UNIVERSITY OF CALIFORNIA,
IRVINE

Factuality and Large Language Models: Evaluation, Characterization, and Correction

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Computer Science

by

Anthony Chen

Dissertation Committee:
Sameer Singh, Associate Professor, Chair
Erik Sudderth, Professor and Chancellor's Fellow
Richard Futrell, Associate Professor

2023

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to my advisor Sameer Singh. We both joined UC Irvine in 2016 and I still remember the first email I sent asking to do research with him. Since then, our field has changed immensely, and I've been fortunate to be privy to his insights into where the field is headed. As a result, I've grown from a bumbling Master's student proposing new ideas every two weeks into a competent researcher. Also, his light hearted approach to life and humour were a breath of fresh air during these years. I look back fondly at our coffee chats about life and research, as well as the late nights furiously working a shared Overleaf document to make a 4 A.M. conference deadline. I'll always be grateful for the chance he took on me as an unproven NLP researcher and his guidance throughout the past seven years.

I would like to thank Professor Erik Sudderth and Professor Richard Futrell for being on my dissertation committee.

I've been blessed to have had labmates who I also consider friends: Ananya, Catarina Belem, Dheerua Dua, Shivanshu Gupta, Tamanna Hossain-Kay, Robert Logan, Kolby Nottingham, Pouya Pezeshkpour, Yasaman Razeghi, Preethi Seshadri, Dylan Slack, Junlin Wang, and Zhengli Zhao. While I wasn't in the lab often, the time I spent with each of you was meaningful to me. I'll miss our times getting lunch and drinks, discussing research, venting about the struggles of being graduate students, and distracting you all from doing work when I was in the lab.

Papers don't write themselves and research isn't done alone. Thank you to the many co-authors I've had that have made me a better researcher and writer: Arun Chaganty, Zhuyun Dai, Chris DuBois, Luyu Gao, Matt Gardner, Pallavi Gudipati, Kelvin Guu, Yicheng Fan, Da-Cheng Juan, Ni Lao, Hongrae Lee, Xiao Ling, Shayne Longpre, Panupong Pasupat, Kartik Perisetla, Nikhil Ramesh, Gabriel Stanovsky, and Vincent Zhao. An extra thank you goes to Matt Gardner who served as an unofficial co-adviser in the early years of my Ph.D. and whose insight into our field I have great respect for.

I've been fortunate to do research internships at Google, Verneek, and Apple. I'd like to thank my hosts Kelvin Guu, Hongrae Lee, Sam Anzaroot, Nasrin Mostafazadeh, and Pallavi Gudipati. You made these internships a fun respite and helped make my choice to pursue a research career in industry post-graduation an easy one.

Doing a Ph.D. wouldn't have been possible without a wonderful group of friends in my personal life. First, thank you to my high school friends, who have been with me for over a decade and whose energy, support, and jokes about me being a perpetual student have kept me afloat. My journey has been a bit unique with constant changes in scenery, and I'd like to also thank the friends I've made during my Ph.D., most notably those in Los Angeles and New York City. You made these places feel like second and third homes. It took a while but I'm joining you all in the real world.

To say doing a Ph.D. is a long journey is an understatement. I'd like to thank my loving family which have been with me every step of the way.

The material presented in this dissertation was funded in part by: the Allen Institute of Artificial

Intelligence (AI2), DARPA MCS program under Contract No. N660011924033 with the United States Office Of Naval Research, and NSF award #IIS-1817183. The views expressed in this dissertation are those of the author and do not reflect views of the funding agencies.

VITA

Anthony Chen

EDUCATION

Doctor of Philosophy in Computer Science **2018-2023**
University of California, Irvine *Irvine, California*

Master of Science in Computer Science **2016-2018**
University of California, Irvine *Irvine, California*

Bachelor of Science in Computer Science **2012-2016**
University of California, Davis *Davis, California*

RESEARCH EXPERIENCE

Graduate Research Assistant **2017–2023**
University of California, Irvine *Irvine, California*

Research Scientist, Intern **2022**
Google *Mountain View, California*

Research Scientist, Intern **2021**
Verneek *New York, New York*

Research Scientist, Intern **2020**
Apple *Remote*

Undergraduate Research Assistant **2015–2016**
University of California, Davis *Davis, California*

TEACHING EXPERIENCE

Teaching Assistant **2017–2022**
ICS139W, CS272

ACADEMIC SERVICE

Reviewer
O-DRUM Workshop @ CVPR: 2023
EMNLP: 2018, 2021

Site Developer
AKBC: 2021

REFEREED CONFERENCE PUBLICATIONS

RARR: Researching and Revising What Language Models Say, Using Language Models **2023**
ACL

Entity-Based Knowledge Conflicts in Question Answering **2021**
EMNLP

Evaluating Entity Disambiguation and the Role of Popularity in Retrieval-Based NLP **2021**
ACL-IJCNLP

MOCHA: A Dataset for Training and Evaluating Generative Reading Comprehension Metrics **2020**
EMNLP

REFEREED WORKSHOP PUBLICATIONS

Evaluating Question Answering Evaluation **2019**
MRQA Workshop @ EMNLP (Best Paper)

UNREFEREED PREPRINTS

PURR: Efficiently Editing Language Model Hallucinations by Denoising Language Model Corruptions **2023**
arXiv

AWARDS

Noyce Fellowhip **2021**

Best Paper **2019**
MRQA Workshop @ EMNLP

INVITED TALKS

RARR: Researching and Revising What Language Models Say **2023**
UC Irvine AI/ML Seminar

Developing Learned Metrics for Generative Reading Comprehension **2020**
Apple

Evaluating Question Answering Evaluation **2019**
MRQA Workshop @ EMNLP

ABSTRACT OF THE DISSERTATION

Factuality and Large Language Models: Evaluation, Characterization, and Correction

By

Anthony Chen

Doctor of Philosophy in Computer Science

University of California, Irvine, 2023

Sameer Singh, Associate Professor, Chair

In the past several years, the field of natural language processing has undergone a paradigm shift driven by a singular piece of technology: large language models. Large language models have emerged as fundamental tools in our field, significantly advancing the state of the art and enabling breakthroughs across a wide range of tasks. The capabilities of large language models are an emergent function of scale - they are trained on terabytes of text data scraped from all corners of the web. By virtue of this training, large language models capture a rough notion of the factual state of the world in their weights. Understanding how language models represent these facts and how these representations are used to generate text is vital given their increasing deployment in the world. In this dissertation, our goal is to provide tooling and technologies to probe and evaluate how language models leverage and generate facts and correct the cases where language models produce inaccurate factual statements.

This dissertation is split into three distinct parts. The first is concerned with evaluating the generations of large language models for correctness. Here, we develop metrics (*i.e.,* LERC) for evaluating the outputs of language models. The second is concerned with characterizing how language models leverage different sources of facts and use them during generation. We develop benchmarks (*i.e.,* AmbER and knowledge conflicts) for assessing the tendencies of language models to over-rely on information stored in their weights and connect this to their propensity to generate factually

inaccurate statements. Our final part is concerned with correcting inaccuracies in language model generations. We develop methodologies for training models for correcting factual inaccuracies that language models generate (*i.e.,* RARR and PURR) as well as concretely establish the task of editing for factuality by introducing new evaluation benchmarks and metrics.

In total, the work presented in this dissertation provides a concrete way to evaluate the facts language models generate, characterize how language models leverage different sources of facts and use them during generation, and correct the facts language models generate.

Chapter 1

Introduction

1.1 Motivation

The past several years have seen a drastic rise in the problems that our field can tackle [160]. This rise in capabilities can be attributed to a singular piece of technology: language models [11, 105, 107]. Language models are certainly not a new creation with seminal research taking place in the 1990's [17, 26, 12]. But in the years prior to the late 2010s, language models saw the most application in re-ranking the likelihood of different language generation systems, such as speech and translation systems, to serve the most plausible generation [77, 16, 113]. However, in recent years, our community has found that by scaling a combination of model size [67], training data [121, 44], and compute [60], this simple method of next-word prediction yields many amazing emergent capabilities [162, 120]. One of these capabilities is that by training on web-corpus sized data, large language models (LLMs) capture a rough and imperfect state of the world in their weights [119, 95]. However, with this growing power to represent facts in the world comes a growing number of questions.

Consider a language model that has been prompted with the question *"What is the fastest marathon*

time set?" which then generates *"The marathon world record is 2:01:39, set by Eliud Kipchoge of Kenya in 2017."*. This coherent and seemingly plausible generation raises a number of questions:

1. Even though this generation *seems* correct, how can we automatically **evaluate** the quality of this answer? Automatic evaluation is vital for quickly and cheaply evaluating the progress of large language models without requiring an abundance of expensive human evaluation.

2. In the event the language model uses knowledge external to its parameters (*e.g.,* a retrieved document from the web), there is a need to **characterize** how much a language model relies on this external (*i.e.,* contextual) knowledge to do generation. Understanding this is vital since internal (*i.e.,* parametric) knowledge is an *imperfect* representation of the world, and over-relying on it can lead to inaccurate generations. These inaccurate generations are colloquially referred to as "*hallucinations*".

3. In the event that the generation is incorrect, how do we **correct** the generation? In this particular example, Kipchoge actually set a new world record in 2022 and thus there is a need to edit this generation to be factually consistent with the current state of the world. As large language models see increasing deployment, there is a reasonable expectation for correct and factual generations, and fixing the outputs is an important step.

These three facets of evaluating, characterizing, and correcting how language models leverage and generate facts form the pillars of work that are encapsulated in this dissertation.

1.2 Contributions and Thesis Overview

In this thesis, we will see concrete ways to address the three aforementioned problems around how large language models leverage and generate facts after first introducing the prerequisite background

in Chapter 2. The author of this dissertation was a primary contributor (*i.e.,* first author or co-first author) on all papers referenced in this section.

1. Part I of this dissertation is concerned with **evaluating** the facts language models generate. In Chapter 3, we will see why evaluating generated text is a challenging language processing task in and of itself [21]. To overcome this challenge, we first introduce MOCHA, a collection of language model generations paired with scores from humans indicating the quality of these generations [22]. We use these human judgement scores in MOCHA to train a NLP model as an evaluation metric which we term LERC.

2. Part II of this dissertation is concerned with **characterizing** how language models capture facts by understanding how language models leverage and balance contextual and parametric knowledge. In Chapter 4, we study how the quality of contextual knowledge affects language model generations by introducing evaluation datasets called AmbER sets to evaluate the retrieval systems used to gather this contextual knowledge [19]. In Chapter 5, we introduce an evaluation paradigm called knowledge conflicts to understand how often and why language models over-rely on parametric knowledge [97]. Using knowledge conflicts, we discover factors that cause language models to ignore contextual knowledge which leads to hallucinations, as well as propose effective strategies to reduce this over-reliance.

3. Part III of this dissertation is concerned with **correcting** the facts language models generate. In Chapter 6, we formalize correction by introducing a new task we call *Editing for Attribution* [45]. In this task, the goal is to *edit* a language model's generation so as to be maximally attributed to a set of evidence, thus reducing the inaccuracies in the generation. As part of this task, we introduce evaluation datasets and metrics, as well as a strong editing baseline, RARR, that works by prompting a large language model to edit out factual inaccuracies. In Chapter 7, we introduce PURR, an efficient editing method that is trained by denoising language model corruptions to text [20].

Finally, in Chapter 8 we conclude by reflecting on the impact and limitations of the work presented in this thesis and look towards future potential continuations.

Chapter 2

Background

2.1 Language Models

A language model is a probabilistic model used to estimate the likelihood of a sequence of words or tokens. Consider a sequence of tokens, $W = (w_1, w_2, \ldots, w_T)$, where w_t represents the t-th token in the sequence and T is the sequence length. The goal of a language model is to estimate the probability of observing the sequence, $P(W)$. This probability distribution is typically factorized via the chain-rule, termed autoregressive language model:

$$P(W) = P(w_T, w_{T-1}, \ldots, w_1) = \prod_{t=1}^{T} P(w_t | w_{t-1}, w_{t-2}, \ldots, w_1)$$

There have been many variations of language models developed. Here we describe some historically relevant variations as well as those pertinent to this dissertation.

N-gram Language Models [17, 12, 71, 51, 26] An n-gram language model makes a simplifying Markov assumption, where the probability of a token depends only on the preceding $n-1$ tokens:

$$P(w_t|w_{t-1}, w_{t-2} \ldots, w_1) \approx P(w_t|w_{t-1}, w_{t-2}, \ldots, w_{t-(n-1)})$$

Learning in an n-gram language model is done using frequency estimation on a corpora of text. For instance, the probability $P(w_t|w_{t-1}, \ldots w_{t-(n-1)})$ can be estimated as the ratio of the count of the n-gram $(w_t, w_{t-1}, \ldots, w_{t-(n-1)})$ to the count of the n-gram $(w_{t-1}, \ldots, w_{t-(n-1)})$ in the training corpus. N-gram language models have notable limitations. The first is limited contextual window to learn dependencies from. The second is a large storage cost for storing all combinations of n-grams which is sparse and grows exponentially as the size of the n-gram grows. Because of these reasons, n-gram language models have fallen out of favor.

Neural Language Models [11, 105, 108] In a neural language model, we parameterize the probability distribution over sequences using a neural network. The neural network is usually a recurrent neural network (RNN) [56, 52, 27] or more recently, a Transformer [157, 34]. Learning in a neural language models is done use back-propagation on a large training corpus of text. Neural language models alleviate the aforementioned issues of n-gram language models. Because neural networks have a much larger contextual window (infinite in the case of RNNs), neural language models learn to leverage history better during training thus capturing richer dependencies from the training data. In addition, neural language models are generally much more space-efficient as they remove the exponential space requirements from their n-gram counterparts.

Conditional Language Models [53, 88, 86] Language models can also be conditioned on external information such as documents, called *contextual information*. These conditional language models take into account past generation history as well as the contextual information when generating the next word. Formally, in a conditional language model, we have additional context information, C,

with the goal to estimate the probability of observing the sequence given the context, $P(W|C)$:

$$P(W|C) = P(w_T, w_{T-1}, \ldots, w_1|C) = \prod_{t=1}^{T} P(w_t|w_{t-1}, w_{t-2} \ldots, w_1, C)$$

Because of the complex dependencies between the sequence and context, conditional language models are parameterized via a neural network. Conditional language models find applications in various conditional language generation tasks. For example, in document summarization, a conditional language model can generate a concise summary given the entire document. In dialogue generation, the model can generate a response based on the conversation history. And in question answering, generation can occur by reasoning first over relevant documents before producing an answer. A benefit of conditional language models is they alleviate the burden of a language model to have to memorize information during training and instead allows the language model to reason over the provided context.

Large Language Models [15, 29, 155] From a probabilistic formulation, a large language model is no different from a smaller language model. However, by scaling both model size and training data, large language models exhibit qualitatively different capabilities compared to their smaller counterparts. By training on extremely large corpora of text, large language models develop emergent capabilities like in-context learning and chain-of-thought prompting [15, 163]. Moreover, by being exposed to web-scale data, large language models learn a great deal of world knowledge that can be extracted via prompting [119, 95]. What constitutes a "large" language model is ever-changing. At the time of writing of this dissertation, "large" is typically defined as having over 10 billion parameters while training on over 100 billion tokens. The process of training a large language model is colloquially termed "pre-training", as these large language models are often further "fine-tuned" [36] or "instruction-tuned" [137, 30, 161]. Large language models can be trained to be conditional or unconditional.

2.2 Language Generation Evaluation Metrics

Evaluation metrics are a critical component in making rapid progress in generative NLP. An evaluation metric can be reference-based or reference-less. In this section we describe reference-based evaluation metrics and describe reference-less metrics when needed in the dissertation.

In a reference-based evaluation metric, a generated piece of text (*i.e.,* candidate) consisting of tokens $c = (c_1, c_2, \ldots, c_N)$ and a reference piece of text that represents a "perfect" generation (*i.e.,* reference) consisting of tokens $r = (r_1, r_2, \ldots, r_M)$ are provided. An evaluation metric $E(c, r)$ produces a score (typically between 0 and 1) that indicates how close the candidate is to the reference. An evaluation metric may also be able to incorporate auxiliary information when computing the score, like the source document for the task of document summarization. The quality of an evaluation metric is evaluated by computing the correlation between the scores the evaluation metric assigns to the scores a human (or group of humans) would assign.

N-gram Evaluation Metrics [116, 92, 7] N-gram metrics compute a similarity score by computing the n-gram overlap between the candidate and reference. These token-based metrics are easy to implement and drove much of generation evaluation in the early 2000's. However, they lack flexibility by focusing only on token overlap and are vulnerable to paraphrasing. Moreover, different generation tasks require evaluating different facets of language (*e.g.,* evaluating summarization is different from evaluating translation) and n-gram metrics cannot be easily adapted to the nuances of different generation tasks.

Model-Based Metrics [33, 142] Model-based metrics train or prompt another NLP model to be aligned to human judgement scores, thus leveraging a model as a metric. These metrics are flexible in that they can be tuned to the specific task of interest. However, they require human judgement scores for the task of interest to train or prompt the model on.

2.3 Retrieval

Retrieval is a critical component for finding relevant information among a large corpora. Given a query, q, and a large corpora of documents, $D = (d_1, d_2, \ldots, d_N)$, the goal is to find the most k most relevant documents $D^* = (d_1, d_2, \ldots, d_k)$. Retrieval is done use an embedding model, E, to create embeddings for queries and documents, $E(q)$ and $E(d_i)$. The score between a query and document under the embedding model, $E(\cdot)$, is computed as the dot product of their embeddings $E(q) \cdot E(d_i)$. D^* under $E(\cdot)$ is defined as:

$$D_E^* = \arg \max_{\substack{d_i \in D \\ |D^*|=k}} E(q) \cdot E(d_i)$$

In many situations, the size of the corpora is so large that exhaustively computing the dot product for each document is prohibitively expensive. In these situations, approximate sub-linear similarity search is employed which employs techniques such as embedding clustering, embedding compression, and embedding quantization [66]. Retrieval is often a critical component for conditional language modeling [53, 88, 86]. For instance, given a question, retrieval is used to find the most relevant documents for the language model to reason over.

We now describe different retrieval methods which differ in how the embedding model, $E(\cdot)$, is implemented.

N-gram Retrieval [136, 32, 135] N-gram retrieval are token-based methods which computes an embedding for a piece of text using n-gram frequencies. Because there are an exponential number of n-grams, the size of n-gram embeddings are exponential in the size of the n-gram and generally extremely sparse. As we previously mentioned with n-gram methods, they also struggle with linguistic phenomena like paraphrasing.

Dense Retrieval [70, 74, 83] Dense retrieval is implemented by using pre-trained language model to produce embeddings for an input piece of text. These embeddings are dense and have a smaller embedding size than n-gram embeddings. Often times, dense retrieval methods are fine-tuned on the target retrieval task of interest by maximizing a query's embeddings to the most relevant document embeddings using a contrastive loss function.

Part I

Evaluation

Chapter 3

Evaluating What Language Models Generate with Learned Metrics

3.1 Introduction

As language models have improved, they have seen wide-spread adaption to many language understanding tasks. One of these such tasks is the long-standing progress of reading comprehension (RC). Reading comprehension (RC) has seen significant progress in the last few years, with a number of question answering (QA) datasets being created [125, 82, 149].

Before generation systems were applied to RC, a majority of datasets were presented using a span-selection or multiple-choice (MC) format. Both formats are easy to evaluate, but in return, have restrictions placed on the questions that can be asked or the answers that can be returned. Furthermore, both formats hinge on distractor spans/choices for learning to be effective. Ensuring high quality distractors is a challenging task in and of itself, which can lead to models that exploit spurious correlations [65, 106, 49]. Posing RC as a generation task addresses the aforementioned issues. Generative RC does not require distractors, circumventing biases that could be introduced

Passage: . . . Behind one door is a lady whom the king has deemed an appropriate match for the accused; behind the other is a fierce, hungry tiger. Both doors are **heavily soundproofed to prevent the accused from hearing what is behind each one**. . .

Question: What feature do the doors have?
Reference: soundproofed
Candidate: They are **heavily soundproofed to prevent the accused from hearing what's behind each one**.

Human Judgement: 5 out of 5
LERC: 0.99 out of 1
BLEU-1: 0.07 out of 1
ROUGE-L: 0.15 out of 1
METEOR: 0.17 out of 1

Figure 3.1: **Generative reading comprehension example.** Properly scoring the candidate requires access to the passage and reference answer. Metrics such as BLEU, ROUGE and METEOR, are agnostic to the end-task while LERC is trained with the passage and question as input. As a result, LERC assigns a score that better reflects human judgement.

by them, and allows arbitrary questions and answers.

Unfortunately, existing metrics for evaluating text generation come with significant shortcomings. Many metrics score *n*-gram overlap, and it is well established that using token overlap as a measure of similarity has drawbacks [21, 40, 159]. Current metrics also only consider the reference and are agnostic to the end-task being evaluated. Figure 3.1 demonstrates that this is problematic for generative RC because scoring a candidate may require a metric to also consider the passage and the question. Without cheap and reliable evaluation, progress in evaluating language model generations will be extremely slow.

To address the need for better evaluation metrics tailored for generation, we present a dataset called MOCHA, aimed at developing *learned* metrics that **MO**del the **C**orrectness of candidates using **H**uman **A**nnotation scores. MOCHA contains human judgement scores on 40K candidates, an order of magnitude larger than prior work [21]. The candidates come from six diverse QA datasets which test a wide range of RC phenomena such as commonsense reasoning and understanding narrative over movie scripts.

Instance	Score
Passage: With the aid of his daughter, Abigail, Barabas recovers his former assets. Barabas then uses his daughter's beauty to embitter Lodowick and Mathias against each other. **Question:** Why did Lodowick and Mathias fight? **Reference:** Over the affection of Abigail **Candidate:** They fight over Barabas's daughter.	5
Passage: Miss Moppet ties a duster about her head and sits before the fire. The mouse thinks she looks very ill and comes down the bell-pull. **Question:** What does the mouse think when she sees the duster on Miss Moppet's head? **Reference:** that Miss Moppet is ill **Candidate:** Miss Moppet thinks it is ill and is trying to sniff him.	2
Passage: Robin took a very long time to clean the windows of her house. **Question:** How would you describe Robin? **Reference:** a neat freak **Candidate:** a clean person	5
Passage: The strangest thing that has happened was when they were singing the Chinese National Anthem she was standing in front of the TV swaying and singing. **Question:** What is probably true about this story? **Reference:** They are watching the Olympics **Candidate:** The Olympics are watching	2

Table 3.1: **Example instances from MOCHA** highlighting the diverse phenomenon that an evaluation metric needs to handle. These phenomenon include resolving coreference, dealing with factual correctness, understanding paraphrases, and understanding semantic roles.

Using MOCHA, we train a **L**earned **M**etric for **R**eading **C**omprehension which we abbreviate as LERC. We compare LERC against two sets of baselines: (1) existing metrics such as METEOR [7] and BERTScore [167]; and (2) a sentence similarity model trained on STS-B [18]. To ensure fair comparison, we evaluate in an out-of-dataset setting: LERC is trained on all datasets *except* the one it is being evaluated on. On the test set, LERC outperforms baselines by as much as 36 Pearson correlation points and on the minimal pairs set, by as much as 26 accuracy points.

3.2 MOCHA: A Dataset of Human Generation Scores

Reading comprehension is the task of probing how well systems can understand passages of text. Framing reading comprehension as a generation problem provides a great deal of flexibility, but introduces the challenging problem of evaluation. The problems of evaluating generated text in a language model are further amplified when applied to generative reading comprehension, where the introduction of a passage and a question can add to the complexity of evaluation (Table 3.1). To handle this challenge, we propose to *train* a generative reading comprehension metric. This first requires a large set of human judgement scores to be gathered.

In this section, we present MOCHA, a dataset that pairs reading comprehension instances, which consists of a passage, question, and reference, with candidates and human judgement scores. We describe the process of gathering candidates, collecting human judgement scores, and creating minimal pairs for evaluation.

