inputs 512, naive implementation would lead to roughly $\approx 131,000$ categories. Therefore, state-of-the-art approaches resort to independence assumption $\mathrm{P}(a_s, a_e|q, D) = \mathrm{P}_\theta(a_s|q, D)\,\mathrm{P}_\theta(a_e|q, D)$. The factorized probability is usually computed by the model with shared parameters θ, as introduced in Wang and Jiang (2017). For most of the systems modeling the *independent objective* with neural networks, the final endpoint probabilities[3] are derived from start/end position passage representations computed via shared model $\boldsymbol{H}_s$,$\boldsymbol{H}_e \in \mathbb{R}^{d\times L}$ as shown for $b \in \{s, e\}$.

$$\mathbf{P}_\theta(a_b) = \mathrm{softmax}(\boldsymbol{w}_b^\top \boldsymbol{H}_b + \boldsymbol{b}_b) \tag{3.2}$$

The passage representations $\boldsymbol{H}_s$, $\boldsymbol{H}_e$ are often pre-softmax layer representations from neural network with passage and question at the input. Symbols d and L denote the model-specific dimension and the passage length, respectively.

Occasionally, the conditional factorization of probabilities $\mathrm{P}(a_s, a_e|q, D) = \mathrm{P}_\theta(a_s)\,\mathrm{P}_\theta(a_e|a_s)$ is considered instead. The probabilities of span's start and end are computed the same way as in equation 3.2. The difference is in the end representations $\boldsymbol{H}_e = f(a_s)$, which now must be a function of span's start a_s.

3.1.1 Joint Assumptions

However, one does not need to apply simplifying assumptions and instead compute joint probability directly. We define a family of *joint* probability functions $\mathbf{P}_\theta(a_s, a_e)$ with an arbitrary vector-to-vector similarity function f_{sim} used for obtaining each span score (e. g., the dot product $\boldsymbol{H}_s^\top \boldsymbol{H}_e$)[4].

$$\mathbf{P}_\theta(a_s, a_e) = \mathrm{softmax}(\mathrm{vec}(f_{sim}(\boldsymbol{H}_s, \boldsymbol{H}_e))) \tag{3.3}$$

[3] For brevity, q, D dependencies are further omitted and bias terms are broadcasted along dimension L.

[4] Here, we slightly abuse the notation for the sake of generality. See Subsection 3.2.2 for specific applications.

Finally, we define a multi-task *compound* objective (3.4) composing the joint and independent probability formulations, computed via a shared model θ.

$$-\sum_{(q,D,a)\in\mathcal{D}} \log \mathrm{P}_\theta(a_s, a_e)\, \mathrm{P}_\theta(a_s)\, \mathrm{P}_\theta(a_e) \tag{3.4}$$

Here probability $\mathrm{P}(a_s)\,\mathrm{P}(a_e)$ can be seen as an auxiliary objective for the more complex joint objective $\mathrm{P}_\theta(a_s, a_e)$ used for decoding in test time. Empirically, we found the *compound* objective to be Pareto optimal to other presented assumptions.

3.2 Experimental Setup

We use Transformers (Wolf et al., 2020) for language representation model (LRM) implementation. Our experiments were done on 16GB GPUs using PyTorch (Paszke et al., 2019). For experiments with LRMs, we used Adam optimizer with a decoupled weight decay (Loshchilov and Hutter, 2017). The used hyperparameters were the same as the SQuADv1.1 default hyperparameters as proposed by specific LRM authors through all our datasets. For BIDAF, we tuned hyperparameters using Hyperopt (Bergstra et al., 2013) separately for independent and compound objectives[5]. See Appendix A.1 for further details.

In all our experiments, we apply *length filtering* (LF). Therefore, probabilities $P(a_s = i, a_e = j)$ are set to 0 iff $j - i > \zeta$, where ζ is a length threshold. Following Devlin et al. (2019), we set $\zeta = 30$ in all of our experiments.

3.2.1 Similarity Functions

Here we sum up the definitions of similarity functions presented in the paper. We experimented with 5 similarity functions. For each start representation $\boldsymbol{h_s} \in \mathbb{R}^d$ and end representation $\boldsymbol{h_e} \in \mathbb{R}^d$, both column vectors from the matrix of boundary vectors $\boldsymbol{H}_s$, $\boldsymbol{H}_e \in \mathbb{R}^{d\times L}$ respectively. Note that d here is model specific dimension, L is passage length, $\circ$ denotes elementwise multiplication and $[;]$ denotes concatenation. The similarity functions above these representations are defined as:

- A dot product:
$$f_{dot}(\boldsymbol{h_s}, \boldsymbol{h_e}) = \boldsymbol{h_s}^\top \boldsymbol{h_e}. \tag{3.5}$$

- A weighted dot product:
$$f_{wdot}(\boldsymbol{h_s}, \boldsymbol{h_e}) = \boldsymbol{w}^\top [\boldsymbol{h_s} \circ \boldsymbol{h_e}]. \tag{3.6}$$

- An additive similarity:
$$f_{add}(\boldsymbol{h_s}, \boldsymbol{h_e}) = \boldsymbol{w}^\top [\boldsymbol{h_s}; \boldsymbol{h_e}]. \tag{3.7}$$

- An additive similarity combined with weighted product:
$$f_{madd}(\boldsymbol{h_s}, \boldsymbol{h_e}) = \boldsymbol{w}^\top [\boldsymbol{h_s}; \boldsymbol{h_e}; \boldsymbol{h_s} \circ \boldsymbol{h_e}]. \tag{3.8}$$

- A multi-layer perceptron (MLP) as proposed by Lee et al. (2019):
$$f_{MLP}(\boldsymbol{h_s}, \boldsymbol{h_e}) = \boldsymbol{w}^\top \sigma(\boldsymbol{W}[\boldsymbol{h_s}; \boldsymbol{h_e}] + \boldsymbol{b}) + \boldsymbol{b}_2, \tag{3.9}$$

  where $\sigma(x) = \ln(\mathrm{ReLU}(x))$ and ln denotes layer normalization (Ba et al., 2016).

[5] We used f_{madd} similarity during parameter tuning.

3.2.2 Applied Models

Our experiments are based on three EQA models:

BERT-base (Devlin et al., 2019) and **ALBERT**-xxlarge (Lan et al., 2020) are LRMs based on the self-supervised pretraining objective (see Section 2.3). During fine-tuning, each model receives the concatenation of question and passage as input. Outputs $\boldsymbol{H} \in \mathbb{R}^{d \times L}$ corresponding to the passage inputs of length L are then reduced to boundary probability distributions by two vectors $\boldsymbol{w_s}$, $\boldsymbol{w_e}$ as $\mathbf{P}(a_b) = \text{softmax}(\boldsymbol{w}_b^\top \boldsymbol{H} + \boldsymbol{b}_b)$ where $b \in \{s, e\}$. To compute joint probability distribution $\mathbf{P}(a_s, a_e)$, start representations are computed using $\boldsymbol{W} \in \mathbb{R}^{d \times d}$ and $\boldsymbol{b} \in \mathbb{R}^d$ (broadcasted) as $\boldsymbol{H}_s = \boldsymbol{W}\boldsymbol{H} + \boldsymbol{b}$ and end representations as $\boldsymbol{H}_e = \boldsymbol{H}$. Unless stated otherwise, a dot product f_{dot} is used as the similarity measure.

$$\mathbf{P}(a_s, a_e) = \text{softmax}(\text{vec}(\boldsymbol{H}_s^\top \boldsymbol{H}_e)) \tag{3.10}$$

For comparison with the conditional objective, we reimplemented the conditional objective used in ALBERT (Lan et al., 2020). First, the probability distribution $\mathbf{P}(a_s)$ for the start position is computed in the same manner as for the independent objective—by applying a linear transformation layer on top of representations $\boldsymbol{H} \in \mathbb{R}^{d \times L}$ from the last layer of the LRM, where d is the model dimension and L denotes the input sequence length.

$$\mathbf{P}(a_s) \propto \exp\left(\boldsymbol{w}_s^\top \boldsymbol{H} + \boldsymbol{b_s}\right) \tag{3.11}$$

During the validation, top-k ($k = 10$ in our experiments) start positions are selected from these probabilities, while in the training phase, we apply teacher forcing by only selecting the correct start position. Representation of i-th start position $\boldsymbol{h}_i$ from the last layer of the LRM corresponding to the selected position is then concatenated with representations corresponding to all the other positions $k = 0..L$ into matrix $\boldsymbol{C}$.

$$\boldsymbol{C} = \begin{bmatrix} — & [\boldsymbol{h_0}; \boldsymbol{h_i}] & — \\ — & [\boldsymbol{h_1}; \boldsymbol{h_i}] & — \\ & \vdots & \\ — & [\boldsymbol{h_n}; \boldsymbol{h_i}] & — \end{bmatrix} \tag{3.12}$$

Subsequently, a layer with tanh activation is applied on this matrix $\boldsymbol{C}$, followed by a linear transformation to obtain the end probability distribution:

$$\mathbf{P}(a_e | a_s = i) \propto \exp\left(\boldsymbol{w}_c^\top \tanh\left(\boldsymbol{W}\boldsymbol{C} + \boldsymbol{b'}\right) + \boldsymbol{b}\right) \tag{3.13}$$

For each start position, we again select top-k end positions, to obtain k^2-best list of answer spans. In contrast to the official ALBERT implementation, we omitted a layer normalization after tanh layer.

BIDAF (Seo et al., 2017) dominated the state-of-the-art systems in 2016 and motivated a lot of following research work (Clark and Gardner, 2018; Yu et al., 2018). Question and passage inputs are represented via the fusion of word-level embeddings from GloVe (Pennington et al., 2014) and character-level word embeddings obtained via a convolutional neural network. Next, a recurrent layer is applied to both. Independently encoded questions and passages are then combined into a common representation via two directions of attention over their similarity matrix $\boldsymbol{S}$. The similarity matrix is computed via multiplicative-additive interaction from equation (3.14) between each pair of question vector $\boldsymbol{q_i}$ and passage vector $\boldsymbol{p_j}$, where $[;]$ denotes row-wise vector concatenation and $\circ$ stands for the Hadamard product.

$$\boldsymbol{S_{ij}} = f_{madd}(\boldsymbol{q_i}, \boldsymbol{p_j}) = \boldsymbol{w}^\top[\boldsymbol{q_i}; \boldsymbol{p_j}; \boldsymbol{q_i} \circ \boldsymbol{p_j}] \tag{3.14}$$

Common representations are then concatenated together with document representations yielding $\boldsymbol{G}$ and passed towards two more recurrent layers producing $\boldsymbol{M}$ and $\boldsymbol{M}^2$—first to obtain answer-start representations $\boldsymbol{H}_s = [\boldsymbol{G}; \boldsymbol{M}]$ and second to obtain answer-end representations[6] $\boldsymbol{H}_e = [\boldsymbol{G}; \boldsymbol{M}^2]$. Unless stated otherwise, the joint probability $\mathrm{P}(a_s, a_e)$ is then computed over scores from vectorized similarity matrix of $\boldsymbol{H}_s$ and $\boldsymbol{H}_e$ using the 2-layer feed-forward network f_{MLP} as a similarity function. We refer reader to read the original publication (Seo et al., 2017) for the in-detail description of BIDAF architecture.

3.2.3 Datasets

Dataset	Train	Test
SQuADv1.1	87,599	10,570
SQuADv2.0	130,319	11,873
Adversarial SQuAD	-	3,560
Natural Questions	104,071	12,836
NewsQA	74,160	4,212
TriviaQA	61,688	7,785

Table 3.1: Number of examples per each dataset used.

We evaluate our approaches on a wide spectrum of datasets. We do not split development datasets, as we use fixed hyperparameters with a fixed amount of steps and use the last checkpoint for our LRM experiments. This also makes our results directly comparable to other works (Devlin et al., 2019; Lan et al., 2020)[7]. The statistics for all datasets are shown in Table 3.1. To focus only on the extractive part of QA and to keep the format the same, we use curated versions of the last 3 datasets as released in MrQA shared task (Fisch et al., 2019). The curation process converted and filtered the dataset according to two criterias into a unified, extractive format. Firstly, the answer to each question must appear as a span of tokens in the passage. Secondly, passages, which may span through multiple paragraphs or documents, are concatenated and truncated to the first 800 tokens. This eases the computational requirements for processing long documents efficiently.

SQuADv1.1 (Rajpurkar et al., 2016) is a popular dataset composed of question, paragraphs, and answer span annotation collected from the subset of Wikipedia passages.

SQuADv2.0 (Rajpurkar et al., 2018) is an extension of SQuADv1.1 with additional 50k questions and passages, which are topically similar to the question, but do not contain an answer.

Adversarial SQuAD (Jia and Liang, 2017) tests, whether the system can answer questions about paragraphs that contain adversarially inserted sentences, which are automatically generated to distract computer systems without changing the correct answer or misleading humans. In particular, our system is evaluated in AddSent adversary setting, which runs the model as a black box for each question on several paragraphs containing different adversarial sentences and picks the worst scoring answer.

[6] For details, see formulas 2 to 4 in Seo et al. (2017).

[7] Submissions on the SQuAD test set need to be manually validated, which often takes months in practice.

Natural Questions (Kwiatkowski et al., 2019) dataset consists of real user queries obtained from Google search engine. Each example is accompanied by a relevant Wikipedia article found by the search engine, and human annotation for long/short answer. The long answer is typically the most relevant paragraph from the article, while short answer consists of one or multiple entities or short text spans. We only consider short answers in this work.

NewsQA (Trischler et al., 2017) is a crowd-sourced dataset based on CNN news articles. Answers are short text spans and the questions are designed such that they require reasoning and inference besides simple text matching.

TriviaQA (Joshi et al., 2017) consists of question-answer pairs from 14 different trivia quiz websites and independent evidence passages collected using Bing search from various sources such as news, encyclopedias, blog posts, and others. Additional evidence is obtained from Wikipedia through entity linker.

Sample from each used dataset can be found within Appendix E.1.

3.2.4 Statistical Testing

To improve the soundness of the presented results, we use statistical testing. An exact match (EM) metric can be viewed as an average of samples from the Bernoulli distribution. As stated with the central limit theorem, a good assumption might be the EM comes from the normal distribution. We train 10 models for each presented LRM's result, obtaining 10 EMs for each sample. Then we use the *one-tailed unpaired unequal variances t-test* to check whether the case of improvement is significant. The improvement is considered significant iff p-value < 0.05.

3.2.5 Metrics

In this chapter, the performance of the proposed method is measured through two evaluation metrics: *Exact Match* (**EM**) and *F1 for Question Answering* (**F1**). Both metrics are described in-depth in Section 2.2.

3.3 Results

We now show the effectiveness of the proposed approaches. Each of the presented results is averaged from 10 training runs.

	EM	**F1**	**EM**	**F1**
I	66.16	76.19	81.31	88.65
I+J	BIDAF		BERT	
f_{dot}	64.30	73.84	**81.83**	**88.52**
f_{add}	66.04	75.10	81.52	88.47
f_{wdot}	66.10	75.16	81.35	88.29
f_{madd}	66.11	75.23	81.45	88.44
f_{MLP}	**66.96**	**75.90**	81.61	88.44

Table 3.2: A comparison of similarity functions in the models trained via *compound* objective (I+J) and *independent* objective (I).

Model	Obj	SQ1	SQ2	AdvSQ	TriviaQA	NQ	NewsQA
BERT	I	81.31/88.65	**73.89**/76.74	47.04/52.62	62.88/69.85	65.66/78.20	52.39/67.17
	J	81.33/88.13	72.66/75.04	48.10/53.54	**63.93**/69.90	**67.75**/78.70	52.73/66.41
	JC	81.22/88.29	71.51/74.38	46.07/51.35	62.82/69.94	66.48/77.34	52.39/67.05
	I+J	***81.83***/88.52	73.53/76.14	***48.32***/53.47	*63.73*/69.75	***67.75***/78.81	***52.96***/66.83
ALBERT	I	88.55/94.62	87.07/90.02	68.12/73.54	74.7/80.33	70.78/83.42	59.95/75.0
	J	88.84/94.64	86.87/89.71	68.90/74.17	75.11/80.41	**73.36/84.01**	60.19/74.28
	JC	88.60/94.59	86.78/89.73	68.0/73.25	-	72.33/83.35	58.52/72.74
	I+J	***89.02***/94.77	**87.13**/89.98	***69.57***/74.76	***75.31***/80.43	*73.32*/84.08	***60.41***/74.46

Table 3.3: EM/F1 results of different objectives through the spectrum of datasets. Bold results mark best EM across the objectives. Italicized I+J results mark a significant improvement over the independent objective.

Similarity Functions . We analyzed the effect of different similarity functions over all models in Table 3.2. We found different similarity functions to work better with different architectures. Namely, for BIDAF, most of the similarity functions work equally or worse than the independent objective. Exceptionally, f_{MLP} works significantly better. This is surprising especially because we tuned the hyperparameters with the f_{madd} function. For BERT, most of the similarity functions performed better than the independent objective, and simple dot-product f_{dot} improved significantly better above all. We choose f_{MLP} for BIDAF and f_{dot} for our LRMs for the rest of the experiments.

Comparison of Objectives . Our main results—the performance of *independent* (I), *joint* (J), *joint-conditional* (JC) and *compound* (I+J) objectives—are shown in Table 3.3. We note the largest improvements can be seen for an exact match (EM) performance metric. In fact, in some cases objectives modeling joint PMF lead to degradation of F1, while improving EM (e.g., on SQuADv1.1 and NewsQA datasets for BERT). Upon manual analysis of BERT's predictions based on 200 differences between independent and compound models on SQuADv1.1, we found a potential explanation. In 10 cases (5 %) the independent model chooses a larger span encompassing multiple potential answers, thus obtaining a nonzero F1 score. In 9 out of 10 of these cases, we found the compound model to pick just one of these potential answers[8], obtaining either full match or no F1 score at all. We found no cases of compound model encompassing multiple potential answers in the analyzed sample.

Next, we remark that compound objective outperformed others in most of our experiments. In the BERT case, the compound objective performed significantly better than the independent objective on 5 out of 6 datasets. In ALBERT case, the compound objective performed significantly better than the independent objective 5 from 6 times and it was on par in the last case. Comparing compound to joint objective in BERT and ALBERT cases, the two objectives produce similar results, with compound objective performing slightly better when significantly outperforming joint objective on the two SQuAD datasets and no significant differences for the other 4 datasets.

Conditional Objective Our implementation of the conditional objective performs even or worse to the independent objective in most cases. Upon investigation, we found the model tends to be overconfident about start predictions and underconfident about its end predictions, often assigning high probability to a single answer-start. In Table 3.4, we analyze the top-5 most probable samples from BERT on each example of SQuADv1.1 dev data. We found that on average the conditional model kept its top-1 start prediction in

[8] For instance, in Table 3.7, row 4, column 3, we consider 2,000; 40,000; 2,200; 1,294 and 427 as potential answers.

90 % of subsequent top-2 to top-5 less probable answers, but kept its top-1 end prediction only in 4 % of top-2 to top-5 subsequent answers. We found this statistic to be on par for start/end prediction for different objectives. Interestingly, Table 3.4 also reveals that independent objective contains less diverse start/end tokens than joint objectives. Based on these results, and assumptions done in the model, we hypothesize that: (a) the conditional objective selects the answer's start regardless of whether a model can confidently identify the end of this answer (b) since the independent model selects the start and end independently, it is prone to select high-probability start and then only change end predictions in subsequent top-2 to top-5 predictions (or vice-versa select end and change starts in subsequent predictions).

Largest Improvements and Degradation Upon closer inspection of the results, we found possible reasons for the result degradation of the compound model on SQuADv2.0, and also its large improvements gained on the NQ dataset.

For SQuADv2.0, the accuracies of no-answer detection for independent/joint/compound objectives in the case of BERT models are 79.89/78.12/79.32. We found the same trend for ALBERT. We hypothesize, that this inferior performance of joint and compound models may be caused by the model having to learn a more complex problem of K^2 classes of all possible spans over the input document, which is often more (e.g. for $K = 512$) than the size of the datasets, leaving the less of "model capacity" to this another task. To confirm that the compound model is better at the answer extraction step, we run all 10 checkpoints trained on *SQuADv2.0 data with an answer*, while masking the model's no-answer option. The results shown in Table 3.5 support this hypothesis.

On the other side, we found the large improvements over NQ might be exaggerated by the evaluation approach of MRQA, where in the case of multi-span answers, choosing one of the spans from multi-span answer counts as correct. Upon closer result inspection, we found that the independent model here was prone to select the start of one span from the multi-span answer and the end of a different span from the multi-span answer. To quantify this behavior, we annotated 100 random predictions with multi-span answers in original NaturalQuestions on whether they pick just one span from the multi-span answer (which follows from the MRQA formulation) or they encompass multiple spans. For independent/-compound objectives we found 59/77 cases of picking just one of the spans and 22/4 cases of encompassing multiple spans from the multi-span answer for the BERT model and 57/81 and 33/10 cases for the ALBERT respectively.

Length Filtering Heuristic Additionally, we found the benefit from the commonly used length filtering (LF) heuristic is negligible for models trained via any joint objective, as shown in Table 3.6. Therefore, we find it unnecessary to use the heuristic anymore. In this experiment, we also include our results with BIDAF, which show significant improvement of compound objective on the SQuADv1.1 dataset from other approaches.

Model	**Start Token**	**End Token**
I	43.76%	45.66%
JC	90.12%	3.9%
J	33.71%	35.91%
I+J	34.95%	37.27%

Table 3.4: Proportion of samples, on which top-1 prediction start/end token was kept as start/end token also in top-2 to top-5 subsequent predictions.

3.4 Analysis

Fixed Error Types Apart from an example in Figure 1.3, we provide more examples of different predictions[9] between models trained with the independent and compound objective in Table 3.7. In general, by doing manual analysis of errors, we noticed three types of trivially wrong errors being fixed by the compound objective model in BERT:

1. Uncertainty of the model causes it to assign high probabilities to two different answer boundaries. During decoding the start/end boundaries of two different answers are picked up (fourth row in Table 3.7).
2. The model assigns high probability to answer surrounded by the paired punctuation marks (e. g. quotes). It chooses the answer without respecting the symmetry between paired punctuation marks (third row of Table 3.7).
3. Uncertainty of the model causes it to assign high probabilities to two spans containing the same answer string. This is the special case of problem (1)—while the model often chooses the correct answer, the boundaries of two different spans are selected (first row of Table 3.7).

To quantify the occurrence of these errors, we study our best BERT and ALBERT checkpoint predictions for SQuADv1.1 validation data. For BERT, we found the most frequently occurring is error type (1), for which we manually annotated 200 random differences between independent and compound model predictions. We found *5 %* of them to be the case of this error of the independent, and no case of this error for the compound model. Interestingly, 4 out of 10 of these cases were questions clearly asking about a single entity, while the independent model answered multiple entities, e.g., *Q: Which male child of Ghengis Khan and Börte was born last? A: Chagatai (1187—1241), Ögedei (1189—1241), and Tolui.* For the error type (2), we filtered all prediction differences (more than 1300 for BERT and ALBERT) down to cases, where either independent or compound prediction contained non-alphanumeric paired punctuation marks, which resulted in less than 30 cases for each. For BERT, *37 %* independent predictions from these cases contained an error type (2), while again no paired punctuation marks errors were observed for the compound objective.

For the error type (3), we filter prediction differences down to cases, where independent or compound prediction contained the same prefix and suffix of length at least 2 (only 9 and 5 cases for BERT and ALBERT). From these, error type (3) occurred in 3 cases for BERT and in 1 case for ALBERT in case of independent and again we found no case for the

	Objective	**EM**	**F1**
BERT	I	80.70	88.71
	J	81.38	81.51
	I+J	81.51	88.69
ALBERT	I	87.40	94.10
	J	87.74	94.31
	I+J	87.90	94.38

Table 3.5: Performance of SQuADv2.0 models on answerable examples of SQuADv2.0.

[9] We chose to analyze the different predictions, as the model is usually more uncertain in these borderline cases.

Model		**I**	**J**	**JC**	**I+J**
BIDAF	-	65.85/75.94	-	-	66.95/75.89
	LF	**66.16/76.19**	58.24/67.42	-	66.96/75.90
BERT	-	80.98/88.40	81.30/88.11	81.16/88.25	81.80/88.50
	LF	**81.31/88.65**	81.33/88.13	81.22/88.29	81.83/88.52
ALBERT	-	88.39/94.51	88.82/94.64	88.57/94.57	89.01/94.77
	LF	**88.55/94.63**	88.84/94.64	88.60/94.59	89.02/94.77

Table 3.6: SQuADv1.1 EM/F1 results with length filtering (LF) computed from the same set of checkpoints. Differences larger than 0.1 are in bold.

Question	**Passage**	**Independent**	**Compound**	**Ground Truth**
What company won a free advertisement due to the QuickBooks contest?	QuickBooks sponsored a „Small Business Big Game“ contest, in which Death Wish Coffee had a 30-second commercial aired free of charge courtesy of QuickBooks. Death Wish Coffee beat out nine other contenders from across the United States for the free advertisement.	Death Wish Coffee had a 30-second commercial aired free of charge courtesy of QuickBooks. Death Wish Coffee	Death Wish Coffee	Death Wish Coffee
In what city's Marriott did the Panthers stay?	The Panthers used the San Jose State practice facility and stayed at the San Jose Marriott. The Broncos practiced at Stanford University and stayed at the Santa Clara Marriott.	San Jose State practice facility and stayed at the San Jose	San Jose	San Jose
What was the first point of the Reformation?	Luther's rediscovery of „Christ and His salvation“ was the first of two points that became the foundation for the Reformation. His railing against the sale of indulgences was based on it.	Christ and His salvation„	Christ and His salvation	Christ and His salvation
How many species of bird and mammals are there in the Amazon region?	The region is home to about 2.5 million insect species, tens of thousands of plants, and some 2,000 birds and mammals. To date, at least 40,000 plant species, 2,200 fishes, 1,294 birds, 427 mammals, 428 amphibians, and 378 reptiles have been scientifically classified in the region. One in five of all the bird species in theworld live in the rainforests of the Amazon, and one in five of the fishspecies live in Amazonian rivers and streams. Scientists have describedbetween 96,660 and 128,843 invertebrate species in Brazil alone.	2,000 birds and mammals. To date, at least 40,000 plant species, 2,200 fishes, 1,294 birds, 427	427	2,000
What was found to be at fault for the fire in the cabin on Apollo 1 regarding the CM design?	NASA immediately convened an accident review board, overseen by both houses of Congress. While the determination of responsibility for the accident was complex, the review board concluded that “deficiencies existed in Command Module design, workmanship and quality control.„ At the insistence of NASA Administrator Webb, North American removed Harrison Storms as Command Module program manager. Webb also reassigned Apollo Spacecraft Program Office (ASPO) Manager Joseph Francis Shea, replacing him with George Low.	deficiencies existed in Command Module design, workmanship and quality control.“	Harrison Storms	deficiencies

Table 3.7: Examples of predictions from SQuADv1.1 using BERT trained with independent and compound objective.

Model		**I**	**J**	**I+J**
BIDAF	LF	66.16/76.19	58.24/67.42	66.96/75.90
	SF	66.20/76.21	-	66.99/75.90
BERT	LF	81.31/88.65	81.33/88.13	81.83/88.52
	SF	81.38/88.68	81.23/87.97	81.65/88.36
ALBERT	LF	88.55/94.63	88.84/94.64	89.02/94.77
	SF	88.53/94.00	88.28/94.10	88.68/94.49

Table 3.8: SQuADv1.1 EM/F1 results with length filtering (LF) and LF + surface form filtering (SF).

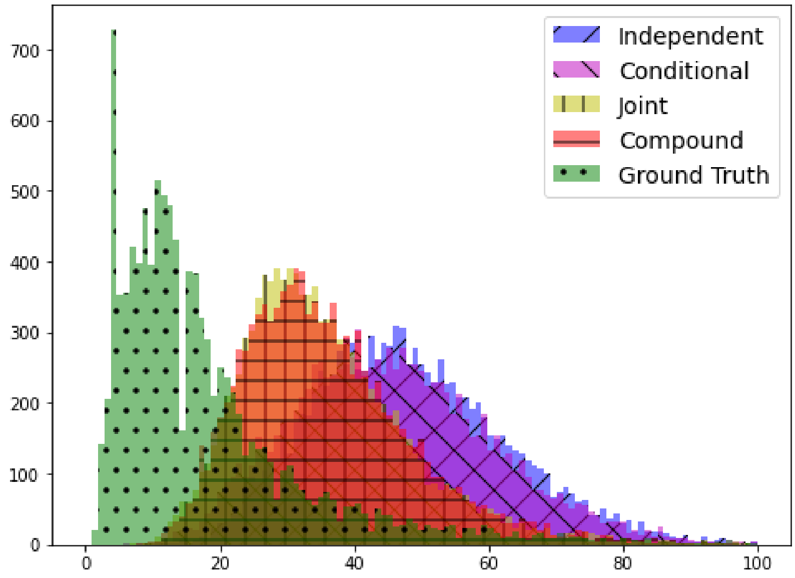


Figure 3.2: Histograms of average character length of top-20 predicted answers from BERT trained with different objectives compared with character length of ground-truth answers.

compound for both models. Note the error type (3) can be fully alleviated by marginalizing over probabilities of top-K answer spans during the inference, as in (Das et al., 2019; Cheng et al., 2020) (read Part “Marginalizing Over the Same String Forms” below for details). Interestingly, for ALBERT, we found only a negligible amount of errors of type (1) and (2) for both objectives[10].

Marginalizing Over the Same String Forms To alleviate the error type (3) from Section 3.4, we experimented with marginalizing over probabilities of top-100 answers (so-called surface form filtering). This is done by summing the probabilities of the same string spans into the most probable string occurrence and setting the probability of the rest to 0. The results for all trained models averaged over 10 checkpoints are presented in Table 3.8. Note this approach sometimes hurts performance, especially in the case of joint probability approaches, where this error type happens very rarely.

[10] The full difference between BERT’s and ALBERT’s predictions and manual analysis can be found in the supplementary.

Bias Towards Longer Answer Spans Finally, during manual analysis, we observed that uncertain models with an independent objective are prone to pick large answer spans. To illustrate, that spans retrieved with approaches modeling joint probability differ, we took the top 20 most probable spans from each model and averaged their length.

This was done for each example in the SQuADv1.1 test dataset. The histogram of these averages is shown in Figure 3.2. For a fair comparison, these predictions were filtered via length filtering.

3.5 Related Work

One of the earliest works in EQA from Wang and Jiang (2017) experimented with generative models based on index sequence generation via pointer networks (Vinyals et al., 2015) and now traditional boundary models that focus on the prediction of start/end of an answer span. Their work showed substantial improvement of conditional factorization boundary models over the index sequence generative models.

Followup work on EQA (Seo et al., 2017; Chen et al., 2017a; Clark and Gardner, 2018; Yu et al., 2018; Devlin et al., 2019; Cheng et al., 2020) and others considered using the assumption of independence in their objectives.

Xiong et al. (2017) explored an iterative boundary model. They used RNN and a highway maxout network to decode the start/end of the span independently in multiple timesteps, each time feeding the RNN with predictions from the previous time step until the prediction was not changing anymore. In their following work Xiong et al. (2018) combined their objective with a reinforcement learning approach, in which the decoded spans from each timestep were treated as a trajectory. They argued that cross-entropy is not reflecting F1 performance well enough, and defined a reward function equal to F1 score. Finally, they used policy gradients as their auxiliary objective, showing 1 % improvement in the terms of F1 score.

Authors of recent LRMs like XLNet (Yang et al., 2019c), ALBERT (Lan et al., 2020) or ELECTRA (Clark et al., 2020) use conditional probability factorization $\mathrm{P}(a_e|a_s)\,\mathrm{P}(a_s)$ for answer extraction in some cases[11]. Although the objective is not described in mentioned papers (except for ELECTRA), we follow the recipe for modeling the conditional probability from their implementation in this work. We believe this is the first official comparison of this objective w.r.t. others.

The most similar to our work is RaSoR system (Lee et al., 2016). In their work, authors compared various objectives—binary answer classification of every input token, BIO sequence classification with CRF layer on top of their model, and most importantly joint objective, which turns out to work the best. However, in our experiments, training with the joint objective alone does not always perform that well. For BIDAF, we failed to find the hyperparameters for the model to converge to results similar to different approaches.

[11] For instance, ALBERT uses conditional objective for SQuADv2.0, but not for SQuADv1.1.

3.6 Chapter Summary

This chapter closely studies the objectives used within the extractive question answering (EQA). It identifies the commonly used *independent* probability model as a source of trivially wrong answers. As a remedy, it experiments with various ways of learning the joint span probability. Finally, it shows how the *joint* objective, and subsequently the *compound* objective—the combination of independent and joint probability in objective—improves statistical EQA systems across 6 datasets without using any additional data. Using the proposed approach, we were able to reach significant improvements through the wide spectrum of datasets, including +1.28 EM on Adversarial SQuAD and +2.07 EM on NaturalQuestions for BERT-base. We performed a thorough manual analysis to understand what happened to trivially wrong answers, and we found most of the cases disappear. We also found that independent models tend to "overfit" to F1 metric by encompassing multiple possible answer spans, which would explain the effect of joint objectives improving the EM far more consistently than the F1. We showed the samples from the joint model contain the greatest start/end token diversity. We further hypothesize that having diverse answers may be especially beneficial towards answer reranking step commonly used in QA (Fajcik et al., 2021b; Iyer et al., 2021). In addition, we also identified the reason for the performance decrease with *compound* objective on SQuADv2.0—no-answer classifier trained within the same model performs worse—and we leave the solution for this deficiency for future work. In the next Chapter 4, the compound objective will be further applied in ODQA setting, where its contributions were also confirmed.

Chapter 4

On Complementarity of Extractive and Abstractive QA

Years 2020–2021 showed rapid progress in neural factoid open-domain question answering based on *retriever-reader* architecture (ODQA). Such ODQA systems (Voorhees et al., 2001; Chen et al., 2017a) seek evidence for answering the questions inside the knowledge source using the *retriever* and then extract the answer from the retrieved knowledge using the *reader*. The knowledge source is often a large corpus of short snippets of natural language, so-called passages (e.g., taken from an encyclopedia).

The progress can be attributed to advances in neural retrieval methods (Karpukhin et al., 2020; Izacard and Grave, 2021a; Khattab et al., 2021b; Luan et al., 2021; Xiong et al., 2020, *inter alia*) that benefit from smarter negative sampling strategies or a better trade-off between complex question-passage interaction and its efficiency. It also can be attributed to reading methods that enable processing large quantities of retrieved passages Izacard and Grave (2021b). They compensate for a certain amount of the retrieval error and enable early aggregation of the answer's evidence between passages.

This work demonstrates a relative improvement of 23-32 % compared to last year's state-of-the-art DPR system (Karpukhin et al., 2020), while using the same knowledge source and the retriever. We propose a state-of-the-art ODQA baseline composed of a retriever, passage reranker, extractive reader, generative reader, and a novel component fusion approach. We follow the practice from information retrieval and show that our moderately sized reranker allows reducing the passage count needed at the input of large reader models about four times. Our readers then take the best from both worlds. The extractive reader proposes a list of salient answer spans. The abstractive reader reranks these spans, seeing all the passages at once, or generates its own answer. The proposed pipeline is heterogeneous and modular, making it an ideal benchmark.

In summary, the contributions of this chapter are three-fold:

1. We present a simple novel approach to aggregate scores from all system components and show that the complementarity of heterogeneous extractive and abstractive approaches is superior to a posterior averaging ensemble of homogeneous models.
2. We show that the extractive reader can sometimes match the performance of the abstractive approaches without taking the advantage of the fusion between retrieved passages. This indicates that the evidence aggregation from multiple passages in the abstractive approaches is either not learned or not necessary to perform well on these datasets.

3. We push the state-of-the-art for two large and popular datasets, demonstrating what is achievable with the proposed approach, having the same knowledge source and the retriever as in the previous works (Karpukhin et al., 2020; Izacard and Grave, 2021b).

4.1 Open-domain Question Answering Pipeline

We propose the R2-D2 (Rank twice, reaD twice), a 4-stage pipelined system that can choose whether to generate or to extract an answer. The parameters of each component in the pipeline are estimated separately. It is composed of DPR passage retriever (Karpukhin et al., 2020), passage reranker (see Subsection 4.1.1), and two readers. Figure 4.1 shows the diagram of our system. The first reader performs an extractive span-selection similar to the systems in the previous Chapter 3. The second reader is based on Fusion-in-Decoder (FiD) (Izacard and Grave, 2021b).