3.2.1 Constituent Datasets of MOCHA

Candidates in MOCHA come from 6 constituent QA datasets that are diverse in their domains and answer types. This ensures that training and evaluation with MOCHA does not overfit to the characteristics of any constituent dataset.

- **NarrativeQA** [76] tests reasoning about events, entities, and their relations on movie scripts and book summaries.
- **MCScript** [115] tests reasoning on stories written for a child-level reader.
- **CosmosQA** [61] tests commonsense reasoning on blogs describing everyday events.
- **SocialIQA** [138] tests social reasoning with passages constructed from a knowledge base.

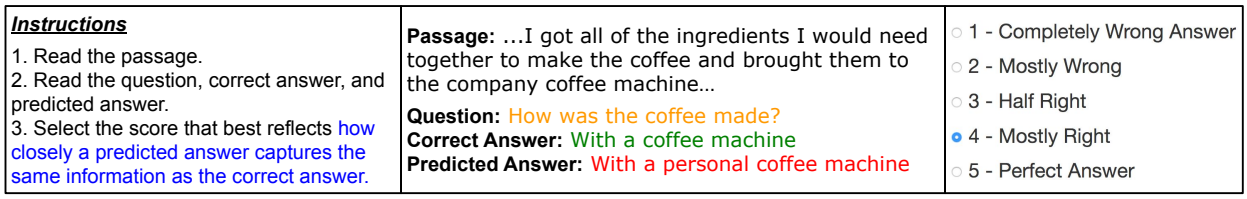


Figure 3.2: **Compressed Mechanical Turk interface** for evaluating answer correctness. Workers were asked to score (1 to 5) how similar a candidate is to a reference using the passage and the question.

- **DROP** [38] tests predicate argument structure and numerical reasoning on Wikipedia articles concerning American football games, census results, and history.
- **Quoref** [35] tests coreferential reasoning on Wikipedia articles.

NarrativeQA was created as a generative RC dataset. CosmosQA, MCScript, and SocialIQA were created as MC datasets which we re-purpose as generative datasets by using the correct choice as the reference. Our motivation for doing this is that the number of generative QA datasets is quite small, which we attribute to the quality of evaluation metrics. While the main focus of this work is in developing and evaluating metrics for generative RC. However, we wanted to see whether a learned metric could do well on span-selection datasets. We collected candidates on two span-based datasets, DROP and Quoref, to test this.

3.2.2 Collecting Candidates and Human Scores for MOCHA

Candidates on all four generative datasets are generated using backtranslation [143] and using a fine-tuned GPT-2 model [120]. We also generate candidates for NarrativeQA and MCScript using a trained MHPG model [8]. We tried using MHPG for CosmosQA and SocialIQA but candidates were of poor quality. Unique to NarrativeQA, each question has two references. We treat the second reference as a candidate to be annotated if it has low n-gram overlap with the first reference. We use a span-selection BERT-based model to generate candidates for Quoref and NAQANET [38] and

NABERT[1] models for DROP.

Models are trained on the training sets of each constituent dataset and candidates are produced on instances from the validation set (and test set if available). We filtered out candidates that exactly matched the reference. We also filtered out instances in DROP where the reference and the candidate are both numbers.[2]

Annotations are collected with Mechanical Turk using the interface in Fig. 3.2. Workers are asked to score candidate answers on an ordinal scale from 1 to 5. We start by collecting a single annotation per candidate. Following this, candidates are split into training, validation, and test sets such that all candidates from a passage are contained within a dataset split. For instances in our validation and test sets, we collect one additional human judgement score per candidate for span-based datasets, and two additional human judgement scores per candidate for generative datasets. Multiple annotations for a given candidate are averaged to form a gold annotation. We calculated inter-annotator agreement using Krippendorff's Alpha-Reliability [78] on the validation set of all 6 constituent datasets. We choose this metric because it applies to our setting, where there are multiple annotators per instance, and the annotators vary between instances. Agreement on our 6 datasets range from 0.71 to 0.92 (average = 0.82), indicating strong agreement.

In total, MOCHA contains 40K candidates, large enough for training a learned metric as well as for evaluating current and future metrics.

3.2.3 Statistics of MOCHA

Statistics of instances and dataset splits in MOCHA are provided in Table 3.2. The number of unique passages varies considerably across datasets. NarrativeQA, which has the longest passages, has few unique passages, while SocialIQA has a unique passage for each question/reference pair.

[1] https://github.com/raylin1000/drop-bert

[2] From inspection, if the reference and candidate are both different numbers the candidate is always wrong.

Dataset	Avg Pass. Len	Avg Ques. Len	Avg Ref. Len	Avg Cand. Len	# Passages			# Ques./Ref. Pairs			# Candidates		
					Train	Dev	Test	Train	Dev	Test	Train	Dev	Test
NarrativeQA	333.0	9.6	5.8	5.9	85	11	18	2249	277	500	7471	890	1707
MCScript	197.1	7.8	4.3	4.1	462	61	93	2940	390	583	7210	978	1409
CosmosQA	72.8	10.8	7.5	8.8	1064	142	212	1139	156	226	5033	683	1017
SocialIQA	15.7	7.2	3.9	3.9	3075	414	611	3075	414	611	7409	1017	1527
DROP	213.4	11.6	3.6	5.1	80	10	17	542	76	117	687	97	152
Quoref	324.0	15.8	2.3	8.2	184	24	38	1098	123	180	3259	344	509
Total					4950	662	989	11043	1436	2217	31069	4009	6321

Table 3.2: **Statistics of MOCHA** broken down by constituent datasets.

The number of candidates also varies across datasets. The most pronounced outlier is DROP, where we collected a tenth of the candidates compared to the other datasets. This is because we filtered out instances when both the candidate and reference were numbers, leaving much fewer candidates to annotate. The number of candidates outnumbers the question/reference pairs because for each pair, we generated multiple candidates using different generation sources (e.g. backtranslation, different model outputs).

3.3 Training a Learned Metric from Human Annotations

We provide details on LERC, our learned metric. LERC is initialized using BERT-base [37] We define as input a tuple consisting of a passage, **p**, a question, **q**, a reference answer, **a**, and a candidate answer, $\hat{\mathbf{a}}$. The input to BERT is structured as:

[CLS] **p** [SEP] **q** [SEP] **a** [SEP] $\hat{\mathbf{a}}$ [SEP]

BERT returns a hidden state for each input token. We use the first hidden state $\mathbf{h}_{[CLS]}$, as the pooled representation of the input.

3.3.1 Fine-Tuning with Human Judgements

Our goal is to train BERT to mimic the human judgements given a set of input tuples, $\{(\mathbf{p}, \mathbf{q}, \mathbf{a}, \hat{\mathbf{a}})\}_{i=1}^{n}$, and a set of human judgment scores, $\{y\}_{i=1}^{n}$, We apply a regression layer on top of our pooled representation (Fig. 3.3) and train with a MSE loss.

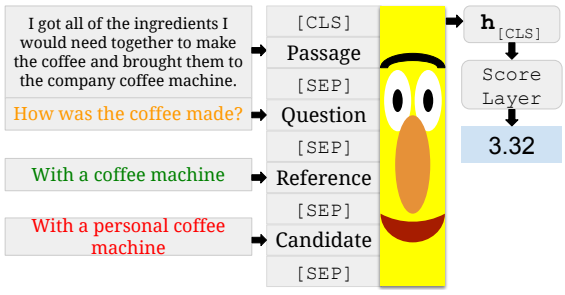


Figure 3.3: **LERC is a fine-tuned BERT model** trained on human judgment scores.

$$\hat{y}_i = \mathbf{W}\, \mathbf{h}_{i\,[CLS]}$$

$$\text{loss}_i = (y_i - \hat{y}_i)^2$$

3.3.2 Pre-Training the Learned Metric

Learning the interactions between the input components can be difficult with only human judgement fine-tuning. To overcome this, we *pre-train* on four multiple-choice QA datasets: BoolQ [31], MCTest [132], RACE [82], and MultiRC [73]. We use the same input structure as fine-tuning, but the reference and candidate are replaced by two answer choices, $\mathbf{a_1}$ and $\mathbf{a_2}$:

[CLS] **p** [SEP] **q** [SEP] $\mathbf{a_1}$ [SEP] $\mathbf{a_2}$ [SEP]

We pre-train BERT via 3-way classification to predict whether: $\mathbf{a_1}$ is the correct answer, $\mathbf{a_2}$ is the correct answer, or $\mathbf{a_1}$ and $\mathbf{a_2}$ are both correct. MultiRC has multiple correct answers per question and we create additional instances where both $\mathbf{a_1}$ and $\mathbf{a_2}$ are correct.

3.4 Experimental Setup and Results

Training LERC We use the PyTorch [117], HuggingFace Transformers [164], and AllenNLP [47] libraries to implement LERC. We pre-train LERC before fine-tuning on MOCHA. We evaluate

Metric	NarrativeQA		MCScript		CosmosQA		SocialIQA		DROP		Quoref		Avg. *r*	
	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test	Dev	Test
BLEU-1	0.403	0.472	0.181	0.260	0.660	0.670	0.595	0.549	0.409	0.387	0.674	0.578	0.487	0.486
METEOR	0.605	0.615	0.461	0.502	0.696	0.711	0.644	0.637	0.664	0.568	**0.729**	0.716	0.633	0.624
ROUGE-L	0.434	0.495	0.224	0.297	0.701	0.701	0.599	0.558	0.480	0.366	0.712	0.604	0.525	0.503
BERTScore	0.419	0.534	0.172	0.194	0.803	0.779	0.604	0.584	0.174	0.328	0.207	0.286	0.396	0.450
BERT STS-B	0.711	0.686	0.364	0.449	0.803	0.789	0.663	0.666	0.690	**0.715**	0.690	**0.750**	0.653	0.676
LERC	**0.772**	**0.738**	**0.666**	**0.694**	**0.852**	**0.824**	**0.777**	**0.799**	**0.760**	0.712	0.704	0.741	**0.755**	**0.751**

Table 3.3: **Pearson correlation to human judgement scores on MOCHA.** LERC results are from a model trained in an out-of-dataset fashion, averaged across three runs.

LERC in two settings, an out-of-dataset (OOD) setting and an all-datasets (AD) setting. In the OOD setting, we train and tune LERC on all datasets in MOCHA *except* the dataset we are evaluating on. This reflects the use case where we want to apply LERC to evaluate a new dataset where we do not have human judgement scores. In the AD setting, we train on all datasets in MOCHA and evaluate on all datasets. All results reported for LERC are the average of three runs using the best set of hyperparameters found on the validation set of MOCHA.

Baselines We compare LERC against BLEU-1 [116], ROUGE-L [92], METEOR [7], and BERTScore [167]. We also compare LERC against a BERT-base model fine-tuned on the sentence similarity task, STS-B [18]. Results for BERT STS-B are the average of three runs using the best set of hyperparameters found on the validation set of STS-B. All baselines are agnostic to the passage and the question.

3.4.1 Correlation Results

We evaluate the baselines and OOD LERC in Table 3.3 using Pearson correlation. LERC outperforms the baseline metrics despite being trained in a out-of-dataset situation. METEOR does surprisingly well despite relying on *n*-gram overlap to do evaluation. Interestingly, the sentence similarity model does better than the baseline metrics while falling behind LERC.

Dataset	Dev *r*
NarrativeQA	0.805
MCScript	0.816
CosmosQA	0.864
SocialIQA	0.820
DROP	0.796
Quoref	0.794

Table 3.4: **LERC trained on all constituent datasets.** Results are on the validation set of MOCHA.

Ablation	Avg. Dev *r*
Ref. Only	0.081
Cand. Only	0.093
Ref. & Cand.	0.742
Ques. & Ref. & Cand.	0.723
Pass. & Ques. & Ref. & Cand.	0.726
LERC (with pre-training)	0.755

Table 3.5: **Partial-input ablations of LERC** trained in an out-of-dataset fashion. Results are Pearson correlation on the validation set, averaged across all constituent datasets.

We also study whether having human judgements for a particular dataset helps. We present results in Table 3.4 on the validation set of MOCHA when LERC is trained in an AD setting. Having human judgements for the target dataset is always helpful.

3.4.2 Error Analysis of LERC

We gather the 10 validation instances per generative dataset (40 instances total) with the highest absolute difference between the human judgement score and LERC score. We categorize the errors made by LERC in Table 3.6. A large source of error is the inability to leverage the passage correctly as well as handling large lexical gaps between references and correctly paraphrased candidates. The "Other" category includes understanding semantic roles and misspellings of the reference.

3.4.3 Ablation Results

We study five ablations of OOD LERC with results in Table 3.5. All ablations do not involve any pre-training. When looking at ablations of LERC, several interesting phenomena emerge. Pre-training is important with such a complex input structure. Removing pre-training while still

Error Source	Example
Passage Use *(22.5%)*	**Passage:** Edward takes charge and the children develop and expand the farmstead, aided by the entrepreneurial spirit of the younger brother Humphrey. They are assisted by a gypsy boy, Pablo, who they rescue from a pitfall trap. **Q:** Who do the children rescue from a trap? **Ref:** Pablo **Cand:** A gypsy kid **Human Score:** 4.6 **LERC:** 1.0
Same Meaning *(35%)*	**Passage:** The story centres on the relationship between Mrs Kitty Warren and her daughter, Vivie. Mrs. Warren, a former prostitute. **Q:** What did Mrs. Warren previously do for work? **Ref:** Prostitution **Cand:** She was an escort. **Human Score:** 4.6 **LERC:** 1.06
Opposite Meaning *(15%)*	**Passage:** Sasha hated her neighbours dog as it barked all day and night so after going to the shop and buying poisonous slug pellets, Sasha gave the dog some pills. **Q:** How would you describe Sasha? **Ref:** mean **Cand:** kind **Human Score:** 1 **LERC:** 4.32
Other *(27.5%)*	**Passage:** The train was slow and ambling, so much so that we were 2 hours late when we arrived in Montreal, missing our connection. **Q:** What might be true if the freight trains didn't cause a delay ? **Ref:** They wouldn't have missed their connection **Cand:** they couldn't help noticing their connection **Human Score:** 1 **LERC:** 4.2

Table 3.6: **Error analysis of LERC.** We take the 10 validation instances per *generative* dataset (40 total) with the largest difference between the score assigned by LERC and the score assigned by humans. We then group the highest error instances by the sources of the error.

using the passage and question as input hurts performance. Ablations of LERC that do not use the passage but still have the reference and candidate as input only fall slightly behind the complete metric. One explanation is that current generative QA models may not generate many candidates that would require the metric to use the passage. Therefore, even the complete version of LERC may have learned to ignore the passage. We explore this in the following section when conducting

Difference Source	Examples
BLEU under-scores paraphrases *(92.5%)*	**Passage:** Tracy took Jesse's students to the park. Jesse had an emergency and asked her to. **Q:** How would Jesse feel afterwards? **Ref:** grateful **Cand:** thankful **LERC:** 5.0 **BLEU-1:** 0 **Human Score:** 5
LERC overly sensitive *(7.5%)*	**Passage:** By 17, Norman is the best swordsman in all of England; by the age of 18, he has a large bounty on his head, and by the age of 19, he leads the largest band of thieves in all of England. **Q:** What age was Norman when there was a bounty on his head? **Ref:** 18 **Cand:** 19 **LERC:** 5.0 **BLEU-1:** 0 **Human Score:** 1

Table 3.7: **Analysis of LERC vs BLEU-1.** We take the 10 validation instances per *generative* dataset (40 total) with the largest difference between the score assigned by LERC and the score assigned by BLEU-1. We then group these instances by the source of the difference.

an error analysis of LERC.

As sanity checks for dataset biases, we also evaluate impoverished ablations that should not perform well: when the model has access only to the reference or to the candidate. These ablations correlate quite poorly with human judgments. The correlation is slightly positive for both, however, perhaps measuring the grammaticality of a candidate, or the difficulty of matching long references.

3.4.4 LERC vs BLEU

To understand the differences in behavior between LERC and the popular BLEU metric, we collect the 10 validation instances per generative dataset with the highest absolute difference between the BLEU-1 and LERC score. We categorize the source of the differences in Table 3.7. In about 90% of the cases, the gap is due to BLEU scoring candidates too low (e.g. not capturing paraphrases). In the remaining cases, the gap is due to LERC over-scoring the candidate, usually due to the reference

and candidate being similar (e.g. both are numbers).

3.5 Summary of Contributions

In this chapter we showed how evaluating generation is a challenging learning problem. This problem is exacerbated as large language models continue to improve. To evaluate generated text, we proposed to use learned metrics. We introduce MOCHA, a dataset of human judgement scores for training and evaluating generative reading comprehension metrics. Using MOCHA, we train a learned metric, LERC, that outperforms all existing metrics and is much more robust when evaluated on a set of minimal pairs. We demonstrate that LERC is a better metric for evaluating generative reading comprehension than any previously existing metric while also showing that considerable work remains in our error analysis.

The text in this chapter is based on the publications:

- *Evaluating Question Answering Evaluation* [21] (MRQA Workshop @ EMNLP 2019).
- *MOCHA: A Dataset for Training and Evaluating Generative Reading Comprehension Metrics* [22] (EMNLP 2020).

The results presented in this chapter can be reproduced at https://github.com/anthonywchen/MOCHA.

Part II

Characterization

Chapter 4

AmbER Sets: Characterizing How Retrieval Bias Affects Language Model Generations

4.1 Introduction

The task of next-word prediction has been shown to imbue large language model with the ability to implicitly capture a surprising amount of world knowledge. However, this world knowledge is incomplete and represented imperfectly in the language model's weights. This is less than desirable, because often we require our language model to generate accurate and up-to-date information. Hence, only relying on parametric knowledge can make langauge model unreliable.

Because of this, there has been growing interests in "open-domain" tasks, where relevant documents are first retrieved from a knowledge source before a language model system performs reasoning over the retrieved documents produce an generation [23, 118]. Because success hinges on finding relevant documents, open-domain progress has been closely tied to improvements in retrieval systems[1] [84, 70, 88].

[1]For example, replacing the BM25 retriever with DPR on Natural Questions increases exact match by 15 points.

Q: Which battle did **Abe Lincoln** fight in?
A: World War II
Wikipedia Documents Ranked by BLINK:
1. Abraham_Lincoln
2. Abraham_Lincoln_in_the_Black_Hawk_War
3. Abraham_Lincoln_(captain)
4. Benjamin_Lincoln
5. Lincoln_Nebraska
6. Lincoln_England

Q: What musical instrument does **Abe Lincoln** play?
A: Trombone
Wikipedia Documents Ranked by BLINK:
1. Abraham_Lincoln
2. John_Wilkes_Booth
3. Abe_(musical)
4. Nebraska
5. Lincoln_Nebraska
6. Abe_Lincoln_(musician)

Figure 4.1: **Entity ambiguity** shown by queries for two entities (president & musician) with the name "Abe Lincoln". Retrieving the **gold document** involves disambiguating which "Abe Lincoln" each query is asking about. BLINK performs sub-optimally on the second query, as it ranks the document of the president first.

A crucial challenge when interacting with a large knowledge source (*e.g.,* Wikipedia) is entity ambiguity, the phenomenon where a single name can map to multiple entities. Resolving this ambiguity is referred to as entity disambiguation and is an important step for effective retrieval. For example, given the query *"What musical instrument does Abe Lincoln play?"*, documents about the musician should rank higher than other entities with the same name (Figure 4.1). Although entity disambiguation has been extensively studied in entity linking [57, 128, 144] and search [5, 6], in open-domain NLP, it is unclear how robust retrieval systems are when faced with ambiguous entity queries. Evaluating entity ambiguity is challenging because the popularity of entities follows a long-tail (Figure 4.2) and rare entities are seldom covered in naturally-occurring datasets.

In this paper we introduce AmbER sets, a benchmark for evaluating the entity disambiguation

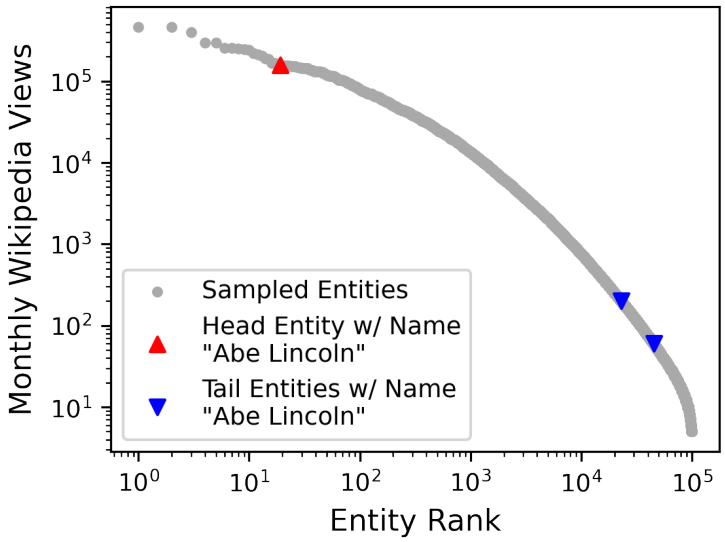


Figure 4.2: **The Long Tail of Entity Popularity:** Graph of the Wikipedia pageviews (in October 2019) for each Wikidata entity, ranked by popularity. Gray are 100k randomly sampled entities, while red/blue are entities with the name "Abe Lincoln".

capabilities of retrievers across multiple NLP tasks. Each AmbER set is a collection of Wikidata entities that share a name, and their corresponding queries for specific NLP tasks. For each set, we define the **head** entity as the most popular entity and **tail** entities as the less popular ones. By creating queries for multiple entities that share a name, AmbER sets provide an accurate test of entity disambiguation capabilities of retrievers. We show examples of AmbER sets for the question answering task in Table 4.1. We automatically create AmbER sets by mining the Wikidata knowledge graph [158] for relevant names and entities, and leveraging task-specific templates to generate inputs for three tasks: fact checking, slot filling, and question answering (Figure 4.3).

We use AmbER sets to conduct a systematic study of various retrieval systems that operate under different principles, such as token overlap and dense embedding similarity. Retrievers perform very differently on AmbER sets in terms of absolute retrieval numbers, but despite these differences, all retrievers exhibit a large degree of popularity bias, under-performing on inputs concerning tail entities. Moreover, we show that when these biased retrievers are used to produce evidence for language models to reason over, the resulting generation quality of the downstream language model is affected as well.

QID	Input	Answer	Gold Document
Q517	What wars did **Napoleon** participate in?	Napoleon Wars	Napoleon
Q3335909	What sport does **Napoleon** play?	Rugby	Napolioni_Nalaga
Q3335909	Which team does **Napoleon** play for?	Fiji National	Napolioni_Nalaga
Q117012	What movement did **Yoko Ono** participate in?	Fluxus	Yoko_Ono
Q16264827	Which sport does **Yoko Ono** participate in?	Judo	Yoko_Ono_(judoka)
Q312	Which industry is **Apple** in?	Electronics	Apple_Inc.
Q532100	What is the record label of **Apple**?	Page One	Apple_(band)
Q7714007	Who acted in **Apple**?	Ray Shell	The_Apple_(1980_film)
Q788822	Who is a cast member on **Her**?	Steve Zissis	Her_(film)
Q788822	Who is **Her**'s screenwriter?	Spike Jonze	Her_(film)
Q28441308	Who performed **Her**?	Aaron Tippin	Her_(song)

Table 4.1: **Examples of QA AmbER sets**. An AmbER set is a collection of entities that share a name, with instantiated queries for each entity. In this work, we use Wikidata to collect entities (QID). We also create queries for two more tasks, fact checking and slot filling (omitted from this table).