Formally, given a question $q \in \mathcal{Q}$ from the set of all possible questions $\mathcal{Q}$ and the corpus $\mathcal{C} = \{p_1, p_2, ..., p_n\}$ composed of passages p_i, the retriever learns a ranking function $\text{rank} : \mathcal{Q} \times \mathcal{C} \rightarrow \mathbb{R}$ that assigns a score to each passage. We assume each passage contains its passage title (e.g., a title from the Wikipedia article).

Taking a top-K scoring passages $\mathcal{C}_r \subset \mathcal{C}$, reranker again rescores $\mathcal{C}_r$ scoring passages by learning a reranking function $\text{rerank} : \mathcal{Q} \times \mathcal{C}_r \rightarrow \mathbb{R}$. Note that while rank and rerank have similar signatures, the computational cost of rerank over the same amount of passages is drastically higher, as it computes fine-grained interaction between tokens of question and passage.

Next, the rescored passages are passed to two readers: the extractive reader reads top-V passages $\mathcal{C}_{rr} \subset \mathcal{C}_r$ independently of each other and computes the probability distribution $\mathbf{P}_e(a_e|q, \mathcal{C}_{rr})$ across each span a_e in the passages (see subsection 4.1.2). The FiD generative reader reads top-V_2 passages $\mathcal{C}'_{rr} \subset \mathcal{C}_r$ jointly and generates an answer from autoregressively factorized probability space $\mathbf{P}_g(a_g|q, \mathcal{C}'_{rr})$ via greedy search.

Finally, R2-D2 aggregates the outputs from all components using two fusions (described in Subsection 4.1.3).

4.1.1 Passage Reranker

The proposed passage reranker is based on a transformer cross-encoder similar to Nogueira and Cho (2019); Luan et al. (2021). The input is composed of question $q \in \mathcal{Q}$ and passage $p \in \mathcal{C}_r$ concatenated with a special `SEP` token. The passage consists of a title and context that are prepended with special start tokens and concatenated together. We denote the contextual representation of input token w obtained by the cross-encoder as $\mathbf{En}(p, q)[w] \in \mathbb{R}^d$.

Now we can define the reranking function for passage rescoring as

$$\text{rerank}(q, p) = \mathbf{En}(p, q)[\texttt{CLS}]^\top \boldsymbol{w} \tag{4.1}$$

where $\boldsymbol{w} \in \mathbb{R}^d$ is a trainable vector and `CLS` is the special token added at the start of an input sequence. Finally, reranking probability that quantifies whether the passage p contains an answer to the question q is defined as

$$\text{P}_{rr}\left(p|q, \mathcal{C}_r\right) = \operatorname*{softmax}_{p' \in \mathcal{C}_r}\left(\text{rerank}\left(q, p'\right)\right)_p . \tag{4.2}$$

Training. The model input for each question is exactly one positive sample supplemented with hard negatives from the retriever. The ground truth passage annotated the same way

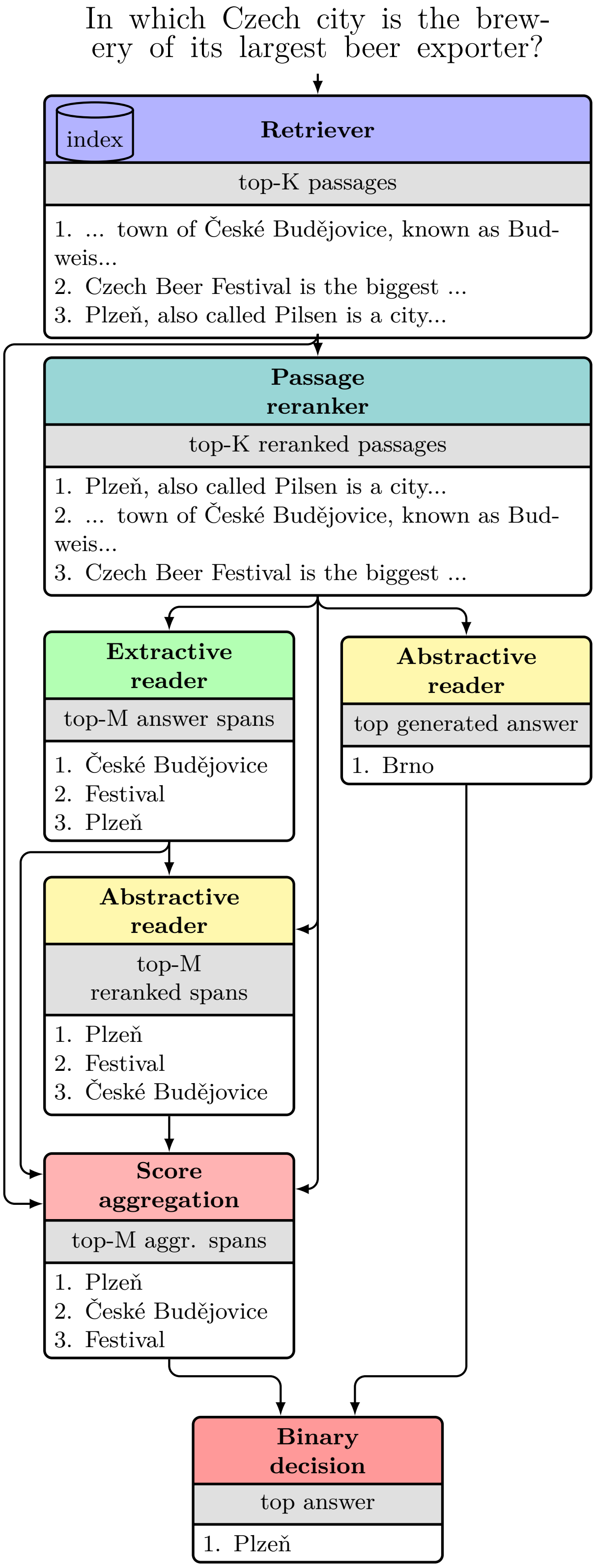


Figure 4.1: R2-D2 pipeline.

as in Karpukhin et al. (2020), is primarily used as a positive sample. If the ground truth is unknown, the positive sample is the top retrieved passage containing an answer. Hard negatives are uniformly sampled from the retriever's top-K results that do not contain the answer. The parameters are estimated via cross-entropy loss.

4.1.2 Extractive Reader

Extractive reader estimates the probability $\mathrm{P}_e(a_e|q, \mathcal{C}_{rr})$. It is the probability of a span a_e from top-V passage $p \in \mathcal{C}_{rr}$ being an answer to a question q. We decompose the $\mathrm{P}_e(a_e|q, \mathcal{C}_{rr})$ into four probabilities of:

- token s being starting token of an answer span,
- token e being ending token of an answer span,
- tokens s and e being boundary tokens of an answer span (Fajcik et al., 2021c),
- passage p containing an answer for the question q (inner reranker) as in Karpukhin et al. (2020).

The final joint probability used in test-time, the product of these probabilities is computed[1]. These probabilities are defined as:

$$\mathrm{P}_b(b_i|q, \mathcal{C}_{rr}) = \mathrm{softmax}(\boldsymbol{s_b})_i \ . \tag{4.3}$$

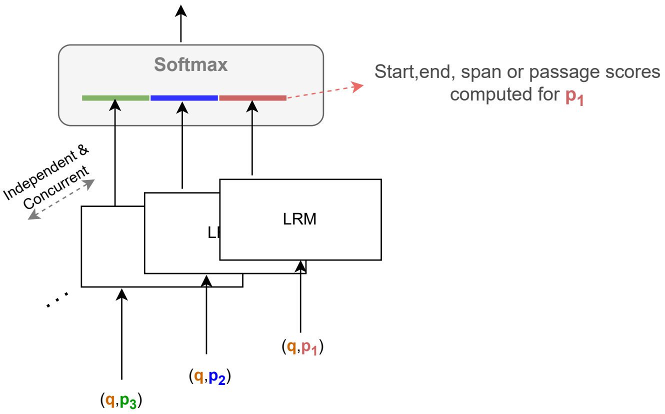


Figure 4.2: Illustration of globally normalized PMF computed by passage-localized LRM.

Here, b may stand for a *start*, *end*, *joint*, and a *passage*. The i is an index of a given element (i-th passage or i-th span or i-th start/end), and the $\boldsymbol{s_b}$ is a vector of scores for each element among all passages in $\mathcal{C}_{rr}$. The model estimates the scores $\boldsymbol{s_b}$ with only a single passage at its input (Clark and Gardner, 2018). However, *softmax normalization sum* sums through all the passages, computing the global probability space (Cheng et al., 2020) (illustrated in Figure 4.2). These scores are computed in scalar form as:

$$\begin{align} s^s_{start} &= \mathbf{En}(p,q)[s]^\top \boldsymbol{w}_{start} \tag{4.4} \\ s^e_{end} &= \mathbf{En}(p,q)[e]^\top \boldsymbol{w}_{end} \tag{4.5} \\ s^{s,e}_{joint} &= (\boldsymbol{W}_j \mathbf{En}(p,q)[s] + \boldsymbol{b}_j)^\top \mathbf{En}(p,q)[e] \tag{4.6} \\ s^p_{passage} &= \mathbf{En}(p,q)[\mathtt{CLS}]^\top \boldsymbol{w}_p \ . \tag{4.7} \end{align}$$

[1] Decoding from subsets of these probabilities showed no significant difference (Table 4.4).

Here, $\boldsymbol{w}_*, \boldsymbol{b}_j \in \mathbb{R}^h$, $\mathbf{En}(p,q)[\cdot] \in \mathbb{R}^h$, and $\boldsymbol{W}_j \in \mathbb{R}^{h \times h}$ are all trainable parameters. In particular, **En** encoder is based on a language representation model (LRM). We omit the spans in the title and question for answer span selection. Therefore the final answer can be selected only from the context. The following training loss $\mathcal{L}$ with independently marginalized components is used per sample:

$$\begin{aligned} \mathcal{L} = &- \log \sum_{s \in starts(C_{rr})} \mathrm{P}_{start}(s|q, \mathcal{C}_{rr}) - \log \sum_{e \in ends(C_{rr})} \mathrm{P}_{end}(e|q, \mathcal{C}_{rr}) \\ &- \log \sum_{j \in boundaries(C_{rr})} \mathrm{P}_{joint}(j|q, \mathcal{C}_{rr}) - \log \sum_{p \in C_{rr}} \mathrm{P}_{passage}(p|q, \mathcal{C}_{rr}) \,. \end{aligned} \tag{4.8}$$

Summation aggregates probabilities from target annotations (starts, ends, etc.) obtained in a distant supervision fashion. The answer span is annotated iff it contains the same string form as the answer string. The passage is annotated if it contains answer span (Clark and Gardner, 2018; Karpukhin et al., 2020). This is similar to surface form filtering, introduced in Section 3.4, but instead used in training.

4.1.3 Component Fusion

To produce the final answer, R2-D2 aggregates the log-probabilities of all system components via linear combinations tuned on validation data.

Firstly, the log-probabilities of all system components for top-M answer spans proposed by the extractive reader are aggregated. Formally, assume the $\mathcal{A}_q$ is the set of top-M answer spans from $\mathbf{P}_e(a|q, \mathcal{C}_{rr})$ for question q. The generative model performs the **answer reranking** re-evaluating the log-probability of the answer spans

$$\{\log \mathrm{P}_g(a|q, \mathcal{C}'_{rr}) : a \in \mathcal{A}_q\}. \tag{4.9}$$

Next a logistic regression loss (4.11) is minimized to perform **score aggregation**. It combines the scores across the R2-D2 components to maximize the correct answer span probability over dataset $\mathcal{D}$. This dataset is composed of the top-M outputs of the extractive reader with the correct answer.

$$x(a) = [\mathrm{P}_e(a) \;\, \mathrm{P}_g(a) \;\, \mathrm{P}_r(p_a) \;\, \mathrm{P}_{rr}(p_a)] \tag{4.10}$$

$$-\sum_{(\mathcal{A}_q, gt) \in \mathcal{D}} \log \operatorname*{softmax}_{a \in \mathcal{A}_q} \left(w^\top \log x(a) + b\right)_{gt} \tag{4.11}$$

Here p_a denotes the passage containing the answer span a, $\mathcal{A}_q$ is a set of proposed answer spans, gt is the correct answer span, probability dependencies, including q, are dropped for clarity and only the logistic regression parameters w, b are tuned in this step.

Finally, we hypothesized the correct answer span might not always be available in the passage set $\mathcal{C}_{rr}$, but the generative reader might be able to generate the answer from its parameters and the evidence given in passages. We introduce the binary classifier, which decides whether to select the best span answer from the answer aggregation step or a free-form answer generated via FiD. Given that $s_{agg}(q) = \max_{a \in \mathcal{A}_q} w^\top x(a) + b$ is the best span score and $s^*_g(q) = \log \mathrm{P}_g(a^*_q|q, \mathcal{C}'_{rr})$ is the log-probability of the answer a^*_q obtained via greedy decoding for question q, a classifier is trained via binary cross-entropy $\mathrm{BCE}(l, t)$ loss with log-odds ratio l and target t to do the **binary decision**

$$\sum_{(e,t) \in \mathcal{D}} \mathrm{BCE}(w^\top [s_{agg}(e); s^*_g(e)] + b, t). \tag{4.12}$$

Here, the training dataset $\mathcal{D}$ contains only cases where either the extractive or exclusively the abstractive prediction is correct (but not both).

4.2 Experimental Setup

Presented models are implemented in PyTorch (Paszke et al., 2019) using Transformers (Wolf et al., 2020). We use 12GB GPU to train the passage reranker, 48GB GPU for the generative reader, and 16x 32GB GPUs to train the extractive reader with $V = 128$ passages at its input. The inference runs on a 12GB GPU. In all experiments, we used Adam optimizer with a decoupled weight decay (Loshchilov and Hutter, 2017).

4.2.1 Metrics

Our models are evaluated by **EM** and **Accuracy@K**. Following the previous work, we use the script from Lee et al. (2022)[2] to measure the EM, and match the answer string exactly as Karpukhin et al. (2020)[3]. The metrics are defined in Section 2.2.

4.2.2 Datasets and Data Pre-processing

We evaluate our models on three datasets. Their statistics are available in Table 4.1. To train the reranker we filter out examples, which do not contain golden passage or exact match in top-K retrieved passages. To train the extractive reader, only examples with an exact match in a golden passage or top-1 retrieved passage are kept. Both filtering strategies are closely described in Appendix A.2.

NQ-Open (Lee et al., 2022) or NaturalQuestions-Open consists of real user queries obtained from the Google search engine. The maximum length of each answer is at most 5 tokens. The released version of the dataset is already tokenized. Each training and development sample contains 1 annotated answer, while the test dataset contains a 5-way answer annotation. Note that this dataset version is slightly different from the one in Subsection 3.2.3. Apart from the slightly different filtering procedure of the original NaturalQuestions dataset (Kwiatkowski et al., 2019), the golden documents—documents with correct answer—are mapped to the 12-20-2018 Wikipedia snapshot. This is useful for document retrieval supervision.

TQ-Open (Joshi et al., 2017) or TriviaQA-Open consists of question-answer pairs from 14 different trivia quiz websites. Each question contains a human-annotated answer and a set of answer aliases gathered from Wikipedia. We use the filtered version. This version is larger than the version from Subsection 3.2.3, as it contains all questions with documents grounded in web, heuristically mapped to Wikipedia articles (Karpukhin et al., 2020).

EfficientQA (Min et al., 2021) is a dataset collected the same way as NQ-Open through 2019. Since the Wikipedia snapshot used within this work is from the year 2018, this dataset likely contains more questions without evidence in our corpus than NQ-Open. We use the officially released dev set for testing[4] models trained and validated on NQ-Open data.

[2] Python code at https://cutt.ly/rkZNIer. Accessed 16.3.2023.

[3] Python code at https://cutt.ly/0luNhx4. Accessed 16.3.2023.

[4] The official test set was not released during our experiments.

Dataset	**Train**	**Dev**	**Test**
NQ-Open	79,168	8,757	3,610
- filtered reranker	71,238	-	-
- filtered extractive reader	61,755	-	-
- with golden passage	58,876	6,515	-
TQ-Open	78,785	8,837	11,313
- filtered reranker	69,346	-	-
- filtered extractive reader	62,332	-	-
- with golden passage	60,413	6,760	-
EfficientQA	-	-	1,800

Table 4.1: Dataset statistics. The filtered lines report how many examples are kept for training the reranker (filtered reranker) and extractive reader (filtered extractive reader). The lines with golden passage denote how many examples from the set contain golden passage annotation.

Additionally, we also report results according to train-test set question/answer overlaps for NQ-Open and TQ-Open discovered by Lewis et al. (2021a) in Appendix D.3.

4.2.3 Models and Pipeline

Retriever. We use BERT-based DPR from the official checkpoint[5]. Each passage is represented via 768-dimensional embedding. We use a multiset checkpoint for TQ-Open—a checkpoint from a model trained across multiple ODQA datasets including TQ-Open—as the checkpoint for TQ directly isn't officially released. We use the same knowledge corpus containing 21,015,320 passages based on the 12-20-2018 Wikipedia snapshot as Karpukhin et al. (2020). In inference time, the retriever passes $K = 200$ passages $\mathcal{C}_r$ to reranker.

Passage Reranker. We use the RoBERTa-base (Liu et al., 2019) and truncate the inputs to a maximum length of 256. The linear scheduler with 0.1 warmup proportion is used, the number of epochs is 5 and the model is validated every 40,000 optimization steps. We use learning rate $1.6 \cdot 10^{-4}$ and minibatch size 8. In training, the model reranks 24 passages per question with negatives uniformly sampled from top-400 passages retrieved by DPR. During the inference, top-K ($K = 200$) retriever passages are rescored and passed to readers.

Extractive Reader. The extractive reader encoder is based on pretrained ELECTRA-large. Its inputs are truncated if they are longer than the allowed maximum size (512 tokens). During the training phase, all spans from all $p \in \mathcal{C}_r$[6] that match[7] with at least one of the known answers are selected as target annotations. Therefore the annotations might appear in the wrong context. The extractive reader reads the top-$V = 128$ passages during the training phase and when it is used without the reranker. To demonstrate the effect of the reranker, the reader reads only the top-$V = 24$ passages if the reranker is used. We use a linear scheduler with a warmup for the first 20,000 steps for all models. The maximum number of training steps is 200,000. The model is validated every 20,000 steps, and the best checkpoint among validations is selected. The learning rate is $2 \cdot 10^{-5}$ and the optimization step was done after each training example.

[5] https://github.com/facebookresearch/DPR. Accessed 16.3.2023.

[6] Note that we train on data from retriever, not reranker.

[7] Matching strategies are described in Appendix A.2.

Generative Reader. We utilize T5-large (Raffel et al., 2020) and use a concatenation of question, passages, and their respective titles at the Fusion-in-Decoder's input the same way as Izacard and Grave (2021a). We truncate each passage to the length of 250 tokens for NQ-Open. For TQ-Open, as questions are significantly longer, we truncate whole inputs to the same size. Following FiD for TQ-Open, we use only human-generated answer. In training, the golden passage always comes first, if available, and we take the rest of passages as ranked by retriever up to V_2 passages. The model is trained using standard cross-entropy loss and considers left-to-right factorization of the sequence probability. Izacard and Grave (2021a) trained FiD with $V_2 = 100$ passages at its input. However, such an approach requires a tremendous amount of GPU memory and thus requires employing speed-memory trade-offs such as gradient checkpointing (Chen et al., 2016a). Unlike the original approach, we use only $V_2 = 25$ passages in our FiD. We note that in practice combining reranker with shorter-context FiD yields results similar to the original implementation with much lower memory consumption and better throughput in the R2-D2 setting[8]. We analyze the speed of our implementation in Appendix B.1. Other hyperparameters are similar to the original work—minibatch size 64, learning rate $5 \cdot 10^{-5}$ but no learning rate schedule. In test time, we decode an answer via greedy decoding.

[8] Due to the numerous decoder computations in answer re-ranking.

4.3 Results and Analysis

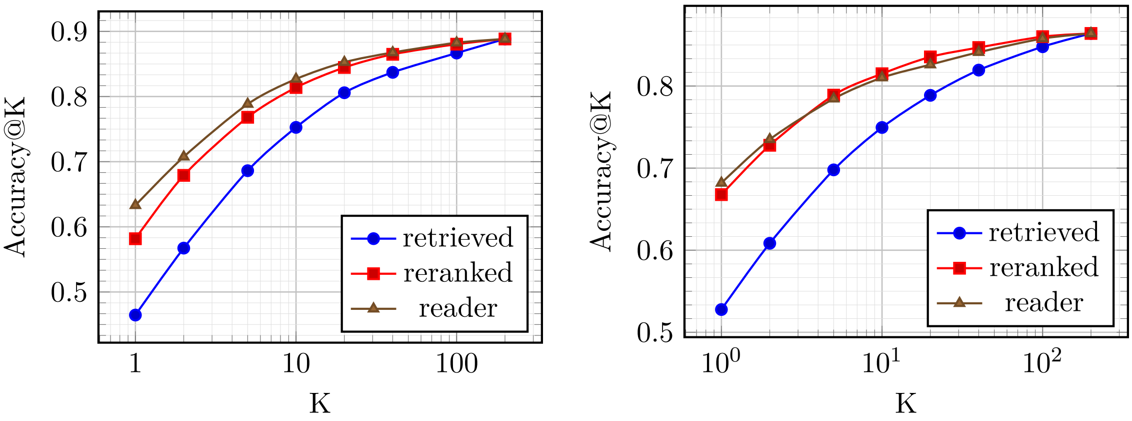
	Method	**NQ**	**TQ**	**#θ**
Extractive	BM25+BERT (Mao et al., 2021a)	37.7	60.1	110M
	Hard EM (Min et al., 2019a)	28.1	50.9	110M
	Path Retriever (Asai et al., 2020)	32.6	-	447M
	Graph Retriever (Min et al., 2019b)	34.5	56.0	110M
	ORQA (Lee et al., 2022)	33.3	45.0	220M
	REALM (Guu et al., 2020)	40.4	-	660M
	ProQA (Xiong et al., 2021b)	34.3	-	220M
	DPR (Karpukhin et al., 2020)	41.5	56.8	220M
	RDR (Yang and Seo, 2020)	42.1	57.0	110M
	GAR+DPR (Mao et al., 2021a)	43.8	-	626M
	ColBERT (Khattab et al., 2021b)	48.2	63.2$^-$	440M
	RIDER (GAR+DPR) (Mao et al., 2021b)	48.3	-	626M
	UnitedQA-E (Cheng et al., 2021b)	51.8	68.9	440M
	FiE (Kedia et al., 2022)	54.9	68.2	220M
	FiE-large (Kedia et al., 2022)	58.4	71.6	440M
Generative	BM25+SSG (Mao et al., 2021a)	35.3	58.6	406M
	T51.1+SSM (Roberts et al., 2020)	35.2	61.6	11B
	RAG (Lewis et al., 2020)	44.5	56.8	516M
	DPR+SSG (Min et al., 2020)	42.2	-	516M
	FiD-base (Izacard and Grave, 2021b)	48.2	65.0	333M
	FiD-large (Izacard and Grave, 2021b)	51.4	67.6	848M
	FiD-large++ (Izacard et al., 2020)	54.7	**73.3**	848M
	UnitedQA-G (Cheng et al., 2021b)	52.3	68.6	880M
	EMDR^2 (Singh et al., 2021)	52.5	71.4	333M
	YONO (Lee et al., 2022)	53.2	72.1	440M
	ReAtt (Jiang et al., 2022)	54.7	-	770M
	Atlas (Izacard et al., 2022)	60.4	79.8	11B
Ens.	UnitedQA (Ens. E+G+G) (Cheng et al., 2021b)	54.7	70.5	1.87B
Ours	R1-D1 (Generative)	49.9	65.4	848M
	R1-D1 (Extractive)	50.8	65.0	445M
	R2-D2 (21M)	**55.0**	69.9	1.29B
	R2-D2 (21M) w/ HN-DPR	**55.9**	-	1.29B

Table 4.2: Comparison with the state-of-the-art in EM. #θ denotes the estimated amount of model parameters (in millions (M)/billions (B)). Results with gray background were published after R2-D2. **Bold** font highlights the best result (not considering subsequent work). The symbol $^-$ reports the result only for a smaller system with $220M$ parameters.

The effectiveness of our approach is compared with the state-of-the-art in Table 4.2. Our system, composed of just the retriever and FiD reader R1-D1 (Generative), shows inferior performance compared to FiD-large. This is most likely caused by 4 times fewer passages at its input, as in Izacard and Grave (2021b). In contrast, our ELECTRA-based extractive reader R1-D1 (Extractive) shows large gains compared to extractive state-of-the-art, while having the same retriever as DPR. We hypothesize this may be caused by the ELECTRA pretraining method, which shows strong performance through a variety of tasks and we further show that it is also due to training and inference with large input size of 128 passages and better objective (discussed in Section 4.3.2).

The only system that matches the performance of our extractive reader is the concurrent work on UnitedQA-E (Cheng et al., 2021b), which uses advanced regularization and HardEM techniques. We note that these are orthogonal to our approach and could potentially lead to further improvements.

Furthermore, we find that our R2-D2 system with 21M passages corpus is competitive even with FiD++, which uses DPR retriever improved via knowledge distillation, and 26M



(a) NQ-Open.

(b) TQ-Open.

Figure 4.3: Accuracy@K on test-data.

passage corpus, which also includes lists. Additionally, we evaluate our model with a better retrieval model (HN-DPR) based on the DPR checkpoint where hard negatives are mined using the retrieval model itself[9]. Note that we do not compare EfficientQA with state-of-the-art, as the previous works didn't report results on the dev set we use for testing.

Finally, the results published after R2-D2 (Fajcik et al., 2021b) are presented in gray rows. We note that all these systems use better-performing retrieval system than DPR.

4.3.1 Reranker Performance

Next, we compare the performance of our retriever, reranker, and reader with Accuracy@K in Figure 4.3. The passage reranker improves the accuracy consistently and we observe the same trend on other datasets (Appendix D.1). We also include analysis, where we rerank each passage p_i according to its $s^i_{passage}$ score from the extractive reader. We observe results similar or even better to reranker for $K < 10$, indicating the extractive reader reranks well on its own. However, in subsequent experiments we do not replace the reranker with reader because: (i) passage reranker has fewer parameters, (ii) extractive reading can run in parallel with reranking and generative reading as the extractive reader is not benefiting from reranking, and (iii) passage reranking scores often improve results during score aggregation (see Section 4.3.4).

[9] https://cutt.ly/Ux5Yt4h. Accessed 16.3.2023.

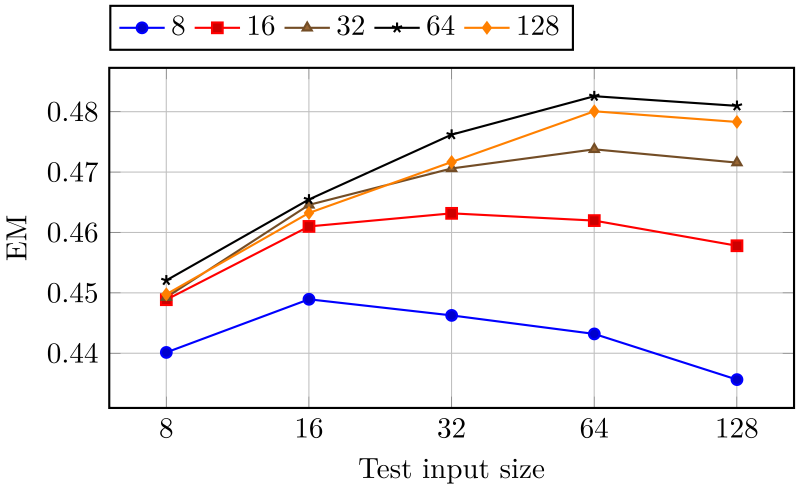


Figure 4.4: Influence of test input size on extractive reader's performance for various train input sizes (different curves) on NQ-Open test dataset.

independent marginalization	**span marginalization**	**joint objective**	**EM**
-	✓	-	45.42
-	✓	✓	45.41
✓	✓	-	45.71
✓	-	✓	**47.09**
✓	✓	✓	47.06

Table 4.3: Ablation of loss components on NQ-Open test dataset using ELECTRA-base model.

4.3.2 Extractive Reader Ablation

In order to investigate the influence of the number of input passages on the extractive reader's performance, we trained multiple ELECTRA-base models, each with a different input size. During test time, we evaluate each, varying the input size. Figure 4.4 shows that increasing train/test input size has a positive influence on the extractive reader's performance. However, input size 128 doesn't seem to increase the performance anymore, causing diminishing returns in test time.

Secondly, we analyze the ablation of loss components in Table 4.3. Following Cheng et al. (2020), we compare the *independent marginalization* over all answer span starts S and ends E

$$-\log \sum_{s \in S} \mathrm{P}_{start}(s) - \log \sum_{e \in E} \mathrm{P}_{end}(e) \tag{4.13}$$

with loss that does *span marginalization* through spans

$$-\log \sum_{c \in C_{rr}} \sum_{(s,e) \in \mathrm{answers}(c)} \mathrm{P}_{start}(s)\, \mathrm{P}_{end}(e). \tag{4.14}$$

Here answers(c) represents all answer spans in the passage c. As explicitly shown in Appendix C.1, independent marginalization sums through the combination of every start-end

Factorization	NQ-dev	NQ-test	EfficientQA
I	48.32	50.58	47.33
J	48.53	**51.25**	**47.83**
I+J	**48.57**	50.83	**47.83**
I+C	48.22	50.55	47.22
J+C	48.49	51.11	47.56
I+J+C	48.50	50.86	47.67

Table 4.4: The results of the pipeline with different types of extractive reader's distribution used for decoding. See text for details.

Readers	Fusion	NQ-Open			TQ-Open			EfficientQA		
		ret.	+rr	Δ	ret.	+rr	Δ	ret.	+rr	Δ
ext	-	50.78	50.72	-0.06	65.01	65.46	0.45	47.00	47.56	0.56
gen	-	49.92	50.69	0.77	65.38	69.14	3.76	44.83	47.33	2.50
ext+gen	naive	51.88	52.44	0.56	66.17	68.01	1.84	47.06	49.11	2.05
ext+gen	aggr	54.13	54.90	0.77	67.42	68.66	1.24	50.44	52.00	1.56
ext+gen	aggr+bd	54.07	**54.99**	0.92	67.37	**69.94**	2.57	49.72	**52.22**	2.50

Table 4.5: Ablation study. We report results for extractive (ext), generative (gen), and both readers (ext+gen) without (ret.) and with reranking (+rr). The Δ column shows the exact match difference caused by passage reranking.

factor $\mathrm{P}_{start}(s)\,\mathrm{P}_{end}(e)$, whereas the span marginalization from equation (4.14) sums only through the subset of these elements corresponding to specific answer span, i.e., the independent marginalization also contains inter-passage spans, which is an obviously wrong assumption. In spite of this, (i) Table 4.3 shows that independent marginalization achieves better performance, which is consistent with Cheng et al. (2020).

Next, (ii) Table 4.3 also shows that the joint probability component from equation (4.6) performs better when combined with the independent marginalization, but has no effect when combined with span marginalization. Additionally, (iii) Table 4.3 shows that the usage of the joint objective is neutral or beneficial to NQ-Open performance, which further agrees with observations in Table 3.3 from Chapter 3.

Finally, we analyze the usage of different distributions (independent (I), joint (J), and passage classifier (C)) for decoding in inference time (Table 4.4). We saw that using just joint (J), or a combination of independent & join probability space (I+J) works marginally better. Note that thorough this work, we reported all results in the I+J+C setting.

4.3.3 Pipeline Ablation

The ablations are listed in Table 4.5. We ablate results without using passage reranker, with separate readers and their combination, and with different stages of component fusion. Namely, performing a *naive* answer re-ranking by generative reader means the system chooses the most probable answer span among the top-M spans provided by the extractive reader according to generative reader log-probabilities as shown in equation (4.9). Analogously, the *aggr* fusion denotes that the system chooses the most probable answer span

	$\boldsymbol{P}_*$	$\emptyset$	$\{r\}$	$\{rr\}$	$\{r, rr\}$
NQ-Open	$\{e\}$	50.72	51.41	51.55	51.69
	$\{g\}$	52.44	52.88	53.35	53.19
	$\{e, g\}$	54.63	**55.10**	54.82	54.90
TQ-Open	$\{e\}$	65.54	65.64	65.60	65.61
	$\{g\}$	68.25	68.17	68.21	68.26
	$\{e, g\}$	68.45	68.57	**68.66**	**68.66**

Table 4.6: Results for different pipeline components used for score aggregation on NQ-Open a TQ-Open. See text for details.

	$\boldsymbol{P}_*$	$\emptyset$	$\{r\}$	$\{rr\}$	$\{r, rr\}$
NQ-Open	$\{e\}$	52.85	53.30	53.10	52.94
	$\{g\}$	52.44	52.77	53.21	53.07
	$\{e, g\}$	54.35	**55.10**	54.46	54.99
TQ-Open	$\{e\}$	69.34	69.28	69.23	69.26
	$\{g\}$	69.76	69.71	69.65	69.77
	$\{e, g\}$	69.80	69.89	69.88	**69.94**

Table 4.7: Results for binary decision on NQ-Open and TQ-Open for different aggregated pipeline components from Table 4.6.

according to aggregated scores, as in equation (4.11). Finally, the *aggr+bd* fusion denotes the binary decision, as shown in equation (4.12).

As expected, we observe that the reranker improves the results consistently for the generative model in all cases. The gains are especially large for TQ-Open (over 3.7 EM, underscored in Table 4.5). In fact, the results are comparable to Izacard and Grave (2021a), suggesting that using the FiD reader with a smaller context window and reranker is a reasonable alternative to memory inefficient FiD with large input size. Furthermore as expected, the extractive reader without reranker already has top-128 passages at the input, and the passage reranking doesn't provide any advantage (less than 1 EM).

Finally, the results on NQ-Open and EfficientQA suggest applying the binary decision does not bring large improvements over the score aggregation if any. However, notice that this is not the case for TQ-Open, where the generative reader performs significantly better compared to the extractive reader, suggesting both component fusions play important roles in the system.

4.3.4 Component Fusion

Furthermore, we analyze the performance of each component combination in the score aggregation and its impact on the component fusion via binary decision. Both fusions are tuned on validation data and reported on the test dataset of the NQ-Open and TQ-Open datasets. See Appendix D.2 for analysis on additional datasets. Table 4.6 shows all relevant combinations of ranker r, reranker rr, extractive reader e, and generative reader g probabilities used in score aggregation. In overall, we observe that combining retriever and

Readers	Ensemble	EM	Δ_{ext}	Δ_{gen}
ext	-	46.79	-	-
	2 models	48.30	1.51	-
	3 models	48.59	1.80	-
gen	-	45.00	-	-
	2 models	46.30	-	1.30
	3 models	46.59	-	1.59
ext+gen	aggr	**49.92**	**3.13**	**4.92**

Table 4.8: Comparison between ensembling via posterior averaging and score aggregation on NQ-Open.

reranker scores with the reader leads to better or equal performance. On NQ-Open, we observe minor improvements up to ~1 EM. However, there is no difference on TQ-Open.

The impact of adding a binary decision after the score aggregation is shown in Table 4.7. Interestingly, the binary decision component significantly improves the performance only without reranked answer scores ($\{e\}$ rows in both tables). However, fusing the generative and extractive reader via binary decision performs significantly worse on NQ-Open than fusing both readers together with score aggregation ($\{e\}$ row in Table 4.7 vs. $\{e, g\}$ row in Table 4.6). As already noted in ablations, we find this to be quite the opposite for TQ-Open. We hypothesize that the binary decision is strong in cases, where the generative reader performs better to extractive reader (the case of TQ-Open). We argue that if the generative reader is better, the abstractive answer should be used far more often, than when it's not. We support the hypothesis by analyzing the proportion of test samples, on which the binary decision component was activated (i.e. an abstractive prediction was selected). On NQ-Open, the component almost never activated (only on 3.5 % samples), but this proportion is much higher (26.6 %) on TQ-Open.