4.2 AmbER Sets

Retrieving relevant documents from large knowledge sources such as Wikipedia is an important first step in the open-domain pipeline. An inherent problem in working with such sources is entity disambiguation: resolving a name (mention) to an entity in the knowledge source. Entity disambiguation can be challenging because many entities share a name, and the popularity of entities follows a long-tail distribution (Figure 4.2). Despite the importance of entity disambiguation, it remains an understudied problem for open-domain NLP. We introduce AmbER sets for evaluating entity disambiguation capabilities of retrievers and analyze the role of entity popularity in disambiguation.

4.2.1 What is an AmbER Set?

We first provide an intuition for an AmbER set before concretely defining one. Consider two entities, a president and a musician, both of which have the name "Abe Lincoln" (Figure 4.1). Now, consider the query "Which battle did Abe Lincoln fight in?" and assume a retriever correctly returns the

article about the president for this query. Simply because the correct document was retrieved does not mean a retriever has the ability to disambiguate between the president and the musician, as the president is much more popular. We should only be confident in its ability to disambiguate entities if we *also* pose a query about the less popular musician and the retriever again returns the correct document (as opposed to the document about the president).

Based on this intuition, we define an AmbER set as a collection of queries that satisfy the following:

- **Criteria 1: Polysemous Name**: The queries in an AmbER set are all about entities that share a common name (*e.g.,* Abe Lincoln).
- **Criteria 2: Disparity in Popularity**: An AmbER set contains queries about both the most popular entity for a name (the head entity), *e.g.,* the president, and the less popular entities (the tail entities), *e.g.,* the musician.
- **Criteria 3: Resolvable Ambiguity**: The content of the query should be sufficient to resolve to the correct entity. The query *"Which battle did Abe Lincoln fight in?"* satisfies this criteria, because there is only one Abe Lincoln that fought in a war, while *"Where was Abe Lincoln born?"* does not since it applies to all Abe Lincolns.

We provide examples of AmbER sets for the task of question answering in Table 4.1.

4.2.2 Open-Domain Tasks

In this work, we create AmbER sets for three tasks: fact checking, slot filling, and question answering. We consider these three tasks for three reasons. First, these three set of tasks are diverse in nature. In this work, slot filling is a generation task, question answering is a span selection task, and fact checking is a classification task. Second, the training sets available for each task are quite disparate. The largest fact checking training set, FEVER [153], has 80k instances, while the slot

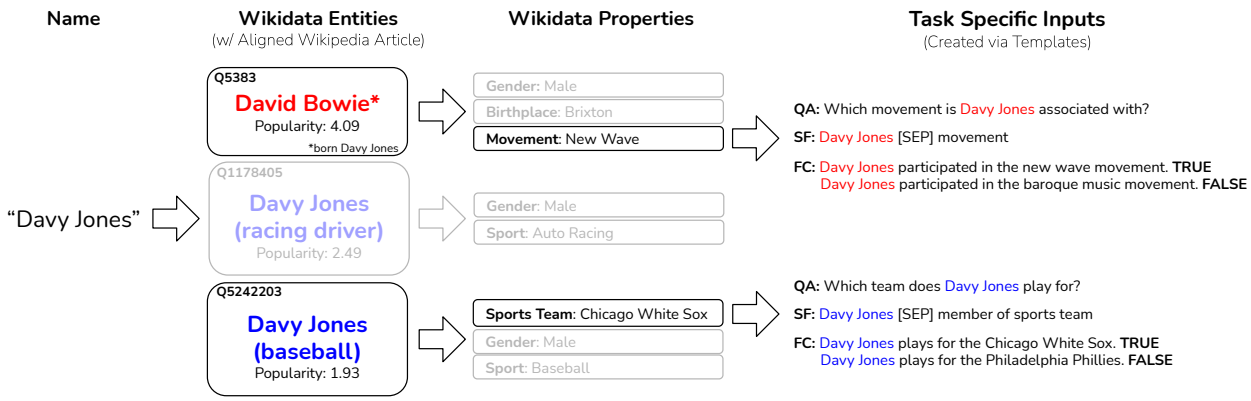


Figure 4.3: **Automated creation of AmbER sets.** We collect sets of entities from Wikipedia that share a name, where the most popular entity is the head entity (in red) and others are tail entities (in blue), along with their properties and associated values. We filter out properties that do not help distinguish entities in the set (gray-ed out), and remove entities that do not have any properties remaining. From the remaining properties, we instantiate queries via templates for three tasks: question answering (QA), slot filling (SF), and fact checking (FC).

filling dataset, T-REx [41], has over 2 million instances. The final reason we study these three tasks is that their inputs are short and easy to create.

4.3 Creating AmbER Sets

While AmbER sets can be manually created, doing so can be time-consuming, requiring a human to manually scour a knowledge base for polysemous names and related entities before manually writing queries for those entities. Instead, we present a pipeline for automatically creating AmbER sets using the Wikidata knowledge graph [158]. In this section, we describe two different *collections* of AmbER sets, and discuss our automatic pipeline for creating AmbER sets.

4.3.1 Two Collections of AmbER Sets

A natural question is "How do retrievers handle entity ambiguity when two entities have the same entity type as opposed when they have different types?". To answer this question, we create two

collections of AmbER sets. The first is AmbER-H, a collection of AmbER sets where all entities are humans. The choice to restrict AmbER-H to humans is motivated by the fact that humans have properties that help distinguish themselves from other humans, generally based on occupation. The second is AmbER-N, a collection of AmbER sets where all entities contained are non-humans, and disambiguation of a name is between non-human entities with different entity types. This is because a non-human entity, like a movie, does not generally have a single distinguishing property to distinguish from other movies. This makes it natural to compare non-human entities to other non-human entities with different types. We specify the entity types in each collection in Table 4.2.

4.3.2 Automatic Creation of AmbER Sets

We now describe a pipeline to automatically create AmbER sets for three tasks: fact checking, slot filling, and question answering. We provide a visualization of the pipeline in Figure 4.3.

Collecting Names and Entities We begin by collecting all entity aliases[2] in Wikidata. From these aliases, we filter for those that are shared by multiple Wikidata entities. Each entity in Wikidata is represented by a unique QID. The entities must have an entity type from Table 4.2 depending on the collection we are collecting AmbER sets for. Each alias and associated entities form the basis for an AmbER set. Within each set, we define the head and tail entities based on the number of Wikipedia page views for the month of October 2019. We filter out AmbER sets where the percentage gap in popularity between the head entity and the most popular tail entity is less than 10% to account for noise in the monthly page views.

Collecting Distinguishing Properties We gather properties and associated values for each entity from Wikidata. We only retain properties that are in a specified list (Table 4.2), as they are useful for resolving ambiguity *(Criteria 3)*. We also filter a property if two entities within an AmbER set

[2]Aliases are all possible names for an entity.

	Entity Type	Property *(PID)*	Percent
AmbER-H	Human	instrument *(P1303)*	17.01
		movement *(P135)*	2.04
		appears in *(P1441)*	0.08
		killed by *(P157)*	0.19
		PhD student *(P185)*	0.42
		military branch *(P241)*	12.22
		sports position *(P413)*	12.82
		sports team *(P54)*	17.25
		battles or wars *(P607)*	12.29
		sport *(P641)*	25.68
AmbER-N	Album	performer *(P175)*	16.57
		record label *(P264)*	7.11
		tracklist *(P658)*	0.21
	Business	industry *(P452)*	0.65
	City	population *(P1082)*	0.24
	Film	cast member *(P161)*	27.14
		screenwriter *(P58)*	18.28
	Literary Work	author *(P50)*	11.13
	Musical Group	record label *(P264)*	2.1
	Song	performer *(P175)*	4.42
		record label *(P264)*	0.62
	TV Series	cast member *(P161)*	2.01
		# seasons *(P2437)*	1.85
		screenwriter *(P58)*	0.21
	Written Work	author *(P50)*	7.43

Table 4.2: **Distinguishing Properties** selected to create queries based on whether they are sufficient to resolve ambiguity. We provide the percent breakdown of how often each property occurs in each AmbER collection.

have that property, ensuring that the remaining properties can be used to disambiguate between entities with the same name. These properties are used to instantiate the queries.

Aligning Entities to Wikipedia We use the KILT Wikipedia snapshot [118] as the knowledge source for AmbER sets for better reproducibility. Each Wikipedia document in KILT has an associated QID. For each entity, we find all Wikipedia documents with that associated QID. After

this alignment, we apply a round of filtering on the tuples. For each tuple, we check that the value of the tuple is within the first 350 tokens of the aligned Wikipedia article. If not, we remove the tuple.[3] Aligned Wikipedia articles that contain the tuple value serve as gold documents.

Instantiating AmbER Instances Recall that our goal was to create AmbER sets for three tasks: fact checking, slot filling, and question answering. We are able to create queries for all three tasks simultaneously using the collected Wikidata tuples. For question answering and fact checking, we use templates based on properties to instantiate inputs. Three of the authors wrote a template each for each property for the two tasks. Duplicate templates are removed, resulting in an average of 3 question answering templates per property and 2.7 fact checking templates per property.

For slot filling, we create a single input from each Wikidata tuple by concatenating the AmbER set name with the property name, and using the value of the tuple as the answer. For question answering, we also create a single input for each tuple by filling in the template with the AmbER set name and using the value of the tuple as the answer. For fact checking, we create two inputs for each tuple, one claim that is true using the tuple value and one claim that is false. The false claim is created by finding the most popular value for the tuple property that does not match the tuple value[4].

4.3.3 Dataset Statistics

We provide statistics for AmbER sets in AmbER-H and AmbER-N in Table 4.3. On average, each AmbER set has about three entities that share the same name. Of these three entities, on average, only two have properties after filtering. In total, our AmbER sets contain about 80k task-specific input queries.

[3]This reduces the number of tuples for AmbER-H from 17,079 to 5,942 and for AmbER-N from 22,219 to 13,804.

[4]The most popular instrument in Wikidata is piano. Therefore, given the true claim *"Abe Lincoln played the trombone."*, the false claim would be *"Abe Lincoln played the piano."*.

	AmbER-H	AmbER-N
# AmbER Sets	2,093	5,237
Averages per AmbER Set		
...# entities	2.98	2.42
...# entities w/ properties	2.03	2.06
...# properties	2.84	2.64
# Input Queries	23,768	55,216
...Question Answering (QA)	5,942	13,804
...Slot Filling (SF)	5,942	13,804
...Fact checking (FC)	11,884	27,608

Table 4.3: **Statistics of AmbER collections.**

4.4 Experimental Setup

Retrieval Systems The primary focus of this work is to evaluate entity ambiguity of retrieval systems. We consider four retrievers based on different retrieval paradigms. The first three are TF-IDF, a token-based retriever using sparse embeddings, DPR [70], a dense embedding based retriever, and BLINK [166], a linker-based retriever which ranks documents based on input entities. These three retrievers have been thoroughly evaluated on a number of open-domain tasks in [118] with no obvious winner across tasks. Encouraged by the disambiguation success on rare entities by [114], we also evaluate a retriever based on Bootleg, another entity linker.

Downstream Models The dominant approach to open-domain tasks is a two-stage process where a retriever first finds relevant documents, followed by a downstream langauge model that processes these documents to produce an answer. We evaluate the end-to-end performance on AmbER sets by fine-tuning downstream language models on our tasks of interest. For fact checking, we fine-tune a BERT classifier [36] on FEVER [153]. For question answering, we fine-tune a RoBERTa model [93] on Natural Questions [81]. For slot filling, a generation task, we fine-tune a BART model [87] on T-Rex [41]. We use the AllenNLP and HuggingFace Transformers library to fine-tune our downstream models [46, 165].

	Retriever	Fact Checking (FC)				Slot Filling (SF)				Question Answering (QA)			
		All	Head	Tail	∀	All	Head	Tail	∀	All	Head	Tail	∀
AmbER-H	TF-IDF	17.3	28.5	8.2	0.0	18.8	31.9	8.1	0.0	16.7	28.2	7.3	0.1
	DPR	18.1	23.9	13.3	0.1	8.0	11.6	5.1	0.3	13.1	19.6	7.9	1.1
	BLINK	**55.9**	**64.4**	**49.0**	**5.6**	38.2	57.0	22.9	11.5	31.7	40.5	24.6	6.6
	Bootleg	34.8	43.0	28.2	0.7	**56.5**	**63.9**	**50.6**	**25.3**	**67.2**	**77.1**	**59.1**	**36.1**
AmbER-N	TF-IDF	9.4	13.6	4.9	0.0	13.4	21.0	5.2	0.2	13.9	21.7	5.4	0.3
	DPR	**36.9**	**48.0**	**24.8**	**4.4**	29.9	40.9	18.0	6.0	36.2	49.2	22.2	9.3
	BLINK	11.7	13.9	9.4	0.0	5.7	7.3	3.9	0.7	35.2	44.7	24.9	10.1
	Bootleg	3.5	4.6	2.4	0.0	**52.3**	**61.3**	**42.5**	**22.4**	**59.8**	**69.5**	**49.3**	**29.0**

Table 4.4: **Top-1 retrieval results** on each collection of AmbER sets. We report accuracy@1 results on all instances as well as results on instances about head entities and instances about tail entities. We also report a set-level metric, *all correct* (∀), the percentage of AmbER sets where *all* inputs had the correct document retrieved.

4.5 Results

In this section, we evaluate existing open-domain NLP pipelines using AmbER sets. We also conduct a user study to evaluate the quality of the queries in the AmbER sets.

Top Document Retrieval We report retrieval performance in Table 4.4 in terms of retriever accuracy@1 (the % of instances where the first retrieved document is the gold document). For each task, we report values on the entire AmbER set ("All"), as well as instances corresponding only to "Head" entities or to "Tail" entities. We also report a metric we call *all correct* (∀), the fraction of AmbER sets in which all queries had the correct document retrieved. All retrievers do better on head entities compared to tail entities. Since BLINK, Bootleg, and DPR are initialized using pre-trained language models, they may have a predisposition towards being biased to more popular entities. However, we find TF-IDF also does better on head entities, perhaps because more popular entities have longer Wikipedia pages, possibly increasing term-frequency scores. Second, there are large discrepancies between a retriever's performance on different tasks for an AmbER collection. For instance, DPR does substantially worse on slot filling compared to its performance on question answering. This is surprising since queries for all tasks are created from the same set of Wikidata

		FC		SF		QA	
		Head	Tail	Head	Tail	Head	Tail
AmbER-H	TF-IDF	19.5	67.5	28.2	75.7	27.9	76.1
	DPR	1.2	10.0	2.3	23.8	2.6	27.0
	BLINK	9.8	32.2	14.0	58.2	4.4	27.6
	Bootleg	6.2	24.7	9.3	30.5	3.7	28.7
AmbER-N	TF-IDF	10.1	49.9	22.0	76.9	23.0	76.8
	DPR	6.2	32.2	9.1	48.3	8.7	44.0
	BLINK	5.8	22.8	5.1	32.2	5.5	31.9
	Bootleg	7.7	26.1	16.1	36.2	7.8	31.6

Table 4.5: **Entity confusion** measures the % of queries the gold document ranks **worse** (lower) than a document for another entity with the same name (*i.e.*, another entity in the AmbER set). Retrievers are four times as likely to exhibit this when dealing tail queries.

tuples. Finally, we find that retrievers are mostly incorrect on getting all the queries in a set correct, with some receiving a $\forall$ score of 0 on some tasks. Overall, we find that the Bootleg retriever on average does the best across tasks, however there is significant scope for improvement.

Entity Confusion To explicitly evaluate whether retrievers get confused by entities in the same AmbER set, we compute *entity confusion* for retrievers defined as the percentage of queries where the retriever ranks a document for an incorrect entity *from the same AmbER set* over the gold document (Table 4.5). We find that across retrievers, tasks, and AmbER collections, entity confusion is twice as high for tail entity inputs. This result indicates that the popularity of an entity for a given name plays a significant role in retrieval performance.

Effect of Popularity Gap Since the difference in popularity between the head and tail entities can vary considerably, these results obfuscate the effect of the *size* of the popularity gap. We explore how the gap in popularity between head and tail entities translates to the gaps in performance on their associated queries. For a head entity with popularity p_h and a tail entity with popularity p_t from the same AmbER set, we calculate popularity gap, $\frac{p_h - p_t}{p_t}$, and bin associated head/tail inputs

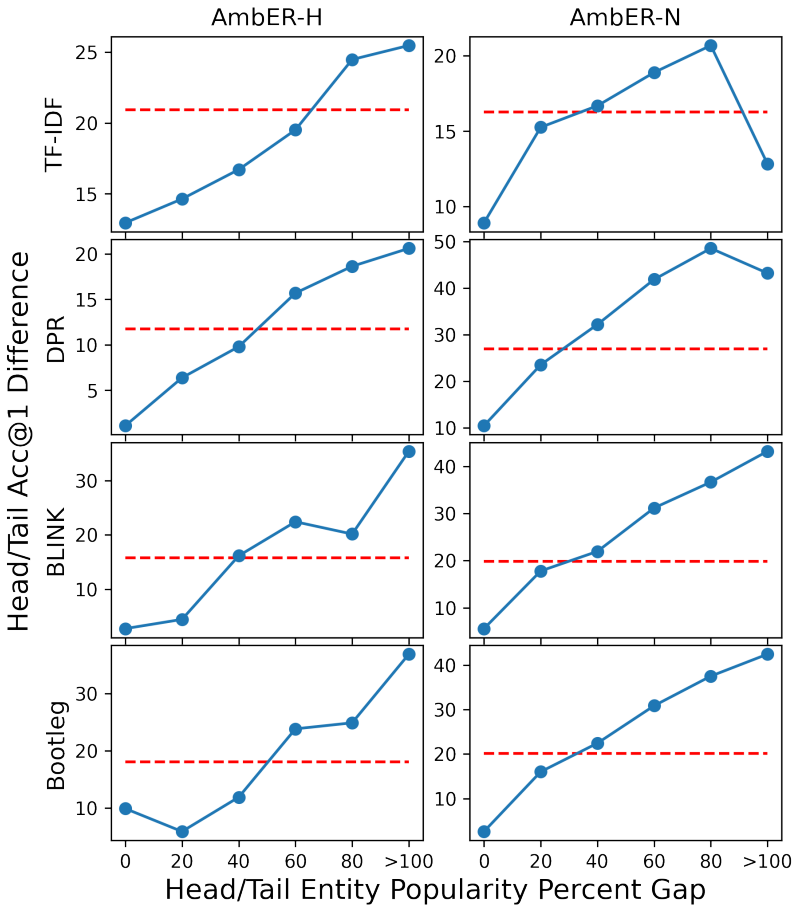


Figure 4.4: **Popularity Gap vs Retrieval Gap.** We bin QA queries of pairs of head and tail entities based on the popularity gap between the entities. For each bin, we calculate the retrieval accuracy@1 difference on the head and tail queries. Larger popularity gaps tend to lead to a wider gaps in retrieval performance. The red line is retrievers' performance gaps between head and tail queries on the entire collection.

based on the gap[5]. For each bin, we calculate the difference in accuracy@1 between the head and tail entity queries. Results for QA AmbER sets (Figure 4.4) show that there is a strong correlation between the popularity gap and the difference in performance.

End to End Results We evaluate end to end performance in several evaluation settings with all results provided in Table 4.6. The metrics used are F1 for slot filling and question answering and accuracy for fact checking. In the "oracle" setting, we directly provide the downstream NLP model

[5]Bin width of 20%. Queries with a popularity gap higher than 100% are binned into the highest bin.

	Task	System	Results		
			All	Head	Tail
H	FC	BERT (Oracle)	77.7	73.6	80.3
		BERT + BLINK	59.8	60.1	57.7
	SF	BART (Oracle)	83.9	85.0	83.5
		BART + BLINK	34.4	38.2	32.6
	QA	BERT (Oracle)	71.4	77.7	83.0
		BERT + BLINK	27.5	33.8	22.3
N	FC	BERT (Oracle)	66.6	63.9	69.5
		BERT + DPR	60.9	61.4	60.4
	SF	BART (Oracle)	82.1	80.1	84.3
		BART + DPR	18.6	18.6	18.6
	QA	BERT (Oracle)	83.5	85.1	81.8
		BERT + DPR	26.0	31.3	20.4

Table 4.6: **End-to-end performance on AmbER sets**. We evaluate systems in an oracle setting, where the gold document is provided, and a retrieval setting, where 20 documents are provided from a retriever.

the gold document, and find that the gap between head entities and tail entities is fairly small. This suggests that in closed NLP settings, where the gold document is known, entity disambiguation is not a major concern.

User Study AmbER sets are created in a largely automatic process, raising questions about data quality. To address these questions, we conduct a small user study on AmbER sets to evaluate whether the queries are resolvable by humans. We present a query from a QA AmbER set along with three documents for the entities from the same AmbER set, one of which is the gold document. We first ask the user to select the relevant document, then we ask the user to select an answer span from the selected document. In total, we asked 7 subjects to examine about 120 queries across AmbER-H and AmbER-N, and computed their accuracy in selecting the correct document and answer (Table 4.7). We also compare retrievers for this task, *i.e.* select from 3 documents for the same queries, and find that humans perform very well on the document selection task compared to retrievers on both sets. We also compare the accuracy of answer selection, and see that the closed

System	AmbER-H		AmbER-N	
	Doc Acc.	EM	Doc Acc.	EM
TF-IDF	43.3	-	50.3	-
DPR	69.1	-	68.3	-
BLINK	69.1	-	74.1	-
Bootleg	79.6	-	73.1	-
BERT	-	71.8	-	75.5
Human	**100**	**78.8**	**97.9**	**77.5**

Table 4.7: **User study on AmbER QA.** Humans are nearly perfect in identifying the correct document for each query (Doc Acc), while existing retrievers frequently fail. When the gold document is provided to downstream NLP models (BERT), they do almost as well as humans in answering the question (EM).

domain NLP model (fine-tuned BERT) is as almost accurate as humans on the same set of queries[6]. This further confirms that closed NLP models are not the source of bias towards head entities, but the retrievers are.

4.6 Summary of Contributions

Because large language model are a fuzzy and imperfect representation of the factual state of the world, open-domain pipelines, where a retrieval systems finds relevant evidence before a language model reasons over it, is vital. However, bias in retrieval systems can lead to faulty evidence retrieval, ultimately affecting the generation capabilities of language models. One such bias is popularity bias, where the most popular entity for a name is preferred over others with the same name. This is termed entity ambiguity, and is an inherent problem in retrieval as many entities can share a name. For evaluating disambiguation capabilities of retrievers, we introduce AmbER sets. An AmbER set is a collection of task-specific queries about entities that share a name, but the queries have sufficient content to resolve the correct entity. Our experiments demonstrate the struggles

[6]The relatively low answer score is due to artifacts in using EM for QA evaluation, and is consistent with human performance on span selection [125]).

of current retrievers in handling entity ambiguity. In particular, we find that the popularity of an entity in relation to other entities that share a name plays a significant role during disambiguation. Moreover, we evaluate how this bias in retrieval systems affects downstream language models and find a similar drop in performance when generating facts about less popular entities when using a biased retrieval system.

The text in this chapter is based on the publications:

- *Evaluating Entity Disambiguation and the Role of Popularity in Retrieval-Based NLP* [19] (ACL 2021).

The results presented in this chapter can be reproduced at https://github.com/anthonywchen/AmbER-Sets.

Chapter 5

Knowledge Conflicts: Characterizing How Language Models Balance Contextual and Parametric Knowledge

5.1 Introduction

As previously mentioned, the world knowledge language models can learn is vast, but is represented in an incomplete and imperfect fashion in the model weights. This type of knowledge is referred to as *parametric knowledge*. Knowledge-dependent tasks, such as open-retrieval question answering (QA), require this expansive "world knowledge", common sense, and reasoning abilities in a precise way. Thus, state-of-the-art approaches typically follow a retrieve-and-read setup [24], where the retriever sources relevant documents, and the reader produces an answer from these. In this sense, there are two sources of knowledge contributing to model inference with an ambiguous and opaque division of labour. The first is the implicit parametric knowledge (*i.e.*, their learned weights) instilled by pre-training and fine-tuning [119]. The second is contextual knowledge, usually sourced as

Question: Who did US fight in world war 1?
Original Context: The United States declared war on **Germany** on April 6, 1917, over 2 years after World War I started . . .
Original Answer: **Germany**

Model Prediction: ***Germany***

Question: Who did US fight in world war 1?
Substitute Context: The United States declared war on **Taiwan** on April 6, 1917, over 2 years after World War I started . . .
Substitute Answer: **Taiwan**

Model Prediction: ***Germany***

Figure 5.1: **Knowledge Substitution:** A **substitute example** is derived from the **original example** by replacing the original answer, **Germany**, with a similar type of answer, *i.e.* **Taiwan**. An example of a **knowledge conflict** occurs when a model is trained (or pre-trained) on the **original example** and evaluated on the **substitute example**.

passages of text from the retriever [42].