4.3.5 Comparison with Posterior Averaging

Finally, we compare our score aggregation with the ensemble computed via posterior probability averaging. In particular, we train three extractive and generative base-sized models initialized with a different random seed. We do not use a reranker in this experiment and set the train/test input size of the extractive reader to 32. We assess the predictions using the averaged posterior probabilities and compare their average performance with score aggregation in Table 4.8. Concretely, we compare with the average of all 2 model ensembles (2 models) and with an ensemble of all 3 checkpoints (3 models). We observe two to three times improvement of score aggregation over the posterior probability averaging on NQ-Open test dataset.

4.4 Related Work

Passage Reranking. Previous work in QA based on neural nets used Bi-LSTM encoders (Wang et al., 2018; Lee et al., 2018) that score each document independently. Over time, Bi-LSTM were replaced by BERT-like transformer encoders (Qiao et al., 2019; Wang et al., 2019). For document ranking, Nogueira et al. (2019) proposed a multi-stage architecture.

The first stage scores each document independently, and the second estimates the more relevant document from all document pairs. Another document ranking approach uses the seq2seq model to generate a true or false answer to the document's relevance to the query (Nogueira et al., 2020). Recent works have often focused on effective reranking. Xin et al. (2020) achieved inference speedup using early exiting, Jang and Kim (2020) proposed a smaller and faster model and Mao et al. (2021b) came up with a method that uses reader's predictions to rerank the passages.

Our reranker is most similar to Nogueira and Cho (2019); Luan et al. (2021), except that unlike in IR, we assume there is just one correct passage and thus train our model via categorical cross-entropy.

Subsequent work on reranking in QA inspired by this work includes YONO (Lee et al., 2022), where authors designed a single-architecture system that does retrieval, reranking, and answer extraction via FiD-like model jointly, unlike the proposed pipelined R2-D2 method. Furthermore, Atlas system (Izacard et al., 2022) studies passage reranking in the domain of retriever-reader domain adaptation. Specifically, top-k documents, retrieved from pretrained retriever, are reindexed and reranked using the new updated retriever. This facilitates training, by possibly providing informative negatives (Xiong et al., 2021a), and avoiding online index refreshing.

Multipassage Reading Comprehension Related work considers generative and extractive approaches toward modeling the reader. The generative reader generates an answer while conditioned on question alone (Roberts et al., 2020), or a question with relevant passages (Lewis et al., 2020; Min et al., 2020). Izacard and Grave (2021b) showed it suffices to concatenate the passages in the decoder of the seq2seq model, increasing the number of top passages the model can depend on dramatically. The extractive reader used in ODQA assumes that the answer is a continuous span string located in retrieved paragraphs (Chen et al., 2017a). Clark and Gardner (2018) proposed to aggregate the probabilities of distantly supervised answer matches via Maximum Marginal Likelihood MML. Lin et al. (2018) proposed to denoise distantly supervised answer string matches in MML via paragraph-ranker. Cheng et al. (2020) experimented with different assumptions for MML, showing improvement when marginalizing over components of span probability independently. Karpukhin et al. (2020) incorporated an independent passage classifier loss to his MML objective.

Our objective is similar to the last work, except that it uses a joint component and also optimizes MML over relevant passages' probabilities. Furthermore, Chapter 3 proposed to model joint span probability directly via compound objective, instead of modeling the probability of span's start and end independently.

Subsequent work from Kedia et al. (2022) further applied joint objective together with the extractive reader, which combines cross-passage information in the encoder, so-called Fusion-in-Encoder (FiE). Work demonstrates the strength of better retrieval from (Izacard et al., 2020) combined with cross-passage fusion and joint objective achieving state-of-the-art across systems with less than 1B parameters at the time of writing.

Component Fusion. Yang et al. (2019b) also combined BM25 ranker and reader scores via linear combination. Our work can be seen as an extension of this idea to combine the scores of all pipeline components. Iyer et al. (2021) proposed a system that directly learns to rerank question-passage-answer triplets proposed via extractive model. However, reranking answers from their large extractive model via large reranker leads to ~1 EM improvement absolute, whereas R2-D2s score aggregation improves 4 to 5 EM w.r.t. the extractive reader. Concurrently with our work, Cheng et al. (2021b) proposed a hard voting ensembling scheme to combine the reader predictions. Firstly, each model from an ensemble produces its best prediction, then the votes for identical predictions are combined, omitting the scores produced by the individual models. The authors obtained the best results using two FiD readers and a single extractive reader, leading to 1.6 and 2.4 EM improvement on TQ-Open and NQ-Open, compared to their best single extractive or generative model.

Subsequent work includes EMDR^2 (Singh et al., 2021), an approach where the retriever and reader are trained jointly, and ReAtt (Jiang et al., 2022), a system where retriever and reader are components of single T5-based architecture trained jointly. Lastly, Izacard et al. (2022) introduced Atlas, a model pretrained for retrieval-augmented tasks. Authors compare different joint retriever-reader training approaches, including EMDR^2, for few-shot learning, and full fine-tuning, across knowledge-intensive tasks. Their fully fine-tuned 11B system achieves state-of-the-art at the time of writing.

4.5 Chapter Summary

This chapter proposed R2-D2, a novel state-of-the-art pipeline for open-domain QA based on 4 components: retriever, reranker, generative reader, and extractive reader. It was shown that employing a reranker is a reasonable alternative to using large passage counts at the input of both the extractive and the generative reader. Our results on NQ-Open and EfficientQA showed that the extractive and the generative reader could perform equally in ODQA, although the generative reader is twice the size of the extractive reader. However, we observe the extractive reader underperforms on TQ-Open. We hypothesize, that the cause is (1) the complexity of trivia questions with many entities, which often require combining evidence from multiple passages—these are impossible to answer for the extractive reader by design—and (2) the expensive hyperparameter search, as we used NQ-Open hyperparameters also for TQ-Open. Contrary to belief based on the results on different datasets (Yang et al., 2019b; Wang et al., 2019; Izacard and Grave, 2021b), we found the extractive reader can also benefit from larger input sizes, both in training and test time. Finally, we proposed a component fusion, which allows the merging of the complementary behavior of generative and extractive approaches along with the ranking components, and found it improves the results significantly. Due to its heterogeneous and modular nature, our pipeline forms an ideal base for past and future research of component integration in modern ODQA.

Chapter 5

Inherent Redundancy in Open-Domain Index

Recent work on retrieval-based open-domain question answering (ODQA) often comes with a system that seeks an answer for the question in the massive index of external passages that scales in the order of millions or even billions (Seo et al., 2019) of natural language passages. Recent advances in neural passage retrieval (Karpukhin et al., 2020; Izacard and Grave, 2021a; Khattab et al., 2021b; Luan et al., 2021, *inter alia*) greatly improved the performance of ODQA. It seems that the retriever has to efficiently represent each of the passages, in order to find the documents containing test answers. However, it is not the case. We find that not all passages are equally important and the current approaches can exploit the passage prior in existing datasets.

In this work, we demonstrate that there often exists a strong prior over golden passages. Our simple approach is based on the apriori passage relevance classifier, which decides for a given passage and its title whether the passage is irrelevant or not. We exploit this prior on popular ODQA datasets—NQ-Open, EfficientQA, and TQ-Open. We tune our simple approach for pruning away the contents of a massive index so that only 8 % of original index content is retained and while losing only up to 3 % exact match. In summary, the contributions of this chapter are 3-fold:

1. Our experiments show there is a concerningly strong prior over passages on studied datasets. Such prior could e.g., ease off the problem of retrieval by not representing the irrelevant passages at all, *assuming they are redundant.* This is probably why choosing in-batch negatives—samples positive for other examples in minibatch—instead of random negatives during training of a neural passage retriever leads to converging towards competitive results (Xiong et al., 2021b; Karpukhin et al., 2020).

2. We empirically verify that the same prior is captured in the recent DPR passage embeddings (Karpukhin et al., 2020), indicating the neural retrieval models could prune the passage space the very same way as our classifier does. This agrees with the observations of the recent work, that reports on DPR's inability to generalize across domains.

3. Exploiting this prior, we demonstrate is possible to build a memory efficient ODQA system that can fit into a 6GiB docker image, losing only a negligible amount of performance. We submit this trivially compressed system to the NeurIPS@EfficientQA challenge (Min et al., 2021), achieving 3rd place in the competition.

5.1 Pruning Approach

To reduce the size of the index, we resort to an apriori relevance classifier that selects the relevant content without seeing a question. Note this is in contrast with the retriever, which considers a question when assigning relevance. Consider the Wikipedia corpus split into 100-word passages. The work of Karpukhin et al. (2020) indicates that the distribution of golden passages—the passages containing an answer from the dataset—differs from the distribution of all passages. This is implicated by the fact that golden passages perform as better negative samples than just any randomly sampled passages when training the retriever. Therefore, given a passage p_i from Wikipedia, we propose an apriori relevance classifier (we call *pruner*) into relevance class r that models the Bernoulli distribution $\mathbf{P}(r|p_i)$. The input of this classifier is the concatenation of a Wikipedia passage (sometimes referred to as context) and its article's title separated with the special `SEP` token and different segment embedding. The classifier is trained via binary cross-entropy on the set of golden passages and non-golden passages extracted from Wikipedia. In test-time, we collect the probabilities $\mathrm{P}(r|p_i)$ for each passage p_i in the corpus and use them for ranking. We keep only passages p_i that satisfy the threshold constraint $\mathrm{P}(r|p_i) > \tau; \tau \in (0, 1)$. Furthermore, we empirically verify in Section 5.3.2 that the passage embeddings from Karpukhin et al. (2020) contain strong features that capture the very same apriori relevance as this classifier does.

5.2 Experimental Setup

LRM, QA baseline, Datasets To estimate the impact of corpus pruning on various ODQA components, we evaluate the approach on the heterogeneous modular ODQA system R2-D2 (Rank twice, reaD twice) introduced in Chapter 4. We implement our pruner in PyTorch (Paszke et al., 2019) using Transformers (Wolf et al., 2020). Pruner training along with R2-D2 inference runs on a 12GB GPU. We evaluate the pipeline by two standard metrics **EM** and **Accuracy@K** (see Section 2.2 for detail). We include three same datasets as used for R2-D2 validation (Subsection 4.2.2), namely NQ-Open, TQ-Open and EfficientQA (Min et al., 2021) dataset, which was also the official development dataset on EfficientQA challenge. We fine-tune the base version of ELECTRA (Clark et al., 2020) with a 2-layer feed-forward network on top of it as a binary classifier (the same way as authors do it in classification tasks, with GeLU (Hendrycks and Gimpel, 2016) and dropout), using Adam optimizer with a decoupled weight decay (Loshchilov and Hutter, 2017).

Pruning Datasets To train the pruner, we create a training set with 2 negative passages per positive passage from the dataset's training examples[1] with golden passage annotation. The negative passages are uniformly sampled from all non-golden Wikipedia passages. To create development and test sets for pruner, we split the subset of the dataset's development set with examples containing golden passage annotation, using a $1 : 2$ ratio. We sample only one negative passage per positive sample for development and test sets so that datasets are balanced. We further refer to these datasets as *Golden*. The procedure is the same for both datasets. The statistics of these datasets are shown in Table 5.1.

Pruner Training The system is trained via weighted cross-entropy in 2 epochs using minibatch size 12 and learning rate $3 \cdot 10^{-5}$ linearly decreasing to 0. The loss per positive

[1] In the preliminary experiment, we tried different ratios to use as a large dataset as possible, and found $1 : 2$ split working better than $1 : n$ for $n > 2$ ratios.

Dataset	**Train**		**Dev**		**Test**
	full	w/ann	full	w/ann	full
NQ-Open	79,168	58,876	8,757	6,515	3,610
TQ-Open	78,785	60,413	8,837	6,760	11,313
EfficientQA	-	-	-	-	1,800
NQ-Golden	-	176,628	-	4,332	8,698
TQ-Golden	-	181,239	-	4,516	9,004

Table 5.1: Dataset statistics. The columns *w/ann* denote how many examples from the *full* set contain golden passage annotation. The *Golden* sets are datasets used to estimate the pruner.

Dataset	**Index**	**Reader**		
		ext	gen	ext+gen
NQ-Open	1.7M	48.92	48.31	52.58
	21M	50.72	50.69	54.99
	Δ	-1.80	-2.38	-2.41
TQ-Open	1.7M	63.51	67.18	67.96
	21M	65.46	69.14	69.94
	Δ	-1.95	-1.96	-1.98
EfficientQA	1.7M	45.06	45.22	49.22
	21M	47.56	47.33	52.22
	Δ	-2.50	-2.11	**-3.00**

Table 5.2: Results with pruned index. The Δ row shows the exact match difference caused by pruning.

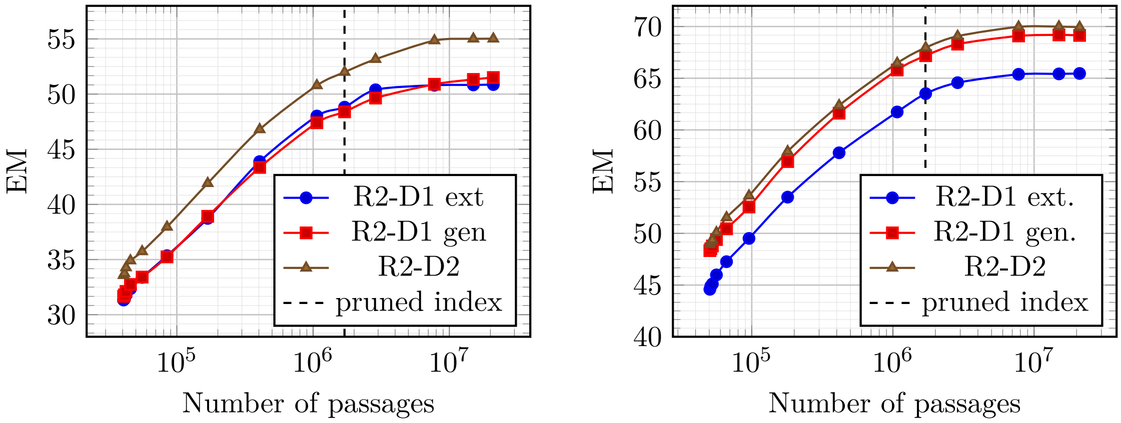
sample has twice the weight of the loss per negative sample. To analyze a particular operating point, the τ threshold is tuned so that we pool the top 1.7M passages. Such an amount conveniently fitted the 6GiB constraint of the EfficientQA challenge. We combine these relevant passages with missing golden passages from the training data, obtaining 1,702,133 passages in total for NQ-Open and EfficientQA and 1,706,676 passages in total for TQ-Open.

5.3 Results and Analysis

5.3.1 Performance Validation

Reading Performance on 1.7M Passages. To understand the effects of pruning, our results show the impact of pruning on the passage ranking and its impact on the system's ability to produce the final answer. The results of the latter with an index of 1.7M passages are shown in Table 5.2. We observe the largest decrease in performance on EfficientQA, which is probably due to the dataset's covariate shift caused by its timeframe. We also tried to train a multiset pruner on the unified NQ-Golden and TQ-Golden dataset. We found only a minor decrease in performance (-1.3 EM on NQ and -0.3 EM on TQ test sets) when pruning 1.7M passages.

In addition to evaluation on the TQ-Open and NQ-Open showed in Table 5.2, we also report results on subsets of these datasets in Table 5.3, as split in Lewis et al. (2021a). We



(a) NQ-Open.

(b) TQ-Open.

Figure 5.1: Index size analysis on test sets. *Pruned index* stands for 1.7M passages, a point we chose to analyze in this thesis. Note the R2-D1 system uses a retriever and reranker (R2) but only one reader (D1), extractive (ext.), or generative (gen.).

compare R2-D1 (with extractive and generative reader, marked as gen and ext respectively) and R2-D2 (ext+gen) to the official result on FiD (Izacard and Grave, 2021b). We find there is no difference on *question overlap subset* between the full and pruned version on both datasets, which is expected as every training question has its golden document in the pruned index. Interestingly, observing *answer overlap only subset* on NQ-Open, we see a performance drop for the gen model but no difference in the R2-D2 setting. Similarly, on TriviaQA, the bad performance of the extractive reader is partially compensated for by the gen model in *answer overlap only* setting. This demonstrates the robustness of the R2-D2's component fusion.

Overall Reader Performance. Furthermore, we analyze the exact match as a function of index size for NQ in Figure 5.1. We start by including all the golden passages from the training data (40,670 for NQ-Open, 50,502 for TQ-Open). Next, we consider adding the passages according to the ranking produced by the pruner. We find the difference between using the full index and only golden passages is about 21 EM (21.27 for NQ-Open, 21.01 for TQ-Open). This means the R2-D2 system that uses only the golden passages from training data achieves a performance comparable to ORQA (Lee et al., 2022), the first system presented on NQ-Open. On the other hand, we do not observe any significant difference in EM up from the 7.7M passages on both datasets, *indicating almost two-thirds from 21M total passages are completely redundant* for R2-D2.

Ranking Performance. Next, we analyze the performance of R2-D2's retrieval on the pruned and full index in Figure 5.2. We consider performance with and without the reranker module in the pipeline. The reranker improves the accuracy consistently for both, pruned and full versions of the pipeline. Remarkably, the pruned version of R2-D2 with reranker (reranked-pruned) performs better than the full version only with retriever (retrieved-full) up to $K = 26$ paragraphs (dashed vertical line). We observe a similar trend on other datasets, e.g., for the TQ-Open test the reranked-pruned improves over retrieved-full up to $K = 116$ paragraphs (additional analyses on dev sets and EfficientQA set are in Appendix D.4).

Model		NQ-Open				TQ-Open			
		Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap
-	FiD	51.40	71.30	48.30	34.50	67.60	87.50	66.90	42.80
21M	ext+gen	54.99	75.00	48.89	39.91	69.94	90.18	71.53	44.83
	gen	50.69	70.06	46.98	34.04	69.14	87.50	70.32	44.83
	ext	50.72	72.53	45.40	35.11	65.46	83.63	66.42	39.46
1.7M	ext+gen	52.58	75.00	48.89	34.88	67.96	90.18	68.61	41.92
	gen	48.31	69.75	43.81	30.24	67.18	88.39	68.86	42.08
	ext	48.92	72.84	45.71	31.23	63.51	85.12	62.77	36.93
Δ	ext+gen	-2.41	0.00	0.00	-5.03	-1.98	0.00	-2.92	-2.91
	gen	-2.38	-0.31	-3.17	-3.80	-1.96	0.89	-1.46	-2.75
	ext	-1.80	0.31	0.31	-3.88	-1.95	1.49	-3.65	-2.53

Table 5.3: Results on the overlapping and non-overlapping parts of test sets for NQ and TQ. *Total* column corresponds to the overall result on the whole dataset, as reported before, *Question Overlap* corresponds to samples with train-test question overlap and answer overlap, *Answer Overlap Only* corresponds to samples with answer overlap, but no question overlap and *No Overlap* corresponds to samples with no overlap between train and test sets.

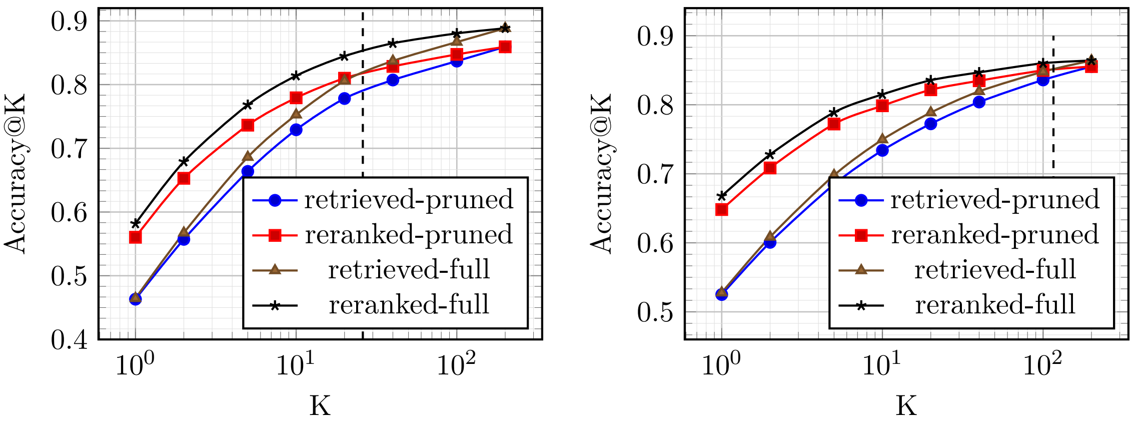
5.3.2 Pruner

Our pruning approach achieved 90.63% accuracy on NQ-Golden test dataset and 86.94% accuracy on TQ-Golden test dataset. This indicates that there exists a strong prior over the passages of Wikipedia in these open-domain QA datasets. Interestingly, the pruner still missed 5.2 % of golden passages from the NQ-Golden training passages and 13.2 % of the TQ-Golden golden passages in the 1.7M index.

What Pruner Learned To offer a glimpse of what our pruner system learned, we turn towards two experiments on NQ-Golden. Firstly, we investigate to what extent the pruner's score assigned to a passage correlates with the number of pageviews of the passage's article. We *hypothesize the pageviews roughly correspond to the topic's popularity*, and thus the stronger the pruner correlates with it, the better it generalizes towards estimating what topics were popular at the time of dataset collection.

Concretely, we collect Wikipedia article's pageviews[2] between 2018-08-01 and 2018-12-31. First, considering pageviews→pruner direction, we tune the pageviews-based threshold classifier on the NQ-Golden training data, directly optimizing for accuracy. The simple approach leads to 76 % accuracy on the dev data, *indicating the pageviews and relevance annotation are correlated.* Correlation is transitive, and thus since annotations and the pruner's decision are also correlated (accuracy ~90 %), a correlation between the pruner and pageviews is likely present. To investigate this directly, we study pageviews←pruner direction. We train a linear regression, which estimates the pageviews from our pruner's hidden state embedding from the `[CLS]` token-level output. We compute the rank correlation between the estimated pageviews and the true pageviews on dev data, finding *a moderate correlation* (Kendall's $\tau = 0.53$).

[2] Available at https://dumps.wikimedia.org/other/pageviews/. Accessed 16.3.2023.



(a) NQ-Open.

(b) TQ-Open.

Figure 5.2: Analysis of Accuracy@K on test sets.

In the second experiment, we compare to what extent might what pruner is learning be *similar to memorizing the training samples and using a K nearest neighbor (K-NN) classifier*. To represent the passage along with its title by the single vector, we average the pretrained ELECTRA's input embeddings (already combined with segment and positional embeddings) created the same way as our pruner's input embeddings are (see Section 5.1). We project the training and validation sample representations into 2-dimensional space using T-SNE (Van der Maaten and Hinton, 2008) and plot them in Figure 5.3. We observe the class of each sample from the development data indeed seems to be similar to the classes of its K-NN from the training data.

To quantify this observation, we match each passage representation from the dev data with the K=5 nearest passages from the training data, according to its cosine similarity. We find predicting the same label as is assigned to the majority of K nearest neighbors from the training data already yields 81 % accuracy on the dev data of NQ-Golden.

Connection between Retrieval Embeddings and Pruner. Next, we analyze whether the DPR embeddings on NQ capture the same phenomena as our pruner does. Starting with basic statistics, we compute the mean and the variance vectors of d-dimensional embeddings representing pruned (1.7M) set of documents P and those which represent the rest of the knowledge base N. We find that computing the L2 distance between both, means and variances, *yields order of magnitude different results than the distance between randomly permuted splits* of $P \cup N$ with the same size. Next, we found *a significant difference between the average length of embedding vectors from P and N*. Conclusively, we train a logistic regression classifier on a balanced dataset constructed from P and a subset of N, which predicts if the passage belongs to the pruned set P or not based on its embedding. We found the classifier achieves 84.1 % accuracy on the dev set, *confirming our hypothesis that the apriori relevance is indeed (in some way) captured* in DPR's embeddings.

5.4 Exploiting the Passage Prior for Memory-Efficient QA

Compressing the R2-D2 System into 6GiB We save models and index in half-precision without significant loss of performance and use off-the-shelf `ZIP`[3] compression to reduce the

[3] https://launchpad.net/ubuntu/+source/zip. Accessed 16.3.2023.

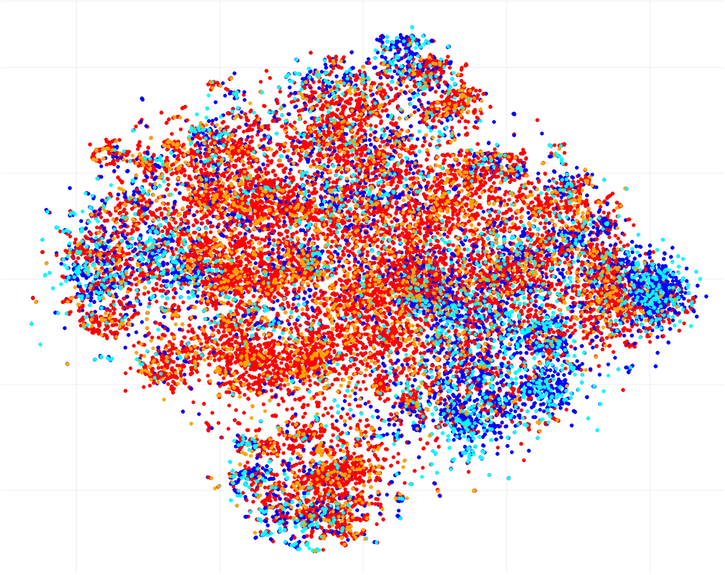


Figure 5.3: Representations of the relevant and irrelevant passages from the NQ-Golden training and development data as obtained via T-SNE. Blue and cyan points correspond to relevant passages in training and development data. Red and orange points correspond to irrelevant passages in training and development data.

size of the models and the corpus. To fit the 6GiB limit, and thus be comparable with recent EfficientQA competition[4] format, we use 100MB CentOS8 docker image[5] and compress python's `site-packages` to reduce the size of PyTorch.

We compare the memory footprint of the compressed system's docker image (pruned system) with the image of the full system in Figure 5.4. The total uncompressed size of an image is 81.01GiB while the size of the pruned image is 5.96GiB (92.6 % less). Here, *codes* are python code and configurations, *corpus* is a sqlite3 database of passages, and *binaries* are the OS with python libraries. We save *dense index* as a raw `h5` matrix. Interestingly, the dense corpus has similar space requirements as the *parameters* of all 4 components of the R2-D2 pipeline used in this work.

[4] https://efficientqa.github.io/. Accessed 15.3.2023.

[5] Docker image id: `nvidia/cuda:10.2-base-centos8`

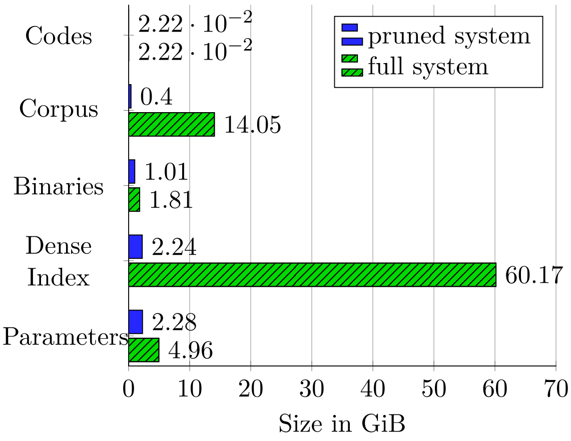


Figure 5.4: Component sizes inside the docker image.

5.4.1 EfficientQA Challenge

EfficientQA challenge (Min et al., 2021) focused on the compression of disk requirements of ODQA systems. In practice, the constraint was measured on the docker image size, so the system could uncompress itself before execution. In this section, we will focus on a single track of this competition, *the best-performing system with a total size under 6 GiB*.

Performance Validation The system performance was measured via two metrics: automatic (EM) and human evaluation. Each question in EfficientQA test set[6] has up to 5 annotated answers. This list is non-exhaustive, thus it can happen that the correctly predicted answer is not in-between the annotated answers considered by automatic metric. Each prediction was rated by three separate human annotators. Each of these analyzed question following a 3-step process:

1. the annotator first works on understanding the meaning and intent of the question (with a web search if necessary),
2. the annotator then determines whether the question is ambiguous, i.e., whether the question can lead to multiple different answers depending on factors such as: when the query was asked; where the query was asked; some unspoken intent of the questioner; or the opinion of the person giving the answer,
3. the annotator determines whether each answer is "*definitely correct*" (correct given a usual interpretation of the question), "*possibly correct*" (could be correct, given some interpretation of the question), or "*definitely incorrect*".

Submitted System The system submitted in EfficientQA slightly differs from the published (and improved) R2-D2 system (Chapter 4). The reranker was based on Longformer, which we later found had an inferior performance on EfficientQA (Fajcik et al., 2021b) compared to the more computationally efficient RoBERTa-based reranker. Next, our pruner was based on a less-performing RoBERTa-based classifier. The component fusion did not include the combination of the retriever's and reranker's probabilities. Lastly, our hyperparameters and preprocessing were not identical to the published version.

Competition Results The results are available in Table 5.4. The percentages in parentheses show relative improvement of human evaluation over automatic evaluation. Our system achieved 3rd place in both, automated metrics and human evaluation. The reason for the judgment match between the two metrics might lie in the systems with similar retriever-reader architecture of these systems. This is reflected in relatively large inter-agreement between the systems (more than 50 % with other systems in track) compared to different competition tracks. For instance, in a track under 500MiB, a system UCLNLP-FB (Lewis et al., 2021b) generated a colossal amount of question-answer pairs. In test time, this system retrieved the question-answer pair most similar to the question at the input and responded with the associated answer. While such a method performed very well in EM, the system performed worse than the competing retriever-reader system NAVER RDR (Yang and Seo, 2021) in human evaluation.

Component Size Comparison The analysis of component sizes of top-3 systems in the track is available in Figure 5.5. The analyzed components are: *passage corpus*—a collection of texts to search from, *dense index*—a set of embeddings for every text from text collection, *sparse index*—a set of sparse embeddings for every text from text collection (only FB system

[6] This is the hidden competition test set, not used for other experiments from Chapter 4, or Chapter 5.

Model	EM	Human Evaluation	
		Definitely	Possibly
FB system (Izacard et al., 2020)	53.33	65.18 (+22.2%)	76.09 (+42.7%)
Ousia-Tohoku Soseki[7]	50.17	62.01 (+23.6%)	73.83 (+47.2%)
BUT R2-D2 (ours)	47.28	58.96 (+24.7%)	70.33 (+49.2%)

Table 5.4: EfficientQA competition results in track under 6 GiB.

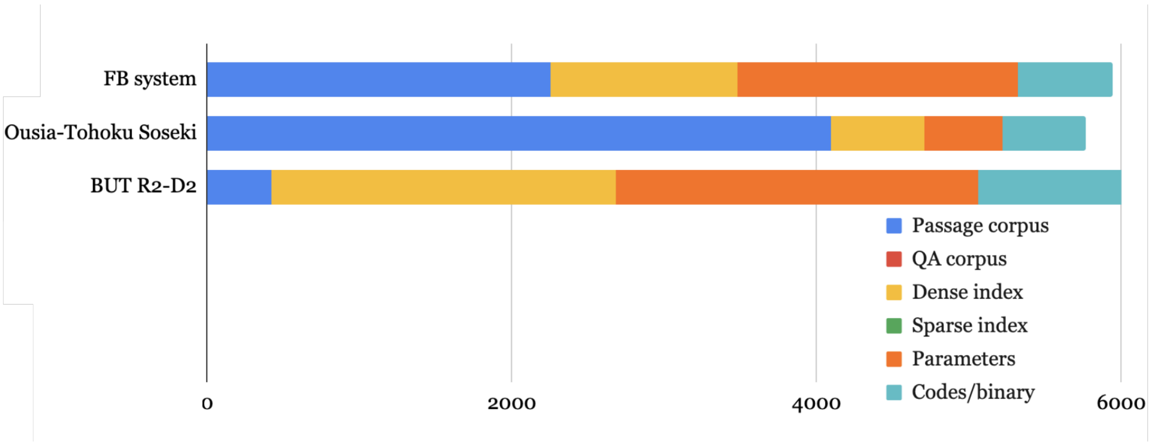


Figure 5.5: Comparison of different system component sizes in GiB in top-3 EfficientQA submissions. *QA corpus* component was not present in this track. Figure edited from Min et al. (2021).

used this), *parameters*—the size of all parameters from neural models and *codes/binary* corresponds to source code, and binaries such as OS, Python, and PyTorch.

5.4.2 Trivia Game: Humans vs AI

Finally, our system was used among the QA systems competing with Trivia experts in the QA competition. According to Turing (1950), the ultimate goal of artificial intelligence is to create machines that answer questions as well as humans (known as the Turing test). In practice, past work has shown that it depends on which humans. Existing comparisons of human question answering ability often use unskilled humans (Rajpurkar et al., 2016), leading to claims of computers "putting millions of jobs at risk" (Cuthbertson, 2018). In order to select a skilled human team, 5 teams, each up to 8 players, which signed up for the competition through social media, were given an hour to compete with the baseline systems. The best team was picked up for the final game. More details on the preliminary round can be found in the original publication (Min et al., 2021).

Competition Rules Each round with a single question could contain up to three phases[8]. At the start of every round, humans and computers switched. If ones did not succeed in the current phase, the others had the opportunity. Most of the answers were answered by winning systems in the track, but rarely moderator selected "the guest system", including R2-D2.

[8] Blogpost and video recording of the game is available at https://go.umd.edu/2020eqa. Accessed 16.3.2023.

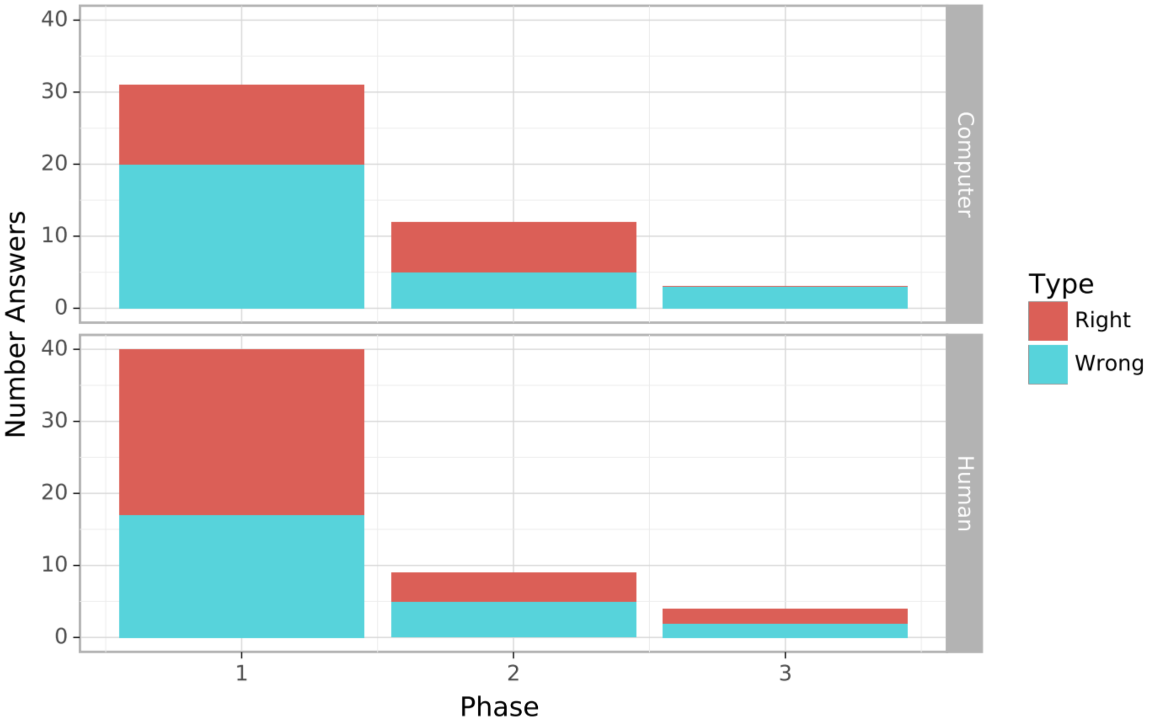


Figure 5.6: Results from Humans vs AI competition. Figure is taken from Min et al. (2021).