As a testament to their memorization abilities, large language models can produce competitive results relying only on their own parametric knowledge, without access to relevant documents [15, 133]. However, this memorization behaviour has manifested in a penchant to *hallucinate*, or parrot answers memorized during training, completely ignoring relevant documents when provided [79, 9]. This memorization behaviour violates the expectation that the reader produce answers consistent with the retrieved information, diminishing interpretability of the system. More problematically, this behaviour inhibits the model's ability to generalize to evolving knowledge and time-dependent answers, not found in training [54, 140].

Our objective is to understand how systems employ parametric and contextual knowledge together by studying knowledge conflicts: situations where the contextual knowledge contradicts with knowledge learned during pre-training or fine-tuning. Because the space of knowledge conflicts is broad, we restrict ourselves to the space of *entity-based* conflicts – restricted to named entity substitutions. We create an automated framework that identifies QA instances with named entity

answers, then substitutes mentions of the entity in the gold document with an alternate entity, thus changing the answer (Figure 5.1). Our framework is extensible and flexible, allowing entities mined from various sources (entities in datasets, or knowledge graphs like Wikidata [158]), and with custom substitution policies.

We use our automated framework to create substitution instances for Natural Questions [81] and NewsQA [156]. Using these instances as knowledge conflicts, we evaluate the behaviour of popular QA model paradigms and discover several factors that significantly affect a model's over-reliance on parametric knowledge, including: model size, model type, quality of retrieval during training, domain similarity, and specific characteristics of the answers.

5.2 Creating Knowledge Conflicts Via Substitution

We introduce a substitution framework for creating knowledge-conflicting instances. The framework maps a QA instance $x = (q, a, c)$, with query q, answer a, and the context passage c in which a appears, to $x' = (q, a', c')$ where a is replaced by substitution answer a' as the gold answer, and where all occurrences of a in c have been replaced with a', producing new context c'.

This substitution framework extends partially-automated dataset creation techniques introduced in Chapter 4 for creating AmbER sets. Our dataset derivation follows two steps: (1) identifying QA instances with named entity answers, and (2) replacing all occurrences of the answer in the context with a substituted entity, effectively changing the answer. We provide tools to identify coherence-preserving substitutions and create substitutions with certain characteristics (e.g. semantic equivalence, or popularity score on Wikipedia).

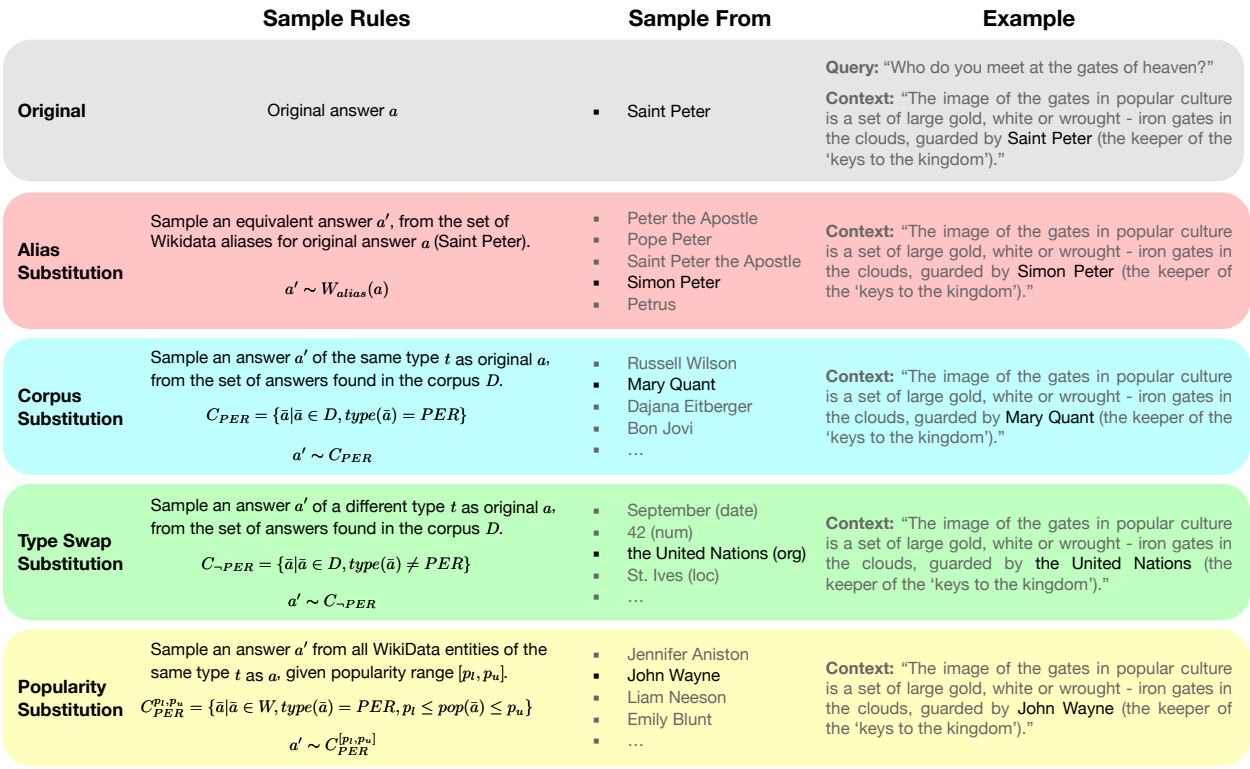


Figure 5.2: **Substitution Methods.** An illustration of substitution types and their rules, whereby the original answer a is replaced by a substitution answer a', sourced either from Wikidata W or the set of answers appearing in the training dataset D. $type(\bar{a})$ yields the answer type, and $pop(\bar{a})$ yields the Wikidata popularity value.

5.2.1 Identifying Named Entity Answers

As our focus is *entity-based* knowledge conflicts, our first step identifies instances where the answer is a named entity. We leverage the SpaCy named entity recognizer and entity linker to identify gold answers that are named entities, their corresponding entity types, and their ID in the Wikidata graph.[1] This allows us to gather auxiliary information about the entity, such as entity popularity.

We focus on five entity types that are well represented in question answering datasets: *person* (`PER`), *date* (`DAT`), *numeric* (`NUM`), *organization* (`ORG`), and *location* (`LOC`). Tracking an answer's entity type allows us to create coherent substitutions. QA instances without a gold answer among these

[1] SpaCy NER: https://spacy.io/usage/linguistic-features#named-entities, EL: https://v2.spacy.io/usage/training#entity-linker.

five entity types are filtered out. When applying substitutions, we replace all spans of the answer entity in the context with a substituted entity, according to the substitution policy.

5.2.2 Types of Substitutions

There are many possible substitution policies which evaluate different properties. In Figure 5.2, we illustrate the versatility of our framework, highlighting the types of knowledge substitutions we experiment with in this work. An advantage of this framework over recent similar work [140] is that it is extensible. Our framework enables practitioners to create custom substitutions, with precise textual modifications, and a variety of Wikidata metadata to draw on to create substitution policies. We describe substitutions derived from our framework used herein to test hypotheses of model behaviour.

Corpus Substitution (CS) replaces answer a with another entity a' from the same dataset (*in-domain*). The substitution entity is randomly sampled from the gold answers found in the same dataset D, such that a and a' share the same entity type (*i.e.*, for $type(\cdot) \in \{\texttt{PER}, \texttt{DAT}, \texttt{NUM}, \ldots\}$, $type(a) = type(a')$).

Type Swap Substitution (TSS) replaces answers a with a nonsensical in-domain entity a'. The substitution entity is randomly sampled from the gold answers found in the same dataset D, such that a and a' have **different** types, $type(a) \neq type(a')$. Nonsensical answer substitutions are useful to test model robustness or common sense.

Popularity Substitution (PS) tests how the popularity of the substituted entity affects reliance on parametric knowledge. We replace a in c with a', which is a randomly sampled Wikidata entity of the same type as a. The popularity of a', $pop(a')$, is between user-specified bounds p_l and p_u,

Sub. Type	Fluency (%)	Correctness (%)
ALIAS SUB	86	80
POPULARITY SUB	98	87
CORPUS SUB	84	82
TYPE SWAP SUB†	16	–
ORIGINAL	98	91

Table 5.1: **Human Evaluation** of 80-100 Natural Questions examples per row. Substitutions yield reasonable fluency and correctness compared to original examples. † Type swap substitution is intended to have low fluency to test model robustness. Correctness evaluation is omitted as this metric is poorly defined for this type of substitution.

measured in monthly Wikipedia page views, as estimated from October 2019.

Alias Substitution (AS) replaces answer a with a semantically equivalent paraphrase a', sampled from the list of a's Wikidata aliases $W_{alias}(a)$.

5.2.3 Substitution Quality

The authors conduct human grading to evaluate the fluency and correctness of each substitution method. For *fluency*, the annotator is asked whether the substituted answer a' is a grammatical replacement within the given context c'. For *correctness*, the annotator is given the query-context pair (q, c') and asked to highlight the span that answers the question. Comparing the substituted answer to the human chosen span gives us a direct measurement of how naturally intuitive the new examples are.

Table 5.1 shows the automated substitution methods retain fluency and correctness just above 80% for Natural Questions — slightly less than the original examples. These metrics suggest the current framework is effective for average-case analysis of model interpretability, and certain training methods. However, there are quality limitations with respect to human-curated resources (0-14% fluency gap, 4-11% correctness gap), and this resource is most effective for tasks and datasets with

entity-based answers, easily classified by a corresponding Named Entity Recognition model. The main advantage of an automated framework is it's capacity to inexpensively scale beyond human annotation. Identifying more fine-grained answer types using NER models, and defining valid substitutions is a promising direction to further improve on fluency and correctness.

5.3 Experimental Setup

5.3.1 Datasets

Training We adopt a common and human-sourced query distribution in open-domain question answering, using [81]'s Natural Questions (NQ) for training. For certain experiments we train with NewsQA [156], a news-oriented dataset with examples whose answers are prone to change over time (susceptible to knowledge conflicts).

Inference At inference time we create knowledge conflicts for (1) the training set (to understand knowledge conflicts on data the models have seen), (2) the development set, as well as (3) an out-of-distribution (OOD) set, either the training set for NQ or NewsQA, depending on which was not used at training time. For simplicity we use the *MRQA* Workshop Shared Task's versions for each of these datasets where the same tokenization and pre-processing are used [42].[2]

[90] show the Natural Questions training and development sets contain many similar queries and answers. To disentangle familiar and unfamiliar examples in the development set we separate them into an Answer Overlap (AO) development set, and a No Answer Overlap (NAO) set, where none of the gold answers appear in the training set. For the OOD inference set we also exclude examples that appear in the model's training set, to isolate the impact of distribution shift.

[2] https://github.com/mrqa/MRQA-Shared-Task-2019.

5.3.2 Models

This work evaluates retrieve-and-read QA systems: the retriever finds relevant documents and the reader produces an answer using these documents.

Retriever We use dense passage retrieval (DPR) [68] as the primary retrieval system. In some experiments we also use a sparse retriever, TF-IDF [127, 100]. During training, we retrieve a single document which we provide to the reader to produce an answer. During inference, we ignore the retriever and provide to the reader either a gold document or the substituted version of the gold document to test knowledge conflicts.

Generative Reader In this setting, a model receives a query concatenated with contextual text and *decodes* a prediction. Our generative model is a T5 model [122] and for simplicity, we train using a single retrieved passage.[3] While training with multiple documents would yield better results [63], training with only a single document as input allows us to better decouple the interactions between the reader and the retriever.

We choose to evaluate a simple T5 reader model because it is the consistent component across high-performing retrieval-based QA models [63, 91, 75], and thus preserves the generality of our findings. Where various implementations differ slightly, we also explore the impact of model size and quality of retrievers used at training time.

Extractive Reader We also experiment with a span-extraction QA model, where the predicted answer is a span of text taken directly from the context c. We use the RoBERTa [94] implementation from HuggingFace [165] and hyperparameters from [96].[4] By necessity, this model is trained with

[3]Default implementation and hyperparameters: https://github.com/google-research/text-to-text-transfer-transformer.

[4]Training pipeline available at https://github.com/huggingface/transformers/tree/master/examples/question-answering.

gold passages that always have a gold span.

5.3.3 Metrics

To understand a model's propensity to rely on memorized answers, we narrow our focus to examples that a model correctly answered on the original, unaltered example. Using the standard SQuAD-based Exact Match measurement [126], we compare model predictions on examples before (x) and after (x') the substitution has been applied. We then measure the fraction of times the model predicts: the *Original* answer (p_o), the *Substitute* answer (p_s), or an *Other* answer altogether, on x'.

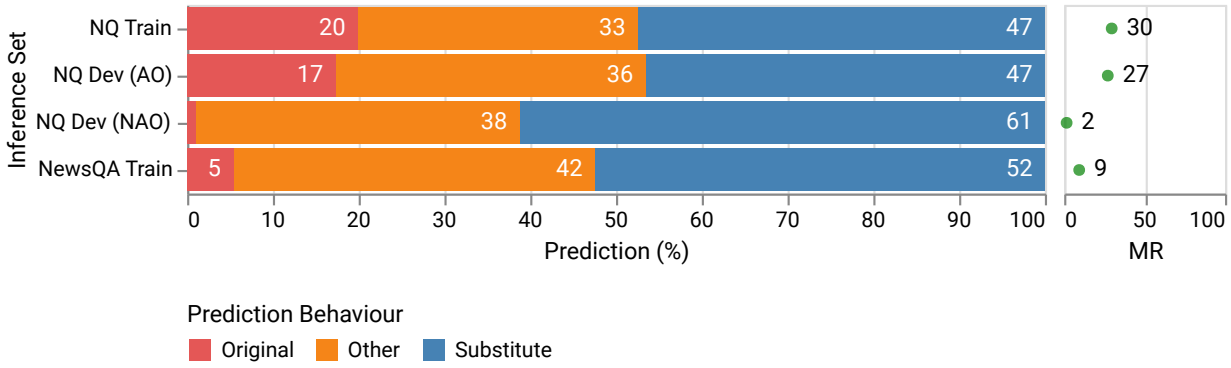
The Memorization Ratio (M_R) measures how often the model generates the original answer (parametric knowledge) as opposed to the answer in the context (contextual knowledge). This estimates the *overstability* of the model — it's brittleness to changing information.

$$M_R = \frac{p_o}{p_o + p_s}$$

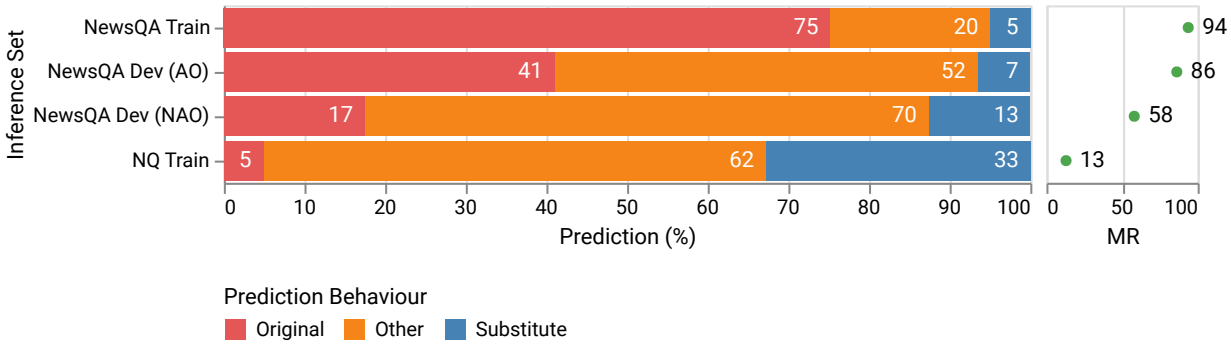
5.4 Results

5.4.1 Primary Results

Our results on *corpus substitution* test how a QA model chooses answers when the substituted answer is in the same distribution as the training set. Figure 5.3 measure how often the model generates the *Original* answer, the *Substitute* answer, or some *Other* answer altogether on x'. To confirm the observed phenomena is not dataset specific, Figure 5.3a presents results for the model trained on Natural Questions (NQ), and Figure 5.3b for the model trained on NewsQA. In each case, we evaluate on the training set, validation set (with and without answer overlap), and an out-of-distribution dataset.



(a) Trained on Natural Questions (NQ) Train



(b) Trained on NewsQA Train

Figure 5.3: **Corpus Substitution.** Inference behaviour and memorization ratio (M_R) of generative models evaluated on corpus substituted instances.

Ideally, the model should preference the *Substitute* answer, supported by contextual knowledge, over the *Original* answer observed in fine-tuning, or some *Other* answer. However, the model predicts the *Substitute* answer a' rarely more than 50% of the time for the NQ model, and significantly less for the NewsQA model. Instead, the model reverts back to predicting the *Original* answer seen in training, ignoring the contextual passage, up to 20% of the time for NQ, and 75% for NewsQA. Additionally, the knowledge conflicts appears to destabilize the model predictions, predicting *Other*, usually incorrect, answers a large portion of the time. (See Section 5.4.3, where the *Other* category is discussed in detail.) These results demonstrate that common generative QA reader models are unlikely to trust the retrieved information over their parametric memory (learned at training time).

The most apparent trend is that the model predicts the memorized *Original* answer more frequently in examples observed at (or similar to) training-time. While the memorization ratio (M_R) falls

Inference Set	Model Prediction Category on x'			
	ORIG.	OTHER	SUB.	AVG.
NQ TRAIN	63.3	87.1	69.9	74.2
NQ DEV (AO)	62.0	85.9	70.2	74.4
NQ DEV (NAO)	66.7	83.5	52.0	64.1
NEWSQA	75.7	77.1	60.8	68.5

Table 5.2: **Model Uncertainty.** For the NQ trained model, we compute the percentage of time in which $p(x) > p(x')$, indicating the model was more confident in it's prediction made for the original example x than the corpus substitution example x'.

significantly for Dev NAO and the out-of-distribution (OOD) sets, it is still non-trivial — nor is the resultant tendency for the model to predict *Other* answers, where it had correctly generated the *Original* answer, when supported by contextual knowledge in x.

How is Model Uncertainty Affected? Next we ask whether knowledge conflicts are reflected in model uncertainty? If model predictions *are* relatively uncertain when knowledge conflicts occur, then confidence thresholds might permit the system to abstain from answering some of these questions. In Table 5.2 we compute how often model confidence is greater on the original example x than the modified example x', broken down by prediction category and inference set.

Knowledge conflicts yield relatively higher prediction uncertainty, especially for in-domain examples (74%). Uncertainty is also elevated for out-of-distribution examples in NQ Dev (NAO) or NewsQA (64% and 69% respectively). In particular, uncertainty is highest for instances where the model predicts *Other*. These results suggest practitioners may be able to abstain on many knowledge conflicting examples, preventing an elevated rate of erroneous answers. However, the abstention solution simply exchanges incorrect answers for no answers, without addressing the primary issue of a model ignoring contextual knowledge.

How is Inference Stability over Semantically Equivalent Answers? *Alias substitution* swaps the answer with a semantically equivalent paraphrase, effectively isolating the impact of a benign

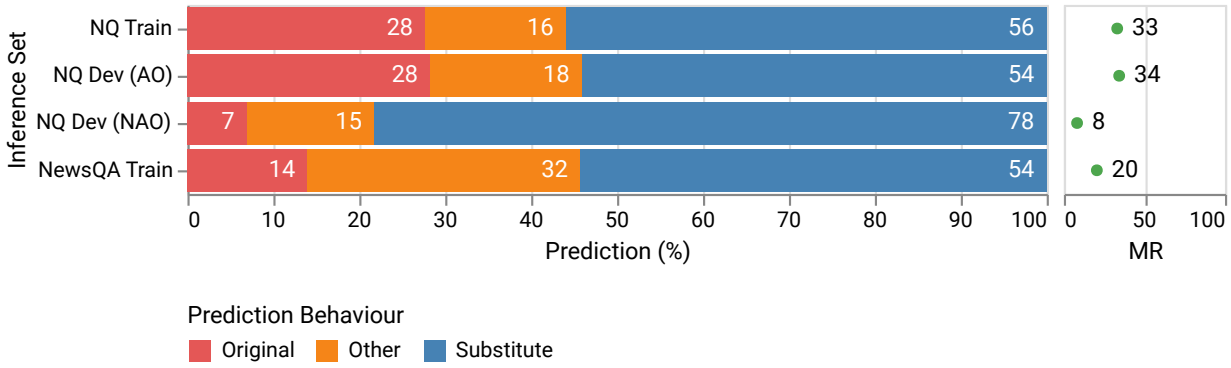


Figure 5.4: **Alias Substitution.** Inference behaviour and memorization ratio (M_R) of a T5 model trained on NQ.

perturbation, without introducing any real conflict in knowledge. As this type of substitution is not a knowledge conflict, we consider both *Original* and *Substitute* predictions correct model behaviour, and examine how often subtle answer paraphrases cause instability in the answers (*i.e.,* predicting *Other*). Figure 5.4 shows an elevated preference to select the *Original* answer than when the knowledge conflicted in corpus substitution, however *Other* is also predicted at least 15% of the time. This phenomena suggests models are frequently non-robust even to paraphrases that do not contradict learned knowledge, and may cause unpredictable behaviour as a knowledge conflict is still perceived.

5.4.2 Factors Impacting Model Behaviour

We've observed model behaviour appears strongly contingent on the domain similarity of presented knowledge conflicts. Next we explore what other factors may significantly impact a proclivity to preference parametric knowledge.

How does Model Size impact Memorization? As [9] has shown, large language models are susceptible to parroting memorized information. Figure 5.5 illustrates notable increases in memorization ratio as a function of the number of parameters. On the Train and Dev (AO) sets, the

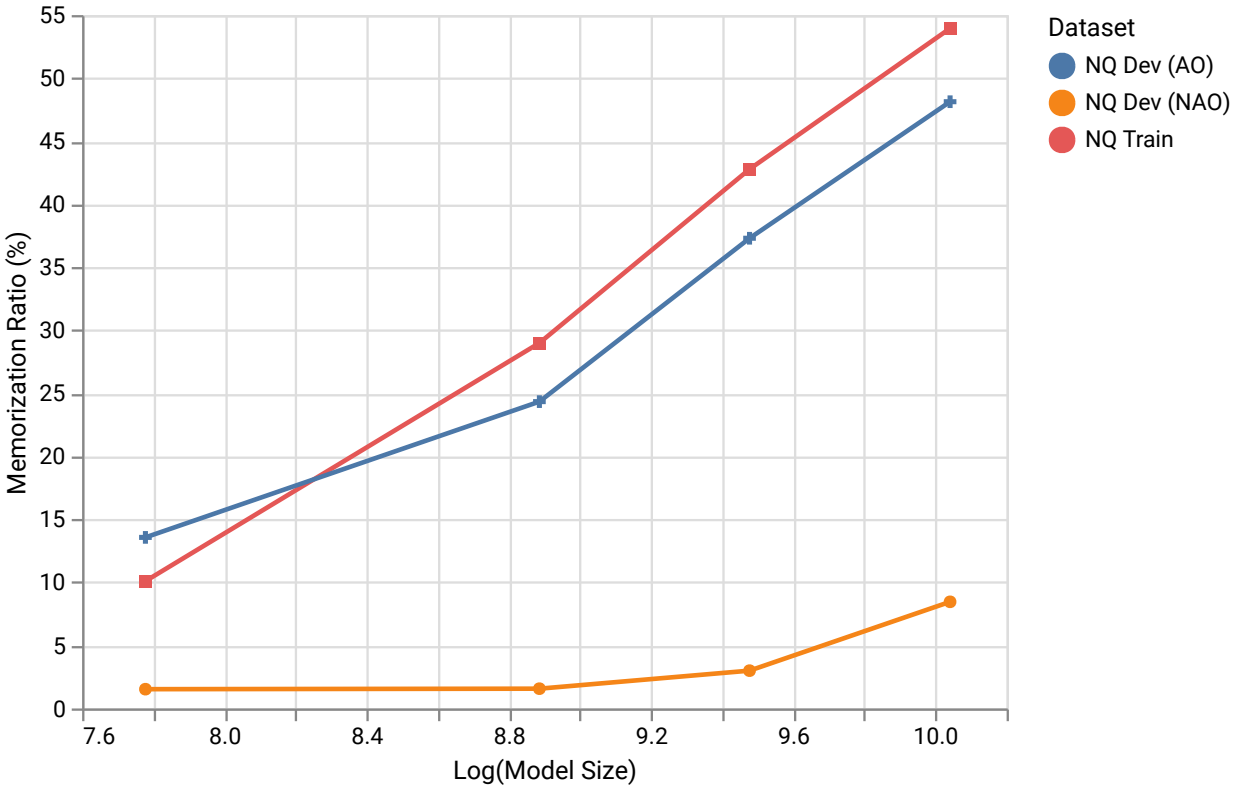


Figure 5.5: **Impact of Model Size on Memorization Ratio.** We finetune T5 small (60M), large (770M), XL (3B), and XXL (11B) models on NQ, finding the memorization ratio increases with model size for all inference sets.

memorization ratio rises from $< 15\%$ to $\geq 50\%$ in just two orders of magnitude, with no sign of diminishing returns. Most striking, the memorization ratio even for the Dev (NAO) set rises for the largest models in our experiments (11B parameters), which remain orders of magnitude smaller than the largest language models available.

How does Retrieval Quality impact Memorization? Until now we've used the highest ranked DPR document during training. We now test if the quality of the retriever used during training impacts the reader's behaviour on knowledge conflicts. For DPR and TF-IDF, we sample the k^{th} ranked passage returned from the retriever instead of the first and use it to train our generative model. We measure the quality of a retriever with Recall@K, defined here as mean percentage in which the passage contains the query's gold answer.

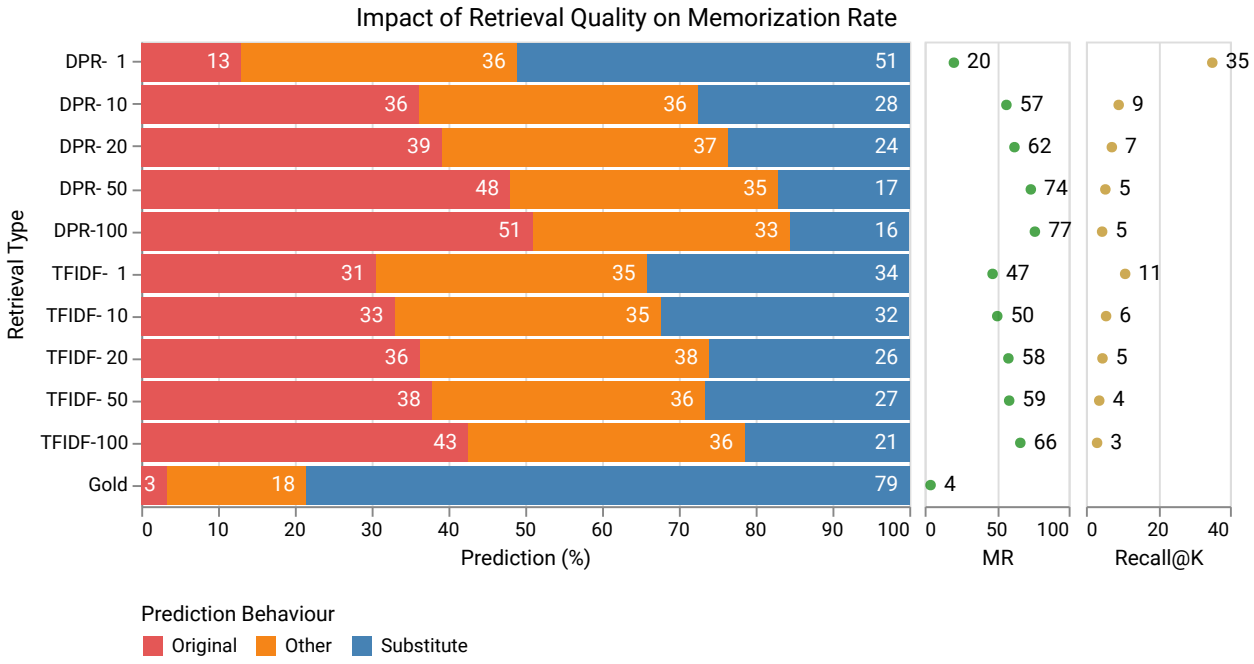


Figure 5.6: **Impact of Retrieval Quality on Memorization.** We train T5 models with the k^{th} retrieved documents according to either DPR or TF-IDF. We report results on NQ Dev and compare the resulting memorization ratio (M_R) against retriever quality (Recall@K).

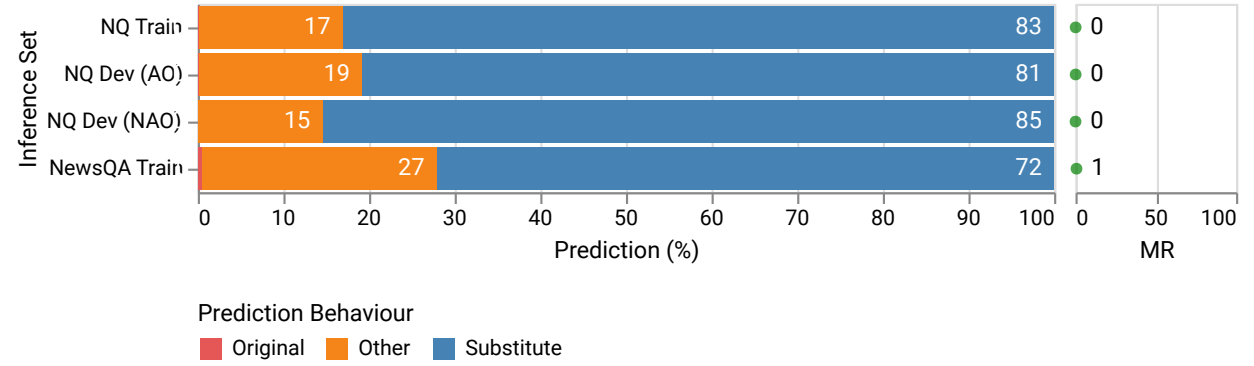


Figure 5.7: **Extractive QA.** Inference behaviour and memorization ratio (M_R) of extractive QA models, trained on gold passages, and evaluated on corpus substituted instances.

Figure 5.6 illustrates a clear inverse relationship between retrieval quality (Recall@K) and the memorization ratio (M_R). For both TF-IDF and DPR, less relevant passages during training causes the model to predict the *Original* answer at inference on x', effectively ignoring the passage. Training with gold passages reduces memorization, as the model is conditioned to expect the answer to always present in the passage.

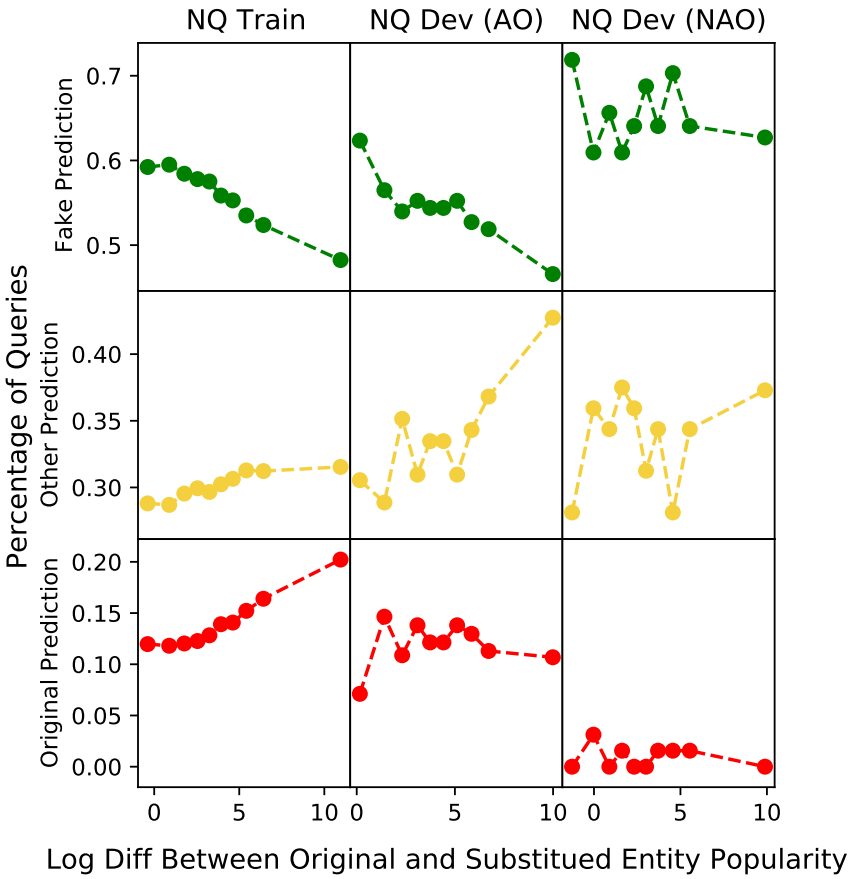


Figure 5.8: **Popularity Substitution.** Inference on queries where documents have been substituted with Wikidata entities of varying popularities. Model is T5 trained on NQ.

While training with gold passages effectively minimizes the memorization ratio, this is not standard practice among state-of-the-art QA models [63, 91, 75]. Typically, these generative QA systems are trained with retrieved passages, more conducive to scalable, and end-to-end training procedures. Consequently, training with gold passages may not present a convenient or viable solution.

Are Extractive QA Models susceptible to Knowledge Conflicts? One potential solution to the aforementioned issues with generative models is to use extractive QA readers which select a span from the passage. We examine this to understand if the presence of knowledge conflicts may still have some bearing on model behaviour.

In Figure 5.7, we replicate the corpus substitution knowledge conflicts from Figure 5.3 but with an

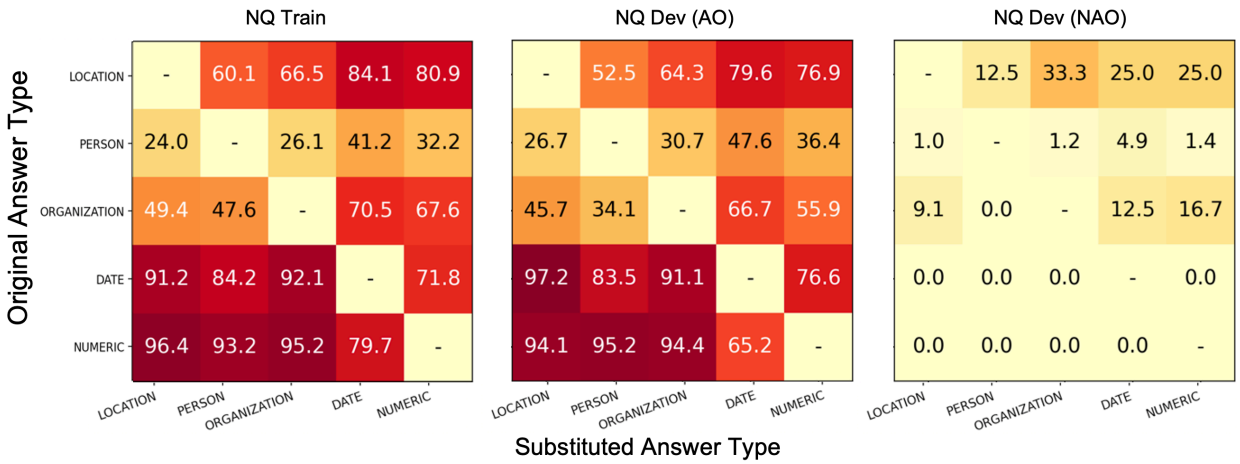


Figure 5.9: **Type Swap Substitution.** A Memorization Ratio (M_R) matrix broken down by answer type, for the NQ generative model. Darker intensity indicates higher M_R. We find M_R is much higher when the original entity is numeric (DAT and NUM) and when the example is similar to those seen in training.

extractive QA model. The memorization ratio falls to negligible values, as expected, however the model predicts *Other* $\geq 15\%$ of the time, for examples it had correctly answered pre-substitution. As discussed further in Section 5.4.3, this is likely symptomatic of greater model uncertainty in the presence of knowledge conflicts. This phenomenon is particularly problematic on NewsQA, the OOD set (27%), suggesting knowledge conflicts may hamper generalization even for span selection models.

How does Popularity of an Answer Entity impact Memorization? Using *popularity substitution* we examine if models are biased towards predicting more popular answers [146, 19]. Limiting our focus to the *Person* answer category, we order all PER Wikidata entities by popularity (approximated by Wikipedia monthly page views) and stratify them into five evenly sized popularity buckets. For each NQ instance with a PER answer, we generate five substituted instances, using a sampled entity for each of the five buckets.

In Figure 5.8, we plot the difference in popularity between the original and substituted answers against the percentage of model predictions on x' that fall into each category. For NQ Train and Dev (AO), the higher the popularity of the substituted entity, the more likely the model is to rely

on contextual knowledge and predict the *Substitute* answer. Conversely, the lower the popularity, the more likely the model is to predict an *Other* or *Original* answer. On the Dev (NAO) set, the popularity of the substituted entity is less predictive of model behavior. This suggests the popularity of a substituted entity plays a role only when the original answer is from a domain very close to training.

How do Models Behave on Nonsensical Knowledge Substitutions? Here we ask if nonsensical (obviously incorrect) substitutions elicit a higher memorization ratio, and whether model behaviour varies for different types of answers. *Type swap substitution* tests this by replacing the original entity with an entity of a different type. While practitioners typically prefer models to produce answers consistent with contextual knowledge, here a model may have good reason to doubt the quality of information. This experiment is relevant to measuring the common sense inherent in models, or robustness to misinformation attacks. We plot the memorization ratio M_R, across the possible range of type substitutions in Figure 5.9.

We again observe elevated memorization ratios across NQ Train and NQ Dev (AO). When the original entity is a string (entity types `LOC`, `PER`, `ORG`), the model is more likely to rely on contextual knowledge and generate the *Substitute* answer. In contrast, when the original entity is numerical (`DAT` and `NUM`), the model is more likely to predict the *Original* answer. The most striking result is when a numeric entity is replaced with a textual one; at least 83% of the time the model predicts the *Original* answer. On NQ Dev (NAO), memorization is low across type-pair substitutions, aligning with our previous experiments demonstrating memorization is lower on unseen data. Overall, these results suggest generative QA models may (inadvertently) be partially robust to index poisoning or misinformation attacks, attempting to elicit obviously false answers.

Sub (%)	**Example of Phenomena**
Grounding to Original	
CS (*7.5%*)	**Context:** The 2017 American Championship Series pit Hodgson against the Yankees
AS (*40%*)	...
TSS (*2.5%*)	**Q:** who won the american league?
XCS (*10%*)	**Orig Ans:** the Houston Astros
	Sub Ans: Hodgson
	Pred: the astros
Grounding to Substitute	
CS (*12.5%*)	**Context:** The Bay of Pigs was a failed invasion defeated by New Amsterdam ...
AS (-)	**Q:** who won the the bay of pigs?
TSS (*7.5%*)	**Orig Ans:** Cuban Revolutionary Forces
XCS (*25%*)	**Sub Ans:** New Amsterdam
	Pred: Amsterdam
Another Correct Answer	
CS (*12.5%*)	**Context:** Abby graduated from Canberra and earned her master from Georgia St...
AS (*2.5%*)	**Q:** where did abby go to college?
TSS (*2.5%*)	**Orig Ans:** Louisiana State
XCS (-)	**Sub Ans:** Canberra
	Pred: georgia state university
Random Passage Span	
CS (*17.5%*)	**Context:** There are 1000 sq metres farmers and 757,900 ag workers in the US ...
AS (*27.5%*)	**Q:** how many farmers are in usa?
TSS	**Orig Ans:** 3.2 million
(*22.5%*)	**Sub Ans:** 1000 sq metres
XCS (*65%*)	**Pred:** 757,900
Hallucinate	
CS (*47.5%*)	**Context:** "El Pollo Loco" means "Chile" ...
AS (*15%*)	**Q:** what does el pollo loco mean?
TSS (*65%*)	**Orig Ans:** The Crazy Chicken
XCS (-)	**Sub Ans:** Chile
	Pred: the oiled bird
Other	
CS (*2.5%*)	**Context:** The His Airness River is a 251-kilometre long river ...
AS (*15%*)	**Q:** what is east of the jordan river?
TSS (*0%*)	**Orig Ans:** Jordan
XCS (-)	**Sub Ans:** His Airness
	Pred: al - qurnah

Table 5.3: **Qualitative Analysis for *Other* predictions.** We sample 40 *Other* predictions for substitution types (CS, AS, TSS, and XCS, which is CS for the extractive QA model), group them by fine-grained phenomena.

5.4.3 Analyzing *Other* Predictions

While the *Original* and *Substitute* answers are well defined, the *Other* category is broad and serves as a catch-all. We perform a qualitative analysis to understand what phenomenon *Other* captures. For corpus, alias, and type-swap substitutions, we sample 40 instances each where *Other* is predicted, then group them into meaningful buckets (Table 5.3).

Part of *Other* predictions are due to the strict *EM* metric. Most prevalent is *alias substitution*; for 40% of cases the predicted answer is grounded to the original answer. Additionally, hallucinating an answer not in the context occurs throughout substitution types. We find that a reason models either hallucinate an answer or picks a random context span is when the substituted answer is implausible, as is designed in the *type-swap substitution*.

We also find interesting behavior within the type-swap substitution. When a textual entity (`PER`, `LOC`, or `ORG`) is replaced by another textual entity (with a different type), models are more likely to predict the substituted entity than when a textual entity is replaced by a numeric entity (`DAT` or `NUM`). This suggests models are able to recognize the plausibility of answers, and fall back to hallucinating an answer when an answer is implausible.

5.4.4 Mitigating Memorization

Our experiments suggest memorization can be mitigated by training with a perfect retriever — the reader learns to trust the passage and ground it's generation in this context. However, perfect retrieval annotations are costly and prohibitive to collect. In the absence of gold documents, we propose a simple method to mitigate memorization: augment the training set with training examples modified by *corpus substitution*. We construct a training set containing NQ examples with DPR passages, and the *corpus substituted* version of all DPR passages *that contain a gold answer to substitute for*. (This works out to 25% of the original training set size for DPR on NQ). The objective

Inference Set	M_R	EM (Δ)
NQ TRAIN	29.5 → 2.6	70.9 → 64.9 (-5.0)
NQ DEV (AO)	27.1 → 1.9	62.7 → 64.2 (+1.5)
NQ DEV (NAO)	1.5 → 0.0	32.9 → 40.0 (+7.1)
NEWSQA	9.3 → 0.6	21.4 → 25.8 (+4.4)

Table 5.4: **Mixed Training with Substitutions** yields reduced memorization (M_R) and improves generalization to OOD data.

of these targeted substitutions is to teach a retrieve-and-generate QA model not to memorize answers, but to rely on the context more often.

Table 5.4 illustrates training with our augmented dataset greatly decreases the memorization ratio on all KC datasets to negligible levels. An important consequence of this: out-of-domain generalization on **original** instances improves for both NQ Dev NAO (7%) and NewsQA (4%). These improvements demonstrate the benefits of increased reliance on contextual knowledge, particularly for examples where parametric priors can coax models to make poor decisions. We hope our substitution framework with this simple training method proves useful for practitioners developing systems which generalize to changing knowledge.

5.5 Summary of Contributions

Parametric knowledge learned during pre-training contains a great deal of world knowledge. However, this knowledge captured in the weights of language models is represented in an incomplete and imperfect fashion. Because of this, knowledge-dependent tasks question answering and fact checking require access to external contextual knowledge. In this work, we examine how conflicts between contextual and parametric knowledge affect langauge models when tasked to do question answering. In formalizing this problem, we contribute a substitution framework for creating knowledge conflicts, and rigorously evaluate model behaviour under this framework. Finally, we propose a method to mitigate memorization and consequently improve generalization on out-of-distribution

examples. Our findings show knowledge conflicts are an under-explored topic, providing valuable insights into model interpretability and generalization to evolving world knowledge.

The text in this chapter is based on the publications:

- *Entity-Based Knowledge Conflicts in Question Answering* [97] (EMNLP 2021).

The results presented in this chapter can be reproduced at https://github.com/apple/ml-knowledge-conflicts.

Part III

Correction

Chapter 6

Editing for Attribution: A New Task for Correcting Language Model Generations

6.1 Introduction

Generative language models and other text generation models are now the backbone of many AI systems. For example, large language models can perform multi-step reasoning [112, 163], generate plans [1], use tools and APIs [145, 150], and answer open-domain questions [119, 134]. Despite these incredible advances, state-of-the-art LMs still frequently produce biased, misleading, or unsupported content, colloquially called "hallucinations" [101, 104]. To make LMs more trustworthy, we want to justify each generation by an *attribution report* [129, 14] that contains supporting evidence from trusted sources (e.g., encyclopedia or articles) where appropriate.

Most existing LMs, such as those based on sequence-to-sequence architectures, lack a built-in mechanism for attribution. Even *retrieval-augmented models* [55, 89], which retrieve relevant documents and then condition on them to generate text, still do not guarantee attribution. Prior work has shown that retrieval-augmented models generate text that either includes additional information

outside the retrieved documents [39], ignores the documents altogether [79], or even contradicts the documents [98]. In fact, occasionally ignoring the retrievals can make the models more robust to bad retrievals [72], illustrating that end-task performance and attribution are not always aligned.

Instead of constraining LMs to generate attributed text, we propose a model-agnostic approach to improve the attribution of any existing LM: *Retrofit Attribution using Research and Revision* (RARR). The approach is inspired by works on fact-checking where simple research-and-revise workflows are effective at attributing or correcting unattributed claims made by humans [154, 141, 152]. As shown in Figure 6.1, after generating text with the LM, RARR does *research* to retrieve relevant evidence, and then *revises* the text to make it consistent with the evidence while preserving qualities like style or structure, enabling the revised text to be seamlessly used in place of the original. RARR can be viewed as a retrieval-augmented model where retrieval happens *after* generation rather than before. This allows RARR to stand on the shoulders of giant LMs without having to modify them to support attribution.

In our effort to expand the scope of Research & Revision models to handle the output of arbitrary LMs, we make the following contributions. First, we formalize the *Editing for Attribution* **task** and propose new **metrics** that evaluate revision models not just on their ability to produce well-attributed revisions, but also on their ability to otherwise *preserve* original properties of the text. Second, we use these metrics to **benchmark** how existing revision models perform on various types of LM outputs such as knowledge-intensive statements, reasoning chains, and dialog responses. Finally, we find that existing revision models do not always generalize across many tasks (and were not originally intended to), and therefore propose a new research-and-revise **model** that leverages the power of few-shot prompting in large language models to robustly generalize across domains.