1. In the first phase, systems under 500 MiB were competing with instant responses from the players. One of the players had to immediately press the buzz-in key in order to sign up for the answer, and could not talk to each other. If no one succeeded in the first round, the second phase continued. A correct answer in this round awarded the team with 3 points.
2. In the second phase, humans had 30 seconds to discuss the answer, and now competed with the systems under 6 GiB. If no one succeeded in the second phase, the last phase continued. A correct answer in this round awarded the team with 2 points.
3. In the last phase, humans were given access to the search result snippets from a search engine using the question as a query and again had 30 seconds for discussion. In this phase, the computer system were not limited in memory. A correct answer in this round awarded the team with 1 point.

Competition Results Numbers of correctly and incorrectly answered questions from each phase from computers and humans are shown in Figure 5.6. While humans performed better in overall—answered more questions correctly in total, and obtained more points—computers managed to take a lead in phase 2. These were scenarios, where *the answer was not guessed by humans instantly, and systems under 500MiB provided incorrect answer.*

5.5 Related Work

Lewis et al. (2021a) showed 60-70 % of answers and 30 % of questions from test-set (including NQ-Open and TQ-Open) have a near-duplicate paraphrase in their respective training sets in studied ODQA datasets. Min et al. (2021) presented a simple baseline that includes an index containing 1.65M passages. These include all passages from the Wikipedia articles assigned to the top-5 positive passages from DPR training data for NQ (Karpukhin et al., 2020) (therefore golden passage, and highest-ranking BM25 passages to each question). However, this pruning approach led to a -6.7 EM decrease, while ours led to at most -3 EM with a similar number of passages. Izacard et al. (2020) employed three strategies to reduce the size of the index, among which was a pruner that filters articles based on their title and list of Wikipedia categories. Nonetheless, their pruning approach leads to ~4 EM loss in performance with FiD-large while still retaining 10M passages. Concurrently with our work Yang and Seo (2021) proposed a RoBERTa-based pruner almost identical to ours as a part of their memory pruning framework. Their DPR-based extractive system with 1,2M passages loses about 5 % exact match on test sets of NQ and TQ. Interestingly, our nearest R2-D2 result with exactly 1M passages leads to slightly lower performance loss (4.23 and 3.48 EM on NQ-Open and TQ-Open, respectively). This could be attributed to (a) different pruner model and dataset construction and (b) the robustness of R2-D2.

5.6 Chapter Summary

This chapter exposed a strong prior that is present in currently popular ODQA datasets NQ-Open and TQ-Open. Exploiting this prior, we showed it's possible to drastically reduce the colossal number of passages commonly used within the knowledge-base of retrieval-based open-domain QA systems (by 92 %) with only minor loss of performance (-3 EM). It was shown the proposed approach can be exploited in reducing the memory footprint of the QA system, and that such a system can beat skilled humans in a certain type of questions. Our R2-D2 system, with just 8 % of Wikipedia knowledge, is also available as an online demo[9].

As a final remark, we note the pruned index size opens up new possibilities, as it means the knowledge-base embedding matrix now fits to most modern GPUs. However, the possibility of such a drastic reduction of knowledge-base is concerning. In practice, removing the system's knowledge is often an undesirable property. *How well do the methods evaluated on these datasets perform in different prior scenarios? Will future search engines ignore unpopular topics altogether? Does this strong prior over passages suggest that the open-domain answering datasets aren't really "open" enough?*. We would like to address these questions in future research.

[9] Available at https://r2d2.fit.vutbr.cz/. Accessed 16.3.2023. The integration into the demo system is not the original work of the author of this thesis.

Part II

Fact-Checking

Chapter 6

Extracting Rumour Stances for Fact Verification

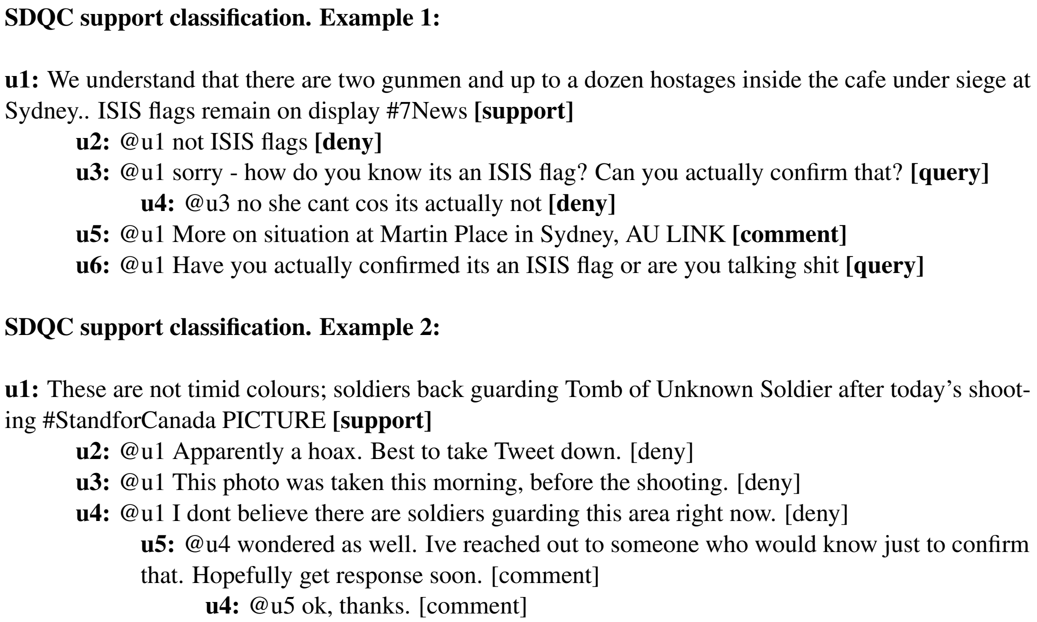
In this chapter, our work on challenge for determining rumor veracity from rumor stances (RumourEval 2019) (Gorrell et al., 2019) is presented. Rumour verification is a particular case of fact-checking. Rumours are "*circulating stories of questionable veracity, which are apparently credible but hard to verify, and produce sufficient skepticism and/or anxiety so as to motivate finding out the actual truth*" (Zubiaga et al., 2016). The challenge focused on capturing the rumor's credibility from superficial sources, i.e., from the text itself, user's metadata from social media, and most importantly *discussion thread of other users' replies* from the respective social media, the rumor comes from (Twitter or Reddit[1]). In this context, this work's contribution lies solely on determining the rumor stance of each reply from the discussion thread, not the veracity of the rumor itself. Past work has demonstrated the importance of this subtask as a first step to veracity identification (Ferreira and Vlachos, 2016; Chen et al., 2016b; Enayet and El-Beltagy, 2017; Li et al., 2019; Glenski et al., 2018; Jin et al., 2016; Liu et al., 2015; Mendoza et al., 2010; Procter et al., 2013; Qazvinian et al., 2011; Wei et al., 2019; Zhao et al., 2015, *inter alia*). In detail, subtask A from SemEval-2019's Task 7 (also referred to as SDQC classification) focused on classifying whether the stance of each post in a given Twitter or Reddit thread *supports*, *denies*, *queries* or *comments* the hidden rumor. The rumor itself is not explicitly annotated, but it is a part of the source post from the discussion thread. In all cases, we refer to the judged post in the discussion thread as the *target post*, the post that started the discussion as the *source post*, and the post to which the target post replies to as the *previous post*. An example of a discussion thread is shown in Figure 6.1. Rarely, the source post refutes the hidden rumor; the example of such a case is shown in Figure 6.2. The discussions collected from Twitter were focusing only on 9 topics popular in 2019—Sydney siege, Germanwings crash, etc., whereas the test dataset contained a different set of topics.

In summary, the contributions of the work described in this chapter are:

1. We establish a strong baseline for rumor stance classification, achieving second place in the competition, only 0.2 % F1 behind the winner (Yang et al., 2019a).

2. Unlike previous work, and the majority of the submissions, including the first-place submission, we demonstrate that no handcrafted features are necessary to obtain a top-performing system.

[1] Websites available at https://twitter.com/ and https://www.reddit.com/ respectively.



SDQC support classification. Example 1:

u1: We understand that there are two gunmen and up to a dozen hostages inside the cafe under siege at Sydney.. ISIS flags remain on display #7News **[support]**

u2: @u1 not ISIS flags **[deny]**

u3: @u1 sorry - how do you know its an ISIS flag? Can you actually confirm that? **[query]**

u4: @u3 no she cant cos its actually not **[deny]**

u5: @u1 More on situation at Martin Place in Sydney, AU LINK **[comment]**

u6: @u1 Have you actually confirmed its an ISIS flag or are you talking shit **[query]**

SDQC support classification. Example 2:

u1: These are not timid colours; soldiers back guarding Tomb of Unknown Soldier after today's shooting #StandforCanada PICTURE **[support]**

u2: @u1 Apparently a hoax. Best to take Tweet down. [deny]

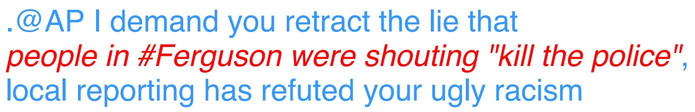
u3: @u1 This photo was taken this morning, before the shooting. [deny]

u4: @u1 I dont believe there are soldiers guarding this area right now. [deny]

u5: @u4 wondered as well. Ive reached out to someone who would know just to confirm that. Hopefully get response soon. [comment]

u4: @u5 ok, thanks. [comment]

Figure 6.1: Example of discussion thread from RumourEval2019 dataset. Each post is accompanied by its stance towards the rumor (in square brackets). Figure taken from Gorrell et al. (2019).



.@AP I demand you retract the lie that
people in #Ferguson were shouting "kill the police",
local reporting has refuted your ugly racism

Figure 6.2: An example of discussion's source post denying the actual rumor which is present in the source post—annotated with red cursive.

	S	D	Q	C	Total
train	925	378	395	3519	5217
in %	18	7	8	67	
dev	102	82	120	1181	1485
in %	7	6	8	80	
test	157	101	93	1476	1827
in %	9	6	5	81	

Table 6.1: Across-class statistics of the training, development, and test dataset. The examples belong to 327/38/81 training/development/test tree-structured discussions.

3. It is shown that the majority of the stances can be correctly judged when considering only the source post, previous post, and the target post itself.

6.1 BUT-FIT's System

6.1.1 Dataset

RumourEval 2019 dataset is composed of Twitter and Reddit dataset. The Twitter dataset was focusing on rumors about 9 particular disasters found on Snopes and Politifact[2]. Reddit threads are more varied and were identified through manually searching debunking forums and current affairs forums. Each thread was annotated by up to 10 humans. The stance inter-agreement was 76.2 % for Twitter and 78 % for Reddit. The statistics of post annotations from the official RumourEval dataset are available in Table 6.1. Additional insights we learned from the dataset are presented in Appendix E.3.

6.1.2 Pre-processing

We parse and replace URLs and mentions with special tokens `$URL$` and `$mention$` using tweet-processor[3]. We use spaCy[4] to split each post into sentences and add the `[EOS]` token to indicate the termination of each sentence. We employ the tokenizer that comes with the Hugging Face PyTorch re-implementation of BERT[5]. The tokenizer lowercases the input and applies the WordPiece encoding (Wu et al., 2016) to split input words into most frequent n-grams present in the pretraining corpus, effectively representing text at the sub-word level while keeping a 30,000-token vocabulary only.

6.1.3 Model

Following (then a recent) trend in transfer learning from language representation models (LRM), we employ the pretrained BERT model. During pretraining, the model's input consists of two documents $\texttt{[CLS]document}_1\texttt{[SEP]document}_2\texttt{[SEP]}$ each represented by a sequence of tokens divided by the special `[SEP]` token and preceded by the special `[CLS]` token. Input tokens are represented by jointly learned token embeddings E_t, segment

[2] Websites available at https://www.snopes.com/ and https://www.politifact.com/ respectively.

[3] https://github.com/s/preprocessor. Accessed 16.3.2023.

[4] https://spacy.io/

[5] https://github.com/huggingface/pytorch-pretrained-BERT

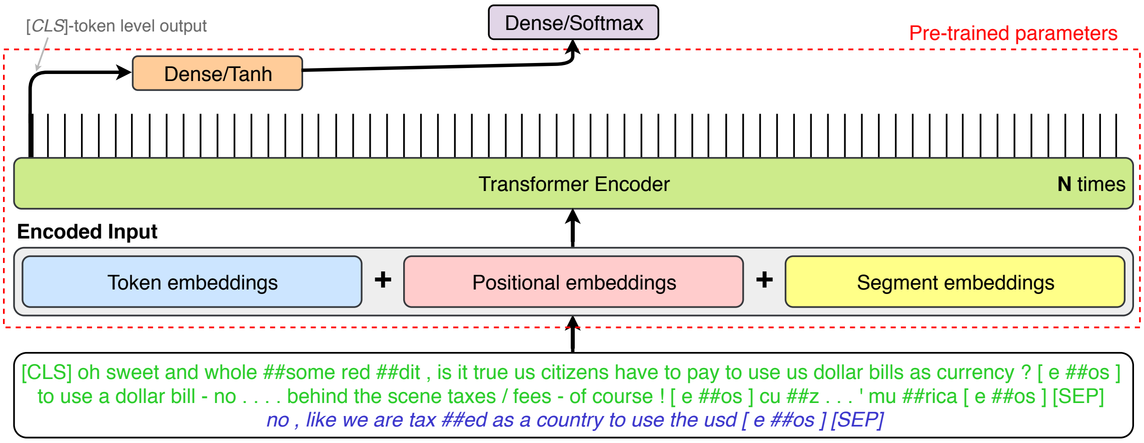

[CLS]-token level output
Dense/Softmax
Pre-trained parameters
Dense/Tanh
Transformer Encoder
N times
Encoded Input
Token embeddings
+
Positional embeddings
+
Segment embeddings
[CLS] oh sweet and whole ##some red ##dit , is it true us citizens have to pay to use us dollar bills as currency ? [e ##os] to use a dollar bill - no . . . . behind the scene taxes / fees - of course ! [e ##os] cu ##z . . . ' mu ##rica [e ##os] [SEP]
no , like we are tax ##ed as a country to use the usd [e ##os] [SEP]

Figure 6.3: An architecture of BUT-FIT's system. The text segment containing `document`$_1$ is green, the segment containing `document`$_2$ (the target post) is blue. The input representation is obtained by summing input embedding matrices $E = E_t + E_s + E_p \in \mathbb{R}^{L \times d}$, L being the input length and d the input dimensionality. The input is passed N times via the transformer encoder. Finally, the `[CLS]` token-level output is fed through two dense layers yielding the class prediction.

embeddings E_s, capturing whether the word belongs into `document`$_1$ or `document`$_2$, and positional embeddings E_p. The full description of the BERT model can be found in Section 2.3.

Our system follows the assumption that the stance of the discussion's post depends only on itself, on the source thread post, and on the previous thread post. Since the pretraining input is composed of two documents, we experimented with different ways of encoding the input (see Section 6.3), ending up with just a concatenation of the source and the previous post as `document`$_1$ (left empty in case of the source post being the target post) and the target post as `document`$_2$. The fine-tuning of BERT is done using the `[CLS]` token-level representation and passing it through two dense layers yielding posterior probabilities as depicted in Figure 6.3. A weighted cross-entropy loss is used to ensure a flat prior over the classes. The class weights are equal to the frequency of the underrepresented class when compared to the majority class, i.e., 2 times less frequent class will be weighted with the weight 2.

6.1.4 Ensembling

Before submission, we trained 100 models differing just by their learning rates. We experimented with 4 different greedy fusion mechanisms in order to maximize the F1 measure on the development set. We note that each of the proposed mechanisms is stochastic, and we re-run it several times and pick the best result.

The `TOP-N` fusion chooses 1 model randomly and adds it to the ensemble. Then, it randomly shuffles the rest of the models and tries to add them into the ensemble one at a time, while iteratively calculating the ensemble's F1 by averaging the output probabilities of every model in the ensemble. If a model increases the total F1 score, the model is permanently added to the ensemble. The process is repeated until no further model improving the ensemble's F1 score can be found. This procedure resulted in a set of 17 best models.

The EXC-N fusion chooses all 100 models into the ensemble and then iteratively drops one model at a time. In each round, a model that results in the largest increase of the ensemble's F1 is dropped. The process stops when dropping any other model cannot increase the F1 score. Using this approach, we ended up using 94 models.

The TOP-N$_s$ fusion is analogous to the TOP-N fusion, but it averages pre-softmax scores instead of output class probabilities.

The OPT-F1 fusion aims at learning weights that sum up to 1 for the weighted average of output probabilities from models selected via the procedure used in the TOP-N strategy. The weights are estimated using modified Powell's method from the SciPy package to maximize the F1 score on the development dataset.

6.2 Experimental Setup

We implemented our models in PyTorch, taking advantage of the Huggingface reimplementation (see Footnote 5), with the "BERT-large-uncased" setting, pretrained using 24 transformer layers, having the hidden unit size of $d = 1024$, 16 attention heads, and $335M$ parameters. When building the ensemble, we picked learning rates from the interval $[1e{-}6, 2e{-}6]$. Each epoch iterates over the dataset in an ordered manner, starting with the shortest sequence. We truncate sequences at maximum length $l = 200$ with a heuristic—firstly we truncate the $\texttt{document}_1$ to length $l/2$, if that is not enough, then we truncate the $\texttt{document}_2$ to the same size. We keep the minibatch size of 32 examples and keep other hyperparameters the same as in the BERT paper (Devlin et al., 2019). We use the same Adam optimizer (Kingma and Ba, 2015) with the L2 weight decay of 0.01 and no warmup. We observed that the BERT models had to cope with a high variance during the training. This might be caused by the problem difficulty, the relatively small number of training examples, or the complexity of the models. To deal with the problem, we decided to discard all models with F1 scores of less than 55 on the development dataset. Our initial experiments used sequences up to the length of 512, but we found no difference when truncating them down to 200. We trained the model on the GeForce RTX 2080 Ti GPU.

6.2.1 Baselines

We compare our system to three baselines. The first is the Branch-LSTM baseline provided by the task organizers[6]—inspired by the winning system of RumourEval 2017. The second baseline FeaturesNN is our re-implementation of the first baseline in PyTorch without the LSTM—posts are classified by means of a 2-layer network (ReLU/Softmax), using only the features defined in Footnote 8. In the third case (BiLSTM+SelfAtt), we use the same input embeddings as in our submitted model but replace the BERT's transformer with a 1-layer BiLSTM network followed by a self-attention and a softmax layer, inspired by Lin et al. (2017). The only difference is no orthogonality constraint, as we didn't find it helpful in preliminary experiments.

6.2.2 Evaluation

The evaluation follows standard classification metrics: **Accuracy** and macro-averaged **F1** score (see Section 2.2).

[6] http://tinyurl.com/y4p5ygn7. Accessed 16.3.2023.

	$\#\theta$	Acc$_{test}$	macro F1$_{dev}$	macro F1$_{test}$	F1$_S$	F1$_Q$	F1$_D$	F1$_C$
Branch-LSTM	453K	84.10	-	49.30	43.80	55.00	7.10	91.30
FeaturesNN	205K	82.84	$45.46 \pm 1e{-}2$	$44.55 \pm 2e{-}2$	40.29	40.12	17.69	80.43
BiLSTM+SelfAtt	28M	83.59	$47.55 \pm 6e{-}3$	$46.81 \pm 6e{-}3$	42.21	45.20	17.75	81.92
BERT$_{base}$	109M	84.67	$51.40 \pm 1e{-}2$	$53.39 \pm 3e{-}2$	43.49	59.88	18.42	90.36
BERT$_{big-noprev}$	335M	84.33	$52.61 \pm 2e{-}2$	$52.91 \pm 4e{-}2$	42.37	55.17	24.44	90.15
BERT$_{big-nosrc}$	335M	84.51	$53.72 \pm 2e{-}2$	$55.13 \pm 3e{-}3$	43.02	56.93	26.53	90.51
BERT$_{big}$	335M	84.08	$56.24 \pm 9e{-}3$	$56.70 \pm 3e{-}2$	44.29	57.07	35.02	90.41
BERT$_{big}$ `EXC-N`*	-	85.50	58.63	60.28	48.89	62.80	37.50	91.94
BERT$_{big}$ `TOP-N`*	-	85.22	62.58	60.67	48.25	62.86	39.74	91.83
BERT$_{big}$ `OPT-F1`	-	85.39	62.68	61.27	48.03	62.26	42.77	92.01
BERT$_{big}$ `TOP-N`$_s$	-	85.50	61.73	**61.67**	49.11	64.45	41.29	91.84
GPT	116M	-	55.51$^{\triangle}$	-	-	-	-	-
GPT+ft	116M×2	-	56.69$^{\triangle}$	-	48	60	48	91
GPT+ft (ens)	-	-	-	61.87	-	-	-	-
ConversBERT+ft	109M×2	-	56.7	-	-	-	-	-
BSAF$^?$	340M	86.10	-	63.47	45.87	60.82	54.86	92.34
BSAF(ens)$^?$	-	-	-	64.66	45.96	61.99	57.93	92.74
jLF$^?$	149M	-	-	57.82	-	-	-	-
jLF+ft$^?$	149M	-	-	63.71	-	-	-	-
jLF+ft+BiLSTM$^?$	160M	-	-	64.87	-	-	-	-
jLF+ft+BiLSTM(ens)$^?$	-	-	-	**67.20**	51.58	65.76	58.90	92.56

Table 6.2: Results on the dev and test dataset. SemEval submissions are denoted by *. Results marked with $^{\triangle}$ report their best—not average—result. Models marked with $^?$ follow unknown result reporting protocol. Results with gray background were published concurrently, or after the publication of the original work (Fajcik et al., 2019).

6.3 Results and Discussion

Main Results The results are shown in Table 6.2. The values for every single model were obtained by averaging the results of 11 models. We report the mean and the standard deviation in these cases. $\#\theta$ denotes the number of parameters. Columns F1$_S$ to F1$_C$ report individual F1 scores for each class on test dataset. All ensemble models have the F1 score optimized on the development dataset. BiLSTM+SelfAtt contains 4.2M parameters, without pretrained BERT embeddings. BERT$_{big-nosrc}$ and BERT$_{big-noprev}$ denote system instantiations with an empty source and an empty target post, respectively. Results marked with +ft use handcrafted features.

Observations We observe that (i) the BERT model offers a substantial performance boost when compared to three baselines; (ii) in the BERT model case, the test set performance actually improves when compared to the dev set performance. This could be caused by its pretraining causing better cross-domain transfer when exposed to different events from the test set. Next, (iii) it can be concluded that both the previous post and source post are essential for the model's performance. Finally (iv) we note that `OPT-F1` achieves excellent results on the dev set, as it is the only fusion trained on the dev set via gradient descent-based procedure. However, in spite of our expectations, this fusion approach did not outperform others on the test set.

What Features Were Not Helpful We tried adding a number of other features, including those indicating positive, neutral, or negative sentiment, and all the features used by the FeaturesNN baseline into a final 2-layer network. We also tried adding jointly learned POS, NER, and dependency tag embeddings, as the third segment embeddings[7]. We also experimented with an explicit `[SEP]` token to separate the source and the previous post in the BERT input. However, none of the mentioned changes led to a statistically significant improvement.

Insights from Subsequent Work In general, we observe two directions of improvements in subsequent work (related work is in the next Section 6.4). Firstly, usage of different handcrafted features significantly improves F1 score of posts that deny the rumor. This is demonstrated by all documented works: the RumourEval's winning system GPT (Yang et al., 2019a), ConversBERT (Radhakrishnan et al., 2020), BSAF (Wang et al., 2021) and jLF (Khandelwal, 2021). For the most extensive set of features, we refer the reader to Khandelwal (2021), where 441 total features were used. This included *structural features* capturing the post's structure (video presence, the ratio of capital letters, . . .), *content features* capturing the presence of indicative words (cue words, swear words, rumor words like 'gossip' or 'hoax'), *conversational features* which capture similarities between target post and other posts in the thread sequence, *affective features* such as the presence of words representing pleasantness, activation, and imagery (Whissell, 2009), *emotion features* such as the presence of words capturing fine-grained emotion spectrum, *dialogue act features* (Pennebaker et al., 2001) and *speech-act features* (Wierzbicka, 1987) capturing the presence of words typical for a predefined set of acts. Unfortunately, none of the works above quantified the importance of its selected features. Secondly, jLF shows that incorporating the very long context of previous discussion posts (not just a source post and a previous post), and jointly training for rumor veracity prediction produces substantial performance improvements. A simple prior from the thread sequences was also uncovered by (Radhakrishnan et al., 2020), who incorporated the beliefs that a query post is less likely to follow another query post, and a support post is more likely to follow a support post via prior distribution interpolated with the estimated distribution.

6.4 Related Work

Previous SemEval Competitions In previous years, there were two SemEval competitions targeting the stance classification. The first one focused on the setting in which the actual rumor was provided (Mohammad et al., 2016). Organizers of SemEval-2016 Task 6 prepared a benchmarking system based on SVM using hand-made features and word embeddings from their previous system for sentiment analysis (Mohammad et al., 2013), outperforming all the challenge participants.

The second competition was the previous RumourEval won by a system based on word vectors, handcrafted features[8] and an LSTM (Hochreiter and Schmidhuber, 1997) summarizing information of the discussion's branches (Kochkina et al., 2017). Other submissions were either based on similar handcrafted features (Singh et al., 2017; Wang et al., 2017; Enayet and El-Beltagy, 2017), features based on sets of words for determining language cues such as Belief or Denial (Bahuleyan and Vechtomova, 2017), post-processing via rule-based

[7] We tried adding the learned representations to the input the same way the segment/positional embeddings are added.

[8] The features included: a flag indicating whether a tweet is a source tweet of a conversation, the length of the tweet, an indicator of the presence of URLs and images, punctuation, the cosine distance to the source tweet and all other tweets in the conversation, the count of negation and swear words, and an average of word vectors corresponding to the tweet.

heuristics after the feature-based classification (Srivastava et al., 2017), Convolutional Neural Networks (CNNs) with rules (García Lozano et al., 2017), or CNNs that jointly learned word embeddings (Chen et al., 2017b).

End-to-end Approaches Augenstein et al. (2016) encode the target text by means of a bidirectional LSTM (BiLSTM), conditioned on the source text. The paper empirically shows that the conditioning of the source text matters. Du et al. (2017) propose target augmented embeddings—embeddings concatenated with an average of source text embeddings—and apply them to compute attention based on the weighted sum of target embeddings, previously transformed via a BiLSTM. Mohtarami et al. (2018) propose an architecture that encodes the source and the target text via an LSTM and a CNN separately and then uses a memory network together with a similarity matrix to capture the similarity between the source and the target text and infers a fixed-size vector suitable for the stance prediction.

Subsequent Work. The concurrent BLCU-NLP team (Yang et al., 2019a) used GPT (Radford et al., 2018) with handcrafted features, data augmentation for similar datasets, and all thread posts preceding the target post prepended (or truncated) and achieved 1st place in the competition. Radhakrishnan et al. (2020) shown that using BERT trained on social media conversations, and post-training probability adjustment with priors can further improve the performance. Wang et al. (2021) shows that score aggregation via an additional self-attention layer could provide marginal improvements. Lastly, Khandelwal (2021) has shown that substantial improvements can be made with extensive feature utilization, the inclusion of the long context preceding the post via LongFormer (Beltagy et al., 2020), and joint training with veracity prediction. Result comparisons with subsequent work can be found in the previous Section 6.3.

6.5 Chapter Summary

The system presented in this chapter achieved the macro F1 score of 61.67, without the usage of any handcrafted features. It improved over the challenge baseline by 12.37 % absolute while using only the source post of discussion, the previous post, and the target post to classify the target post's stance to a rumor. The system achieved 2nd place in the competition. Furthermore, this chapter explored different ensembling techniques and provided evidence showing that `TOP-N`$_s$ fusion is a good choice for probabilistic model ensembling.

Next, it was shown that despite the advances of the current pretrained models, the best performance of the subsequent work at the time of writing achieves 67.2 F1. The inability to achieve better performance may by hindered by the low human inter-agreement—76.2 % for Twitter data and 78 % for Reddit data respectively (Gorrell et al., 2019). Up to this day, the RumourEval dataset is a challenging testing ground for testing language understanding capabilities of future NLP models.

Lastly, the amount of subsequent work which took advantage of the approach and the code released in our original work (Fajcik et al., 2019) demonstrates that our submission became a strong baseline for the community and served as a point of comparison for these works (Radhakrishnan et al., 2020; Wang et al., 2021; Khandelwal, 2021).

6.6 Comparison with Systems Grounded with Trusted Evidence

Approaches to automated fact-checking based on capturing superficial cues of credibility, as the way the claim is written, and metadata such as author, source, or common news-concluding phrases, are definitely not very trustworthy cues for decision-making. However, in prior work, there is an abundance of work (Zubiaga et al., 2016; Derczynski et al., 2017; Li et al., 2019) in this direction. A possible explanation for this such directions is the following. Firstly, it is much easier to implement superficial systems when compared to their trustworthy evidence-grounded counterpart, as the latter includes an extra retrieval step and keeping an up-to-date database of trustworthy information. Secondly, collecting dataset with trustworthy evidence annotation for disproving the false information is expensive, whereas there is plenty of easy-to-scrape FC websites picking up a specific claim, from a specific post of news-site, and verifying its potential falsehood[9]. Thirdly, despite not being faithful, such methods are useful for early intervention, where the purpose of the system is merely to filter out the sheer amount of content for manual fact-checks in social media. Lastly, sometimes no evidence-grounding information is found, and only superficial information is available. Hence, in this thesis, we argue that such systems are *not subordinate, but complementary* to evidence-grounded systems, such as the one presented in the next Chapter 7.

[9] For instance https://www.politifact.com/.

Chapter 7

Interpretable Evidence-Grounded Fact-Checking

Today's automated fact-checking systems are moving from predicting the claim's veracity by capturing the superficial cues of credibility, such as the way the claim is written, the statistics captured in the claim author's profile, or the stances of its respondents on social networks (Zubiaga et al., 2016; Derczynski et al., 2017; Gorrell et al., 2019; Fajcik et al., 2019; Li et al., 2019) towards evidence-grounded systems which, given a claim, identify relevant sources and then use these to predict the claim's veracity (Thorne et al., 2018; Jiang et al., 2020; Park et al., 2022). In practice, providing precise evidence turns out to be at least as important as predicting the veracity itself. Disproving a claim without linking it to factual evidence often fails to be persuasive and can even cause a "backfire" effect—refreshing and strengthening the belief into an erroneous claim (Lewandowsky et al., 2012)[1].

For evidence-grounded fact-checking, most of the existing state-of-the-art systems (Jiang et al., 2021; Stammbach, 2021; Khattab et al., 2021a) employs a 3-stage cascade approach; given a claim, they retrieve relevant documents, rerank relevant evidences (sentences, paragraphs or larger text blocks) within these documents, and predict the claim's veracity from the top-k (usually k=5) relevant evidences.

This comes with several drawbacks; firstly, *the multiple steps of the system lead to error propagation*, i.e. the input to the last system might often be too noisy to contain any information. Some previous work focused on merging evidence reranking and veracity prediction into a single step (Ma et al., 2019; Schlichtkrull et al., 2021). Secondly, in an open-domain setting, *number of relevant evidences can be significantly larger than k*[2], especially when there is a lot of repeated evidence. Thirdly, in an open-domain setting, *sometimes there is both, supporting and refuting evidence.* The re-ranking systems often do not distinguish whether evidence is relevant because it supports or refutes the claim, and thus may select the evidence from one group based on the in-built biases.

Another problem is that existing evidence-grounded datasets for fact-checking often annotate relevant evidences on the different granularity of text. Some datasets provide only noisy paragraph-level annotation (Park et al., 2022), others finer sentence-level annotation (Thorne et al., 2018, 2021). To further strengthen the persuasive effect of the evidences and understand the model's reasoning process, some of these systems provide token-level

[1] Further discussion in Appendix F.2.

[2] e.g.,~3.7 % of FEVER's non-exhaustive annotations.

cues of interpretability (Popat et al., 2018; Liu et al., 2020). However, the token-level interpretability in the mentioned work was often considered a useful trait, which was evaluated only qualitatively, as the labor-intensive human evaluation was out of the scope of their focus.

To this extent, in this chapter we propose Claim-Dissector (CD), a latent variable model which:

1. jointly ranks top-relevant, top-supporting, and top-refuting evidences, and predicts the veracity of the claim in an interpretable way, where the probability of the claim's veracity is estimated using the linear combination of per-evidence probabilities (Subsection 7.1.2),

2. can provide fine-grained (sentence-level or token-level evidence) while using only coarse-grained supervision (on block-level or sentence-level respectively),

3. can be parametrized from a spectrum of language representation models (such as RoBERTa or DeBERTaV3 (Liu et al., 2019; He et al., 2021)).

Finally, we collect a 4-way annotated dataset TLR-FEVER of per-token relevance annotations. This serves as a proxy for evaluating interpretability: we measure how similar are the cues provided by the model to the ones from humans. We believe future work can benefit from our quantitative evaluation approach while maintaining focus.

7.1 Model Description

We present a 2-stage system composed of the *retriever* and the *verifier*. The documents are ranked via retriever. Each document is split into blocks. The blocks from top-ranking documents are passed to the verifier and jointly judged. Our interpretable CD verifier is capable of re-ranking documents for any granularity of relevant evidence (e.g., document, block, sentence, token). Jointly, the same model predicts the claim's veracity. The overall schema of our approach is depicted in Figure 7.1.

7.1.1 Retriever

Given a claim $c \in \mathcal{C}$ from the set of all possible claims $\mathcal{C}$ and the corpus $\mathcal{D} = \{d_1, d_2, ..., d_n\}$ composed of documents d_i, the retriever produces a ranking using function $\text{rank} : \mathcal{C} \times \mathcal{D} \rightarrow \mathbb{R}$ that assigns a claim-dependent score to each document in the corpus. In this work, we focus on the verifier; therefore, we take the strong retriever from Jiang et al. (2021). This retriever interleaves documents ranked by BM25 (Robertson and Zaragoza, 2009) $(a_1, a_2, ...a_n)$ and Wikipedia API $(b_1, b_2, ...b_m)$ following Hanselowski et al. (2018) as $(a_1, b_1, a_2, b_2, ...)$, while skipping duplicate articles. Each document is then split into non-overlapping blocks of size L_x, respecting sentence boundaries[3]. Our verifier then computes its veracity prediction from top-K_1 such blocks. To keep up with similar approaches (Hanselowski et al., 2018; Stammbach and Neumann, 2019; Subramanian and Lee, 2020), we also experiment with expanding evidence with documents hyperlinked to the top retrieved articles. We rank these documents according to the rank and sequential order in the document they were hyperlinked from. We then process these extra ranked documents the same way as retrieved documents, adding top-K_2 blocks to the verifier's input. As discussed more closely in

[3] Every block contains as many sentences as can fit into L_x tokens, considering the verifier's tokenization.

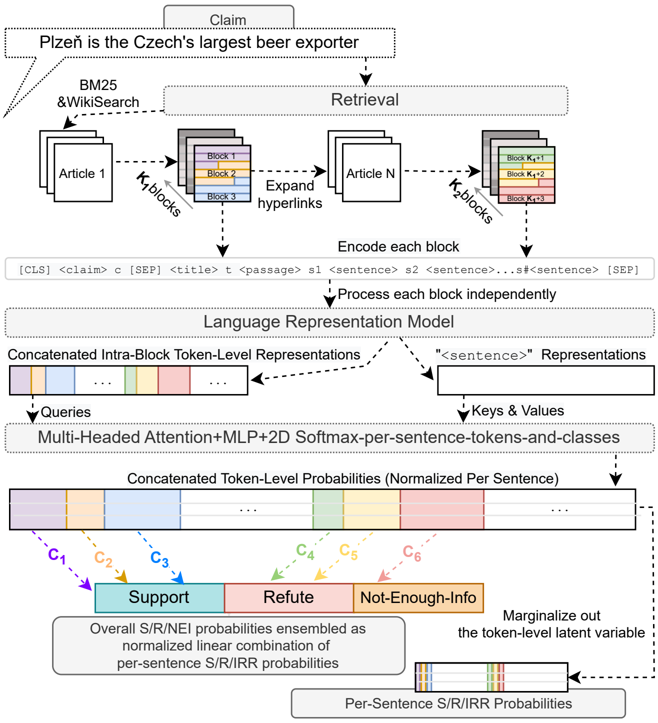

Claim
Plzeň is the Czech's largest beer exporter
BM25
&WikiSearch
Retrieval
Article 1
Block 1
Block 2
Block 3
K_1blocks
Expand
hyperlinks
Article N
Block K_1+1
Block K_1+2
Block K_1+3
K_2blocks
Encode each block
[CLS] <claim> c [SEP] <title> t <passage> s1 <sentence> s2 <sentence>...s#<sentence> [SEP]
Process each block independently
Language Representation Model
Concatenated Intra-Block Token-Level Representations
"<sentence>" Representations
Queries
Keys & Values
Multi-Headed Attention+MLP+2D Softmax-per-sentence-tokens-and-classes
Concatenated Token-Level Probabilities (Normalized Per Sentence)
c_1
c_2
c_3
c_4
c_5
c_6
Support
Refute
Not-Enough-Info
Overall S/R/NEI probabilities ensembled as normalized linear combination of per-sentence S/R/IRR probabilities
Marginalize out the token-level latent variable
Per-Sentence S/R/IRR Probabilities

Figure 7.1: Diagram of Claim-Dissector's workflow. Abbreviations S, R, IRR, NEI stand for support, refute, irrelevant, and not-enough-information. MLP function from the figure is defined by equation 7.1.

Stammbach and Neumann (2019), some relevant documents are impossible to retrieve using just the claim itself, as their relevance is conditioned on other relevant documents. However, we stress that such approaches also mimic the way the FEVER dataset was collected, and thus the improvements of such an approach on "naturally collected" datasets might be negligible if any.

7.1.2 Verifier

The verifier first processes each block independently using a language representation model (LRM) and then aggregates cross-block information via multi-head attention (Vaswani et al., 2017), computing matrix $\boldsymbol{M}$. This matrix is used to compute both, the probability of each evidence's relevance and the probability of the claim's veracity. Furthermore, the way the model is constructed allows learning a linear relationship between these probability spaces.

Formally given a claim c and $K = K_1 + K_2$ blocks, K input sequences x_i for each block i are constructed as

[CLS] <claim> c [SEP] <title> t <passage> s_1 <sentence> s_2
<sentence>...$s_\#$<sentence> [SEP],

where [CLS] and [SEP] are transformer special tokens used during the LRM pretraining (Devlin et al., 2019). Each block is paired with its article's title t and split into sentences $s_1, s_2, ..., s_\#$. Symbols $c, t, s_1, s_2, ..., s_\#$ thus each denote a sequence of tokens. We further introduce new special tokens <claim>, <title>, <passage>, <sentence> to separate different input parts. Crucially, every sentence is appended with a <sentence> token. Their respective embeddings are trained from scratch. Each input x_i is then encoded via LRM $\boldsymbol{E}_i = \text{LRM}(x_i) \in \mathbb{R}^{L_B \times d}$, where L_B is an input sequence length, and d is LRM's hidden dimensionality. The representations of every block are then concatenated into $\boldsymbol{E} = [\boldsymbol{E}_1; \boldsymbol{E}_2; ...; \boldsymbol{E}_K] \in \mathbb{R}^{L \times d}$, where L is the number of all tokens in the input sequences from all retrieved blocks. Then we index-select all representations from $\boldsymbol{E}$ corresponding to positions of sentence tokens in $s_1, s_2, ..., s_\#$ into score matrix $\boldsymbol{E_s} \in \mathbb{R}^{L_e \times d}$, where L_e corresponds to the number of all tokens in all input sentences (without special tokens). Similarly, we index-select all representations at the same positions as the special <sentence> tokens at the input from $\boldsymbol{E}$ into matrix $\boldsymbol{S} \in \mathbb{R}^{L_S \times d}$, where $L_S \ll L_e$ is the total number of sentences in all inputs x_i. The matrix $\boldsymbol{M} \in \mathbb{R}^{L_e \times 3}$ is then given as

$$\boldsymbol{M} = \text{SLP}(\text{MHAtt}(\boldsymbol{E}_s, \boldsymbol{S}, \boldsymbol{S}))\boldsymbol{W}. \tag{7.1}$$

The MHAtt : $\mathbb{R}^{L_e \times d} \times \mathbb{R}^{L_S \times d} \times \mathbb{R}^{L_S \times d} \rightarrow \mathbb{R}^{L_e \times d}$ operator is a multi-head attention (Vaswani et al., 2017) with queries $\boldsymbol{E}_s$, and keys and values $\boldsymbol{S}$. $\boldsymbol{W} \in \mathbb{R}^{d \times 3}$ is a linear transformation, projecting resulting vectors to the desired number of classes (3 in case of FEVER). SLP operator is defined as

$$\text{SLP}(\boldsymbol{x}) = \text{GELU}(\text{dp}(\boldsymbol{W'}\,\text{lnorm}(\boldsymbol{x}))). \tag{7.2}$$

The operator dp denotes the dropout (Srivastava et al., 2014) used in training, $\boldsymbol{W'}$ is a trainable matrix, GELU is the Gaussian Error Linear Unit (Hendrycks and Gimpel, 2016) and lnorm is the layer normalization (Ba et al., 2016).

To compute the per-evidence probabilities we split the matrix $\boldsymbol{M}$ according to tokens belonging to each evidence. For instance, for sentence-level evidence granularity we do split $\boldsymbol{M} = [\boldsymbol{M}^{s_1,1}; \boldsymbol{M}^{s_2,1}; ...; \boldsymbol{M}^{s_\#,K}]$ along dimension L_e into submatrix representations corresponding to sentence s_1 in block 1 up to last sentence $s_\#$ in block K. We then independently normalize each such matrix of i-th evidence of j-th block as[4]:

$$\text{P}^{i,j}(w, y) = \frac{\exp \boldsymbol{M}^{i,j}_{w,y}}{\sum_{w'} \sum_{y'} \exp \boldsymbol{M}^{i,j}_{w',y'}}. \tag{7.3}$$

Note that $w \in \{1, 2, ..., |s_{i,j}|\}$ is a token index in the (i,j)-th evidence and $y \in \{\text{S}, \text{R}, \text{IRR}\}$ is the relevance class label. Then we marginalize over latent variable w to obtain the marginal log-probability per evidence.

$$\log \text{P}^{i,j}(y) = \log \sum_{w'} \text{P}^{i,j}(y, w') \tag{7.4}$$

Then objective $\mathcal{L}_R$ is computed for evidences annotated in label set $\mathbb{A} = \{(y_1^*, (i_1, j_1)), ...\}$ (usually $|\mathbb{A}| \ll L_S$) for a single claim[5].

$$\mathcal{L}_R = \frac{1}{|\mathbb{A}|} \sum_{y^*, (i,j) \in \mathbb{A}} \log \text{P}^{i,j}(y^*) \tag{7.5}$$

In training, $\mathbb{A}$ contains the same amount of relevant and irrelevant labels. For relevant, the log-probability $\log \text{P}^{i,j}(y = y^*)$ is maximized, based the overall claim's veracity label

[4] Note that the probability also depends on input sequences $\{x_i\}_{i \in \{1,2,...,K\}}$, but we omit this dependency for brevity.

[5] If example has NEI veracity in FEVER, $\mathcal{L}_R = 0$.

$y^* \in \{S, R\}$. For irrelevant evidences, $y^* = \text{IRR}$ is maximized. As FEVER contains only annotation of relevant sentences, we follow the heuristic of Jiang et al. (2021) and sample irrelevant sentences ranked between 50 and 200, in order to avoid maximizing the objective for false negatives. In test-time, we rank the evidence (i, j) according to its combined probability of supporting or refuting relevance $score_{i,j} = \sum_{y \in \{S,R\}} \mathrm{P}^{i,j}(y)$.

Next, we compute the probability of the claim's veracity $y \in$ {S, R, NEI}. First notice that scores in $\boldsymbol{M}$ are logits (proof in Appendix C.2)

$$\boldsymbol{M}^{i,j}_{w,y} = \log(C^{i,j}\,\mathrm{P}^{i,j}(w, y)). \tag{7.6}$$

Therefore, we use a learnable extra non-negative degree of freedom $C^{i,j}$ to compute a linear ensemble[6] producing the final probability

$$\mathrm{P}(y) = \frac{\sum_{i,j,w} C^{i,j}\,\mathrm{P}^{i,j}(w, y)}{\sum_{y'} \sum_{i,j,w} C^{i,j}\,\mathrm{P}^{i,j}(w, y')}. \tag{7.7}$$

Lastly, we bias the model to focus only on some tokens in each evidence by enforcing an $\mathcal{L}_2$ penalty over the scores in $\boldsymbol{M}$ by

$$\mathcal{L}_2 = \frac{1}{L_e} ||\boldsymbol{M}||_F^2, \tag{7.8}$$

where $||\cdot||_F$ denotes Frobenius norm. We show empirically that prior imposed by this objective is crucial for obtaining weakly-supervised token-level cues (Section 7.4.2). Therefore the final per-sample loss with hyperparameters λ_R, λ_2 is

$$\mathcal{L} = -\log \mathrm{P}(y) - \lambda_R \mathcal{L}_R + \lambda_2 \mathcal{L}_2. \tag{7.9}$$

7.1.3 Baseline

Apart from previous work, we propose a baseline bridging the proposed system and the recent work of Schlichtkrull et al. (2021). In order to apply this recent work for FEVER, we introduce a few necessary modifications[7]. We normalize all scores in $\boldsymbol{M}$ to compute joint probability across all blocks

$$\mathrm{P}(w, y) = \frac{\exp \boldsymbol{M}_{w,y}}{\sum_{w'} \sum_{y'} \exp \boldsymbol{M}_{w',y'}}. \tag{7.10}$$

Following previous work, we marginalize out per-token probabilities in each evidence $s_{i,j}$.

$$\mathrm{P}(s_{i,j}, y) = \sum_{w' \in s_{i,j}} \mathrm{P}(w', y) \tag{7.11}$$

Using this sentence probability formulation, the objective is computed for every relevant evidence.

$$\mathcal{L}_{b0} = \frac{1}{|\mathbb{A}_p|} \sum_{s_{i,j}, y \in \mathbb{A}_p} \log \mathrm{P}(s_{i,j}, y) \tag{7.12}$$

Next, unlike Schlichtkrull et al. (2021), we interpolate objective $\mathcal{L}_{b0}$ with objective

$$\mathcal{L}_{b1} = \log \mathrm{P}(y) = \log \sum_{s_{i,j}} \mathrm{P}(s_{i,j}, y) \tag{7.13}$$

[6] Assuming y=IRR=NEI.

[7] The necessity of these is explained in Appendix F.3.

by computing their mean. Like the CD, we use $\mathcal{L}_{b1}$ objective to take advantage of examples from NEI class for which we have no annotation in $\mathbb{A}_p$ (and thus $\mathcal{L}_{b0}$ is virtually set to 0). Unlike CD, the annotations $\mathbb{A}_p$ in $\mathcal{L}_{b0}$ contain only relevant labels where $y^* \in \{S, R\}$[8].

In order to not penalize non-annotated false negatives, we compute global distribution in $\mathcal{L}_{b0}$ during training only from representations of tokens from labeled positive and negative sentences in $\boldsymbol{M}$. In test time, we rank evidences according to $score_{i,j} = \sum_{y \in \{S,R\}} \mathrm{P}(s_{i,j}, y)$, and predict claim's veracity according to $\mathrm{P}(y) = \sum_{s_{i,j}} \mathrm{P}(s_{i,j}, y)$. We also considered different model parametrizations discussed in Appendix D.5.

7.1.4 Transferring Supervision to Finer Language Granularity

The proposed model can benefit from annotation on the coarser granularity of the language than tested in inference time. For example, evidence annotation can be done at the document, block, paragraph, or token level. In Section 7.4.2, we show despite the fact that the model is trained on coarse granularity level, the model still shows a moderate performance of relevance prediction when evaluated on finer granularity. We demonstrate this with two experiments.

First, *the model is trained with sentence-level supervision and it is evaluated on a token-level annotation.* For this we leave model as it is—reminding that prior over per-token probabilities enforced by the objective $\mathcal{L}_2$ is crucial (Table 7.8).

Secondly, *we assume only block-level annotation is available in training and we evaluate on sentence-level annotation.* Here, we slightly alter the model, making it rely more on its sentence-level representations. In Section 7.4.2, we show this simple alteration significantly improves the performance at the sentence level. To compute the block-level probability, the block is the evidence, therefore the evidence index can be dropped. The probability of the j-th block b_j is obtained by marginalizing out the per-token/per-sentence probabilities.

$$\begin{aligned} \mathrm{P}(b_j, y) = \sum_{s_{i,j} \in b_j} \mathrm{P}(s_{i,j}, y) = \\ \sum_{s_{i,j} \in b_j} \sum_{w' \in s_{i,j}} \mathrm{P}(w', y) \end{aligned} \tag{7.14}$$

In practice, we found it helpful to replace the block-level probability $\mathrm{P}(b_j, y)$ with its lower-bound $\mathrm{P}(s_{i,j}, y)$ computed for 1 sentence sampled from the relevant sentence likelihood.

$$\mathrm{P}(b_j, y) \approx \mathrm{P}(s_{i,j}, y); s_{i,j} \sim \mathrm{P}(\boldsymbol{s}_{i,j}, y \in \{S, R\}) \tag{7.15}$$

Intuitively, making a single sentence estimate (SSE) forces the model to invest the mass into a few sentence-level probabilities. This is similar to HardEM[9]. In $\mathcal{L}_R$ we then maximize the probabilities of positive blocks computed as in equation 7.15, and negative sentences[10] computed (and normalized) on sentence level as in equation 7.5.

[8] Maximizing NEI class for irrelevant sentences leads to inferior accuracy. This makes sense, since it creates "tug-of-war" dynamics between $\mathcal{L}_{b0}$ and $\mathcal{L}_{b1}$. The former objective tries to allocate mass of joint space in NEI class, since most documents are irrelevant, whereas the latter objective tries to allocate the mass in the dimension of labeled veracity class.

[9] In preliminary experiments, we also tried HardEM, but the results over multiple seeds were unstable.

[10] Indices of irrelevant sentences are mined automatically (see Section 7.1.1), therefore this supervision comes "for free".

Baseline for Token-level Rationales

Similarly to Shah et al. (2020); Schuster et al. (2021), we train a masker—a model which learns to replace the least amount of token embeddings at the Claim-Dissector's input with a single learned embedding[11] in order to maximize the NEI class probability. We compare the unsupervised rationales given by the masker with the unsupervisedly learned rationales provided by the Claim-Dissector on-the-fly.

Our masker follows the same architecture as Claim-Dissector, except that the multi-headed layer from the equation (7.1) is omitted. It receives K_1 blocks at its input, encoded the very same way as for the Claim-Dissector. Instead of computing matrix $\boldsymbol{M}$—which contains three logits per evidence token, the masker predicts two logits $[l_0^i, l_1^i]$—corresponding to keep/mask probabilities $[p_0^i, p_1^i]$ for i-th token in evidence of every block. The mask $[m_0^i, m_1^i]$ is then sampled for every token from concrete distribution via Gumbel-softmax (Jang et al., 2017). During training, i-th token embedding e_i at the Claim-Dissector's input e_i' is replaced with a linear combination of itself and a learned mask-embedding $e_m \in \mathbb{R}^d$, tuned with the masker.

$$e_i' = m_0^i e_i + m_1^i e_m \tag{7.16}$$

The masker is trained to maximize the Claim-Dissector's log-likelihood of NEI class while forcing the mask to be sparse via L1 regularization. Per-sample objective to maximize with sparsity strength hyperparameter λ_S is given as

$$\mathcal{L} = \log \mathrm{P}(y = NEI) - \frac{\lambda_S}{L_e} \sum_i |m_0^i|. \tag{7.17}$$

7.2 Related Work

Datasets. Previous work in supervised open-domain fact-checking often focused on large datasets with the evidence available in Wikipedia such as FEVER (Thorne et al., 2018), FEVER-KILT (Petroni et al., 2021), FAVIQ (Park et al., 2022), HoVer (Jiang et al., 2020), REALFC (Thorne et al., 2021) or TabFact (Chen et al., 2020). We follow this line of work, and validate our approach mainly on FEVER because of its sentence-level annotation, open-domain setting, 3 levels of veracity (into S/R/NEI classes), and controlled way of construction—verification should not require world knowledge, everything should be grounded on trusted, objective, and factual evidence from Wikipedia.

Open-domain Fact-Checking (ODFC) Unlike this work, most of the previous work includes 3-stage systems that retrieve evidence, rerank each document independently, and then make a veracity decision from top-k documents (Thorne et al., 2018; Nie et al., 2019; Zhong et al., 2020).

Jiang et al. (2021) particularly distinguished the line of work which aggregates the final decision from independently computed per-sentence veracity probabilities (Zhou et al., 2019; Soleimani et al., 2020; Pradeep et al., 2021b, *inter alia*) and the line of work where the top-relevant sentences are judged together to compute the final veracity probability (Stammbach and Neumann, 2019; Pradeep et al., 2021a, *inter alia*). Jiang et al. (2021) compares a similar system against these two assumptions, showing that joint judgment of relevant evidence is crucial when computing final veracity. We stress that our system falls into the joint judgment category. Although relevance is computed per sentence, these

[11] Schuster et al. (2021) used mask token representation, but we get singificantly better performance on TLR-FEVER, if we learn the "noise embedding" instead.

probabilities along with linear combination coefficients depend on all inputs, so the model conditioned on all (often hundreds) of input sentences.

To deal with multi-hop evidence (evidence which is impossible to mark as relevant without other evidence) Subramanian and Lee (2020); Stammbach (2021) iteratively rerank evidence sentences to find minimal evidence set, which is passed to the verifier. Our system jointly judges sentences within a block, while the multi-head attention layer could propagate cross-block information. Our overall performance results suggest that our system is about on par with these iterative approaches while requiring only single forward computation. However, further analysis shows our model underperforms on multi-hop evidence (more in Section 7.4.2).

Interpretability Popat et al. (2018); Liu et al. (2020) both introduced systems with an interpretable attention design and demonstrated their ability to highlight important words through a case study. In our work, we take a step further and propose a principled way to evaluate our system quantitatively. We note that Schuster et al. (2021) proposed a very similar quantitative evaluation of token-level rationales, for data from the VitaminC dataset. The dataset, constructed from factual revisions on Wikipedia, assumed that the revised part of facts is the most salient part of the evidence. In contrast, we instruct annotators to manually annotate terms important to their judgment (Section 7.3.1). The VitaminC dataset is not accompanied by the evidence corpus, thus we deemed it as unsuitable for open-domain knowledge processing.

Krishna et al. (2022) proposed a system that parses evidence sentences into natural logic-based inferences (Angeli and Manning, 2014). These provide deterministic proof of the claim's veracity. Authors verify the interpretability of the generated proofs by asking humans to predict veracity verdict from them. However, the model is evaluated only on the FEVER dataset and its derivatives, which contain potential bias to their approach—the claims in this dataset were created from facts through „mutations" according to natural logic itself.

Joint Reranking and Veracity Prediction Schlichtkrull et al. (2021) proposed a system similar to our work for fact-checking over tables. The system computes a single joint probability space for all considered evidence tables. The dataset however contains only claims with true/false outcomes, typically supported by a single table. While our work started ahead of its publication, it can be seen as an extension of this system.

7.3 Experimental Setup

Unless said otherwise, we employ DeBERTaV3 (He et al., 2021) as LRM. In all experiments, we firstly pretrain the model on MNLI (Williams et al., 2018). We use maximum block-length $L_x = 500$. Our recipe for implementation and model training including baselines is closely described in Appendix A.3.

7.3.1 Datasets

We evaluate our system on 5 datasets. Samples from these are available in Appendix E.1.

FEVER. We mainly validate our approach on FEVER (Thorne et al., 2018) and our newly collected dataset of token-level rationales, which extends a subset of FEVER with extra annotations. FEVER is composed of claims constructed from Wikipedia. Each claim was annotated as SUPPORTED, REFUTED or there is NOT-ENOUGH-INFORMATION for

FEVER	Total	Supported	Refuted	Not-Enough-Info
Train	145,449	80,035	29,775	35,639
- FEVER$_{MH}$	12,958 (8.91%)	10,003	2,955	-
- FEVER$_{MH_{ART}}$	11,701 (8.04%)	8,959	2,742	-
Dev	19,998	6,666	6,666	6,666
- FEVER$_{MH}$	1,204 (6.02%)	675	529	-
- FEVER$_{MH_{ART}}$	1,059 (5.30%)	596	463	-
Test	19,998	-	-	-

Table 7.1: FEVER dataset and its subsets.

HoVer	Total	Supported	Not-Supported
Train	18,171	11,023	7,148
- 2 Hop	9,052	6,496	2,556
- 3 Hop	6,084	3,271	2,813
- 4 Hop	3,035	1,256	1.779
Dev	4,000	2,000	2,000
Test	4,000	2,000	2,000

Table 7.2: HoVER dataset and its subsets.

its veracity, achieving 68 % inter-annotator agreement. Each annotator was presented with an evidence sentence and the first sentence of articles from hyperlinked terms. In FEVER, examples in the development set contain multi-way annotation of relevant sentences, i.e., each annotator selected set of sentences (evidence group) he considered relevant. To analyze the performance of our components on samples that might need multi-hop reasoning, we further created subsets of training/development set. FEVER$_{MH}$ contains only examples where all annotators agreed on that more than 1 sentence is required for verification, whereas FEVER$_{MH_{ART}}$ contains only examples, where all annotators agreed that *sentences from different articles* are required for verification. As majority of examples of FEVER$_{MH}$ are from FEVER$_{MH_{ART}}$, we only evaluate on FEVER$_{MH}$. We include the subset statistics in Table 7.1.

TLR-FEVER To validate token-level rationales, we collect our own test dataset on a random subset of the validation set (only considering examples with gold sentence annotation). We collect a 4-way annotated set of token-level rationales inside the already annotated relevant sentences (each token is annotated with a RELEVANT/IRRELEVANT label). The annotators were colleagues with NLP background from our lab. We instruct every annotator via written guidelines, and then we had 1-on-1 meeting after annotating a few samples, verifying that the contents of the guidelines were understood correctly. We let annotators annotate 100 samples, and resolve reported errors manually, obtaining 94 samples with fine-grained token-level annotation. In guidelines, we simply instruct annotators to *highlight the minimal part of the text they find important for supporting/refuting the claim. There should be such a part in every golden sentence (unless an annotation error happened).* The complete guidelines are available in Appendix E.2.

To establish the performance of the average annotator, in place of inter-agreement, we compute the performance of each annotator compared to other annotators on the dataset and then compute the average annotator performance. We refer to this as *human baseline lower-bound*, as each annotator was compared to only 3 annotations, while the system

FAVIQ	**Total**	**Support**	**Refute**
Train	17,008	8,504	8,504
Dev	4,260	2,130	2,130
Test	4,688	2,344	2,344

Table 7.3: FAVIQ dataset statistics.

RealFC	**Total**	**Supported**	**Refuted**	**Neutral**
Train	50,902	13,122 (25.78%)	6,236 (12.25%)	31544 (61.97%)
- Bipolar Evidence	3,193 (6.27%)	1,453	1,422	318
Dev	5,832	1,578 (27.06%)	724 (12.41%)	3530 (60.53%)
- Bipolar Evidence	381 (6.53%)	190	162	29
Test	6,265	1,742 (27.81%)	817 (13.04%)	3706 (59.15%)
- Bipolar Evidence	473 (7.55%)	209	209	55

Table 7.4: RealFC dataset and its subsets.

is compared to 4 annotations. We measure performance via F1 metric and report it in Table 7.8.

HoVer We study the limitations of our system when applied to a dataset that requires multi-hop reasoning on HoVer (Jiang et al., 2020). The dataset is created from HotspotQA (Yang et al., 2018), a Wikipedia-based QA dataset for multi-hop reasoning. Annotators first manually converted question-answer pairs into claims. The subset of original 2-hop claims is further extended to require 3 to 4 hops, using related Wikipedia pages. Then the claims are mutated, by word substitutions, the inclusion of negated words (not necessarily making claim refuting (Schuster et al., 2019)), entity substitution, or making the claim more specific or more general. Next, the claims are labeled into SUPPORTED and NOT-SUPPORTED classes. Unlike in FEVER, due to low annotator inter-agreement between *refuting* and NEI classes, authors merge these into NOT-SUPPORTED class, achieving inter-agreement 90 % dataset statistics, including a number of examples requiring a different amount of reasoning hops, are shown in Table 7.2.

FAVIQ-A. To assess the performance of our system when using silver-mined passage annotations, following Asai et al. (2022), we test our system on FAVIQ-A (Park et al., 2022). FAVIQ-A is a fact-checking dataset created by automatically converting disambiguated question-answer pairs from AmbigQA (Min et al., 2020). The conversion is done via 11B text-to-text language model T5-XXL (Raffel et al., 2020) trained on QA to natural language inference format conversion data from Demszky et al. (2018) and further finetuned on small AmbigQA-specific dataset created by the authors. Having two disambiguated questions q_1, q_2 of question q, authors convert them into claims using their corresponding answers a_1, a_2 to create SUPPORTED claims and pair them with mismatched answers (q_1,a_2), (q_2,a_1) to create REFUTED claims. As evidence labels, we use labels automatically mined Wikipedia passage labels from Asai et al. (2022). The statistics of this dataset are shown in Table 7.3.

RealFC To validate our system on a natural dataset with exhaustive annotations and bipolar evidence annotated, we resort to (Thorne et al., 2021). RealFC is a dataset created from yes/no questions collected from Google search queries (Clark et al., 2019; Kwiatkowski et al., 2019). Questions are manually converted to claims. The system is given the claim along with a specific section of the Wikipedia page. The section is split into non-overlapping blocks, as in FEVER. Thus, this dataset does not include the retrieval step. Each sentence of a section is annotated for being SUPPORTING/REFUTING/NEUTRAL towards the claim. The verdict[12] for each section can be SUPPORT/REFUTE/NEUTRAL. About 6–7 %[13] of these sections contain both, supporting and refuting evidence (so-called bipolar evidence).

7.3.2 Evaluation Metrics

Recall@Input (RaI). We evaluate retrieval w.r.t. recall at the model's input while considering the different amount of K_1+K_2 blocks at the input, i.e. the score hit counts iff any annotated evidence group was matched in K_1+K_2 input blocks.

Number of Sentences@Input (#SaI) denotes an average number of sentences at the model's input under the corresponding $K_1 + K_2$ retrieval setting.

Accuracy (A) The proportion of correctly classified samples, disregarding the predicted evidence.

Recall@5 (R@5) The proportion of samples for which any annotated evidence group is matched within top-5 ranked sentences.

FEVER-Score (FS). The proportion of samples for which (i) any annotated evidence group is matched within top-5 ranked sentences, and (ii) the correct class is predicted.

$\mathbf{F}_1$ Score measures unigram overlap between predicted tokens and reference tokens, disregarding articles (same way as in QA). Having multiple references, the maximum F1 between prediction and any reference is considered per sample. More details can be found in Section 2.2.

$\mathbf{F}_1$ Score for binary relevance classification computes the harmonic mean of precision and recall for less frequent class (Section 2.2).

EM Unlike in QA, the EM results reported on HoVer, used for assessment of performance on relevance predictions, are proportional to the number of cases when *all evidences annotated as ground truth* were selected as relevant.

Conditional scoring Similarly to FS, conditional scoring validates both; the performance of reranking and veracity prediction. Specifically, it computes average accuracy/F1 score across samples, while setting per-sample hit/F1 to 0, if the model predicted the wrong veracity. The exact definition of conditional scores used is in Appendix E.4.

In practice, both CD and masker model infer continuous scores capturing relevance for every token[14]. When evaluating F1, we select only tokens with scores greater than threshold τ. We tune the optimal threshold τ w.r.t. F1 on TLR-FEVER.

7.4 Results and Analysis on FEVER

7.4.1 Retrieval Performance

We evaluate the retrieval method from Jiang et al. (2021) and the proposed hyperlink expansion method in Table 7.5. We focus on analyzing the effect of hyperlink expansion,

[12] The inter-agreement for section-level annotations is unknown.

[13] Depending on the data split.

[14] We consider mask-class logits as scores for masker.

K_1+K_2	FEVER	FEVER$_{MH}$	FEVER$_{MH_{ART}}$	#SaI
35+0	94.2	52.0	45.8	239.9
100+0	95.1	58.5	53.1	649.4
35+10	95.2	61.9	57.0	269.6
35+20	95.9	69.0	65.2	309.0
35+30	96.7	77.5	74.7	388.6
35+35	97.5	84.1	82.3	506.7
35+50	97.7	86.5	85.0	624.3
35+100	98.4	93.0	92.4	1008.8
100+100	98.6	93.4	92.7	1431.0

Table 7.5: Retrieval performance in RaI on FEVER dev set and its subsets.

varying K_2, while keeping $K_1 = 35$ in most experiments, which is setting similar to previous work—Jiang et al. (2021) considers reranking top-200 sentences. We observe that setting $K_1 + K_2 = 35 + 10$ already outperforms retrieval without hyperlink expansion and $K_1 = 100$ blocks. Such observation is thus consistent with previous work which used hyperlink signal (Hanselowski et al., 2018; Stammbach and Neumann, 2019).

7.4.2 Claim-Dissector's Performance

We report the results of base-sized models based on a 3-checkpoint average. We train only a single large model. We further evaluate retrieval in Appendix D.6 as it is a non-essential part of our contribution.

Performance. We compare the performance of our system with previous work in Table 7.6. Results marked with ? were unknown/uncertain, and unconfirmed by authors. We note that apart from HAN (Ma et al., 2019), all previous systems were considering two separate systems for reranking and veracity prediction. Next, we note that only the ProofVer system uses additional data. It leverages rewritten-claim data for fact-correction paired with original FEVER claims (Thorne et al., 2021).

We observe that (i) even our base-sized RoBERTa-based CD model outperforms base-sized HESM on dev data, and its large version outperforms large-sized KGAT, DREAM, and HESM on test data, (ii) our base-sized DebertaV3-based CD model matches large-based DREAM and even KGAT with oracle inputs on dev set, (iii) model version with hyperlink expansion (suffixed \w HE) further improves the overall performance, (iv) using larger model improves mostly its accuracy, (v) Claim-Dissector$_L$ \w HE achieves better FEVER score than T5-based approach (with two 3B models) and better accuracy than LongFormer+DebertaXL, but it is not Pareto optimal to these previous SOTA, (vi) our model is outmatched by recent ProofVer-SB, though it is more efficient as ProofVer-SB requires two rounds of reranking and autoregressive decoding. We still consider this a strong feat, as our system was focusing on modeling reranking and veracity prediction jointly in an interpretable way. Finally, we inject blocks with missing golden evidence into inputs of Claim-Dissector$_L$ \w HE at random positions and measure oracle performance. We observe that items missed by retrieval are still beneficial to the performance.

Ablations. We ablate components of Claim-Dissector (CD) in Table 7.7. Firstly, we resort to single-task training. We drop veracity classification (VC) objective $\log \mathrm{P}(y)$ or relevance classification (RC) objective $\mathcal{L}_R$ from the loss. We observe an overall trend—single-task model performs slightly better to multi-task model. The advantages of the multi-task model, however, lie in its efficiency and ability to provide explanations between per-evidence rele-

	System	FS	A	R@5	HA	#θ
Development Set	TwoWingOS (Yin and Roth, 2018)	54.3	75.9	53.8	✗	?
	HAN (Ma et al., 2019)	57.1	72.0	53.6	✓	?
	UNC (Nie et al., 2019)	66.5	69.7	86.8	✓	408M
	HESM (Subramanian and Lee, 2020)	73.4	75.8	90.5	✓	39M
	KGAT[OR] (Liu et al., 2020)	76.1	78.3	94.4	✗	465M
	DREAM (Zhong et al., 2020)	-	79.2	90.5	✓$^?$	487M
	T5 (Jiang et al., 2021)	77.8	**81.3**	90.5	✗	5.7B
	LF+D$_{XL}$ (Stammbach, 2021)	-	-	90.8	✗	1.2B
	LF$_{2-iter}$+D$_{XL}$ (Stammbach, 2021)	-	-	**93.6**	✓	1.2B
	ProofVer-MV (Krishna et al., 2022)	78.2	80.2	-	✓	515M
	ProofVer-SB (Krishna et al., 2022)	**79.1**	80.7	**93.6**	✓	765M
	Baseline$_{joint}$	75.2	79.8	90.0	✗	187M
	Claim-Dissector$_{RoBERTa}$	74.6	78.6	90.4	✗	127M
	Claim-Dissector$_{RoBERTaL}$	75.1	79.1	90.6	✗	360M
	Claim-Dissector$_{RoBERTaL}$ \w HE	76.1	79.4	91.7	✓	360M
	Claim-Dissector	76.2	79.5	91.5	✗	187M
	Claim-Dissector \w HE	76.9	79.8	93.0	✓	187M
	Claim-Dissector$_{L}$	76.9	80.4	91.8	✗	439M
	Claim-Dissector$_{L}$ \w HE	$\underline{78.0}$	$\underline{80.8}$	$\underline{93.3}$	✓	439M
	Claim-Dissector$_{L}$ \w HE [OR]	78.9	81.2	94.7	✓	439M
Test Set	KGAT (Liu et al., 2020)	70.4	74.1	-	✗	465M
	DREAM (Zhong et al., 2020)	70.6	76.9	-	✓$^?$	487M
	HESM (Subramanian and Lee, 2020)	71.5	74.6	-	✓	58M
	ProofVer-MV (Krishna et al., 2022)	74.4	79.3	-	✓	515M
	T5 (Jiang et al., 2021)	75.9	79.4	-	✗	5.7B
	LF$_{2-iter}$+D$_{XL}$ (Stammbach, 2021)	76.8	79.2	-	✓	1.2B
	ProofVer-SB (Krishna et al., 2022)	**76.8**	**79.5**	-	✓	765M
	Claim-Dissector$_{RoBERTaL}$	73.1	76.4	-	✗	360M
	Claim-Dissector$_{RoBERTaL}$ \w HE	74.3	77.8	-	✓	360M
	Claim-Dissector$_{L}$	74.7	78.5	-	✗	439M
	Claim-Dissector$_{L}$ \w HE	$\underline{76.5}$	$\underline{79.3}$	-	✓	439M

Table 7.6: Performance on dev and test splits of FEVER. #θ denotes the number of parameters in the model. Model names suffixed with [OR](as Oracle) inject missing golden evidence into its input. Models using any kind of hyperlink augmentation (HA) are marked. Our models with hyperlink expansion are suffixed with (\w HE). **Overall best** and $\underline{\text{our best}}$ result are in bold and underlined respectively (disregarding oracle results).

	FEVER			**FEVER**$_{MH}$		
System	**FS**	**A**	**R@5**	**FS**	**A**	**R@5**
CD$_{LARGE}$ \w HE [OR]	78.9	81.2	94.8	50.3	81.9	58.9
CD$_{LARGE}$ \w HE	78.0	80.8	93.4	44.7	81.2	53.1
CD \w HE	76.9	79.8	93.2	41.3	80.8	49.9
CD \w HE \wo MH	76.5	79.5	93.0	41.7	80.8	50.2
Baseline	75.2	79.8	90.0	28.9	80.9	34.7
CD	76.2	79.6	91.7	30.0	79.2	36.4
CD \wo $\mathcal{L}_2$	76.0	79.7	91.6	30.4	79.5	36.2
CD \wo VC	-	-	91.9	-	-	37.8
CD \wo RC	-	79.9	-	-	81.5	-

Table 7.7: Ablation Study. Minor differences to Table 7.6 are caused by different early-stopping (Appendix A.3).

vances and final conclusion. Next, we observe that dropping the $\mathcal{L}_2$ objective doesn't affect the performance significantly. Further, we study the effect of hyperlink expansion (HE) and the effect of the multi-head (MH) attention layer. As expected, hyperlink expansion

System	F1
Select All Tokens	52
Select Claim Overlaps	63
Masker	71
Claim-Dissector \wo $\mathcal{L}_2$	60
Claim-Dissector	77
Human Performance LB	85

Table 7.8: Token-level relevance on TLR-FEVER.