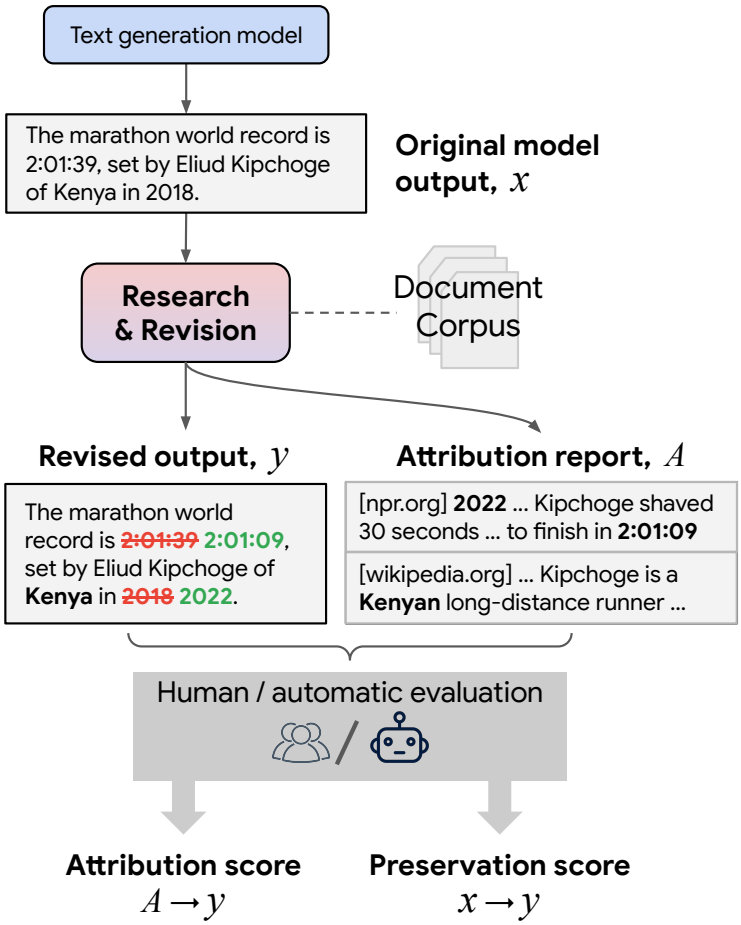


Figure 6.1: **The *Editing for Attribution* task.** The input x is a text passage produced by a generation model. Our *Research & Revision* model outputs an attribution report A containing retrieved evidence snippets, along with a revision y whose content can be *attributed* to the evidence in A while *preserving* other properties of x such as style or structure.

6.2 Task formulation

We propose the task of *Editing for Attribution* as follows. As Figure 6.1 shows, the input to the system is a text passage x produced by a generation model. The output is a revised text passage y along with an *attribution report* A, which contains *evidence snippets* $e_1, \ldots, e_M$ that support the content in y. Optionally, the attribution report can contain additional information such as the alignment between evidence snippets and relevant parts in y.

We propose to measure the quality of the revised text y and attribution report A along two dimensions:

(1) **attribution**: how much of the revised text y can be attributed to the evidence in A, and (2) **preservation**: how much the revised text y preserves aspects of the original text x.

6.2.1 Measuring attribution

Previously, [129] proposed *Attributable to Identified Sources* (AIS), a human evaluation framework which considers a binary notion of attribution. Roughly speaking, a text passage y is attributable to a set A of evidence if a generic hearer would affirm the statement "According to A, y" under the context of y. A system either receives full credit (1.0) if *all* content in y can be attributed to A, and no credit (0.0) otherwise.

We propose a more fine-grained, sentence-level extension of AIS. We ask annotators to give an AIS score for each sentence s of y, and then report the average AIS score across all sentences:

$$\text{Attr}_{\text{AIS}}(y, A) = \operatorname*{avg}_{s \in y} \text{AIS}(s, A).$$

Since the AIS score is binary, this effectively measures the percentage of sentences in y that are fully attributed to A. When judging each sentence, we also give annotators access to the surrounding sentences and other necessary context, such as the question that the text passage responded to. We also impose the maximum number of evidence snippets in the attribution report A to make it concise enough for both the annotator and downstream users. By manually inspecting 30 examples from our benchmarks, we found $M = 5$ snippets to be sufficient for full attribution.

During model development, we define an automated metric, auto-AIS ($\text{Attr}_{\text{auto}}$), that approximates human AIS judgments. We utilize the natural language inference (NLI) model from [58], which correlates well with AIS scores. For each sentence s of y, and for each evidence snippet e in A, let

$\text{NLI}(e, s)$ be the model probability of e entailing s. We then define

$$\text{Attr}_{\text{auto}}(y, A) = \underset{s \in y}{\text{avg}} \max_{e \in A} \text{NLI}(e, s).$$

To improve accuracy, we decontextualize [28] each sentence based on the entire context of y before computing the scores.

6.2.2 Measuring preservation

To measure preservation, we first ask annotators to decide if the revision preserves the text's original intent (completely, somewhat, or not at all). Like AIS evaluation, we give annotators the necessary surrounding context. We define the binary metric $\text{Pres}_{\text{intent}}(x, y)$ to be 1.0 if the revision completely preserves the original intent, and 0.0 otherwise.

However, even if a revision preserves intent, it may still make superfluous modifications, such as reordering words, changing textual style, or including unnecessary additional information [152]. Different tasks have different requirements for what should be preserved. Here, we desire a simple metric that can be readily computed for many tasks and that generally penalizes unnecessary changes. We thus define a metric based on the character-level Levenshtein edit distance [85] between x and y:

$$\text{Pres}_{\text{Lev}}(x, y) = \max\left(1 - \frac{\text{Lev}(x, y)}{\text{length}(x)}, 0\right)$$

This metric is 1.0 if x and y are the same, and 0.0 if y completely overwrites all parts of x. Pres_{Lev} is generally sensitive to any kind of change, but certainly does not capture all notions of preservation (e.g., preserving rhyme schemes or puns).

We want the revision to preserve the original intent while avoiding superfluous edits. To reflect this,

we finally combine the two metrics as

$$\text{Pres}_{\text{comb}}(x, y) = \text{Pres}_{\text{intent}}(x, y) \cdot \text{Pres}_{\text{Lev}}(x, y).$$

which is 0.0 if the revision changes the intent and equal to $\text{Pres}_{\text{Lev}}(x, y)$ otherwise. Since $\text{Pres}_{\text{intent}}$ requires human annotation, we use Pres_{Lev} as an automated metric for model development.

6.2.3 Discussion

Optimizing for attribution alone cannot ensure a good revision: for example, an adversarial editor could ensure 100% attribution by simply replacing the input x with the text of any arbitrary retrieved document, which is trivially attributable to itself. Ideally, we want to maximize both attribution and preservation, while navigating any tradoffs between the two. In our experiments, we report both metrics, as well as their harmonic mean (F1_{AP}, analogous to how recall and precision are combined in F1).

We emphasize that this evaluation scheme does not require any "gold" or "reference" edits (unlike many prior evaluations of text revision models), which are often only available for specialized domains. This enables us to broaden the scope to a much wider range of generation tasks.

6.3 RARR: A Baseline For *Editing for Attribution*

We now present *Retrofit Attribution using Research and Revision* (RARR), a simple method for solving the *Editing for Attribution* task. As illustrated in Figure 6.2, given an input passage x, the research stage first generates a set of queries $\{q_1, ..., q_N\}$, each investigating one aspect of x that potentially requires attribution. For each query q_i, it retrieves web documents and selects the best evidence snippets $\{e_{i1}, e_{i2}, \dots\}$. The revision stage then revises the original text x using the

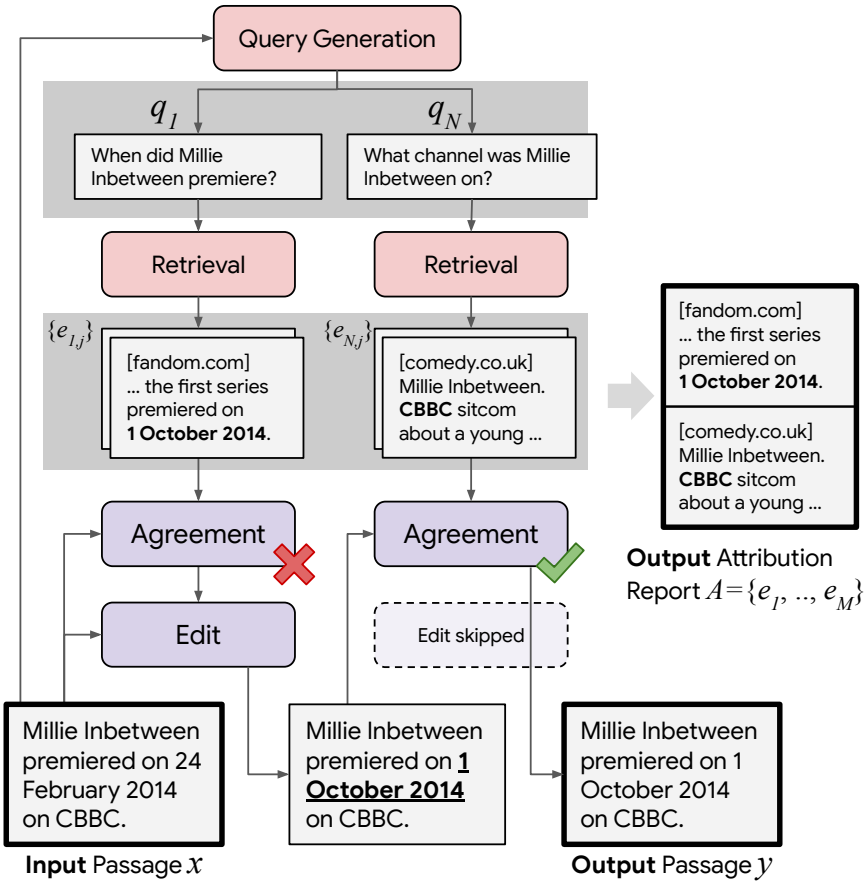


Figure 6.2: **An overview of RARR,** which improves attribution for a text passage via *Research & Revision*. Given the input text passage, the **research stage** uses a *query generator* to raise questions about different aspects of the text. The *retriever* then searches for evidence to investigate each query. The **revision stage** first runs an *agreement model* to detect disagreement between the text and the evidence, then runs an *edit model* to revise the text if needed. Finally, M evidence snippets are selected to form an attribution report.

retrieval results $\{(q_1, e_{11}), \dots\}$, yielding a revised text y.

Most components for RARR are implemented using few-shot prompting [15]. We use PaLM [29] as our language model. Figure 6.3 shows some few-shot examples we use.

(a) **Query generation** $x \rightarrow \{q_1, \ldots, q_N\}$

You said: Your nose switches back and forth between nostrils. When you sleep, you switch about every 45 minutes. This is to prevent a buildup of mucus. It's called the nasal cycle.
To verify it,
a) I googled: Does your nose switch between nostrils?
b) I googled: How often does your nostrils switch?
c) I googled: Why does your nostril switch?
d) I googled: What is nasal cycle?

(b) **Agreement model** $(y, q, e) \rightarrow \{0, 1\}$

You said: Your nose switches . . . (same as above). . . nasal cycle.
I checked: How often do your nostrils switch?
I found this article: Although we don't usually notice it, during the nasal cycle one nostril becomes congested and thus contributes less to airflow, while the other becomes decongested. On average, the congestion pattern switches about every 2 hours, according to a small 2016 study published in the journal PLOS One.
Your nose's switching time is about every 2 hours, not 45 minutes.
This disagrees with what you said.

(c) **Edit model** $(y, q, e) \rightarrow$ new y

You said: Your nose switches . . . (same as above). . . nasal cycle.
I checked: How often do your nostrils switch?
I found this article: Although we . . . (same as above). . . PLOS One.
This suggests 45 minutes switch time in your statement is wrong.
My fix: Your nose switches back and forth between nostrils. When you sleep, you switch about every 2 hours. This is to prevent a buildup of mucus. It's called the nasal cycle.

Figure 6.3: **Examples of few-shot examples** used to prompt the PaLM model (blue = input; red = output).

6.3.1 Research stage

Query generation We perform *comprehensive question generation* (CQGen) which produces a sequence of questions covering *all aspects* of the passage x that need to be verified and attributed. A similar strategy has been employed to train text-planning models [109]. A prompt with six human demonstrations was sufficient for PaLM to adequately learn the task. To increase diversity and coverage, we sample from our CQGen model three times and take the union of the resulting queries.

Evidence retrieval For each query from CQGen, we use Google Search to retrieve $K = 5$ web pages. We extract candidate evidence snippets from each web page by running a sliding window of four sentences across the page, breaking at document headings. The evidence snippets for each query are then ranked based on their relevance to the query. For this, we use an existing query-

document relevance model trained following [111], which computes a relevance score $S_{\text{relevance}}(q, e)$ between a query q and an evidence snippet e. We then keep the top $J = 1$ evidence for each query. The final retrieval result is $[(q_1, e_{11}), \ldots, (q_1, e_{1J}), \ldots, (q_N, e_{N1}), \ldots, (q_N, e_{NJ})]$, where e_{ij} denotes the j^{th} evidence for the i^{th} query, and N denotes the total number of queries from CQGen (which can be different for each input x).

6.3.2 Revision stage

After retrieving evidence, certain parts of x may now be properly attributed, but other parts remain unattributed and should be revised. As illustrated in Figure 6.2, the revision stage initializes the output $y = x$. Then for each retrieved $(q, e) = (q_i, e_{ij})$, the *agreement model* checks if the evidence e disagrees with the current output y regarding the issue in query q. If a disagreement is detected, the *edit model* edits y to agree with e; otherwise, it does nothing. The process continues until all retrievals are processed.

Agreement model The agreement model takes the partially edited passage y, a query q, and the evidence e as input. It then decides whether both y and e imply the same answer to the question in q. This form of question-guided agreement was previously explored by [59]. We implement this by few-shot prompting PaLM using a chain-of-thought style prompt [163], where we ask the model to explicitly state the implied answers for both y and e before producing its judgment about their agreement.

Edit model The edit model is run only if a disagreement is detected. The model takes y, q and e as input, and outputs a new version of y that aims to agree with e while otherwise minimally altering y. We again use few-shot prompting and chain-of-thought, where we ask the model to first identify a particular span in y that needs to be edited before generating the revised y. This helps

PaLM outputs on NQ (factoid statements)

Millie Inbetween is a British comedy television series. It premiered on 24 February 2014 on BBC One. The first series was produced by John Yorke and Phil Clymer.

PaLM outputs on SQA (reasoning chains)

The highest point of Mount Wycheproof is 70 metres. Edmund Hillary climbed Mount Everest, which is 8,848 metres. So Mount Wycheproof would be a breeze for Edmund Hillary.

LaMDA outputs on QReCC (knowledge-intensive dialogs)

When was Welsh social reformer Robert Owen born?
Robert Owen was born on 14 May 1771
...
Did he have another job? } context
In 1810 he moved to Manchester and established a draper's shop.

Figure 6.4: **Examples of editing input passages.** For QReCC, prior dialog turns are also given as the context.

reduce the editor's deviation from the current y.[1]

6.3.3 Attribution report

Finally, we select at most $M = 5$ evidence snippets to form an attribution report A. Note that during evidence retrieval and revision, we may have encountered and used more than M snippets. Our goal is to find a subset of snippets that maximizes *coverage* over the potentially attributable points in the passage, as represented by the queries $q_1, \ldots, q_N$. We use the relevance model from Section 6.3.1 as a proxy for measuring how much an evidence e covers the point raised by a query q. Then, we exhaustively search for $A \subseteq \{e_{11}, \ldots, e_{NJ}\}$ of size at most M that maximizes

$$\text{Cover}(A, q_{1:N}) := \sum_{i=1}^{N} \max_{e \in A} S_{\text{relevance}}(q_i, e)$$

[1]The editor occasionally produces large edits that bring the new revision close to e but far from the current y. Since this is rarely desirable, we reject edits with edit distance above 50 characters or 0.5 times the original text length.

6.4 Experimental Setup

6.4.1 Creating Evaluation Sets

RARR aspires to be a general-purpose method for improving the attribution of any text generation model in any text domain. We thus construct evaluation benchmarks by taking the task input from three diverse datasets, and prompting different generation models to produce *long-form outputs* which may contain "hallucinations," as demonstrated in Figure 6.4. These long-form outputs serve as input text passages to RARR. We generate 150 development and 150 test passages for each combination of generation model and source dataset.

Factoid statements We prompt PaLM 540B and GPT-3 text-davinci-002 to generate long-form answers to questions from the Natural Questions dev set [81]. The resulting passages are mostly coherent but often contain factual errors. This setup examines the ability to attribute a diverse range of factoid knowledge.

Reasoning chains Language models can generate reasoning chains to answer complex questions [163]. We use PaLM and GPT-3 to generate reasoning chains for the StrategyQA train set [50]. This setup tests whether the revision model can provide better attribution for intermediate steps of reasoning, while preserving the overall reasoning process.

Knowledge-intensive dialogs We consider the conversational QA task from the QReCC dev set [2]. Given the previous dialog turns of questions and answers $(Q_1, A_1, Q_2, A_2, \ldots, Q_k)$, we use LaMDA and GPT-3 to answer to the final question Q_k conditioned on the dialog history. The answer tends to be context-dependent, featuring pronouns and implicit references. All dialog turns are given alongside the answer as inputs to the revision model.

6.4.2 Models

We compare RARR to several systems that have a research-and-revise workflow.

EFEC We consider EFEC [152] as a representative fine-tuned editor. EFEC fine-tunes a T5-based model to revise text conditioned on multiple evidence snippets using both semi-supervised and fully-supervised approaches. We compare against their fully-supervised approach, which performed best in their experiments. EFEC uses a neural retrieval model [69] to retrieve from Wikipedia; however, not all passages in our experiments are supported by Wikipedia articles. To more fairly compare the editing capabilities of EFEC, we instead use the evidence retrieved by our research stages (CQGen and web search). Note that the EFEC editor conditions on multiple pieces of evidence at once, while our editor iteratively conditions on one at a time.

LaMDA LaMDA [150] generates responses in three steps: 1) generate a "base response"; 2) generate search queries from the base response; 3) generate a "revised response" conditioned on the base response and retrieved evidence. To apply LaMDA on a given text x, we simply set the base response in step 1 to x, and then run steps 2 and 3 (we call these latter two stages "LaMDA Research"). LaMDA was trained as a dialog system, and always expects a dialog context where the user speaks first. So, for non-dialog tasks, we insert an artificial user utterance as dialog history: *"Tell me something interesting."* For the attribution report, we take all evidence documents retrieved by LaMDA during its research process.

RARR Our model uses few-shot prompting on PaLM 540B for query generation, the agreement model, and the edit model. We use the same prompts for all tasks except when the context comes from a dialog, where we slightly modify the prompts to use the dialog context (e.g., CQGen now maps dialog context + x to queries). The query-evidence relevance model $S_{\text{relevance}}$ is a pretrained T5-large model [124] fine-tuned following [111] on MS MARCO [110].

	Attribution		Preservation			
Model	auto-AIS	AIS	intent	Lev	comb	$F1_{AP}$
PaLM outputs on NQ						
EFEC	45.6 → 64.3	35.4 → 48.3	16.0	39.1	10.4	17.1
LaMDA	39.5 → 49.9	18.3 → 30.4	26.0	39.6	21.1	24.9
RARR	45.6 → 54.9	35.4 → 43.4	90.0	89.6	83.1	**57.0**
PaLM outputs on SQA						
EFEC	37.8 → 58.6	24.5 → 51.7	6.0	31.0	3.8	7.1
LaMDA	32.7 → 43.2	15.8 → 27.0	40.0	46.4	33.7	30.0
RARR	37.6 → 45.1	24.5 → 31.5	92.6	89.9	84.6	**45.9**
LaMDA outputs on QReCC						
EFEC	19.1 → 47.4	13.2 → 48.7	39.7	39.4	23.7	31.9
LaMDA	16.4 → 36.2	16.0 → 27.1	21.3	24.8	12.0	16.6
RARR	18.8 → 29.4	13.2 → 28.3	95.6	80.2	78.1	**41.5**

Table 6.1: ***Editing for Attribution* results.** For attribution, we report the AIS scores of the texts both before and after editing (before → after). For preservation, we report intent preservation $Pres_{intent}$, Levenshtein similarity $Pres_{Lev}$, and the combined $Pres_{comb}$. We summarize $Attr_{AIS}$ and $Pres_{comb}$ using their harmonic mean ($F1_{AP}$).

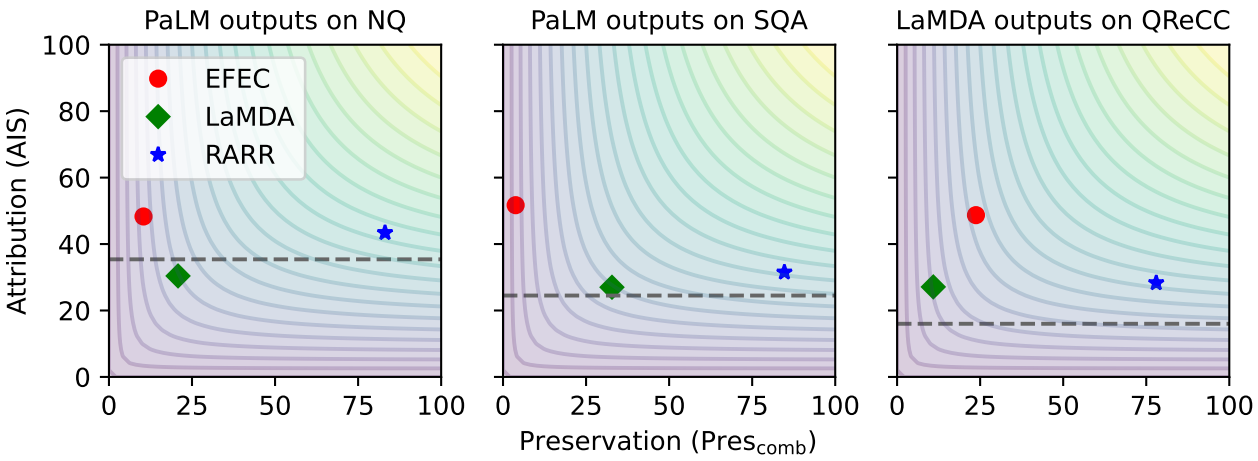


Figure 6.5: **Attribution and preservation scores.** Dashed lines indicate the highest attribution score obtained by any of the models *before* editing: points above the line have better attribution after revision. The contours are $F1_{AP}$ level curves: points along a contour have equivalent $F1_{AP}$. Different models make very different trade-offs between attribution and preservation. Only RARR has a robust $F1_{AP}$ across all tasks.

6.5 Results

6.5.1 Primary Results

For the main experiments, we report results on passages generated by PaLM and LaMDA. Table 6.1 and Figure 6.5 show attribution and preservation results for each model and dataset. We also report F1_{AP}, the harmonic mean of the two metrics, which is shown as level curves in Figure 6.5.

RARR significantly improves attribution while preserving most of the original text. In terms of F1_{AP}, RARR is the only method that performs robustly across all three datasets, and significantly outperforms prior methods on NQ and SQA.

We found that RARR is the only method that preserves the original intent of x over 90% of the time — EFEC and LaMDA only manage to preserve the original intent 6–40% of the time. We also see that editing is crucial to improve attribution: if we only retrieve evidence to support the original response x without editing, attribution ranges from the low 10s to mid 30s. After editing, RARR can increase attribution by up to 13% absolute, while changing only 10–20% of the text.

As noted in Section 6.2, one can sacrifice preservation for higher attribution. EFEC is able to obtain strong F1_{AP} on QReCC by making larger changes to the text in exchange for a higher attribution score. However, it occupies a very different point from RARR on the attribution-preservation trade-off curve, as visualized in Figure 6.5.