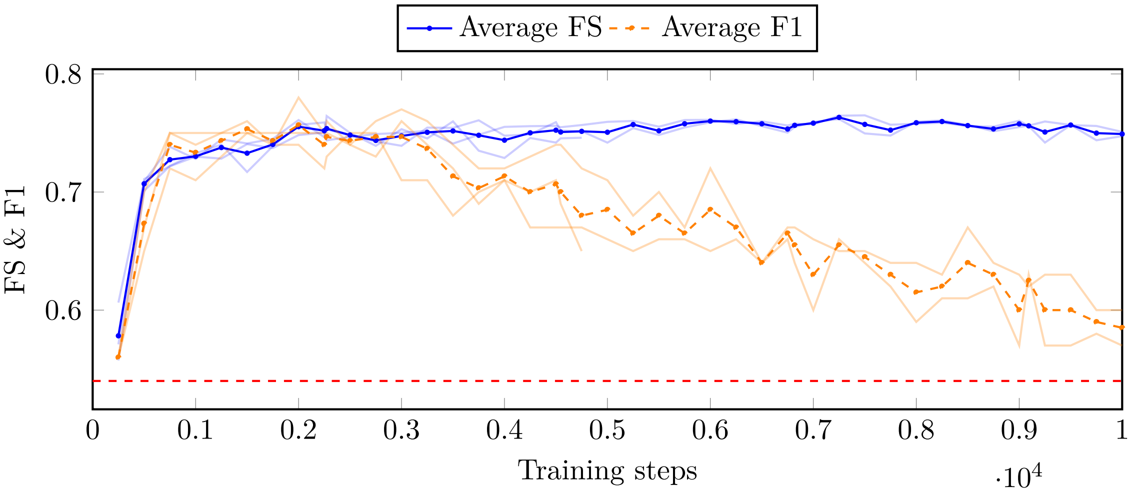


Figure 7.2: Average FEVER-Score (FS) and F1 performance on dev sets during training. Red dashed horizontal line marks the F1 performance when selecting all tokens (Select All Tokens) from Table 7.8. Opaque lines show the performance of individual checkpoints.

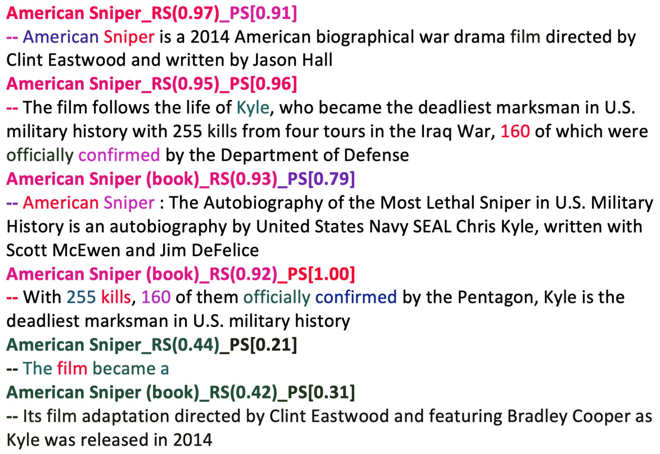
dramatically increases performance on FEVER$_{MH}$. The multi-head attention also brings marginal improvements to results on FEVER. However, contrary to our expectations, there is no effect of the MH layer on FEVER$_{MH}$, the improvements happened in the non-multihop part. Additional experiments with CD on the HoVer dataset (see Section 7.5) confirm this, CD does not work well on examples with cross-article multihop evidence. Improving cross-article reasoning was not our aim, and we leave the investigation to future research.

Transferring Sentence-Level Supervision to Token-Level Performance. We evaluated the performance of token-level rationales[15] on our dataset in Table 7.8. We considered two baselines. The first was to select all tokens in golden evidences (Select All Tokens). The second was to select only tokens that overlap with claim tokens (Select Claim Overlaps). We found that our model with weakly-supervised objective produces token-level rationales significantly better[16] than the masker—a separate model trained explicitly to identify tokens important to the model's prediction. Furthermore, the results also demonstrate the importance of $\mathcal{L}_2$ objective. However, human performance is still considerably beyond the performance of our approach.

Furthermore, in Figure 7.2, we analyze how the performance on FEVER-Score and F1 changes over the course of training on FEVER and TLR-FEVER sets. We find our scores

[15] We visualized predicted token-level rationales on 100 random dev set examples in the supplementary material.

[16] See Appendix F.4 for our F1 significance testing protocol.



American Sniper_RS(0.97)_PS[0.91]

-- American Sniper is a 2014 American biographical war drama film directed by Clint Eastwood and written by Jason Hall

American Sniper_RS(0.95)_PS[0.96]

-- The film follows the life of Kyle, who became the deadliest marksman in U.S. military history with 255 kills from four tours in the Iraq War, 160 of which were officially confirmed by the Department of Defense

American Sniper (book)_RS(0.93)_PS[0.79]

-- American Sniper : The Autobiography of the Most Lethal Sniper in U.S. Military History is an autobiography by United States Navy SEAL Chris Kyle, written with Scott McEwen and Jim DeFelice

American Sniper (book)_RS(0.92)_PS[1.00]

-- With 255 kills, 160 of them officially confirmed by the Pentagon, Kyle is the deadliest marksman in U.S. military history

American Sniper_RS(0.44)_PS[0.21]

-- The film became a

American Sniper (book)_RS(0.42)_PS[0.31]

-- Its film adaptation directed by Clint Eastwood and featuring Bradley Cooper as Kyle was released in 2014

Figure 7.3: Example of interpretable refuting evidence from Claim-Dissector for claim "*American Sniper (book) is about a sniper with 170 officially confirmed kills.*".

reach the top performance and then quickly deteriorate. It is thus necessary to do the early stopping on both, the performance metric and the interpretability metric. In addition, our experiments shown that tuning of λ_2 is crucial, i.e., $\lambda_2 = 2e-3$ tuned for DeBERTa-base, achieves no interpretability for the large version (where we use $5e-4$)[17].

Lastly, interpretable refuting example[18] is available in Figure 7.3. Example shows top-6 refuting sentences ranked by their refuting relevance probability $\mathrm{P}^{i,j}(y=R)$. Each sentence is prefixed with its Wikipedia article title, refuting relevance probability (RS) and prediction score (PS). The prediction score is the corresponding non-negative linear coefficient $C^{i,j}$ max-normalized between 0 and 1 based on maximum $C^{i,j}$ for this sample. The token-level relevance, sentence relevance, and sentence prediction score are highlighted on a red-to-black scale (low relevance is black, high relevance is red). Interestingly, the prediction score is highest for sentences containing crucial refuting evidence—the number of confirmed kills[19].

Are Prediction Scores Useful? In Figure 7.3, a maximum prediction score is assigned to a sentence, that has a lower relevance score than the most relevant document. However, we argue that the sentence with the highest prediction score contains the most relevant information. Hence we formulate the hypothesis that *top prediction score better ranks relevance towards final judgment, whereas top relevance score only reflects the model's confidence of the sentence being somehow relevant.* First, we note that scores are highly correlated, but not the same (average Kendall-τ 0.84). Next, we turn to the A/B testing (details in Appendix F.5), where we select 100 random samples such that: (i) each was correctly predicted, (ii) has a verdict *supported* or *refuted.* From these, we select (a) a sentence with the highest prediction score and (b) a sentence with equal or better relevance probability than (a); if there is no such sentence, we don't include it. We employ 5 annotators to say their preference for sentence (a), sentence (b), or neither (c).

We find that (i) 80 % of annotators unanimously agreed on *not preferring* (b), (ii) 3 or more annotators in 73 % of cases preferred (a) over (b,c) and finally (iii) the worst

[17] The sensitivity to λ_2 is analyzed in Appendix D.7.

[18] We visualized token-level rationales on 100 random dev set examples at shorturl.at/beTY2.

[19] We further visualized token-level rationales on 100 random dev set examples at shorturl.at/beTY2. Accessed 16.3.2023.

Model	**FS**	**A**	**R@5**
Full Supervision	76.2	79.5	91.5
Block Supervision	65.5	76.9	77.8
Block Supervision + SSE	69.7	78.1	83.0

Table 7.9: Sentence-level performance on FEVER dev set under different kinds of supervision.

single annotator preference for (a) over (a+b) cases was 86 %, demonstrating that human preferences strongly support declared hypothesis.

Transferring Block-Level Supervision to Sentence-Level Performance. The performance of our model on the sentence-level evidences is evaluated in Table 7.9. We notice that even our vanilla Claim-Dissector trained with block supervision reaches competitive recall@5 on the sentence level. However, adding SSE from equation 7.15 leads to further improvements both in the recall, but also in accuracy. We expected the recall will be improved because the model now focuses on assigning high probability mass only to some sentences within the block since high entropy of the per-sentence distribution would be penalized in the loss. However, we have not foreseen the damaging effect on accuracy, that block-level supervision causes. Interestingly, the accuracy without any evidence supervision reported in the last row of Table 7.7 was increased.

Detecting Samples with Bipolar Evidence We manually analyzed whether we can take advantage of the model's ability to distinguish between evidence, which is relevant because it supports the claim, and evidence that is relevant because it refutes the claim. To do so, we try to automatically detect examples from the validation set, which contain both, supporting and refuting evidence (which we refer to as bipolar evidence). We note that there were no examples with explicitly annotated bipolar evidence in the training data.

We select all examples where the model predicted at least 0.9 probability for any supporting and any refuting evidence[20]. We found that out of 72 such examples, 66 %(48) we judged as indeed having the bipolar evidence. We observed that about half (25/48) of these examples had bipolar evidence because of the entity ambiguity caused by the open-domain setting. E.g., claim "*Bones is a movie*" was supported by sentence article "*Bones (2001 film)*" but also refuted by a sentence from article "*Bones (TV series)*" and "*Bone*" (a rigid organ).

7.5 Results and Analysis on Other Datasets

In this section, we study the application of CD on datasets of non-FEVER origin. Unless said otherwise, we use hyperparameters from FEVER. The experimental details of the application on each dataset are explained along with the results.

HoVER To further study how limited is our model on claims, which require multihop information, we trained and tested our system on HoVer in Table 7.10. In particular, we follow the recipe for Baleen (Khattab et al., 2021a) and retrieve 4×25 top articles using official Baleen implementation with quantized index[21], which achieves about 2 % lower retrieval@100 on supported samples than reported in the paper. We split top-5

[20] Annotations are available at shorturl.at/qrtIP. Accessed 16.3.2023.

[21] https://github.com/stanford-futuredata/Baleen. Accessed 16.3.2023.

	Hops	**A**	**EM**
Baleen	2	-	47.3
	3	-	37.7
	4	-	33.3
	All	84.5	39.2
CD	2	81.3	48.0
	3	80.1	16.9
	4	78.1	7.7
	All	79.9	23.3

Table 7.10: Results on HoVer dataset (dev split).

	Test	**Dev**	**Δ**	θ
BART$_{LARGE}$ (Park et al., 2022)	64.9	66.9	2.0	374M
EGG (Asai et al., 2022)	65.7	69.6	3.9	336M
CD$_{RoBERTa}$	58.6	69.8	11.2	127M
CD$_{RoBERTaL}$	66.9	73.3	6.4	360M
CD	69.8	76.3	6.5	187M
CD \wo RC	71.2	73.3	2.1	187M
CD$_{LARGE}$	72.0	79.7	7.7	439M

Table 7.11: Performance on FAVIQ-A.

documents from each iteration into blocks, padding input with further documents from the first retrieval iteration when necessary. We keep input size at $K_1 = 35$, and we do not use hyperlink expansion. We compute the probability of the NOT-SUPPORTED class by summing NEI and REFUTE classes. Furthermore, we assume simplified conditions, infuse inputs with oracle information when necessary (achieving RaI 100 %), and predict as many evidences, as there was annotated. We refer the reader to Khattab et al. (2021a) for further information about setup and evaluation metrics.

Nevertheless, our system lags behind Baleen on 3 and 4-hop examples. We hypothesize that similarly to Baleen, an autoregressive process is necessary to match its multi-hop performance. We leave the question of interpretable multi-hop fact-checking with Claim-Dissector open for our future work.

FAVIQ-A The performance of our system on FAVIQ-A, a dataset with silver-mined annotations, is shown in Table 7.11. The model is given top-20 passages retrieved via the DPR system (Karpukhin et al., 2020). We compare to the evidentiality-guided generator (EGG), a t5-based FiD (Izacard and Grave, 2021b) with two decoders from Asai et al. (2022). Similarly to HoVER, we compute the probability of the FAVIQ-REFUTE class by summing NEI and REFUTE class probs from the original FEVER-made model architecture.

The results shown in Table 7.11 demonstrate that DebertaV3-based Claim-Dissector reaches solid state-of-the-art results on the dataset. The domain mismatch (measured by difference **Δ**) between development and test set is likely caused by the domain shift of NaturalQuestions test set, from which FAVIQ's test set was created (see Appendix B in Min et al. (2020)). However, despite our best efforts, we have not uncovered the cause of massive degradation between the dev and test set for `roberta-base` based Claim-Dissector (the standard deviation on the test set was only ±0.4 accuracy points). Interestingly, the gap for DeBERTa-based CD is negligible if we do not use relevance supervision (CD \wo RC).

Dataset	Model	EviF1	VA	VF1	CondAcc	CondF1
Full	3-way+$\mathbb{C}$ (Thorne et al., 2021)	46.8	75.5	65.5	64.2	51.8
	Any-Best+$\mathbb{C}$ (Thorne et al., 2021)	48.6	75.5	65.5	64.2	52.2
	CD$_{RoBERTa}$\W	48.7	76.7	66.6	64.8	52.4
	CD$_{RoBERTa}$	53.0	76.3	68.6	65.0	55.3
	CD	54.7	79.1	72.2	67.6	58.5
	CD$_{LARGE}$	56.3	80.8	73.7	69.6	60.7
Bipolar	3-way+$\mathbb{C}$ (Thorne et al., 2021)	48.1±3.3	43.0±1.1	40.3±1.2	23.0±1.4	28.7±1.4
	CD$_{RoBERTa}$	56.4±0.7	51.9±1.8	47.7±2.9	32.4±0.6	38.7±1.2

Table 7.12: Performance on RealFC test set.

RealFC The results on RealFC are shown in Table 7.12. For relevant evidence classification, binary F1 (EviF1) computed from the concatenation of all relevant/irrelevant decisions for all sentences is reported. Here supporting or refuting evidence counts as relevant, neutral as irrelevant. For verification, accuracy/macro F1 (denoted VA, VF1) are reported. Lastly, we also report conditional scores CondAcc and CondF1, which are aggregated from per-sample binary F1 relevance, set to 0 if the target veracity was not predicted correctly[22]. We compare CD with a pipelined baseline composed from a 3-class relevance classification (assuming either supporting or refuting evidence is relevant) followed by a veracity classifier (first row). Both components are based on `roberta-base` (Liu et al., 2019). We also report best number for the corresponding column across all (RoBERTa-based) baselines from Thorne et al. (2021) (second row). We find that in a similar setup, CD improves only marginally over baseline. The early stopping of baselines and the third row is performed on VA. Row 4 and further report on results early stopped on CondF1, as we found CondF1 to correlate with the majority of the metrics. Furthermore, from row 4, we use veracity class weighting similar to one from the previous Chapter (Subsection 6.1.3). Using identity-weighting (each weight is 1), we observe accuracy to be maximal, whereas using the inverse-class-prior weighting exactly as in the previous chapter we found F1 to perform the best. However, for maximizing conditional scores, we found the average of identity-weighting and inverse-class-prior to work the best, and so we report these further.

We find that CD-based systems with class weighting set a new state-of-the-art on the dataset. Additionally, we uncover the large performance boost specifically on the subset of the dataset with annotated bipolar evidence (with at least 1 supporting, and 1 refuting evidence annotated) (last two rows).

7.6 Chapter Summary

In this chapter, we proposed Claim-Dissector, an interpretable probabilistic model for fact-checking and fact-analysis. Our model jointly predicts the supporting/refuting evidence and the claim's veracity. It achieves results comparable to the state-of-the-art on FEVER, and sets new state-of-the-art on FAVIQ-A and RealFC datasets while providing three layers of interpretability. Firstly, it identifies salient tokens important for the final prediction. Secondly, it allows disentangling the ranking of relevant evidences into ranking of supporting evidence and the ranking of refuting evidence. This allows detecting bipolar evidence without being exposed to such bipolar evidence sets during training. Thirdly, it combines the per-token relevance probabilities via linear combination into a final veracity assessment.

[22] Due to rather complicated exact definition, official conditional scores are documented in Appendix E.4

Therefore it can be identified to what extent the relevance of each token/sentence/block-/document contributes to the final assessment. Conveniently, this allows to differentiate between the concept of evidence relevance and its contribution to the final assessment. Our work was however limited in experiments with these coefficients, and we would like to analyze what they can learn, and how to inject features, such as satire assessment or source trustworthiness, through these coefficients in our future work.

Finally, it was shown that a hierarchical structure of our model allows making predictions on even finer language granularity, than the granularity the model was trained on. We believe the technique proposed in this chapter is transferable beyond fact-checking.

Chapter 8

Epilogue

The last chapter of the thesis focuses on summarizing the takeaways of this work in Section 8.1. In the end„ it carefully points out the limitations of our research and its conclusions, and possible future directions (Section 8.2). The overview of published research work is documented in Section 8.2.

8.1 Conclusion

The major takeaways from this thesis touch the topics such as the better understanding of objectives and complementarity underlying factoid QA, the problem of under-represented passages, for which the model is not being "interested-in" in retrieval, or learning of fine-grained rationales from coarse-grained data necessary to support fact-checkers in their work and persuade the ever-growing masses from believing the falsehoods by providing tools for an interpretable explanation.

Recapitulating back to rethinking the EQA objectives, we clearly demonstrated that the further usage of independent formulation of the span probability is unjustified. Using joint probability, possibly in a loss with compound objective, provides a span distribution that peaks around spans that are not a "trivially wrong answer"—i.e., a text spanning between considered positions, resulting in a nonsense (see Chapter 3 for definition). We also note this formulation is crucial for high-entropy span distributions, such as multi-span answer setting, or apriori answer sampling—sampling a potential answer to a not-yet-known question—in data augmentation, as supported by Lewis et al. (2021b). Finally, it was shown the joint formulation does not need to be inefficient and even demonstrated that a well-optimized dot product similarity works very well with transformer-based systems.

Moving towards R2-D2, we have shown exploiting the complementarity of the LRMs trained with a different objective, and pretrained on different datasets can achieve very strong performance. The combination of extractive and abstractive fusion is important in practice because some information, though captured within the model's parameters, can be missing from the explicit grounding data. So despite having a weaker extractive reader on TQ-Open wasn't originally our intention, having our classifier yield 26.6 % of answers as abstractive answers showcases, how extractive LRMs can fall back to large generative models. Furthermore, we've practically demonstrated that scores from the previous components, which basically come for free, are not to be thrown away, as using them sometimes increased, but never decreased the performance of the system.

Investigating the priors in current ODQA, we found that large ODQA datasets NQ-Open and TQ-Open do not require, and likely do not utilize, the large knowledge bases of unstructured documents they ought to work with. It was found, the majority (about two-thirds) of this knowledge can be pruned out, given the texts found in the training set, with no impact at all, and 92 % can be removed with degradation up to -3 EM. Then we analyzed the DPR embeddings (Karpukhin et al., 2020), claiming strong performance on these datasets, and found their statistics exhibit strong cues pointing at differences between the documents our pruning approach identified as relevant, and those it identified as non-relevant. Lastly, we have exploited this prior in the EfficientQA competition and unlike other teams, we achieved strong performance without the usage of dimensionality reduction or precision reduction techniques, such as quantization.

The work on RumourEval19 has shown that the stances of the conversations towards the implicit rumor can be efficiently detected when considering only the source post, and previous post of each stance. As demonstrated in subsequent work, this does not mean, that the other posts and metadata are not beneficial. However, 3 years after the competition, the best systems still achieve achieving less than 70 % F1 score on the task (Khandelwal, 2021). This could be connected with the ambiguity of the task setup itself, and its annotation procedure (reported annotator inter-agreement 76.2 %).

Lastly, Chapter 7 unveiled a LRM-independent approach, that allows determining the cues of relevance for the particular prediction up to a specific token's contribution towards it. Moreover, there is an explicit link between the relevance probability computation and the final veracity assessment. This approach is not only performing well across datasets, but we argue that it's also necessary for the system being useful to the fact-checkers, and thus interpretable, in practice.

8.2 Research Limitations and Future Directions

Metrics for EQA In Chapter 3, we've only considered automatic metrics for evaluation. A possible extension of this work is to turn towards human preferences and make humans rate the correctness, the soundness, and the form of the answers.

Large LRM Scaling In Chapter 3, we've considered BERT with 110M parameters and ALBERT 235M parameters from the transformers family. We haven't noticed lesser improvements for the larger ALBERT model. However, the trends can be different when considering e.g., 20B models or larger.

Tuning R2-D2's Hyperparameters R2-D2's hyperparameters on TQ-Open were not tuned. We presume this caused the degradation observed, especially for the extractive reader. Different work (Cheng et al., 2021b; Kedia et al., 2022) however showed substantially better performance of extractive reader on TQ-Open.

Retrieval Outside of Wikipedia The open-domain systems presented in this work were all trained and tested only on unstructured knowledge from Wikipedia. The language of Wikipedia is factual, has neutral sentiment, does not contain different views, opinions, or conflicting knowledge[1], and is well structured into chapters, where the first chapter is always a summary.

[1] At least not on purpose.

The web is a quite different domain. As expected based on findings from Chapter 5, semantic retrieval does not seem to perform well for knowledge-intensive tasks (including QA and FC)(Piktus et al., 2021).

As an example, in an ongoing work, we integrated Claim-Dissector (a Wikipedia-based dataset) with retrieval over news media. We found that the veracity assessments from the system are unusable because the system does not consider other factors important for relevance assessment in practice, such as source, its reliability, or its narrative. This is the area of active research, as human fact-checkers also need to deal with lies (Uscinski and Butler, 2013).

Data Biases Claim-Dissector, trained on FEVER, does not exhibit a strong relevance identification accuracy in certain real scenarios. On real data, the system often struggles to recognize what facts are refuting, and what is irrelevant, especially when applied out-of-domain. In the downstream application described in the point above, we tested claims such as the claim „*Weapons are being smuggled into Estonia*". Our system discovered an article with facts about „*Weapons being smuggled into Somalia*", and used it as main refuting evidence to predict REFUTE veracity.

Multilinguality Neither of the research conducted within this thesis was done outside of the English language. It is especially interesting for languages that follow *scriptio continua* with a south-Asian origin (Thai, Lao, Myanmar, Javanese, . . .), for which the classical retrieval approaches such as BM25, based on the lexical overlap, tend to perform poorly (Slávka, 2021; Asai et al., 2021). Another advantage is the sheer amount of information available across multilingual content. The initial indication for this is the work of my student Slávka (2021), which shows that even when using questions originated in English (dataset MKQA (Longpre et al., 2021)), the multilingual reader system tends to work better if retrieval is performed in all MKQA's languages, not just English.

Author Publications Related to This Thesis

The findings reported in this thesis were published at the ACL venues (EMNLP, NAACL, ACL?[2]), under Proceedings of Machine Learning (PMLR), or freely at arXiv. Core-2018 ranking[3] for these venues is A for EMNLP and NAACL, A* for ACL. PMLR itself does not have a core rank, but traditionally, workshops from NeurIPS (A*) publish their proceedings here.

1. Findings of Chapter 3 are published under **The 3rd Workshop on Machine Reading for Question Answering**[4] workshop organized at EMNLP 2021 (Fajcik et al., 2021c). Our paper also received the *Honorable Mention award.*

2. Findings of Chapter 4 are published (Fajcik et al., 2021b) at the main EMNLP 2021 conference under **Findings of the Association for Computational Linguistics: EMNLP 2021**.

3. Findings of Chapter 5 are presented at the EfficientQA@Neurips 2021 challenge, and published (Min et al., 2021) under **Proceedings of Machine Learning Research**. Our part in this work was further extended a published as a standalone paper freely available at arXiv (Fajcik et al., 2021a).

4. Findings of Chapter 6 are published under **International Workshop on Semantic Evaluation 2019** at NAACL 2019 (Fajcik et al., 2019).

5. Findings of Chapter 7 are published (Fajcik et al., 2023) at the main ACL 2023 conference under **Findings of the Association for Computational Linguistics: ACL 2023**.

[2] Final decision for CD will be available on 1.5.

[3] http://portal.core.edu.au/conf-ranks/. Accessed 20.3.2023.

[4] https://mrqa.github.io/. Accessed 20.3.2023.

Bibliography

Gabor Angeli and Christopher D. Manning. 2014. NaturalLI: Natural logic inference for common sense reasoning. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 534–545, Doha, Qatar. Association for Computational Linguistics. 82

Akari Asai, Matt Gardner, and Hannaneh Hajishirzi. 2022. Evidentiality-guided generation for knowledge-intensive NLP tasks. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2226–2243, Seattle, United States. Association for Computational Linguistics. 14, 84, 91

Akari Asai, Kazuma Hashimoto, Hannaneh Hajishirzi, Richard Socher, and Caiming Xiong. 2020. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. 46

Akari Asai, Xinyan Yu, Jungo Kasai, and Hanna Hajishirzi. 2021. One question answering model for many languages with cross-lingual dense passage retrieval. *Advances in Neural Information Processing Systems*, 34:7547–7560. 96

Isabelle Augenstein, Tim Rocktäschel, Andreas Vlachos, and Kalina Bontcheva. 2016. Stance detection with bidirectional conditional encoding. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 876–885, Austin, Texas. Association for Computational Linguistics. 73

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *ArXiv preprint*, abs/1607.06450. 27, 78

Hareesh Bahuleyan and Olga Vechtomova. 2017. UWaterloo at SemEval-2017 task 8: Detecting stance towards rumours with topic independent features. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 461–464, Vancouver, Canada. Association for Computational Linguistics. 72

Iz Beltagy, Matthew E. Peters, and Arman Cohan. 2020. Longformer: The long-document transformer. *ArXiv preprint*, abs/2004.05150. 73

James Bergstra, Dan Yamins, David D Cox, et al. 2013. Hyperopt: A python library for optimizing the hyperparameters of machine learning algorithms. In *Proceedings of the 12th Python in science conference*, volume 13, page 20. Citeseer. 27

Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer. 18, 122

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics. 10, 13, 14, 24, 36, 38, 52

Tianqi Chen, Bing Xu, Chiyuan Zhang, and Carlos Guestrin. 2016a. Training deep nets with sublinear memory cost. *ArXiv preprint*, abs/1604.06174. 22, 45

Weiling Chen, C. Yeo, Chiew Tong Lau, and Bu-Sung Lee. 2016b. Behavior deviation: An anomaly detection view of rumor preemption. *2016 IEEE 7th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 1–7. 66

Wenhu Chen, Hongmin Wang, Jianshu Chen, Yunkai Zhang, Hong Wang, Shiyang Li, Xiyou Zhou, and William Yang Wang. 2020. Tabfact: A large-scale dataset for table-based fact verification. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. 81

Yi-Chin Chen, Zhao-Yang Liu, and Hung-Yu Kao. 2017b. IKM at SemEval-2017 task 8: Convolutional neural networks for stance detection and rumor verification. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 465–469, Vancouver, Canada. Association for Computational Linguistics. 73

Hao Cheng, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2020. Probabilistic assumptions matter: Improved models for distantly-supervised document-level question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5657–5667, Online. Association for Computational Linguistics. 13, 24, 35, 36, 41, 48, 49, 52

Hao Cheng, Xiaodong Liu, Lis Pereira, Yaoliang Yu, and Jianfeng Gao. 2021a. Posterior differential regularization with f-divergence for improving model robustness. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1078–1089, Online. Association for Computational Linguistics. 24

Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, and Jianfeng Gao. 2021b. UnitedQA: A hybrid approach for open domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3080–3090, Online. Association for Computational Linguistics. 14, 46, 52, 95

Davide Chicco and Giuseppe Jurman. 2020. The advantages of the matthews correlation coefficient (mcc) over f1 score and accuracy in binary classification evaluation. *BMC genomics*, 21:1–13. 18

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for*

Computational Linguistics (Volume 1: Long Papers), pages 845–855, Melbourne, Australia. Association for Computational Linguistics. 28, 36, 41, 42, 52

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics. 84

Kevin Clark, Minh-Thang Luong, Quoc V. Le, and Christopher D. Manning. 2020. ELECTRA: pre-training text encoders as discriminators rather than generators. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. 20, 24, 36, 55

Anthony Cuthbertson. 2018. Robots can now read better than humans, putting millions of jobs at risk. 62

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, and Andrew McCallum. 2019. Multi-step retriever-reader interaction for scalable open-domain question answering. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA, May 6-9, 2019*. OpenReview.net. 35

Dorottya Demszky, Kelvin Guu, and Percy Liang. 2018. Transforming question answering datasets into natural language inference datasets. *arXiv preprint arXiv:1809.02922*. 84

Leon Derczynski, Kalina Bontcheva, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Arkaitz Zubiaga. 2017. SemEval-2017 task 8: RumourEval: Determining rumour veracity and support for rumours. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 69–76, Vancouver, Canada. Association for Computational Linguistics. 11, 15, 74, 75

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics. 13, 15, 19, 24, 27, 28, 29, 36, 70, 77, 118

Jiachen Du, Ruifeng Xu, Yulan He, and Lin Gui. 2017. Stance classification with target-specific neural attention networks. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence*, pages 3988–3994. International Joint Conferences on Artificial Intelligence. 73

Omar Enayet and Samhaa R. El-Beltagy. 2017. NileTMRG at SemEval-2017 task 8: Determining rumour and veracity support for rumours on Twitter. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 470–474, Vancouver, Canada. Association for Computational Linguistics. 15, 66, 72

Oren Etzioni. 2011. Search Needs a Shake-Up. *Nature*, 476(7358):25–26. 9

Martin Fajcik, Martin Docekal, Karel Ondrej, and Pavel Smrz. 2021a. Pruning the Index Contents for Memory Efficient Open-Domain QA. *ArXiv preprint*, abs/2102.10697. 14, 97

Martin Fajcik, Martin Docekal, Karel Ondrej, and Pavel Smrz. 2021b. R2-D2: A modular baseline for open-domain question answering. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 854–870, Punta Cana, Dominican Republic. Association for Computational Linguistics. 11, 13, 14, 37, 47, 61, 97

Martin Fajcik, Josef Jon, and Pavel Smrz. 2021c. Rethinking the Objectives of Extractive Question Answering. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 14–27, Punta Cana, Dominican Republic. Association for Computational Linguistics. 11, 12, 41, 97

Martin Fajcik, Motlicek Petr, and Pavel Smrz. 2023. Claim-Dissector: An Interpretable Fact-Checking System with Joint Re-ranking and Veracity Prediction. In *Findings of the Association for Computational Linguistics: ACL 2023*, Toronto, Canada. Association for Computational Linguistics. 12, 15, 97

Martin Fajcik, Pavel Smrz, and Lukas Burget. 2019. BUT-FIT at SemEval-2019 task 7: Determining the rumour stance with pre-trained deep bidirectional transformers. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1097–1104, Minneapolis, Minnesota, USA. Association for Computational Linguistics. 12, 14, 15, 71, 73, 75, 97

William Ferreira and Andreas Vlachos. 2016. Emergent: a novel data-set for stance classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1163–1168, San Diego, California. Association for Computational Linguistics. 15, 66

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics. 29, 130, 131

Marianela García Lozano, Hanna Lilja, Edward Tjörnhammar, and Maja Karasalo. 2017. Mama edha at SemEval-2017 task 8: Stance classification with CNN and rules. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 481–485, Vancouver, Canada. Association for Computational Linguistics. 73

Maria Glenski, Tim Weninger, and Svitlana Volkova. 2018. Identifying and understanding user reactions to deceptive and trusted social news sources. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 176–181, Melbourne, Australia. Association for Computational Linguistics. 66

Genevieve Gorrell, Elena Kochkina, Maria Liakata, Ahmet Aker, Arkaitz Zubiaga, Kalina Bontcheva, and Leon Derczynski. 2019. SemEval-2019 task 7: RumourEval, determining rumour veracity and support for rumours. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 845–854, Minneapolis, Minnesota, USA. Association for Computational Linguistics. 5, 12, 15, 66, 67, 73, 75, 132

Cyril Goutte and Eric Gaussier. 2005. A probabilistic interpretation of precision, recall and f-score, with implication for evaluation. In *European conference on information retrieval*, pages 345–359. Springer. 139

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *ArXiv preprint*, abs/2002.08909. 46

Andreas Hanselowski, Hao Zhang, Zile Li, Daniil Sorokin, Benjamin Schiller, Claudia Schulz, and Iryna Gurevych. 2018. UKP-athene: Multi-sentence textual entailment for claim verification. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 103–108, Brussels, Belgium. Association for Computational Linguistics. 76, 86

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2021. Debertav3: Improving deberta using electra-style pre-training with gradient-disentangled embedding sharing. 16, 21, 76, 82, 119

Dan Hendrycks and Kevin Gimpel. 2016. Gaussian error linear units (gelus). *ArXiv preprint*, abs/1606.08415. 19, 55, 78

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780. 72

Srinivasan Iyer, Sewon Min, Yashar Mehdad, and Wen-tau Yih. 2021. RECONSIDER: Improved re-ranking using span-focused cross-attention for open domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1280–1287, Online. Association for Computational Linguistics. 37, 52

Gautier Izacard and Edouard Grave. 2021a. Distilling knowledge from reader to retriever for question answering. In *ICLR 2021-9th International Conference on Learning Representations*. 38, 45, 50, 54

Gautier Izacard and Edouard Grave. 2021b. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics. 4, 14, 21, 22, 38, 39, 46, 52, 53, 57, 91, 125

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2022. Few-shot Learning with Retrieval Augmented Language Models. *ArXiv preprint*, abs/2208.03299. 14, 46, 52, 53

Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Sebastian Riedel, and Edouard Grave. 2020. A memory efficient baseline for open domain question answering. *ArXiv preprint*, abs/2012.15156. 46, 52, 62, 64

Eric Jang, Shixiang Gu, and Ben Poole. 2017. Categorical reparametrization with gumble-softmax. In *International Conference on Learning Representations (ICLR 2017)*. OpenReview. net. 81, 119

Youngjin Jang and Harksoo Kim. 2020. Document re-ranking model for machine-reading and comprehension. *Applied Sciences*, 10(21). 52

Robin Jia and Percy Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2021–2031, Copenhagen, Denmark. Association for Computational Linguistics. 29, 130

Kelvin Jiang, Ronak Pradeep, and Jimmy Lin. 2021. Exploring listwise evidence reasoning with t5 for fact verification. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 402–410, Online. Association for Computational Linguistics. 15, 75, 76, 79, 81, 85, 86, 87, 127

Yichen Jiang, Shikha Bordia, Zheng Zhong, Charles Dognin, Maneesh Singh, and Mohit Bansal. 2020. HoVer: A dataset for many-hop fact extraction and claim verification. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 3441–3460, Online. Association for Computational Linguistics. 15, 75, 81, 84, 134

Zhengbao Jiang, Luyu Gao, Zhiruo Wang, Jun Araki, Haibo Ding, Jamie Callan, and Graham Neubig. 2022. Retrieval as attention: End-to-end learning of retrieval and reading within a single transformer. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 2336–2349, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 14, 46, 53

Zhiwei Jin, Juan Cao, Yongdong Zhang, and Jiebo Luo. 2016. News verification by exploiting conflicting social viewpoints in microblogs. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2972–2978. AAAI Press. 66

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77. 20, 21

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 2017. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics. 14, 30, 43, 131

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics. 14, 19, 38, 39, 41, 42, 43, 44, 46, 52, 54, 55, 64, 91, 95, 118

Akhil Kedia, Mohd Abbas Zaidi, and Haejun Lee. 2022. FiE: Building a global probability space by leveraging early fusion in encoder for open-domain question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4246–4260, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 14, 46, 52, 95

Anant Khandelwal. 2021. Fine-tune longformer for jointly predicting rumor stance and veracity. In *Proceedings of the 3rd ACM India Joint International Conference on Data Science & Management of Data (8th ACM IKDD CODS & 26th COMAD)*, pages 10–19. 15, 72, 73, 95

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021a. Baleen: Robust multi-hop reasoning at scale via condensed retrieval. *Advances in Neural Information Processing Systems*, 34. 15, 75, 90, 91

Omar Khattab, Christopher Potts, and Matei Zaharia. 2021b. Relevance-guided supervision for OpenQA with ColBERT. *Transactions of the Association for Computational Linguistics*, 9:929–944. 24, 38, 46, 54

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*. 70

Bret Kinsella. 2020. Streaming music, questions, weather, timers and alarms remain smart speaker killer apps, third-party voice app usage not growing. 9

Elena Kochkina, Maria Liakata, and Isabelle Augenstein. 2017. Turing at SemEval-2017 task 8: Sequential approach to rumour stance classification with branch-LSTM. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 475–480, Vancouver, Canada. Association for Computational Linguistics. 72

Amrith Krishna, Sebastian Riedel, and Andreas Vlachos. 2022. Proofver: Natural logic theorem proving for fact verification. *Transactions of the Association for Computational Linguistics*, 10:1013–1030. 82, 87

Kalpesh Krishna, Aurko Roy, and Mohit Iyyer. 2021. Hurdles to progress in long-form question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4940–4957, Online. Association for Computational Linguistics. 9

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur 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 Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466. 8, 14, 30, 43, 84, 130

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A lite BERT for self-supervised learning of language representations. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net. 20, 24, 28, 29, 36

Haejun Lee, Akhil Kedia, Jongwon Lee, Ashwin Paranjape, Christopher Manning, and Kyoung-Gu Woo. 2022. You only need one model for open-domain question answering. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 3047–3060, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics. 14, 43, 46, 52, 57

Jinhyuk Lee, Seongjun Yun, Hyunjae Kim, Miyoung Ko, and Jaewoo Kang. 2018. Ranking paragraphs for improving answer recall in open-domain question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 565–569, Brussels, Belgium. Association for Computational Linguistics. 51

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, Florence, Italy. Association for Computational Linguistics. 9, 10, 24, 27, 138

Kenton Lee, Shimi Salant, Tom Kwiatkowski, Ankur Parikh, Dipanjan Das, and Jonathan Berant. 2016. Learning recurrent span representations for extractive question answering. *ArXiv preprint*, abs/1611.01436. 24, 26, 36

Stephan Lewandowsky, Ullrich KH Ecker, Colleen M Seifert, Norbert Schwarz, and John Cook. 2012. Misinformation and its correction: Continued influence and successful debiasing. *Psychological science in the public interest*, 13(3):106–131. 12, 15, 75, 138

Patrick Lewis, Pontus Stenetorp, and Sebastian Riedel. 2021a. Question and answer test-train overlap in open-domain question answering datasets. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 1000–1008, Online. Association for Computational Linguistics. 44, 56, 63, 125

Patrick Lewis, Yuxiang Wu, Linqing Liu, Pasquale Minervini, Heinrich Küttler, Aleksandra Piktus, Pontus Stenetorp, and Sebastian Riedel. 2021b. PAQ: 65 million probably-asked questions and what you can do with them. *Transactions of the Association for Computational Linguistics*, 9:1098–1115. 61, 94

Patrick S. H. Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. 14, 46, 52

Quanzhi Li, Qiong Zhang, and Luo Si. 2019. eventAI at SemEval-2019 task 7: Rumor detection on social media by exploiting content, user credibility and propagation information. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 855–859, Minneapolis, Minnesota, USA. Association for Computational Linguistics. 12, 15, 66, 74, 75

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. 2018. Denoising distantly supervised open-domain question answering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1736–1745, Melbourne, Australia. Association for Computational Linguistics. 52

Zhouhan Lin, Minwei Feng, Cícero Nogueira dos Santos, Mo Yu, Bing Xiang, Bowen Zhou, and Yoshua Bengio. 2017. A structured self-attentive sentence embedding. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. 70

Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. 2015. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1867–1870. ACM. 66

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 preprint*, abs/1907.11692. 16, 20, 44, 76, 92

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics. 16, 76, 82, 87

Shayne Longpre, Yi Lu, and Joachim Daiber. 2021. MKQA: A linguistically diverse benchmark for multilingual open domain question answering. *Transactions of the Association for Computational Linguistics*, 9:1389–1406. 96

Ilya Loshchilov and Frank Hutter. 2017. Fixing weight decay regularization in adam. *ArXiv preprint*, abs/1711.05101. 27, 43, 55, 119

Yi Luan, Jacob Eisenstein, Kristina Toutanova, and Michael Collins. 2021. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*. 13, 14, 38, 39, 52, 54

Jing Ma, Wei Gao, Shafiq Joty, and Kam-Fai Wong. 2019. Sentence-level evidence embedding for claim verification with hierarchical attention networks. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2561–2571, Florence, Italy. Association for Computational Linguistics. 15, 75, 86, 87

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021a. Generation-augmented retrieval for open-domain question answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4089–4100, Online. Association for Computational Linguistics. 46

Yuning Mao, Pengcheng He, Xiaodong Liu, Yelong Shen, Jianfeng Gao, Jiawei Han, and Weizhu Chen. 2021b. Reader-guided passage reranking for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 344–350, Online. Association for Computational Linguistics. 46, 52

Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. 2010. Twitter under crisis: Can we trust what we rt? In *Proceedings of the first workshop on social media analytics*, pages 71–79. 66

Sewon Min, Jordan Boyd-Graber, Chris Alberti, Danqi Chen, Eunsol Choi, Michael Collins, Kelvin Guu, Hannaneh Hajishirzi, Kenton Lee, Jennimaria Palomaki, Colin Raffel, Adam Roberts, Tom Kwiatkowski, Patrick Lewis, Yuxiang Wu, Heinrich Küttler, Linqing Liu, Pasquale Minervini, Pontus Stenetorp, Sebastian Riedel, Sohee Yang, Minjoon Seo, Gautier Izacard, Fabio Petroni, Lucas Hosseini, Nicola De Cao, Edouard Grave, Ikuya Yamada, Sonse Shimaoka, Masatoshi Suzuki, Shumpei

Miyawaki, Shun Sato, Ryo Takahashi, Jun Suzuki, Martin Fajcik, Martin Docekal, Karel Ondrej, Pavel Smrz, Hao Cheng, Yelong Shen, Xiaodong Liu, Pengcheng He, Weizhu Chen, Jianfeng Gao, Barlas Oguz, Xilun Chen, Vladimir Karpukhin, Stan Peshterliev, Dmytro Okhonko, Michael Schlichtkrull, Sonal Gupta, Yashar Mehdad, and Wen-tau Yih. 2021. Neurips 2020 efficientqa competition: Systems, analyses and lessons learned. In *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, volume 133 of *Proceedings of Machine Learning Research*, pages 86–111. PMLR. 5, 7, 10, 14, 43, 54, 55, 61, 62, 63, 97

Sewon Min, Danqi Chen, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. A discrete hard EM approach for weakly supervised question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2851–2864, Hong Kong, China. Association for Computational Linguistics. 46

Sewon Min, Danqi Chen, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Knowledge guided text retrieval and reading for open domain question answering. *ArXiv preprint*, abs/1911.03868. 46

Sewon Min, Julian Michael, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2020. AmbigQA: Answering ambiguous open-domain questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5783–5797, Online. Association for Computational Linguistics. 14, 46, 52, 84, 91

Takeru Miyato, Shin-ichi Maeda, Masanori Koyama, and Shin Ishii. 2018. Virtual adversarial training: a regularization method for supervised and semi-supervised learning. *IEEE transactions on pattern analysis and machine intelligence*, 41(8):1979–1993. 21

Saif Mohammad, Svetlana Kiritchenko, Parinaz Sobhani, Xiaodan Zhu, and Colin Cherry. 2016. SemEval-2016 task 6: Detecting stance in tweets. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 31–41, San Diego, California. Association for Computational Linguistics. 72

Saif Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. NRC-Canada: Building the state-of-the-art in sentiment analysis of tweets. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, pages 321–327, Atlanta, Georgia, USA. Association for Computational Linguistics. 72

Mitra Mohtarami, Ramy Baly, James Glass, Preslav Nakov, Lluís Màrquez, and Alessandro Moschitti. 2018. Automatic stance detection using end-to-end memory networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 767–776, New Orleans, Louisiana. Association for Computational Linguistics. 73

Preslav Nakov, David P. A. Corney, Maram Hasanain, Firoj Alam, Tamer Elsayed, Alberto Barr'on-Cedeno, Paolo Papotti, Shaden Shaar, and Giovanni Da San Martino. 2021. Automated fact-checking for assisting human fact-checkers. In *International Joint Conference on Artificial Intelligence*. 11

Yixin Nie, Haonan Chen, and Mohit Bansal. 2019. Combining fact extraction and verification with neural semantic matching networks. In *The Thirty-Third AAAI Conference on Artificial Intelligence, AAAI 2019, The Thirty-First Innovative Applications of Artificial Intelligence Conference, IAAI 2019, The Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2019, Honolulu, Hawaii, USA, January 27 - February 1, 2019*, pages 6859–6866. AAAI Press. 81, 87

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *ArXiv preprint*, abs/1901.04085. 13, 39, 52

Rodrigo Nogueira, Zhiying Jiang, Ronak Pradeep, and Jimmy Lin. 2020. Document ranking with a pretrained sequence-to-sequence model. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 708–718, Online. Association for Computational Linguistics. 14, 52

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. 2019. Multi-stage document ranking with BERT. *ArXiv preprint*, abs/1910.14424. 14, 51

Jungsoo Park, Sewon Min, Jaewoo Kang, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2022. FaVIQ: FAct verification from information-seeking questions. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5154–5166, Dublin, Ireland. Association for Computational Linguistics. 7, 15, 16, 75, 81, 84, 91, 133

Razvan Pascanu, Tomás Mikolov, and Yoshua Bengio. 2013. On the difficulty of training recurrent neural networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, volume 28 of *JMLR Workshop and Conference Proceedings*, pages 1310–1318. JMLR.org. 119

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 8024–8035. 27, 43, 55

James W Pennebaker, Martha E Francis, and Roger J Booth. 2001. Linguistic inquiry and word count (liwc): Liwc2001 manual. 72

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics. 28

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics. 81

Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Dmytro Okhonko, Samuel Broscheit, Gautier Izacard, Patrick Lewis, Barlas Oguz, Edouard Grave, Wen-tau Yih, and Sebastian Riedel. 2021. The web is your oyster - knowledge-intensive NLP against a very large web corpus. *CoRR*, abs/2112.09924. 96

Kashyap Popat, Subhabrata Mukherjee, Andrew Yates, and Gerhard Weikum. 2018. DeClarE: Debunking fake news and false claims using evidence-aware deep learning. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 22–32, Brussels, Belgium. Association for Computational Linguistics. 16, 76, 82

Ronak Pradeep, Xueguang Ma, Rodrigo Nogueira, and Jimmy Lin. 2021a. Scientific claim verification with VerT5erini. In *Proceedings of the 12th International Workshop on Health Text Mining and Information Analysis*, pages 94–103, online. Association for Computational Linguistics. 81

Ronak Pradeep, Xueguang Ma, Rodrigo Nogueira, and Jimmy Lin. 2021b. Vera: Prediction techniques for reducing harmful misinformation in consumer health search. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2066–2070. 81

Rob Procter, Farida Vis, and Alex Voss. 2013. Reading the riots on twitter: methodological innovation for the analysis of big data. *International journal of social research methodology*, 16(3):197–214. 66

Vahed Qazvinian, Emily Rosengren, Dragomir R. Radev, and Qiaozhu Mei. 2011. Rumor has it: Identifying misinformation in microblogs. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1589–1599, Edinburgh, Scotland, UK. Association for Computational Linguistics. 66

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of BERT in ranking. *ArXiv preprint*, abs/1904.07531. 51

Alec Radford, Karthik Narasimhan, Tim Salimans, Ilya Sutskever, et al. 2018. Improving language understanding by generative pre-training. 73

Karthik Radhakrishnan, Tushar Kanakagiri, Sharanya Chakravarthy, and Vidhisha Balachandran. 2020. “a little birdie told me ... ” - inductive biases for rumour stance detection on social media. In *Proceedings of the Sixth Workshop on Noisy User-generated Text (W-NUT 2020)*, pages 244–248, Online. Association for Computational Linguistics. 72, 73

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67. 21, 22, 45, 84

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don’t know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics. 29, 129

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics. 10, 19, 24, 29, 62, 129

Dongning Rao, Xin Miao, Zhihua Jiang, and Ran Li. 2021. Stanker: Stacking network based on level-grained attention-masked bert for rumor detection on social media. In *Conference on Empirical Methods in Natural Language Processing*. 15

Adam Roberts, Colin Raffel, and Noam Shazeer. 2020. How much knowledge can you pack into the parameters of a language model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5418–5426, Online. Association for Computational Linguistics. 46, 52

Stephen Robertson and Hugo Zaragoza. 2009. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, 3(4):333–389. 76

Carolina Scarton, Diego Silva, and Kalina Bontcheva. 2020. Measuring what counts: The case of rumour stance classification. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing*, pages 925–932, Suzhou, China. Association for Computational Linguistics. 15

Michael Sejr Schlichtkrull, Vladimir Karpukhin, Barlas Oguz, Mike Lewis, Wen-tau Yih, and Sebastian Riedel. 2021. Joint verification and reranking for open fact checking over tables. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 6787–6799, Online. Association for Computational Linguistics. 15, 75, 79, 82, 138

Tal Schuster, Adam Fisch, and Regina Barzilay. 2021. Get your vitamin C! robust fact verification with contrastive evidence. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 624–643, Online. Association for Computational Linguistics. 81, 82

Tal Schuster, Darsh Shah, Yun Jie Serene Yeo, Daniel Roberto Filizzola Ortiz, Enrico Santus, and Regina Barzilay. 2019. Towards debiasing fact verification models. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3419–3425, Hong Kong, China. Association for Computational Linguistics. 84

Colleen M Seifert. 2002. The continued influence of misinformation in memory: What makes a correction effective? In *Psychology of learning and motivation*, volume 41, pages 265–292. Elsevier. 138

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1715–1725, Berlin, Germany. Association for Computational Linguistics. 20

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. 13, 24, 28, 29, 36

Minjoon Seo, Jinhyuk Lee, Tom Kwiatkowski, Ankur Parikh, Ali Farhadi, and Hannaneh Hajishirzi. 2019. Real-time open-domain question answering with dense-sparse phrase index. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4430–4441, Florence, Italy. Association for Computational Linguistics. 54

Darsh Shah, Tal Schuster, and Regina Barzilay. 2020. Automatic fact-guided sentence modification. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 8791–8798. 81

Devendra Singh, Siva Reddy, Will Hamilton, Chris Dyer, and Dani Yogatama. 2021. End-to-end training of multi-document reader and retriever for open-domain question answering. *Advances in Neural Information Processing Systems*, 34:25968–25981. 46, 53

Vikram Singh, Sunny Narayan, Md Shad Akhtar, Asif Ekbal, and Pushpak Bhattacharyya. 2017. IITP at SemEval-2017 task 8 : A supervised approach for rumour evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 497–501, Vancouver, Canada. Association for Computational Linguistics. 72

Michal Slávka. 2021. Multilingual open-domain question answering. Diplomová práce, Vysoké učení technické v Brně, Fakulta informačních technologií. 96

Amir Soleimani, Christof Monz, and Marcel Worring. 2020. Bert for evidence retrieval and claim verification. In *European Conference on Information Retrieval*, pages 359–366. Springer. 81

Ankit Srivastava, Georg Rehm, and Julian Moreno Schneider. 2017. DFKI-DKT at SemEval-2017 task 8: Rumour detection and classification using cascading heuristics. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 486–490, Vancouver, Canada. Association for Computational Linguistics. 73

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958. 78

Dominik Stammbach. 2021. Evidence selection as a token-level prediction task. In *Proceedings of the Fourth Workshop on Fact Extraction and VERification (FEVER)*, pages 14–20, Dominican Republic. Association for Computational Linguistics. 15, 75, 82, 87

Dominik Stammbach and Guenter Neumann. 2019. Team DOMLIN: Exploiting evidence enhancement for the FEVER shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 105–109, Hong Kong, China. Association for Computational Linguistics. 76, 77, 81, 86

Shyam Subramanian and Kyumin Lee. 2020. Hierarchical Evidence Set Modeling for automated fact extraction and verification. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7798–7809, Online. Association for Computational Linguistics. 76, 82, 87

Elior Sulem, Jamaal Hay, and Dan Roth. 2022. Yes, no or IDK: The challenge of unanswerable yes/no questions. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1075–1085, Seattle, United States. Association for Computational Linguistics. 4, 11

James Thorne, Max Glockner, Gisela Vallejo, Andreas Vlachos, and Iryna Gurevych. 2021. Evidence-based verification for real world information needs. *ArXiv preprint*, abs/2104.00640. 16, 75, 81, 84, 86, 92, 134

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics. 7, 8, 12, 15, 16, 75, 81, 82, 133

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics. 30, 131

Alan Mathison Turing. 1950. Mind. *Mind*, 59(236):433–460. 62

Joseph E Uscinski and Ryden W Butler. 2013. The epistemology of fact checking. *Critical Review*, 25(2):162–180. 96

Laurens Van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of machine learning research*, 9(11). 59

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 5998–6008. 9, 19, 77, 78

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pages 2692–2700. 36

Ellen M Voorhees et al. 2001. Overview of the trec 2001 question answering track. In *Trec*, pages 42–51. 10, 38

Feixiang Wang, Man Lan, and Yuanbin Wu. 2017. ECNU at SemEval-2017 task 8: Rumour evaluation using effective features and supervised ensemble models. In *Proceedings of the 11th International Workshop on Semantic Evaluation*

(SemEval-2017), pages 491–496, Vancouver, Canada. Association for Computational Linguistics. 72

Shuohang Wang and Jing Jiang. 2017. Machine comprehension using match-lstm and answer pointer. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. 10, 12, 24, 26, 36

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerry Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced ranker-reader for open-domain question answering. In *AAAI*, pages 5981–5988. 51

Xi Wang, Weisen Feng, and Fei Wang. 2021. Determining the rumour stance with ensemble method based on bsaf model. In *Proceedings of the 4th International Conference on Computer Science and Software Engineering*, pages 6–12. 15, 72, 73

Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, and Bing Xiang. 2019. Multi-passage BERT: A globally normalized BERT model for open-domain question answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5878–5882, Hong Kong, China. Association for Computational Linguistics. 51, 53

Penghui Wei, Nan Xu, and Wenji Mao. 2019. Modeling conversation structure and temporal dynamics for jointly predicting rumor stance and veracity. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 4787–4798, Hong Kong, China. Association for Computational Linguistics. 66

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural QA as simple as possible but not simpler. In *Proceedings of the 21st Conference on Computational Natural Language Learning (CoNLL 2017)*, pages 271–280, Vancouver, Canada. Association for Computational Linguistics. 24

Cynthia Whissell. 2009. Using the revised dictionary of affect in language to quantify the emotional undertones of samples of natural language. *Psychological reports*, 105(2):509–521. 72

A Wierzbicka. 1987. English act verbs: a semantic dictionary. 72

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics. 82, 119

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System*

Demonstrations, pages 38–45, Online. Association for Computational Linguistics. 22, 27, 43, 55, 119

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. 2016. Google's neural machine translation system: Bridging the gap between human and machine translation. *ArXiv preprint*, abs/1609.08144. 19, 68

Ji Xin, Rodrigo Nogueira, Yaoliang Yu, and Jimmy Lin. 2020. Early exiting BERT for efficient document ranking. In *Proceedings of SustaiNLP: Workshop on Simple and Efficient Natural Language Processing*, pages 83–88, Online. Association for Computational Linguistics. 52

Caiming Xiong, Victor Zhong, and Richard Socher. 2017. Dynamic coattention networks for question answering. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net. 13, 24, 36

Caiming Xiong, Victor Zhong, and Richard Socher. 2018. DCN+: mixed objective and deep residual coattention for question answering. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. 36

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul Bennett, Junaid Ahmed, and Arnold Overwijk. 2020. Approximate nearest neighbor negative contrastive learning for dense text retrieval. *ArXiv preprint*, abs/2007.00808. 38

Lee Xiong, Chenyan Xiong, Ye Li, Kwok-Fung Tang, Jialin Liu, Paul N Bennett, Junaid Ahmed, and Arnold Overwijk. 2021a. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *International Conference on Learning Representations*. 52

Wenhan Xiong, Hong Wang, and William Yang Wang. 2021b. Progressively pretrained dense corpus index for open-domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2803–2815, Online. Association for Computational Linguistics. 46, 54

Ruoyao Yang, Wanying Xie, Chunhua Liu, and Dong Yu. 2019a. BLCU_NLP at SemEval-2019 task 7: An inference chain-based GPT model for rumour evaluation. In *Proceedings of the 13th International Workshop on Semantic Evaluation*, pages 1090–1096, Minneapolis, Minnesota, USA. Association for Computational Linguistics. 66, 72, 73

Sohee Yang and Minjoon Seo. 2020. Is retriever merely an approximator of reader? *ArXiv preprint*, abs/2010.10999. 46

Sohee Yang and Minjoon Seo. 2021. Designing a minimal retrieve-and-read system for open-domain question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 5856–5865, Online. Association for Computational Linguistics. 61, 64

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019b. End-to-end open-domain question answering with BERTserini. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics (Demonstrations)*, pages 72–77, Minneapolis, Minnesota. Association for Computational Linguistics. 52, 53

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019c. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pages 5754–5764. 24, 36

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics. 84

Wenpeng Yin and Dan Roth. 2018. TwoWingOS: A two-wing optimization strategy for evidential claim verification. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 105–114, Brussels, Belgium. Association for Computational Linguistics. 87

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. 2018. Qanet: Combining local convolution with global self-attention for reading comprehension. In *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net. 13, 24, 28, 36

Zhe Zhao, Paul Resnick, and Qiaozhu Mei. 2015. Enquiring minds: Early detection of rumors in social media from enquiry posts. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 1395–1405. ACM. 66

Wanjun Zhong, Jingjing Xu, Duyu Tang, Zenan Xu, Nan Duan, Ming Zhou, Jiahai Wang, and Jian Yin. 2020. Reasoning over semantic-level graph for fact checking. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6170–6180, Online. Association for Computational Linguistics. 81, 87

Jie Zhou, Xu Han, Cheng Yang, Zhiyuan Liu, Lifeng Wang, Changcheng Li, and Maosong Sun. 2019. GEAR: Graph-based evidence aggregating and reasoning for fact verification. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 892–901, Florence, Italy. Association for Computational Linguistics. 81

Arkaitz Zubiaga, Maria Liakata, Rob Procter, Geraldine Wong Sak Hoi, and Peter Tolmie. 2016. Analysing how people orient to and spread rumours in social media by looking at conversational threads. *PloS one*, 11(3):e0150989. 11, 15, 66, 74, 75

Appendices

List of Appendices

Appendix A

Hyperparameters & Preprocessing

A.1 Hyperparameters for the Objectives of Extractive QA

The exact hyperparameters used in this work are documented in our code. We note that for BERT and ALBERT, we simply followed the hyperparameters proposed by the authors for SQuADv1.1. In the case of LRM models, each input context is split into windows as proposed by Devlin et al. (2019). Each input sequence has a maximum length of 384, questions are truncated to 64 tokens, and context is split with overlap stride 128. For SQuADv2.0, we follow BERT's approach for computing the no-answer logit in test-time. Having the set of k windows W_e for each example e, we compute the null-score $ns_w = \text{logit}P(a_s = 0) + \text{logit}P(a_e = 0)$ for each window $w \in W_e$. For joint and compound objectives $ns_w = \text{logit}P(a_s = 0, a_e = 0)$. Defining that for each window w the best non-null answer logit is a_w, the no-answer logit is then given by the difference of lowest null-score and best-answer score $\Gamma = \min_{w \in W_e}(ns_w) - \max_{w \in W_e}(a_w)$ among all windows of example e. The threshold for Γ is determined on the validation data via an official evaluation script.

A.2 Data Pre-processing in R2-D2

This section describes how the training datasets for the reranker and extractive reader are filtered, and how the distant supervision labeling is generated. Note not each example contains the golden passage, as not each example can be mapped to the used dump of Wikipedia. We use the same golden passage mapping as Karpukhin et al. (2020).

For passage reranking, the input must contain at least one positive example. We meet this condition either by adding a golden passage or searching for the passage with an answer in the top 400 results retrieved by DPR. In detail about the search, first, the Simple tokenizer proposed in DrQA[1] tokenizes each passage and golden answer. The positive example is the best-scored tokenized passage that contains an exact match with one of the tokenized answers. Note the search proceeds in the same way as in DPR's Accuracy@K implementation[2].

The extractive reader is trained only on samples that contain an exact match to at least one of the annotated answers in the top-1 passage, or golden passage if it is available. The exact match is performed on the subword token level (i.e. in ELECTRA's tokenization).

[1] https://github.com/facebookresearch/DrQA

[2] https://github.com/facebookresearch/DPR

Next, the span annotations are extracted from the passages at the reader's input. Note each sample may contain multiple answers. The annotations for each answer in each sample are obtained differently in retrieved passages and in the golden passage. For retrieved passages, we search for the answer's exact matches in passages and use each match as target annotation. For the golden passage, we also search for the answer's exact matches in it. If there is none, the answer is soft-matched with a single sub-sequence of golden passage, which yields the highest non-zero F1 score. The F1 soft match is also performed on the subword token level. Therefore answers with zero highest F1 soft match with a golden passage and no exact match in any of the reader's input passages are discarded.

A.3 Hyperparameters & Data Pre-processing in Claim-Dissector

We base our implementation of pretrained language representation models on Huggingface (Wolf et al., 2020). Unless said otherwise, we employ DeBERTaV3 (He et al., 2021) as LRM. In all experiments, we firstly pretrain the model on MNLI (Williams et al., 2018). While we observed no significant improvement when using a MNLI-pretrained checkpoint, we found that without MNLI pretraining, our framework sometimes converges to poor performance. We train the model on FEVER with minibatch size 64, learning rate $5e-6$, and maximum block-length $L_x = 500$. We schedule a linear warmup of the learning rate for first 100 steps and then keep a constant learning rate. We use Adam with decoupled weight decay (Loshchilov and Hutter, 2017) and clip gradient vectors to a maximal norm of 1 (Pascanu et al., 2013). In all experiments, the model is trained and evaluated in mixed precision. We keep $\lambda_R = \lambda_2 = 1$. We use 8x Nvidia A100 40GB GPUs for training. We validate our model at every 500-th step and select the best checkpoint according to FEVER-Score (see Subsection 7.3.2). We have not used any principled way to tune the hyperparameters.

To train the model with SSE, we decrease the strength of block-level supervised $\mathcal{L}_R$ objective to $\lambda_R = 0.7$. We switch between vanilla objective and SSE objective randomly on a per-sample basis. Training starts with replacement probability $p_{sse} = 0.$ for first $1,000$ steps. The probability is then linearly increased up to $p_{sse} = 0.95$ on step $3,000$, after which it is left constant.

All results except for Table 7.7 and Table 7.8 were early-stopped based on the best FS. For Table 7.7, we report the best result for each metric early-stopped independently, to be comparable with ablations where FS was not available. For Table 7.8, we record best F1 during training.

Masker is implemented using DeBERTaV3 architecture. We keep most hyperparameters the same as for Claim-Dissector. The only difference is learning rate $2e-6$, adaptive scheduling on Gumbel-softmax (Jang et al., 2017) temperature τ, and training model/-masker on different dataset split. Training starts with temperature $\tau = 1$ and after initial 100 steps, it is linearly decreasing towards $\tau = 0.1$ at step 700. Then we switch to hard Gumbel-softmax—sampling 1-hot vectors in the forward pass while computing gradients as we would use a soft sample with $\tau = 0.1$ at the backward pass. We randomly split the training set and we train the model on 75 % of the data, and masker on the remaining 25 % of the data.

Appendix B

Released Code

All code released within this dissertation is documented within Table B.1.

B.1 Inference Speed of R2-D2's Implementation

While optimizing the R2-D2's inference speed was not the main focus of this paper, we show that even our unoptimized implementation can be used in practice in a small scale. We analyze the speed of our implementation on NQ-Open test data in Table B.2. The times were measured on a workstation with Intel Xeon Silver 4214 48-core CPU, 188GB RAM, and Nvidia 2080Ti 12GB GPU. Table columns show settings with and without passage reranker. Table rows are split into two parts; *intermediate* rows show time spent by the pipeline's single component (e.g., row ext. reader shows what time the pipeline spent by running just the ext. reader), and *total* rows show the total time taken by the whole pipeline. The retriever and reranker infer with batch sizes 32 and 100 respectively, and the readers run with minibatch size 1.

We note that in retrieval, we do not use any approximate K-NN algorithm to facilitate retrieval of top-K nearest passages and instead do the dot product with the matrix of passages directly on the CPU. Secondly, we note that we do not parallelize the inference of generative reader and extractive reader. Thirdly, notice the difference in the extractive reader's speed with and without passage reranker is caused by its different input size (see details of extractive reader's experiments setup in subsection 4.2.3). Finally, we compare

Project	Location	What is available
Rethinking the Objectives of Extractive QA	https://github.com/KNOT-FIT-BUT/JointSpanExtraction	Code, manual annotations from analysis, all result measurements.
R2-D2	https://github.com/KNOT-FIT-BUT/R2-D2	Code, checkpoints, preprocessed data.
R2-D2 Pruning	https://github.com/KNOT-FIT-BUT/scalingQA	Code, checkpoints, preprocessed data, prior relevance probabilities.
Stance Detection for RumourEval	https://github.com/MFajcik/RumourEval2019	Code, introspections
Claim-Dissector	https://github.com/KNOT-FIT-BUT/ClaimDissector	Code, checkpoints, preprocessed data, manual analyses

Table B.1: Code Availability.

the speed of our approach using FiD with 25 and 100 input passages, like in the original FiD implementation[1]. The ratios of our measurements are compared explicitly in Table B.3.

	Modules	**Rankers**	
		retriever	+reranker
intermediate	retriever	0.21	0.21
	passage reranker	-	1.94
	ext. reader	2.21	0.35
	gen. reader (25)	0.55	0.55
	answer reranker (25)	3.11	3.11
	gen. reader (100)	1.85	-
	answer reranker (100)	11.67	-
total	ext	2.41	2.19
	gen (25)	0.76	2.70
	ext+gen (25)	6.08	6.16
	gen (100)	2.06	-
	ext+gen (100)	15.94	-

Table B.2: Inference times on NQ-Open in seconds per question. See text for details.

Setup ratios	**Modules**			
	only gen.	ans. reranker	gen pipe.	ext+gen pipe.
gen(100) / gen(25)	3.36x	3.75x	2.71x	2.62x
rr+gen(25) / gen(25)	*1.00x	*1.00x	3.55x	1.01x
gen(100) / rr+gen(25)	*3.36x	*3.75x	0.76x	2.58x

Table B.3: Ratios of inference times on NQ-Open. First, two columns compare the speed in the stage of generating abstractive answer (only gen.) and answer reranking (ans. reranker). The subsequent columns compare the speed of the whole pipeline just with a generative reader and no component fusion (gen pipe.) and full R2-D2 pipeline (ext+gen pipe.). Row gen(100)/gen(25) compares the speedup of the pipeline when using just 25 passages in FiD's input (denoted as gen(25)) instead of 100 (denoted as gen(100)). Row rr+gen(25)/gen(25) shows speedup gained from not using passage reranker (denoted as rr). Row gen(100)/rr+gen(25) compares the speed of using rr and gen(25) instead of gen(100) (with no passage reranking). Results marked with * are not affected by the passage reranking component, as they only measure the speed of the pipeline's individual component. For instance, the table shows that doing an answer reranking with a generative reader with just 25 passages at its input runs 3.75x faster than doing an answer reranking with a generative reader that uses 100 passages.

[1] We simply pass 100 input passages to the model trained with 25 passages in the experiment.

Appendix C

Proofs

C.1 Independent Marginalization Sums All Start-End Combinations

The start and end components in the extractive reader's loss from R2-D2 (see equation 4.8) perform summation over all start-end combinations (even inter-passage combinations). This can be trivially shown as

$$\begin{aligned} -\log \sum_{s \in S} \mathrm{P}_{start}(s) - \log \sum_{e \in E} \mathrm{P}_{end}(e) = \\ = -\log \sum_{s \in S} \mathrm{P}_{start}(s) \sum_{e \in E} \mathrm{P}_{end}(e) = \\ = -\log \sum_{s \in S, e \in E} \mathrm{P}_{start}(s)\, \mathrm{P}_{end}(e)\,. \end{aligned} \tag{C.1}$$

C.2 Logit Proof

The link between equation 7.3 and equation 7.6 can be easily proved as follows. Applying logarithm to equation 7.3 we get

$$\log \mathrm{P}^{i,j}(w,y) = \boldsymbol{M}^{i,j}_{w,y} - \log \sum_{w'} \sum_{y'} \exp \boldsymbol{M}^{i,j}_{w',y'}. \tag{C.2}$$

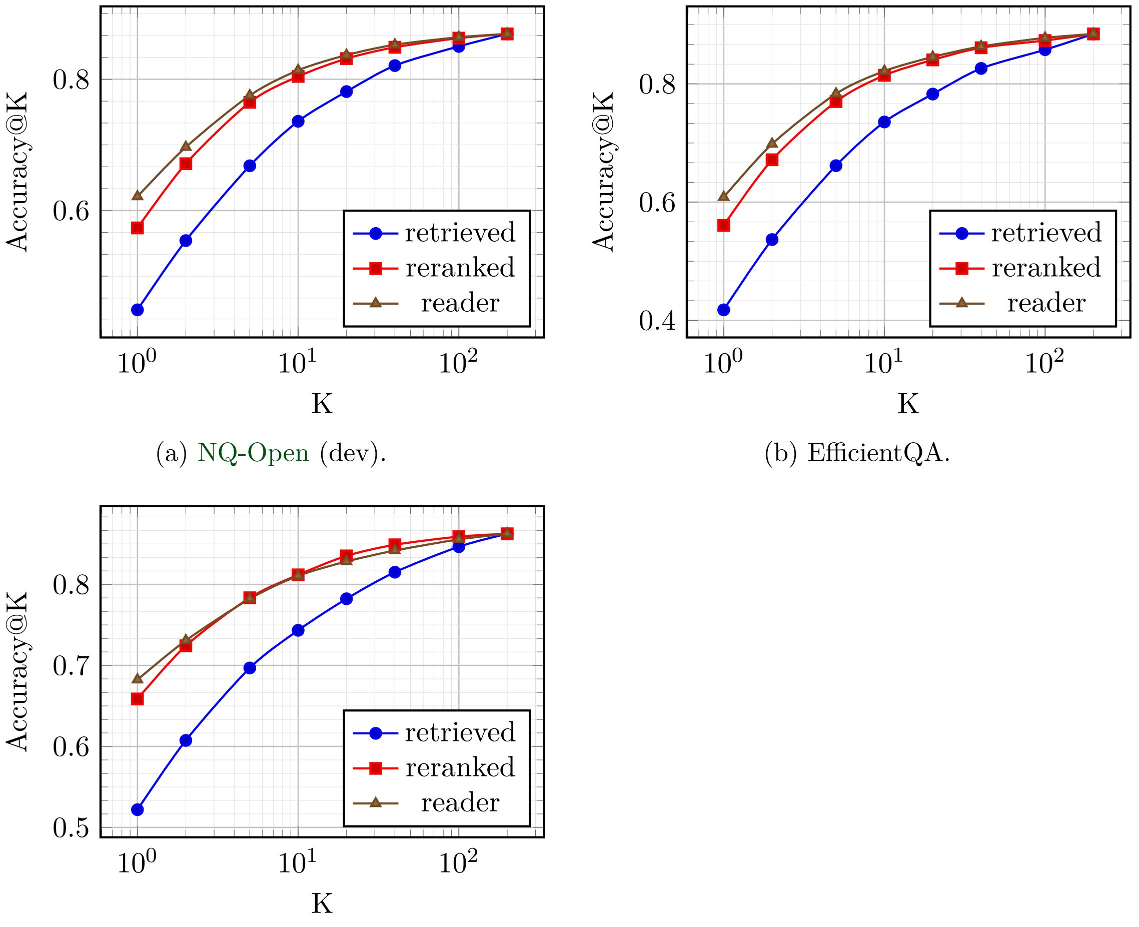
Expressing $\boldsymbol{M}^{i,j}_{w,y}$, substituting $C^{i,j} = \sum_{w'} \sum_{y'} \exp \boldsymbol{M}^{i,j}_{w',y'}$, and merging the logarithms, we arrive to equation 7.6. We recommend Bishop (2006), chapter 4.2 for further information.