6.5.2 Qualitative analysis

Human oracle To understand the remaining headroom in our task, we ask: *what is the minimal amount of editing needed to make a text passage fully attributed?* The answer would depend on the quality of the LM that generated the text as well as the task difficulty. As an approximation, we

x: Justice Ashok Kumar Mathur headed the 7th central pay commission in India. It was created in 2014 and submitted its report in **2016**.	
Attribution: 50%	**Preservation:** 100%
EFEC: The 7th central pay commission in India was created in 2014.	
Attribution: 100%	**Preservation:** 0%
LaMDA: I heard the 7th CPC made recommendations for increasing the minimum salary pay from Rs 7066 to 18k per month for new central government employees.	
Attribution: 0%	**Preservation:** 0%
RARR: Justice Ashok Kumar Mathur headed the 7th central pay commission in India. It was created in 2014 and submitted its report in **2015**.	
Attribution: 100%	**Preservation:** 100%
evidence: The 7th Central Pay Commission (Chair: Justice A. K. Mathur) **submitted its report on November 19, 2015**. The Commission had been **appointed in February 2014**, to look at remuneration for central government employees. ...	

Figure 6.6: **Example model outputs and human judgment of their attribution and preservation scores.** EFEC reduces the passage x into a single sentence. LaMDA changes the writing style. RARR preserves the structure of the input passage. We show one evidence retrieved by RARR to help explain the example.

manually edited 30 examples in our NQ benchmark until we judged them to be 100% attributable. We achieved a preservation score of 88%, which (when combined with 100% attribution) translates to 93.6 $F1_{AP}$, indicating a significant headroom.

Analyzing the baselines As exemplified in Figure 6.6, EFEC frequently attempts to summarize the entire passage into one sentence, or drops later sentences. This is likely due to EFEC's training data, which was limited to single sentences. This behavior generally increases the attribution score, because it is usually easier to make one sentence fully attributable than many sentences. However, in datasets where the claim contains multiple sentences (NQ and SQA), such a behavior yields low preservation scores, and also results in outputs that are less informative. We expect that EFEC could perform much better if its training data were augmented to include multiple sentences. LaMDA Research achieves similar attribution scores to RARR. But as mentioned in Section 6.4.2, the intent and linguistic style of the output tend to deviate from the input, resulting in lower preservation scores (Figure 6.6). We emphasize that this is not a purely apples-to-apples comparison since LaMDA was not optimized for preservation. Overall, these experiments are mainly meant to illustrate that prior

models were simply not designed for the task of *Editing for Attribution*, rather than to mark RARR as the best method.

Analyzing RARR For the research stage, the question generation model had comprehensive coverage: a manual inspection of 40 examples shows ¿ 80% with questions that fully cover all aspects of the input text. The retriever was strongest at researching content involving distinct entities (e.g., a movie, a major event, or a person). In contrast, we found significant headroom for better attribution of statements involving generic objects and more abstract claims (e.g. *"Video games require electricity."*— since this is obvious to most humans, retrieved articles from the web tend to address related but different topics). We suspect that a significant amount of attribution headroom on our benchmarks would benefit from a better research stage.

For the revision stage, RARR was able to revise many unattributed claims, especially those involving entities and numbers (Figures 6.7a and 6.7b). It can also perform larger revisions when necessary (Figure 6.7c). Moreover, RARR abstains from editing when the claim is already well-attributed: on NQ, among the inputs with near-perfect attribution (pre-edit $\text{Attr}_{\text{AIS}} > 0.9$), RARR does not make an edit in 90% of the cases. However, the system also has several shortcomings. Some erroneous edits arise from misleading irrelevant evidence (Figure 6.7d). We also observed an interesting challenge when revising reasoning chains, where the model successfully revised an incorrect claim, but did not revise subsequent reasoning steps that depend on the earlier claim (Figure 6.7e). In this case, further editing to improve logical coherence could help.

6.5.3 Ablations

Ablating query generation RARR uses generated questions as search queries for evidence retrieval. We consider two natural alternatives: using the entire input passage as a single search query, or using each sentence as a search query. For the former, we retrieve $J = 3$ evidence snippets

(a) **Correctly revising an entity**

y: If She Knew What She Wants was written by **Henry Roth**.

e: [en.wikipedia.org] "If She Knew What She Wants" is a song written by American singer-songwriter **Jules Shear** and introduced on . . .

y': If She Knew What She Wants was written by **Jules Shear**.

(b) **Correctly revising a number**

y: God Save the Queen became the British national anthem in **1745**. . . .

e: [www.britannica.com] The oldest national anthem is Great Britain's "God Save the Queen," which was described as a national anthem in **1825**, . . .

y': God Save the Queen became the British national anthem in **1825**. . . .
(The year 1745 was when the song was first performed.)

(c) **Performing a necessary larger revision**

y: "It's My Party" is a song written and composed by American singer-songwriter and producer **Walter Gold**. The song was published in **1962** and first recorded by American singer-songwriter, **producer and arranger** Lesley Gore.

e: [secondhandsongs.com] Song: It's My Party written by **John Gluck, Wally Gold, Seymour Gottlieb, Herbert Weiner**. . .

y': "It's My Party" is a song written and composed by **John Gluck, Wally Gold, Seymour Gottlieb and Herb Weiner**. The song was published in **1963** and first recorded by American **singer** Lesley Gore.

(d) **Confused by misleading evidence**

y: In the Heat of the Night is a police procedural crime drama TV series produced by MGM Television. It aired on **NBC** from September 8, 1988 to May 3, 1992. . . .

e: [metv.com] Watch In the Heat of the Night on **MeTV**.

y': In the Heat of the Night is a police procedural crime drama TV series produced by MGM Television. It aired on **MeTV** from September 8, 1988 to May 3, 1992. . . .
(The show did actually air on NBC in the 1980s, but the model fails to realize that MeTV is a re-run much later in time.)

(e) **Improved attribution, but reasoning doesn't adapt to new info**

(Q: Does Homer Simpson need two hands worth of fingers to count to 5?)

y: Homer Simpson has **5** fingers and 2 hands. Therefore, he does not need two hands worth of fingers to count to 5. He only need one hand to count to 5.

e: [refinery29.com] Other iconic animated characters, such as Homer Simpson and his brood, . . . only have **four fingers**, too.

y': Homer Simpson has **4** fingers and 2 hands. Therefore, he does not need two hands worth of fingers to count to 5. He only need one hand to count to 5.
(Having only 4 fingers per hand, he does need two hands to count to 5.)

Figure 6.7: **Example revisions from RARR, both good and bad.** y = partially edited passage; e = evidence; y' = passage after editing with e.

to make the amount a closer match to other methods.

The results are in Table 6.2. Using the entire input passage as the query gives poor results, as the retrieved evidence tends to not focus on potentially unattributed parts in the passage. Using

Model	PaLM outputs on NQ			PaLM outputs on SQA			LaMDA outputs on QReCC		
	$\text{Attr}_{\text{auto}}$	Pres_{Lev}	F1_{AP}	$\text{Attr}_{\text{auto}}$	Pres_{Lev}	F1_{AP}	$\text{Attr}_{\text{auto}}$	Pres_{Lev}	F1_{AP}
Full RARR	45.6 → 54.9	89.6	**68.1**	37.6 → 45.1	89.9	60.0	18.8 → 29.4	80.2	**43.1**
no agreement model	45.6 → 50.6	82.6	62.8	37.8 → 46.9	83.4	60.0	18.8 → 28.8	72.0	41.2
query = input	45.4 → 47.2	98.4	63.8	39.4 → 30.3	98.8	46.4	19.7 → 20.6	96.3	34.0
query = sentence	49.1 → 52.1	97.0	67.8	43.7 → 44.3	98.8	**61.2**	19.0 → 19.6	97.0	32.6

Table 6.2: **Ablation results.** We report the automatic metrics: $\text{Attr}_{\text{auto}}$, Pres_{Lev}, and harmonic mean between the two (F1_{AP}). We show auto-AIS scores both before and after editing (before → edit), with respect to the attribution report A produced by the model. Even though sentence-as-queries may achieve similar F1_{AP} as RARR, it is less robust to corpus shifts and tends to retrieve passages that may encourage confirmation bias.

Model	NQ F1_{AP}		SQA F1_{AP}	
	orig	no wiki	orig	no wiki
Full RARR	**68.1**	**64.3**	60.0	**57.6**
query = sentence	67.8	60.3	**61.2**	56.7

Table 6.3: **The impact of excluding Wikipedia from the retrieval corpus.** CQGen (full RARR) is more robust to Wikipedia's absence, while using sentences-as-queries suffers a bigger drop in performance.

sentences as queries gives results closer to the full CQGen, but a closer analysis reveals two caveats.

First, sentences-as-queries are more effective when such sentences "mimic" content on the Web, and are less effective otherwise. In Table 6.3, we test this by excluding all of Wikipedia from web search results (since many PaLM outputs for NQ have a Wikipedia style). The attribution performance of sentences-as-queries drops significantly, while CQGen is more robust.

Second, sentence-as-queries tends to retrieve passages that may encourage confirmation bias. Consider the example *"Georgia is called the Peach State, but California actually produces the most peaches."* Retrieval using sentences-as-queries found an article echoing that California produces the most peaches, while CQGen generated the more impartial query *"Which state produces the most peaches?"* and found a newer article saying that South Carolina replaced California as the top peach producer. In this case, RARR using CQGen needs to sacrifice more preservation score to edit the text, leading to a lower F1_{AP} score. This underscores that attribution alone cannot measure

x: The Crown-of-thorns starfish is native to the Great Barrier Reef... The starfish was introduced to the Great-Barrier-Reef by **ocean currents**.

e: [invasivespeciesinfo.gov] **Ballast water** is one of the major pathways for the introduction of nonindigenous marine species...

y: The Crown-of-thorns starfish is native to the Great Barrier Reef... The starfish was introduced to the Great-Barrier-Reef by **ballast water**.

Figure 6.8: **Disabling the agreement model leads to over-edits.** Here, the evidence e does not explicitly disagree with x, but without an agreement model to detect this, the edit model makes an unsupported change.

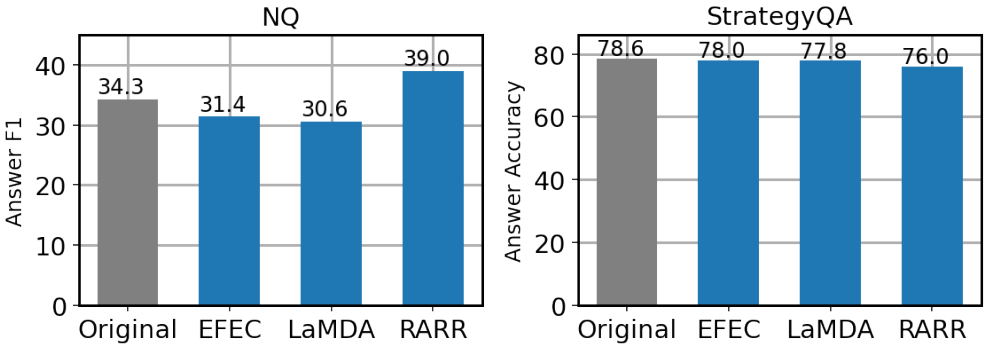


Figure 6.9: **Downstream task performance** on NQ and SQA. RARR's revisions lead to better answer accuracy on NQ. No models improved answer accuracy on SQA.

"correctness" since not all evidence is up-to-date or reliable.

Ablating agreement model We try removing the agreement model, which effectively forces the model to revise the passage based on every retrieved evidence. The results are shown in Table 6.2. As expected, more revision leads to less preservation score and spurious changes to the text passage, as demonstrated in Figure 6.8.

Impact on downstream task performance We have measured preservation using the metric defined in Section 6.2.2. However, another measure of preservation is whether the revised text can still be used to perform the task that it was originally generated for. Following EFEC, we

quantitatively evaluate this on short answer tasks NQ and SQA, and we summarize the result in Figure 6.9.

For NQ, each original text x is a long-form response to a factoid question. To determine whether the revised text y still serves this purpose, we feed the factoid question and y back into PaLM and prompt it to extract a *short* answer from y. We find that RARR not only preserves the short answer accuracy but actually improves it by roughly 5%.

For SQA, each original text is a reasoning chain that helps to answer a yes/no question. We feed the SQA question and y back into PaLM and prompt it to output a yes/no answer, and evaluate answer accuracy. Here, we find that increasing attribution comes at a slight cost in downstream task performance: answer accuracy drops modestly for *all* revision models (up to 2.6%). We suspect that this may be due to noisy retrievals, which sometimes provide misleading evidence (exemplified in Figure 6.7d). Furthermore, even though revisions can address factoid errors in the passage (e.g., *"Homer Simpson has 5 fingers"* from Figure 6.7e), RARR currently does not try to modify subsequent reasoning steps which may no longer be logically entailed (e.g., *"He only needs one hand to count to 5"*).

6.6 Summary of Contributions

Language models have become increasingly good at capturing general facts about that world, however, they store this knowledge imprecisely in their weights. Because of this, they often struggle to precisely memorize "factoid" knowledge which leads to them producing unsubstantiated claims. To address this, we propose editing a language models output to be factually consistent against a set of retrieved evidence. We formalize this task as *Editing for Attribution*, present datasets for evaluating on, and develop automatic evaluation metrics for this task. In addition, we present RARR, a retrieve-and-edit method for revising language model generations to make them attributable to the

researched evidence. From experiments on text passages generated by different models on various domains, we showed that RARR can revise the passages to improve attribution while preserving other desirable properties such as writing style or structure. Furthermore, RARR sits on top of existing generation models without needing to re-design or re-train LMs.

The text in this chapter is based on the publications:

- *RARR: Researching and Revising What Language Models Say, Using Language Models* [45] (ACL 2023).

The results presented in this chapter can be reproduced at https://github.com/anthonywchen/RARR.

Chapter 7

PURR: Training Efficient Editors By Denoising Language Models Corruptions

7.1 Introduction

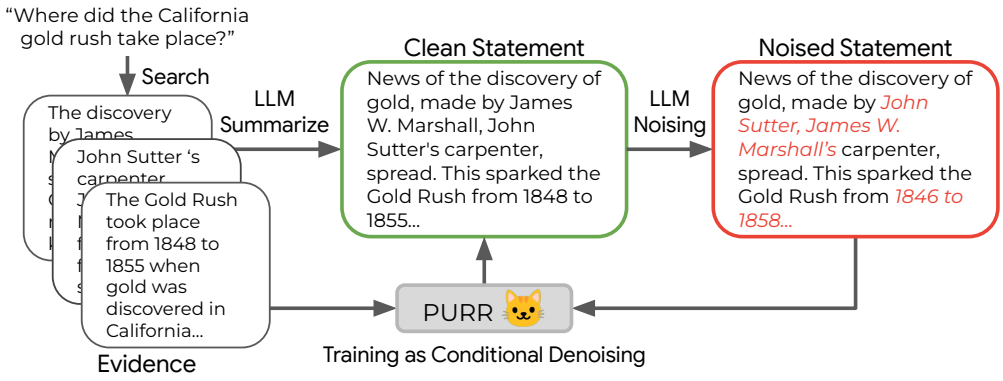
As the strengths of large language models (LLMs) have become prominent [15, 29, 155], so too have their weaknesses [10]. As we showed in Chapter 6, a glaring weakness of LLMs is their penchant for generating false, biased, or misleading claims in a phenomena broadly referred to as "hallucinations" [102, 80, 99, 131]. Most LLMs also do not ground their generations to any source, exacerbating this weakness [130].

Post-hoc attribution and edit strategies offer promising solutions to tackle the problems of grounding and hallucination in language models [151, 45]. These approaches retrieve supporting evidence to attribute the output (referred to as a claim) of a language model, followed by an editor that corrects factual errors in the claim, ensuring consistency with the evidence. A notable advantage of post-hoc methods is their modularity: they can be easily applied to any text regardless of their generation source. However, existing editors exhibit distinct strengths and weaknesses. Sufficiently large

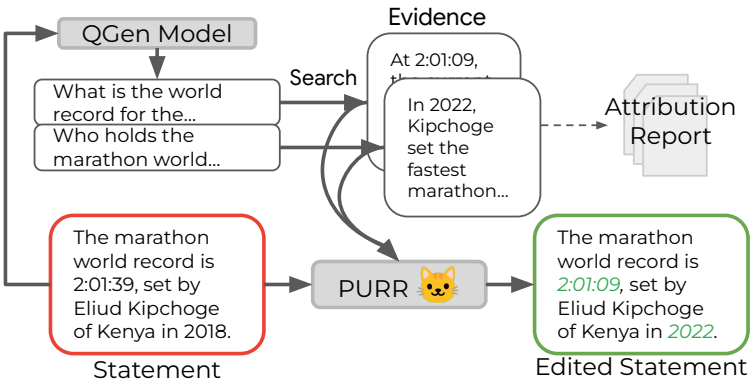
language models can be few-shot prompted to perform editing [3, 45]. However, there is currently a steep compute-quality tradeoff, where only the largest, most expensive models can perform this task well. We showed this when we developed RARR, a prompted editor that works best with langauge models in the hundreds of billions of parameters. In contrast, much smaller, cheaper models can be fine-tuned to perform editing, but are limited to specific domains where adequate training data is available [62, 139].

Instead of utilizing LLMs as prompted editors, in this chapter we leverage their general-purpose capabilities to introduce challenging corruptions (*i.e.*, noise) to clean pieces of text. Subsequently, we fine-tune compact editors to denoise these corruptions by grounding onto relevant evidence. While text to corrupt is readily available, we do not assume that paired relevant evidence is provided. To tackle this, our data generation pipeline first searches for a collection of topically related evidence. We then employ an LLM summarize the evidence into a claim which is then noised (Fig. 7.1a). The evidence is then used to ground the denoising. In contrast to existing work that assumes access to relevant paired evidence to ground the edit when training [4] or assumes edit data is provided for training [139, 62], our approach eliminates these assumptions. Furthermore, unlike distillation where a challenging distillation set is vital and the student model generally under-performs the teacher [13, 148], our noising process introduces challenging corruptions and our resulting editor trained on these corruptions surpasses the performance of the LLM used for noising when the same LLM is employed for prompted editing on multiple datasets.

Our *Petite Unsupervised Research and Revision* model, PURR, is built by fine-tuning a fusion-in-decoder T5 model on denoising data from our data generation pipeline [123, 64]. We evaluate PURR on outputs of large language models following the evaluation setup in Chapter 6. On all benchmarks, PURR outperforms much more expensive LLM-prompted editors in improving attribution while being orders of magnitude faster.



(a) **Training PURR.** Given a seed query, we search for relevant evidence and summarize them into a claim which we corrupt. PURR is trained to denoise the corruption conditioned on the evidence.



(b) **Using PURR.** Given an ungrounded statement, we generate questions to search for relevant evidence which is then used to produce an edit.

Figure 7.1: **Training and Using PURR.**

7.2 Efficient Editing via Denoising

In this section, we present an overview of PURR, highlight its distinguishing features compared to baselines, and describe the denoising training strategy.

7.2.1 Overview of PURR at Inference Time

We first describe how PURR is used at inference time and highlight the differences between PURR and baselines (Fig. 7.1b). Similar to EFEC, PURR is built upon on the T5 model, specifically

T5-large. Furthermore, our editing framework adopts a similar approach to EFEC in terms of incorporating all available evidence simultaneously when making an edit. However, instead of concatenating the evidence in the input, we employ fusion-in-decoder (FiD) to effectively aggregate information across evidence [64]. This approach has demonstrated superior performance in merging information and allows us to surpass the context length limits imposed by modern language models. Finally, rather than employing a prompted language model for query generation during the research stage, we employ distillation to train a T5-large query generation model. Although our primary focus lies in enhancing the editing process, we opt for distillation during query generation as well to ensure that our editing pipeline does not rely on prompting.

7.2.2 Creating Training Data via Noising

To train an editor to fix hallucinations, we need a dataset consisting of a clean statements, y, which are paired with a set of supporting evidence $E = \{e_1, e_2, \ldots, e_n\}$, as well as a corrupted statement, x. While collecting this data manually is feasible, doing so can be expensive, requiring scouring for evidence to ground an LLM generation followed by removing any inaccuracies in the generation. Instead, we remove this bottleneck by leveraging the general purpose generation capabilities of LLMs to create a training set in a completely fashion. We generate clean statements by providing a set of topically related evidence to the LLM, and then corrupt the statements to create simulated hallucinations (Fig. 7.1a).

Generating Clean Statements With Evidence The first step is to create a statement, y, paired with a set of evidence, E, that attributes (*i.e.,* grounds) the statement. Our pipeline only requires a set of queries in the domain of interest to get started. We start with a query, q, and use a search engine to find evidence related to the question. We take the top web pages from the search engine and chunk them into passages. Using the same cross-encoder from the attribution report scoring module, we bin the passages that have the highest relevant score (beyond some threshold) to q

q:	Who will be the new coach of the Detroit lions?
E^+:	- On Jan. 20, 2021 the Detroit Lions named Dan Campbell the franchise's new head coach... - Campbell possesses 23 years of NFL experience, including 12 years as a coach and 11 as a player. In his first year... - On Jan. 20, 2021 the Detroit Lions named Dan Campbell the franchise's new head coach...
$\boldsymbol{x}/y$:	Dan Campbell was appointed the new ~~head~~ ***assistant*** coach of the Detroit Lions on January 20, 2021. With ~~23~~ ***19*** years of NFL experience, 12 as a coach and ~~11~~ ***7*** as a player...
q:	What is the neurological explanation for why people laugh when they're nervous or frightened?
E^+:	- A 2015 Yale study found people respond with a variety of emotions to strong outside stimuli... - Vilayanur Ramachandran states "We have nervous laughter because we want to make ourselves think what horrible thing we encountered isn't really as horrible as it appears"... - Stanley Milgram conducted one of the earliest studies about nervous laughter in the 1960s. His study revealed that people often laughed nervously in uncomfortable situations...
$\boldsymbol{x}/y$:	Yale researchers in 2015 found people often respond to strong external stimuli with a variety of emotions, including ~~nervous laughter~~ ***anger***. ~~Stanley Milgram's~~ ***Vilayanur Ramachandran's*** 1960s study also observed this in uncomfortable situations. Neuroscientist ~~Vilayanur Ramachandran~~ ***Stanley Milgram*** theorizes that people laugh when....

Table 7.1: **Training Examples**. Our editing data covers a variety of domains and introduces challenging corruptions (*e.g.,* numerical, entity, and semantic role). q is the seed query, E^+ is the gold evidence set used to generate the clean statement, y is the clean statement and ***$\boldsymbol{x}$ is the corrupt statement***.

into a set of gold evidence $E^+ = \{e_1^+, e_2^+, \ldots, e_i^+\}$ and the rest of the passages into a set of hard negative evidences $E^- = \{e_1^-, e_2^-, \ldots, e_j^-\}$. In our pipeline, we restrict the size of E^+ to contain at most four pieces of evidence. The resulting evidence set is the union of the gold and hard negative evidences $E = E^+ \cup E^-$. We then prompt a large language model to do zero-shot multi-document summarization of the gold evidence set, E^+. We use the resulting summary as the clean statement, y, and upon manual inspection, the summary has a high degree of faithfulness to the evidence set.

Noising and Conditional Denoising We take the clean statement, y, and noise it by prompting a large language model to corrupt the text resulting in the corruption x. Our prompt contains examples of corruptions, and covers a wide range of linguistic phenomena we observe when it comes to LLM hallucinations. These include incorrect dates and entities, semantic role errors, and quantification errors. Once noised claims paired with evidence is available, an editor can be trained by fine-tuning a sequence-to-sequence model to maximize $P(y|x, E)$. We call the resulting editor from denoising PURR.

7.2.3 Dataset Statistics and Training Details

We utilized GPT-3.5 `text-davinci-003` to facilitate the process of generating summaries and introducing corruption. Our choice of this particular model ensures that our generation strategy can be easily replicated. We started with roughly 6,000 seed queries covering a variety of domains and topics resulting in an edit dataset of 6,000 instances (Table 7.1). We reserve 10% for validation and use the resulting 90% for training. Each instance cost roughly 4 cents to generate and in total cost of roughly $250.