Appendix D

Additional Results

D.1 Reranking Performance in R2-D2

Analysis of Accuracy@K, reranking top-200 retrieved results, on NQ-Open development data is shown in Figure D.1a, on EfficientQA data is shown in Figure D.1b, and on TQ-Open development data in Figure D.1c.



(a) NQ-Open (dev).

(b) EfficientQA.

(c) TQ-Open (dev).

Figure D.1: Analysis of Accuracy@K on different datasets.

D.2 Additional Component Fusion Analysis in R2-D2

This section includes results analogical to Tables 4.6, 4.7 on EfficientQA and development data of NQ-Open and TQ-Open (Tables D.1, D.2).

	$\boldsymbol{P}_*$	$\emptyset$	$\{r\}$	$\{rr\}$	$\{r, rr\}$
NQ-Open	$\{e\}$	48.38	48.94	48.85	49.14
	$\{g\}$	49.99	50.49	50.35	50.47
	$\{e, g\}$	51.79	**51.97**	51.82	51.80
TQ-Open	$\{e\}$	65.07	65.21	65.16	65.24
	$\{g\}$	67.68	67.72	67.73	67.76
	$\{e, g\}$	68.13	**68.19**	68.17	68.12
EfficientQA	$\{e\}$	47.56	48.33	48.89	48.72
	$\{g\}$	49.11	49.56	50.22	50.11
	$\{e, g\}$	50.78	51.67	50.89	**52.00**

Table D.1: Score aggregation on validation data of NQ-Open, TQ-Open and EfficientQA.

	$\boldsymbol{P}_*$	$\emptyset$	$\{r\}$	$\{rr\}$	$\{r, rr\}$
NQ-Open	$\{e\}$	50.65	51.24	51.01	51.17
	$\{g\}$	50.36	50.91	50.68	50.90
	$\{e, g\}$	52.24	**52.29**	52.27	52.07
TQ-Open	$\{e\}$	69.03	69.03	69.01	68.99
	$\{g\}$	69.54	69.46	69.62	69.70
	$\{e, g\}$	69.77	**69.79**	69.67	69.61
EfficientQA	$\{e\}$	48.33	50.06	49.39	49.67
	$\{g\}$	48.94	49.50	50.06	49.72
	$\{e, g\}$	50.78	51.83	50.94	**52.22**

Table D.2: Binary decision on NQ-Open, TQ-Open and EfficientQA.

Model	NQ-Open				TQ-Open			
	Total	Question Overlap	Answer Overlap Only	No Overlap	Total	Question Overlap	Answer Overlap Only	No Overlap
FiD	51.40	71.30	48.30	34.50	67.60	87.50	66.90	42.80
ext+gen	54.99	75.00	48.89	39.91	69.94	90.18	71.53	44.83
gen	50.69	70.06	46.98	34.04	69.14	87.50	70.32	44.83
ext	50.72	72.53	45.40	35.11	65.46	83.63	66.42	39.46
Δ_{gen}	4.30	4.94	1.91	5.87	0.80	2.68	1.21	0.00
Δ_{ext}	4.27	2.47	3.49	4.80	4.48	6.55	5.11	5.37

Table D.3: Results on the overlapping and non-overlapping parts of test sets for NQ and TQ. *Total* column corresponds to the overall result on the whole dataset, as reported before, *Question Overlap* corresponds to samples with train-test question overlap and answer overlap, *Answer Overlap Only* corresponds to samples with answer overlap, but no question overlap and *No Overlap* corresponds to samples with no overlap between train and test sets.

D.3 R2-D2 Results According to Question and Answer Test-Train Overlap

In addition to evaluation on the TQ-Open and NQ-Open shown in Table 4.2, we also report results on subsets of these datasets in Table D.3, as split by Lewis et al. (2021a). We compare R2-D1 (retriever, reranker, and extractive or generative reader, marked as gen and ext respectively) and R2-D2 (ext+gen) to official results on FiD (Izacard and Grave, 2021b).

D.4 Additional Results for Pruning Effect on Reranking

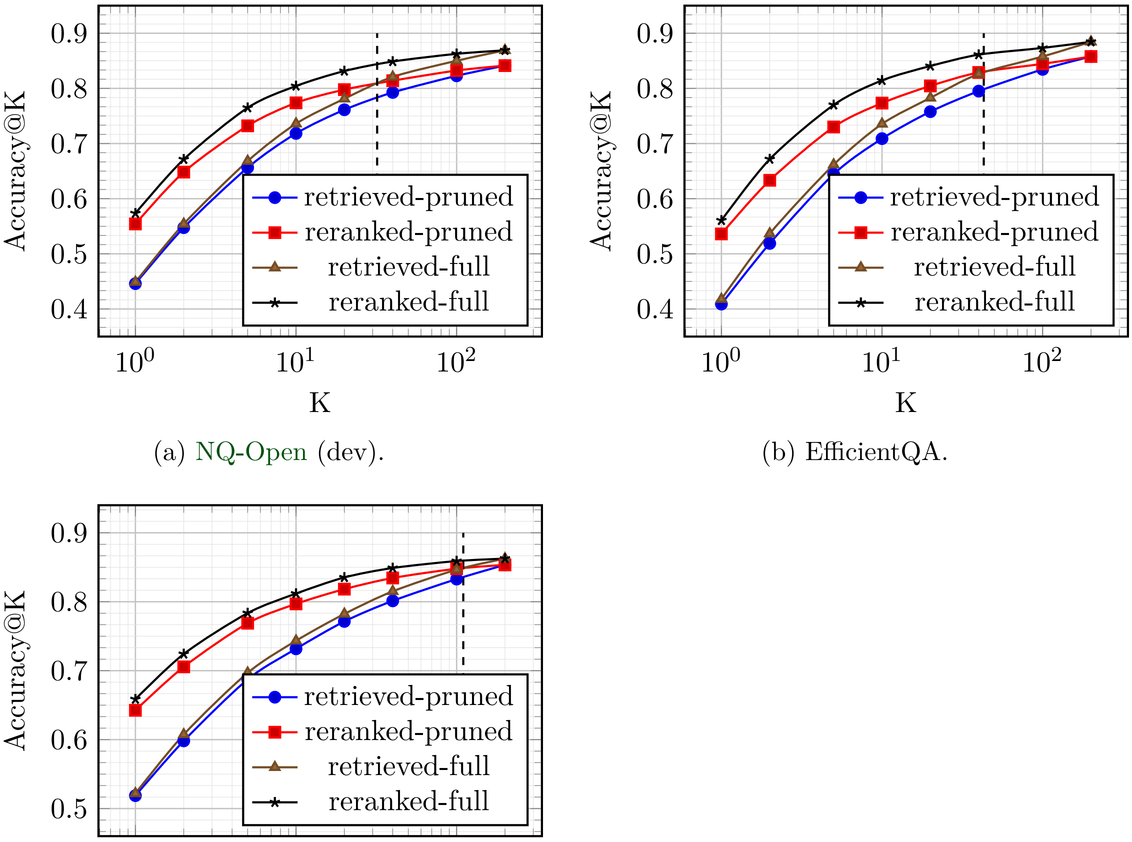
Results on additional datasets are shown in Figures D.2a, D.2b and D.2c.

D.5 Experiments with Different Model Parametrizations of Claim-Dissector

Apart from parametrizations provided in the main paper, we experimented with several different parametrizations described below. We keep the training details the same as for our baseline (Section 7.1.3). Starting off with a baseline system formulation, we will consider replacing L_{b0} with different objective functions.

$$\mathcal{L}_{b2} = \frac{1}{|\mathbb{A}|} \sum_{s_{i,j}, y \in \mathbb{A}} \log \mathrm{P}(s_{i,j}, y) \tag{D.1}$$

With L_{b2}, the annotation set $\mathbb{A}$ contains both relevant and irrelevant annotations. We found in practice this does not work - recall@5 during training stays at 0. This makes sense since if annotation exists, the final class is likely SUPPORT or REFUTE. Drifting the probability mass towards NEI for irrelevant annotations is adversarial w.r.t. total veracity probability.



(a) NQ-Open (dev).

(b) EfficientQA.

(c) TQ-Open (dev).

Figure D.2: Analysis of Accuracy@K on different datasets.

	FEVER		
Model	**FS**	**A**	**R@5**
CD	76.2	79.5	91.5
Baseline	75.2	79.8	90.0
L_{b3}	76.0	79.0	91.2
L_{b4}	75.7	79.7	90.4

Table D.4: Different model parametrizations.

K$_1$	**Recall**	**Recall**$_{MH}$	**Recall**$_{MH_{ART}}$	**#SaI**
10	90.4	40.1	33.0	68.8
20	93.4	48.0	41.5	132.9
30	94.1	51.3	45.0	196.8
35	94.2	52.0	45.8	239.9
50	94.5	54.3	48.4	325.4
100	95.1	58.5	53.1	649.4

Table D.5: Retrieval performance on FEVER dev set.

$$\mathcal{L}_{b3} = \log \sum_{s_{i,j}, y \in \mathbb{A}_p} \mathrm{P}(s_{i,j}, y) \tag{D.2}$$

Instead of maximizing the multinomial probability, L_{b3} objective marginalizes relevant annotations.

$$\mathcal{L}_{b4} = \log \sum_{s_{i,j} \in \mathbb{A}_p} \sum_{y} \mathrm{P}(s_{i,j}, y) \tag{D.3}$$

Additionally to L_{b3}, L_{b4} also marginalizes out the class label y.

The results in Table D.4 reveal only minor differences. Comparing L_{b3} and L_{b4}, marginalizing out label improves the accuracy, but damages the recall. Baseline parametrization achieves the best accuracy but the worst recall. Claim-Dissector seems to work the best in terms of FS, but the difference to L_{b3} is negligible if any.

D.6 Claim-Dissector: Retrieval Performance on FEVER

An in-depth evaluation of the retrieval method adopted from Jiang et al. (2021) is available in Table D.5.

D.7 Sensitivity to λ_2 Weight

In Figure D.3, we analyze the sensitivity of λ_2 parameter on F1 performance on TLR-FEVER during training. Choosing the large weight may lead to instabilities and vanishing of interpretable rationales, choosing smaller weight delays the peak performance in terms of F1. Note that for DeBERTaV3-large we chose $\lambda_2 = .0005$, as the one we used for base version ($\lambda_2 = .002$) leaded to such vanishing.

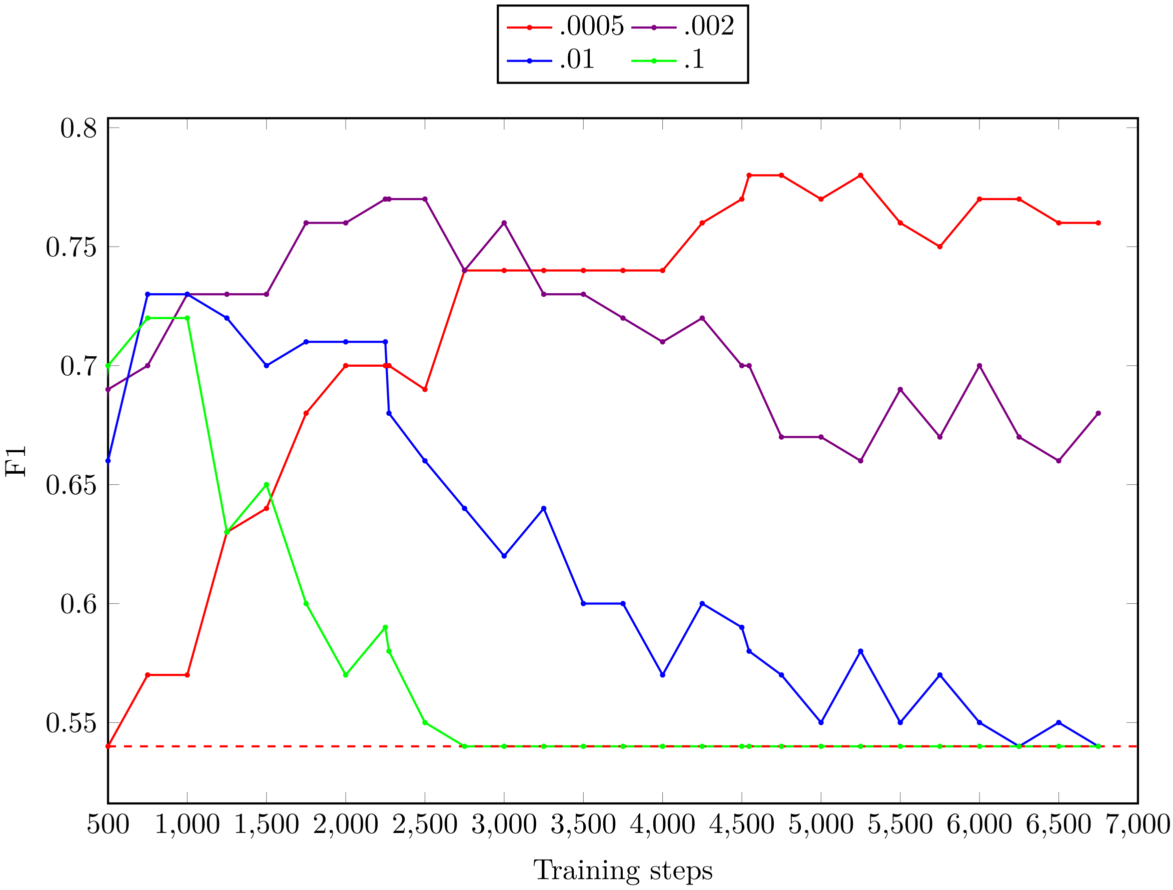


Figure D.3: Sensitivity to λ_2 weight selection for Deberta-V3-base model. Red dashed horizontal line marks the F1 performance when selecting all tokens (Select All Tokens) from Table 7.8.

Appendix E

Dataset Details

E.1 Samples

This section provides samples from every dataset used in the thesis. Each sample comes from the development set.

SQuAD (Rajpurkar et al., 2016)

> **Context**:
>
> `The first recorded travels by Europeans to China and back date from this time. The most famous traveler of the period was the Venetian Marco Polo, whose account of his trip to ”Cambaluc‟, the capital of the Great Khan, and of life there astounded the people of Europe. The account of his travels, Il milione (or, The Million, known in English as the Travels of Marco Polo), appeared about the year 1299. Some argue over the accuracy of Marco Polo’s accounts due to the lack of mentioning the Great Wall of China, tea houses, which would have been a prominent sight since Europeans had yet to adopt a tea culture, as well the practice of foot binding by the women in capital of the Great Khan. Some suggest that Marco Polo acquired much of his knowledge through contact with Persian traders since many of the places he named were in Persian.`
>
> **Question**:
>
> `How did some suspect that Polo learned about China instead of by actually visiting it?`
>
> **Answers from 3 annotators**:
>
> `through contact with Persian traders`
>
> `through contact with Persian traders`
>
> `through contact with Persian traders`

SQuAD2.0 (Rajpurkar et al., 2018) contains unanswerable questions for contexts from SQuAD. The unanswerable question for the context above was

> `Who was the last known European to visit China and return?`

Adversarial SQuAD (Jia and Liang, 2017) contains SQuAD context with a appended distractor sentence. Each context is appended 5 times with a distractor sentence, and worst-of-5 performance is considered during evaluation. The appended distractor sentence is underlined.

Context:

The connection between macroscopic nonconservative forces and microscopic conservative forces is described by detailed treatment with statistical mechanics. In macroscopic closed systems, nonconservative forces act to change the internal energies of the system, and are often associated with the transfer of heat. According to the Second law of thermodynamics, nonconservative forces necessarily result in energy transformations within closed systems from ordered to more random conditions as entropy increases.
The law of thermodynamics is associated with an open system heat exchange.

Question:

What is the law of thermodynamics associated with closed system heat exchange?

Answers from 3 annotators:

Second law

Second law of thermodynamics

Second

Natural Questions (Kwiatkowski et al., 2019) is a dataset used in two different ways in the thesis. In chapter 3, the MRQA version (Fisch et al., 2019) of the original dataset was used (example below). In ODQA, (Chapter 4 and Chapter 5), a NQ-Open version is used. NQ-Open contains only samples with short answers up to 5 whitespace tokens, and no context. Instead the contexts are to be retrieved from the Wikipedia dump. The example below (without context) is also part of NQ-Open. We note that during the conversion of this dataset, the multi-span answers were converted into multiple answers (the sample below contains such a multi-span answer). However in both versions of the dataset these were considered as multiple answers—selecting only one of the spans is sufficient for correct validation. As discussed in Chapter 3, joint probability is biased to pick only one of these spans, thus achieving great gains on Natural Questions.

Context:

<P> According to Unfinished Tales , at the start of the War of the Elves and Sauron , Celebrimbor gave Narya together with the Ring Vilya to Gil - galad , High King of the Noldor . Gil - galad entrusted Narya to his lieutenant Círdan , Lord of the Havens of Mithlond , who kept it after Gil - galad ’s death . According to The Lord of the Rings , Gil - galad received only Vilya , while Círdan received Narya from the very beginning along with Galadriel receiving Nenya from the start . </P>

Question:

who were the three elves who got rings

Answers:

Gil - galad

Galadriel

Círdan

NewsQA (Trischler et al., 2017)

Context:

WASHINGTON (CNN) - Department of Homeland Security Secretary Janet Napolitano apologized Friday for a department assessment that suggested returning combat veterans could be recruited by right-wing extremist groups. Homeland Security Secretary Janet Napolitano says she offered her "sincere apologies for any offense." She met with American Legion National Commander David Rehbein at Homeland Security headquarters. "The secretary started the meeting with an apo-llogy to me personally, to the American Legion and to the entire veterans community," Rehbein told reporters after the meeting. In a statement issued by the department, Napolitano said, "We connected meaningfully about the important issues that have emerged over recent days, and I offered him my sincere apologies for any offense to our veterans caused by this report. ... I pledge that the department has fixed the internal process that allowed this document to be released before it was ready." The report was an unclassified assessment sent to law enforcement agencies. It was titled "Rightwing Extremism: Current Economic and Political Climate Fueling Resurgence in Radicalization and Recruitment." The mention of combat veterans surfaced on a conservative radio program earlier this month, and it drew the scorn of commentators and conservative members of Congress. Rep. John Carter, R -Texas, called on Napolitano to resign. Rehbein said Friday it is time to move forward. "In the mind of the American Legion, I think her apology was sufficient," he said. "The way the Vietnam veterans were treated once they came home, that's what drives the sensitivity to this, because those things start small and then grow from there, and we need to make sure anytime something like that happens we need to step on that and make sure it goes away very quickly."

Question:

Who could be recruited by right-wing extremists?

Answer:

returning combat veterans

TriviaQA (Joshi et al., 2017) also provides automatically produced aliases for annotated answer. These are not present for MRQA version (Fisch et al., 2019). For , the aliases are available, and contexts are omitted.

Context:

WASHINGTON (CNN) - Department of Homeland Security Secretary Janet Napolitano apologized Friday for a department assessment that suggested returning combat veterans could be recruited by right-wing extremist groups. Homeland Security Secretary Janet Napolitano says she offered her "sincere apologies for any offense." She met with American Legion National Commander David Rehbein at Homeland Security headquarters. "The secretary started the meeting with an apo-llogy to me personally, to the American Legion and to the entire veterans community," Rehbein told reporters after the meeting. In a statement issued by the department, Napolitano said, "We connected meaningfully about the important issues that have emerged over recent days, and I offered him my sincere apologies for any offense to our veterans caused by this report. ... I pledge that the department has fixed the internal process that allowed this document to be released before it was ready." The report was an unclassified assessment sent to law enforcement agencies. It was titled "Rightwing Extremism: Current Economic and Political Climate Fueling Resurgence in Radicalization and Recruitment." The mention of combat veterans surfaced on a conservative radio program earlier this month, and it drew the scorn of commentators and conservative members of Congress. Rep. John Carter, R -Texas, called on Napolitano to resign. Rehbein said Friday it is time to move forward. "In the mind of the American Legion, I think her apology was sufficient," he said. "The way the Vietnam veterans were treated once they came home, that's what drives the sensitivity to this, because those things start small and then grow from there, and we need to make sure anytime something like that happens we need to step on that and make sure it goes away very quickly."

Question:

Because he held the earth on his shoulders, for what Greek figure did Gerardus Mercator name his book of maps?

Answer:

Atlas

Aliases:

"Atlases", "Atlas (cartography)", "Atlas (geography)", "The Atlases", "Atlas", "Atlas (book)", "Atlas the book"

RumourEval19 (Gorrell et al., 2019). The example was already shown in Figure 6.1.

FEVER and TLR-FEVER (Thorne et al., 2018). A set of annotated relevant sentences is unified across all FEVER annotators. Relevant tokens, as judged by different annotators of TLR-FEVER are shown with different colors. Each sentence is prepended with the article title in brackets.

Relevant Sentences (annotator #1):

„(Billboard Dad) **Billboard Dad (film)** is a 1998 American direct-to-video comedy film, **directed by Alan Metter** starring Mary-Kate and Ashley Olsen.", „(Alan Metter) **Alan Metter is an American film director** whose most notable credits include Back to School starring Rodney Dangerfield, and Girls Just Want to Have Fun with Sarah Jessica Parker.", „(Alan Metter) He also **produced and directed the** 1983 television special The Winds of Whoopie for Steve Martin."

Relevant Sentences (annotator #2):

„(Billboard Dad) **Billboard Dad** (film) is a 1998 American direct-to-video comedy film, **directed by Alan Metter** starring Mary-Kate and Ashley Olsen.", „(Alan Metter) **Alan Metter** is an American film **director** whose most notable credits include Back to School starring Rodney Dangerfield, and Girls Just Want to Have Fun with Sarah Jessica Parker.", „(Alan Metter) He also produced and **directed the** 1983 television special The Winds of Whoopie for Steve Martin."

Relevant Sentences (annotator #3):

„(Billboard Dad) Billboard Dad (film) is a 1998 American direct-to-video comedy film, **directed by Alan Metter** starring Mary-Kate and Ashley Olsen.", „(Alan Metter) **Alan Metter** is an **American film** director whose most notable credits include Back to School starring Rodney Dangerfield, and Girls Just Want to Have Fun with Sarah Jessica Parker.", „(Alan Metter) He also **produced and directed** the 1983 television special **The Winds of Whoopie for** Steve Martin."

Relevant Sentences (annotator #4):

„(Billboard Dad) **Billboard Dad** (film) is a 1998 American direct-to-video comedy film, **directed by Alan Metter** starring Mary-Kate and Ashley Olsen.", „(Alan Metter) **Alan Metter is an American film director** whose most notable credits include Back to School starring Rodney Dangerfield, and Girls Just Want to Have Fun with Sarah Jessica Parker.", „(Alan Metter) He also produced and directed the 1983 television special The Winds of Whoopie for Steve Martin."

Claim:

Billboard Dad was directed by a parrot.

Verdict:

REFUTE

FAVIQ (Park et al., 2022).

Claim:

the 76th season was the last time that the bills won their division.

Verdict:

SUPPORTS

HOVER (Jiang et al., 2020) contains facts requiring up to 4 hops of reasoning for verification (shown below).

Claim:

James Hewitt and another solicitor have in common the English country of origin. The other English solicitor was a chair of a human rights group. This group was a member of the International Corporate Accountability Roundtable Steering Committee.

Annotated Evidence:

(Mark Stephens (solicitor)) Mark Howard Stephens CBE (born 7 April 1957) is an English solicitor specialising in media law, intellectual property rights and human rights with the firm Howard Kennedy LLP.

(James Hewitt) Mark Howard Stephens CBE (born 7 April 1957) is an English solicitor specialising in media law, intellectual property rights and human rights with the firm Howard Kennedy LLP.

(Global Witness) Gillian Caldwell joined the organisation as Executive Director in July 2015 and Mark Stephens (solicitor) was appointed Chair in March 2016.

(International Corporate Accountability Roundtable) ICAR's Steering Committee includes EarthRights International, Human Rights Watch, Human Rights First, Global Witness and Amnesty International.

Verdict:

SUPPORTED

REALFC (Thorne et al., 2021) contains Wikipedia sections with verdicts and per-sentence level annotations.

Claim:

puerto rican citizens pay u.s. taxes

Wikipedia Section

(**supporting**) The Commonwealth of Puerto Rico is a territory of the United States and Puerto Ricans are US citizens.

(**refuting**) However, Puerto Rico is not a US state.

(**refuting**) Because of this, only Puerto Rican residents who are federal government employees, and those with income sources outside of the territory, pay federal income tax.

(**refuting**) All other employers and employees pay no federal income taxes.

(**supporting**) However, residents of Puerto Rico and businesses operating in Puerto Rico do pay some federal taxes, and the commonwealth's government has its own taxes as well.

(**neutral**) In July 2018, approximately 21% of the labor force on Puerto Rico were employed by the government, however this includes both the commonwealth and federal governments.

Section Verdict:

refuted

E.2 TLR-FEVER Guidelines

Annotation guidelines
Welcome to the “Pilot annotation phase” and thank you for your help!
How to start annotate
If you haven’t done so, simply click on „Start Annotation“ button, and the annotation will start.
Annotation process & guidelines

- In pilot annotation, we are interested in annotator’s disagreement on the task. So whatever disambiguity you will face, do not contact the organizers but judge it yourself.
- Your task is to annotate 100 samples. In each case, you will be presented with list of sentences divided by | character. The sentences do not need to (and often do not) directly follow each other in text. Be sure that in each case you:
- read the claim (lower-right corner)
- read metadata - to understand the context, you also have access to other metadata (lower-right corner), such as
  - titles - Wikipedia article names for every sentence you are presented with, split with character |,
  - claim_label - Ground-truth judgment of the claim’s veracity.
- **highlight minimal part of text you find important for supporting/refuting the claim. There should be such part in every sentence (unless annotation error happened). PLEASE DO NOT ANNOTATE ONLY WHAT IS IMPORTANT IN THE FIRST SENTENCE.**
- Use **„RELEVANT“** annotation button highlight the selected text spans.
- In some cases, you can find **errors in the ground-truth judgment**, in other words, either document is marked as supported and it should be refuted according to your judgment or vice-versa. If you notice so, please annotate any part of the document with **CLAIM_ERROR** annotation.
- In case you would like to comment on some example, use comment button (message icon). If the comment is example specific, please provide specific example’s id (available in-between metadata).

FAQ

- *The example does not contain enough information to decide whether it should be supported or refuted. Should I label it as a CLAIM_ERROR?*
  No. In such a case, please annotate parts of the input, which are at least partially supporting or refuting the claim. Please add comment to such examples. If there are no such input parts, only then report the example as CLAIM_ERROR.

That is it. Good luck!

E.3 RumourEval 2019 Dataset Insights

During data analysis, we gained insights shared within this section.

Empty datapoints 12 data points do not contain any text. According to the dataset authors, they were deleted by users at the time of data download and were left in the data not to break the conversational structure.

Domain-specific context dependency The query stance of some examples taken from subreddit DebunkThis[1] is dependent on domain knowledge. Examples from the subreddit DebunkThis have all the same format „Debunk this: [statement]“, e.g. *“Debunk this: Nicotine isn’t really bad for you, and it’s the other substances that make tobacco so harmful.”*. All these examples are labeled as queries.

Multiple stances within the same tweet The single class of some posts is ambiguous, as they contain more statements that could be classified into different classes. The source/previous post *“This is crazy! #CapeTown #capestorm #weatherforecast https://t.co/3bcKOKrCJB”* and the target post *“@RyGuySA Oh my gosh! Is that not a tornado?! Cause wow, It almost looks like one!”*, labeled as a comment in the dataset, might be seen as a query as well.

Metadata The dataset contains a whole tree structure and metadata for each discussion from Twitter and Reddit. Obviously, the nature of the data differs across platforms. For example, the Reddit subset includes upvotes.

E.4 Conditional Scoring on RealFC Dataset

First, binary F1 is computed for each i-th example from the dataset—section with several sentences, and every sentence is predicted to be relevant (i.e., supporting or refuting) or irrelevant (i.e., neutral)—obtaining F1 score $s(i)$. Define $\mathbb{I}[m(i) = l_i]$ as an indicator function yielding 1 if model m_v predicted the veracity correct class l_i for an i-th example, and 0 otherwise. With dataset of size N, the CondAcc is defined as an *average F1 score for samples with correct veracity*

$$\text{CondAcc} = \frac{1}{N}\sum_{i=1}^{N} s(i)\mathbb{I}[m_v(i) = l_i]. \tag{E.1}$$

Next, assume that TP_s, TP_r, and TP_n and similarly FP_x and FN_x are TPs, FPs, and FNs computed for each veracity class—support veracity (s), refute veracity (r), or neutral veracity (n), and $x \in \{s, r, n\}$. Further, assume that wTP_s, wTP_r, wTP_n are computed as $wTP_x = \sum_{i=1}^{N} s(i)\mathbb{I}[m_v(i) = l_i \wedge x = l_i]$, where $x \in \{s, r, n\}$. Firstly, weighted F1 $wF1_x$ is computed for each class separately from precision p_x and recall r_x as shown in Equation E.2. The CondF1 score is obtained by averaging $wF1_x$ across all classes $x \in \{s, r, n\}$.

$$\begin{aligned} p_x = \frac{wTP_x}{TP_x + FP_x} \qquad r_x &= \frac{wTP_x}{TP_x + FN_x} \\ wF1_x &= 2\frac{p_x r_x}{p_x + r_x} \end{aligned} \tag{E.2}$$

[1] https://www.reddit.com/r/DebunkThis/. Accessed 20.3.2023.

Appendix F

Extra Dicussions

F.1 Addressing the Complexity of Extractive QA Objectives

One may ask what complexity joint modeling objectives come with independently of the underlying architecture. Given that L is the length of the input's passage and d is the model dimension, the independent objective contains only linear transformation and is in $\mathrm{O}\big(dL\big)$ for time and memory, assuming the multiplication and addition are constant operations. For the rest of this analysis, we will denote both time and memory complexities as just complexity, as they are the same for the analyzed cases.

The conditional objective increases the complexity for both only constantly, having an extra feed-forward network for end token representations. However, one may experience a significant computational slowdown, because of the beam search. Having a beam size k and a minibatch size b, the end probabilities cannot be computed in parallel with start probabilities, and have to be computed for the kb cases.

For direct joint probability modeling, the complexity largely depends on the similarity function. The easiest case is f_{dot}, where in theory the complexity rises to $\mathrm{O}\big(dL^2\big)$, but in practice, the dot product is well optimized and has a barely noticeable impact on the speed or memory.

For the f_{add} the complexity is given by the linear projection $\boldsymbol{H_*w_*}$ being in $\mathrm{O}\big(dL\big)$ and outer summation of two vectors $\boldsymbol{H_sw_1} \oplus \boldsymbol{H_ew_2}$, which is in $\mathrm{O}\big(L^2\big)$, where $\boldsymbol{w} = [\boldsymbol{w_1}, \boldsymbol{w_2}]$ and $\boldsymbol{H_*} \in \mathbb{R}^{n \times d}$ are the start/end representation matrices. Therefore the complexity is $\mathrm{O}\big(dL + L^2\big)$. We observed that in practice this approach is not very different from f_{dot}, probably due to d being close to L.

Next, a weighted product f_{wdot} can be efficiently implemented as $\boldsymbol{H_s}(\boldsymbol{w} \circ \boldsymbol{H_e})$, where $\boldsymbol{w}$ is broadcasted over every end representation in $\boldsymbol{H_e}$. In this case, the complexity stays the same as for f_{dot}.

To demonstrate that in practice the speed and memory requirements between independent and joint approach are comparable, one BERT epoch on SQuADv1.1 took about 47 minutes and 4.2GB of memory with the same minibatch size 2 on 12GB 2080Ti GPU with both objective variants. We observed the same requirements for all direct joint probability modeling methods mentioned so far.

Finally, the most complex approach is clearly f_{MLP}. While a theoretical time and memory complexity of an efficient implementation[1] is in $\mathrm{O}\big(d^2L + dL^2\big)$, the complexity

[1] The linear transformation $d \times 2d$ can be applied to each start or end vector separately, and only then the start/end vectors have to be outer-summed.

of this approach can be improved by pruning down the number of possible spans (and the probability space). Assuming the maximum length of the span is $k \ll L$, one can reduce the complexity to $\mathrm{O}\big(d^2L + dLk\big)$ (an approach adopted in Lee et al. (2019)). To illustrate this complexity, the BERT model with the full probability space on SQuADv1.1 with minibatch size 2 took 76 minutes per epoch while allocating 8.2GB of GPU memory (we were unable to fit larger minibatch size to 12GB GPU).

F.2 The Continued Influence Effect: Retractions Fail to Eliminate the Influence of Misinformation

Lewandowsky et al. (2012) summarizes the research paradigm, which focuses on credible retractions in neutral scenarios, in which people have no reason to believe one version of the event over another. In this paradigm, people are presented with a factious report about an event unfolding over time. The report contains a target piece of information (i.e. a claim). For some readers, the claim is retracted, whereas for readers in a control condition, no correction occurs. Then the readers are presented with a questionnaire to assess their understanding of the event and the number of clear and uncontroverted references to the claim's veracity.

In particular, a stimulus narrative commonly used within this paradigm involves a warehouse fire, that is initially thought to have been caused by gas cylinders and oil paints there were negligently stored in a closet. A proportion of participants is then presented with a retraction such as „the closet was actually empty“. A comprehension test follows, and the number of references to the gas and paint in response to indirection inference questions about the event (e.g., „What caused the black smoke?“) is counted.

Research using this paradigm has consistently found that retractions rarely, if ever, had the intended effect of eliminating reliance on misinformation, even when people remember the retraction when later asked. Seifert (2002) have examined whether clarifying the correction might reduce the continued influence effect. The correction in their studies was strengthened to include the phrase „paint and gas were never on the premises“. Results showed that this enhanced negation of the presence of flammable materials backfired, making people even more likely to rely on the misinformation in their responses. Some other additions to the correction were found to mitigate to a degree, but not eliminate, the continued influence effect. For instance, when participants were given a rationale about how misinformation originated, such as „a truckers' strike prevented the expected delivery of the items“, they were less likely to make references to it. Even so, the influence of the misinformation could still be detected. The conclusion drawn from studies on this phenomenon is that it is extremely difficult to return the beliefs of people who have been exposed to misinformation to a baseline similar to those of people who have never been exposed to it. We recommend reading Lewandowsky et al. (2012) for a broader overview of *the misinformation and its correction.*

F.3 The Necessity of Claim-Dissector's Baseline Modifications

The reason for the modification is that (i) the original model (Schlichtkrull et al., 2021) (without L_{b1}) could not benefit from NEI annotations present on FEVER, resulting in an unfair comparison with our models and previous work, as TabFact does not contain such annotations (ii) the original model is not able to distinguish the attribution from the repeated relevant evidence, because equations (6)/(9) in their work just sum the probabilities of relevant items supervised independently—they do not use the supervision of overall veracity for the claim. This is problematic, especially in a FEVER setting comparable to ours, where the relevance of hundreds of sentences is considered (many of them possibly relevant) as compared to TabFact where only top-5 retrieved tables were considered, and often only a single is relevant.

F.4 Statistical Testing on F1 Measure

To compare CD with masker in F1, we follow Goutte and Gaussier (2005), sum TPs, FPs, FNs across the dataset, estimate recall (R) and precision (P) posteriors, and sample F1 distributions. To obtain a sample of average F1 from multiple checkpoints, we estimate the P and R posteriors for each checkpoint separately, sample F1 for each checkpoint and then average these. We estimate $p \approx \mathrm{P}(\mathrm{F1}_a > \mathrm{F1}_b)$ via Monte-Carlo using 10,000 samples, and consider the significance level at $p > 0.95$.

F.5 Details on Analysis of Prediction Scores

We define relevance score (RS) as $\mathrm{P}^{i,j}(y = l)$ where $l \in \{S, R\}$ is the ground truth label. For A/B testing, we shuffle (a) and (b) cases for annotators, so they cannot determine which sentence comes from which source. An example of 1 annotation is available at https://shorturl.at/hiCLX. Since we found many annotators hesitated with no preference option (c) when computing Krippendroff's α, we assume two classes, and empty preference when the annotator does not have a preference (case (c)) (we do not consider it a separate nominal category). Krippendorff's α=0.65 achieved moderate agreement.

To compute average Kendall τ, we select sentences with RS>0.7 for each example; this creates relevance ranking. Then we reorder selected sentences according to PS, obtaining prediction ranking. Kendall τ is computed for every sample, and the resulting statistic is averaged across all samples in the FEVER validation set.