We fine-tune T5-large on our dataset using the validation loss to tune hyperparameters and determine training stoppage. During training, we pair each corrupted statement, x, with four pieces of evidence from the accompanying gold evidence set, E^+, to ground the edit and produce the clean statement, y. In the event that the gold evidence set has fewer than four evidence snippets, we randomly sample evidence from the negative evidence set, E^-, until we hit four snippets. We found adding negative evidence during training helps PURR ignore irrelevant evidence during inference.

Model	Attr. ($x \rightarrow y$)	Pres.	$F1_{AP}$
	PALM outputs on NQ		
EFEC	44.7 $\rightarrow$ **63.9**	39.6	48.5
RARR	44.7 $\rightarrow$ 53.8	89.6	67.2
PURR	44.8 $\rightarrow$ 59.8	**91.0**	**72.2**
	PALM outputs on SQA		
EFEC	37.2 $\rightarrow$ **58.2**	31.0	40.4
RARR	37.2 $\rightarrow$ 44.6	89.9	59.6
PURR	36.9 $\rightarrow$ 47.1	**92.0**	**62.3**
	LaMBDA outputs on QreCC		
EFEC	18.4 $\rightarrow$ **47.2**	39.0	42.7
RARR	18.4 $\rightarrow$ 28.7	80.1	42.2
PURR	16.8 $\rightarrow$ 33.0	**85.8**	**47.7**

Table 7.2: **Results on the *Editing for Attribution* task.** We report the attribution of the statement before and after editing, preservation after editing, and $F1_{AP}$ which combines attribution and preservation. Results are on LLM outputs on factoid question answering, long reasoning question answering, and dialog.

7.3 Results

7.3.1 Primary Quantitative Results

We provide results on the editing-for-attribution task in Table 7.2. We report the attribution of the claim before and after editing and the preservation of the edited claim. Our primary metric, $F1_{AP}$, is the harmonic mean between the attribution and preservation of the edited claim. We first turn our attention to the baselines. EFEC, the editor that was fine-tuned with evidence largely from Wikipedia, struggles on this task. While EFEC improves attribution, this comes at the large expense of preservation and we see this in practice as EFEC tends to make large changes to the claim. RARR, the prompted editor, does not improve attribution as much as EFEC. However it is significantly better at preserving the intent of the original claim. Because of this, RARR is much better on the unified $F1_{AP}$ metric.

PURR improves upon the results of RARR by generally making smaller changes to the claim while

improving the attribution in this more limited edit. Because of this, PURR pushes the state-of-the-art on the unified $F1_{AP}$ metric an all three tasks. Moreover, PURR is significantly more efficient to use by virtue of its size.

7.3.2 Breaking Down the Numbers

We dig into the edits to get a better sense of where PURR improves on the baselines. Based on the preservation, $\text{Pres}_{(x,y)}$, and attribution scores of the original statement, $\text{Attr}_{(x,A)}$, and edited statement, $\text{Attr}_{(y,A)}$, we say an edit can fall into one of the following sets:

- **Huge Edit**: We say an edit is "huge" if preservation is low: $\text{Pres}_{(x,y)} < 0.5$.
- **Bad Edit**: We say an edit is "bad" if the attribution after editing is lower than before: $\text{Attr}_{(y,A)} - \text{Attr}_{(x,A)} < -0.1$.
- **Unnecessary Edit**: We say an edit is "unnecessary" if it is a bad edit and also $\text{Attr}_{(x,A)} > 0.9$. This means the editor made a poor edit when the attribution was already near perfect before editing.
- **Good Edit**: We say an edit is "good" if attribution has significantly improved while preservation is high: $\text{Attr}_{(y,A)} - \text{Attr}_{(x,A)} > 0.3$ and $\text{Pres}_{(x,y)} > 0.7$.

Note that unnecessary edits are a subset of bad edits. We take the 150 instances in the Natural Questions test set and categorize the edits each editor makes in Figure 7.2. On a majority of claims, EFEC makes large edits while rarely making edits that improve attribution while preserving the original claim. RARR does a much better job at minimizing large edits but there are still cases where RARR edits a claim in a way that reduces the attribution. PURR almost never makes large edits and never edits a claim when it is near-perfect in a way that reduces attribution. PURR also makes more good edits compared to the baselines.

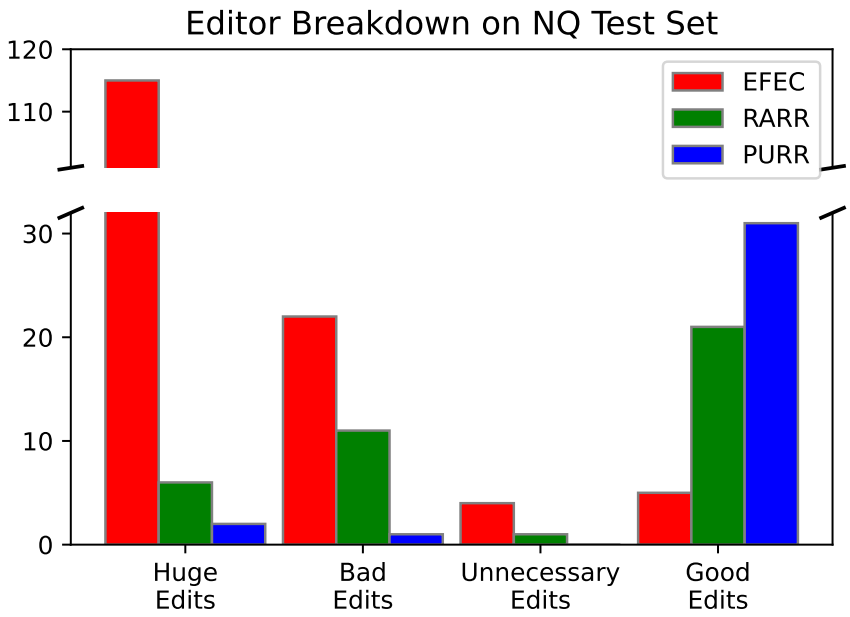


Figure 7.2: **Breakdown of edit types each editor makes** on the Natural Questions test set. EFEC makes huge edits while RARR sometimes over edits. PURR does a much better job at balancing attribution and preservation while rarely over-editing.

7.3.3 Qualitative Analysis

We then dig into the PURR predictions and diagnose the strengths of PURR and examine where there is room for improvement. We show examples in Table 7.3 that we found are representative of the strengths of PURR and areas of potential improvement. We find that PURR is extremely strong at fixing entity and numerical hallucinations as well as longer spans. Additionally, because PURR uses fusion-in-decoder, it is adept at merging information across multiple pieces of evidence to make an edit. We noticed several instances where there are challenging distractors in evidence that can lead to an erroneous edit. Future work will introduce stronger corruptions in the data generation pipeline to better handle this case.

We next analyze the entire inference pipeline of PURR (Fig. 7.1b), which includes the question generation model, the search engine, and the editor itself. Our goal is to see when there is an error, which component is responsible. On the Natural Questions subset of the evaluation, we examine 20 instances where the attribution after editing, $\text{Attr}_{(y,A)}$, is less than 0.30. Our qualitative analysis

is provided in Table 7.4. Roughly 80% of the instances have low attribution after editing because either the question generation model we used did not fully cover the information in the claim or our search procedure did not find the best evidence for editing. We believe the question generation is the easier problem to fix while search is a much harder problem. Editing is a fairly small issue in comparison. Finally, there are some claims that fall into a "miscellaenous" category, either because it was not contextualized enough to properly edit or because the automatic metric erroneously assigned a low score.

7.3.4 Inference Speed and Cost Comparisons of Fine-tuned vs Prompted Editors

A key advantage of PURR over prompt-based editors are the lower computational costs. RARR, a prompt-based editor built upon 540B PALM, runs on dozens of TPUs and takes approximately 40 seconds to edit a single statement. In comparison, PURR can run on a 12GB GPU and takes approximately 2 seconds to edit a single statement on a Titan-RTX. Considering generating our training set costs less than $300 USD which is quickly amortized, we recommend our synthetic data generation strategy for large-scale deployment given the speed and cost savings of PURR.

7.4 Summary of Contributions

Factuality and attribution are vital for the safe deployment of large language models. However, these mechanisms are inherently lacking in LLMs. In the last chapter, we proposed augmenting the outputs of LLMs by retrieving evidence to attribute their outputs followed by prompting another LLM to edit the outputs to remove inconsistencies. However, there is a heavy computational cost which bottleneck these methods which motivates a need to develop efficient editors, but this is hindered by training data scarcity. To overcome these challenges, we use LLMs to corrupt text and

fine-tune compact editors to denoise these faux hallucinations. Our denoising method is completely unsupervised and our resulting editor, PURR, improves attribution performance across various datasets over prompted editors, while being order of magnitude faster to execute.

The text in this chapter is based on the unreferreed manuscripts:

- *PURR: Efficiently Editing Language Model Hallucinations by Denoising Language Model Corruptions* [20] (arXiv:2305.14908)

Correctly revising an entity/span.
x: The US stopped printing $2 bills in 1966. The reason was that vending machines would not accept it. In 1976, the $2 bill was reintroduced...
E: During that time, most goods and services were less than a dollar, making paper currency impractical. As need and use declined, the Federal Reserve stopped printing $2 bills in 1966. In 1976, the Federal Reserve brought back the $2-dollar bill.
y: The US stopped printing $2 bills in 1966. The reason was that **most goods and services were less than a dollar.** In 1976, the $2 bill was reintroduced...
$Pres_{(x,y)}$: 0.81 $Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.61 $\rightarrow$ 0.84
Reasoning across multiple pieces of evidence
x: In 1899, the first driver's license was issued in France. The first license in the United States was issued in 1902 in Massachusetts.
E: **-** France introduced the first driving licences under the Paris Police Ordinance of 14 August 1893... **-** Massachusetts and Missouri were the first states to require a license in 1903, but there was no test associated with it...
y: In **1893**, the first driver's license was issued in France. The first license in the United States was issued in **1903** in Massachusetts.
$Pres_{(x,y)}$: 0.90 $Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.03 $\rightarrow$ 0.96
Challenging distractors in evidence
x: The first orca put in captivity was in 1965. It was at Seattle, Washington's Seattle Aquarium. The orca's name was Wanda.
E: **-** On Nov. 18, 1961 a killer whale had been spotted and a collection crew from the **Marineland of the Pacific** scrambled to capture her. Never had an orca been captured alive, and Wanda was trying to avoid being the first... **-** The first orca to be captured in 1961 was found off the coast of California and given the name Wanda.
y: The first orca put in captivity was in **1961**. It was at **Marineland of the Pacific**. The orca's name was Wanda.
$Pres_{(x,y)}$: 0.77 $Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.33 $\rightarrow$ 0.77

Table 7.3: **Example of good and bad revisions with PURR.** x = claim; E = relevant evidence; y = edited claim using E. PURR can handle hallucinated entities and spans as well as merge information across evidence to edit. PURR can struggle when there are challenging distractors in a piece of evidence.

Query Generation Missing Coverage (*35%*)
x: Legends of Tomorrow season 3 finale aired on April 9, 2018. It's title is No Country for Old Dads and is a 42-minute episode.
Q: **-** When did the season 3 finale of Legends of Tomorrow air? **-** ~~*What's the title of Legends of Tomorrow season 3 finale?*~~ **-** ~~*How long is the season 3 finale of Legends of Tomorrow?*~~
y: Legends of Tomorrow season 3 finale aired on October 10, 2017. It's title is Freakshow and is a 43-minute episode.
$Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.01 $\rightarrow$ 0.01
Search Not Finding Best Evidence (*45%*)
x: Bryce Dallas Howard plays the character of the mother of Cindy Lou Who in the 2018 animated movie The Grinch.
E: ~~*Donna Lou Who is one of the characters of the 2018 Illumination animated film, Dr. Seuss' The Grinch. She's the mother of Cindy Lou Who and twin infants sons, Buster and Bean. She's voiced by Rashida Jones .*~~
y: No edit.
$Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.05 $\rightarrow$ 0.05
Editor Error/Confusion (*5%*)
x: James Marion Sims was the first to perform the first c section in 1846. This was in the US.
E: **-** The first recorded cesarean where both mother and child survived was done in Switzerland, in 1500. That was also a husband-wife affair, although in this case Jacob Nufer was a swine gelder, not a doctor. **-** The first documented cesarean section on a living woman was performed in 1610; she died 25 days after the surgery.
y: James Marion Sims was the first to perform the first c section in **1610**. This was in the UK.
$Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.00 $\rightarrow$ 0.20
Miscellaneous (*15%*)
x: The song was written by Tony Macaulay and John Macleod and recorded by Helen Shapiro in 1961.
y: The song was written by Dub Allbritten and Ronnie Self and recorded by The Jackson 5 in 1960.
Claim not contextualized enough to properly edit.
$Attr_{(A,x)} \rightarrow Attr_{(A,y)}$: 0.00 $\rightarrow$ 0.01

Table 7.4: **Error Analysis of PURR Inference Pipeline.** We sample **20** edits from the NQ set where attribution is low after editing and categorize why by component. x = claim; Q = generated queries used for search, E = relevant evidence; y = edited claim using E. ~~*Strike-through text*~~ represents a query that wasn't generated or evidence that wasn't retrieved but should have been.

Chapter 8

Conclusions

8.1 Summary of Contributions

As large language models continue to grow in size and capacity, they will continue to see increasing deployment in real-world applications. Because of this, it is imperative that language models are reliable and generate factually accurate statements consistent with the current state of the world. The work presented in this dissertation is motivated towards making progress towards this goal

In Part I of this dissertations, we aim to create automatic metrics to **evaluate** the facts that language models generate. Automatic evaluation is vital for quickly and cheaply evaluating the progress of large language models. Our contributions here include MOCHA, a dataset consisting of human judgement scores on language model generations. We use MOCHA to train a model-based metric, LERC. We show that LERC surpasses metrics that existed at the time for the task of reading comprehension.

In Part II of this dissertation, we aim to **characterize** how language models leverage and balance contextual and parametric knowledge. Understanding this is imperative since work has shown that

models which over-rely on parametric are prone to generating factually inaccurate statements. We first introduced AmbER Sets, a dataset that evaluates how biased retrieval systems are. We find that modern retrieval systems, which are built upon language models themselves, are prone to popularity bias. We then find that this affects downstream language model generations. We then introduced knowledge conflicts to probe how often language models over-rely on parametric knowledge. We discover several facts that exacerbate this over-reliance, including model size and retriever quality, as well as propose mitigation strategies.

Part III of this dissertation is concerned with **correcting** the facts that language models generate so that they can be more safely deployed in real-world applications. For formalize this problem as *Editing for Attribution* and introduce evaluation datasets and metrics to tackle this problem. We then introduce strong baselines for this problem, RARR, a prompt-based editor, and PURR, a fine-tuned editor that surpasses RARR on the task while being orders of magnitude faster to use.

8.2 Impact

This thesis provides concrete methodologies in evaluating and fixing how language models represent and generate facts. For evaluation, recent work has adapted model-based metrics via prompting instead of fine-tuning [43]. In the years since our knowledge conflicts work came out, characterizing how language models capture knowledge and use that knowledge has becoming increasingly important. Our framework of probing at how models use contextual and parametric knowledge have become a popular methodology for evaluating language models and recent work has found incorporating knowledge conflicts in few-shot prompts can improve robustness [147, 168]. Moreover, our findings from our knowledge conflicts work on facts that exacerbate over-reliance on parametric knowledge (such as model size) have been shown in many instances [103]. While our work on correcting language model generations has happened fairly recently, it has rapidly grown in popularity as a research topic, inspiring a flurry of research in this area [48, 25].

Correction

8.3 Limitations and Future Work

In this section, we reflect on the research presented in this dissertation and identify limitations of the author's work. We then detail ideas for future research continuing on the theme of evaluation, characterization, and correction.

Evaluation As described in Chapter 2, learned model-based metrics are flexible in that they can be adapted specifically to the generation task of interest. However, such metrics generally require human judgements scores to train on. In the case of MOCHA, our data collection process cost hundreds of dollars as well as hundreds of hours of manual inspection to ensure high data quality. This makes training model-based metrics expensive and time-consuming to create. An another note, because learned metrics are tailored to the task of interest, our learned metric, LERC which was developed for evaluating question answering, cannot be easily adapted to evaluate a new task, such as summarization. These drawbacks limit the ability to quickly develop and broadly apply metrics like LERC.

To circumvent the aforementioned issues, an interesting line of future research is prompt-based metrics. In prompt-based methods, one only needs to provide several examples of human judgement scores as in-context examples, thus achieving an evaluation metric that can be easily and quickly adapted to any new task of interest. While some have been proposed [43], they currently work only on the largest language models, which makes using them for evaluation computationally expensive. One research area to circumvent this requirement is to tailor smaller language models to be better prompted evaluation metrics via instruction tuning [137, 30, 161].

Characterization Our work on knowledge conflicts was presented primarily as an evaluation benchmark for measuring a language model's tendency to hallucinate information. There are few note-worthy limitations of this work. The first is that our substitution framework only considered substituting named-entities when creating knowledge conflicting instances. This leaves open questions as to how large language models handle situations when presented with longer spans that contradict parametric knowledge. Moreover, because our substitution framework for creating knowledge conflicts was automated, it sometimes generated incomprehensible instances. Finally, our method was primarily for evaluation, and leveraging knowledge conflicts to train more faithful language models is an open and exciting area of future research.

Odddne interesting finding in our knowledge conflicts work showed that by fine-tuning a language model on knowledge conflicts (*i.e.,* counterfactual information), hallucination rates were drastically mitigated. This leads to an exciting future line of work on pre-training language models with knowledge conflicts to make them less prone to hallucinations without the need for additional fine-tuning.

Correction Our work on editing was focused primarily on fixing the output of large language models to be more factual. And in this area, our methods, RARR and PURR, were able to greatly improve the factuality of language model generations. However, generated text can often include things like opinions (*e.g.,* red is the best color) or conjectures (*e.g.,* I think it will rain tomorrow), which do not need to be checked for factuality. However, our methods currently have no way of distinguishing between these statements and factual statements that do need to be checked. In addition, there are many other aspects that one may want to edit for that does not revolve around factuality. For instance, one may want to edit a large language model's output for creative tasks, such as song generation. In these cases, a use may wish to correct aspects like rhythm, length, and theme, which RARR and PURR cannot currently do.

Correction beyond factuality is an exciting line of work that requires more methodologies to be

developed. On this note large language models are prone to biased and toxic generations [10], and one interesting area to extend our editing work is to remove these undesirable aspects from language model generations.

Bibliography

[1] M. Ahn et al. Do as I can, not as I say: Grounding language in robotic affordances. *ArXiv*, abs/2204.01691, 2022.

[2] R. Anantha, S. Vakulenko, Z. Tu, S. Longpre, S. G. Pulman, and S. Chappidi. Open-domain question answering goes conversational via question rewriting. In *NAACL*, 2021.

[3] Y. Bai, S. Kadavath, S. Kundu, A. Askell, J. Kernion, A. Jones, A. Chen, A. Goldie, A. Mirhoseini, C. McKinnon, C. Chen, C. Olsson, C. Olah, D. Hernandez, D. Drain, D. Ganguli, D. Li, E. Tran-Johnson, E. Perez, J. Kerr, J. Mueller, J. Ladish, J. Landau, K. Ndousse, K. Lukosuite, L. Lovitt, M. Sellitto, N. Elhage, N. Schiefer, N. Mercado, N. DasSarma, R. Lasenby, R. Larson, S. Ringer, S. Johnston, S. Kravec, S. E. Showk, S. Fort, T. Lanham, T. Telleen-Lawton, T. Conerly, T. Henighan, T. Hume, S. R. Bowman, Z. Hatfield-Dodds, B. Mann, D. Amodei, N. Joseph, S. McCandlish, T. Brown, and J. Kaplan. Constitutional ai: Harmlessness from ai feedback, 2022.

[4] V. Balachandran, H. Hajishirzi, W. Cohen, and Y. Tsvetkov. Correcting diverse factual errors in abstractive summarization via post-editing and language model infilling. In *Conference on Empirical Methods in Natural Language Processing*, 2022.

[5] K. Balog, P. Serdyukov, and A. D. Vries. Overview of the trec 2010 entity track. In *TREC*, 2010.

[6] K. Balog, P. Serdyukov, and A. D. Vries. Overview of the trec 2011 entity track. In *TREC*, 2011.

[7] S. Banerjee and A. Lavie. Meteor: An automatic metric for mt evaluation with improved correlation with human judgments. In *IEEvaluation@ACL*, 2005.

[8] L. Bauer, Y. Wang, and M. Bansal. Commonsense for generative multi-hop question answering tasks. In *ACL*, 2018.

[9] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? In *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, FAccT '21, page 610–623, New York, NY, USA, 2021. Association for Computing Machinery.

[10] E. M. Bender, T. Gebru, A. McMillan-Major, and S. Shmitchell. On the dangers of stochastic parrots: Can language models be too big? *Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency*, 2021.

[11] Y. Bengio, R. Ducharme, P. Vincent, and C. Janvin. A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155, 2003.

[12] A. L. Berger, S. D. Pietra, and V. J. D. Pietra. A maximum entropy approach to natural language processing. *Comput. Linguistics*, 22:39–71, 1996.

[13] L. Beyer, X. Zhai, A. Royer, L. Markeeva, R. Anil, and A. Kolesnikov. Knowledge distillation: A good teacher is patient and consistent. *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10915–10924, 2021.

[14] B. Bohnet, V. Q. Tran, P. Verga, R. Aharoni, D. Andor, L. B. Soares, J. Eisenstein, K. Ganchev, J. Herzig, K. Hui, T. Kwiatkowski, J. Ma, J. Ni, T. Schuster, W. W. Cohen, M. Collins, D. Das, D. Metzler, S. Petrov, and K. Webster. Attributed question answering: Evaluation and modeling for attributed large language models. *ArXiv*, 2022.

[15] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. J. Henighan, R. Child, A. Ramesh, D. M. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen, E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In *NeurIPS*, 2020.

[16] M. Carpuat and D. Wu. Improving statistical machine translation using word sense disambiguation. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 61–72, Prague, Czech Republic, June 2007. Association for Computational Linguistics.

[17] W. B. Cavnar and J. M. Trenkle. N-gram-based text categorization, 1994.

[18] D. M. Cer, M. T. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. Semeval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. *ArXiv*, abs/1708.00055, 2017.

[19] A. Chen, P. Gudipati, S. Longpre, X. Ling, and S. Singh. Evaluating entity disambiguation and the role of popularity in retrieval-based NLP. 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 4472–4485, Online, Aug. 2021. Association for Computational Linguistics.

[20] A. Chen, P. Pasupat, S. Singh, H. Lee, and K. Guu. Purr: Efficiently editing language model hallucinations by denoising language model corruptions. *ArXiv*, 20223.

[21] A. Chen, G. Stanovsky, S. Singh, and M. Gardner. Evaluating question answering evaluation. In *EMNLP Workshop on Machine Reading and Question Answering (MRQA)*, 2019.

[22] A. Chen, G. Stanovsky, S. Singh, and M. Gardner. MOCHA: A dataset for training and evaluating generative reading comprehension metrics. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6521–6532, Online, Nov. 2020. Association for Computational Linguistics.

[23] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading wikipedia to answer open-domain questions. In *ACL*, 2017.

[24] D. Chen, A. Fisch, J. Weston, and A. Bordes. 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, 2017. Association for Computational Linguistics.

[25] J. Chen, G. Kim, A. Sriram, G. Durrett, and E. Choi. Complex claim verification with evidence retrieved in the wild. *ArXiv*, abs/2305.11859, 2023.

[26] F. ChenStanley and GoodmanJoshua. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 1999.

[27] K. Cho, B. van Merrienboer, Çaglar Gülçehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *Conference on Empirical Methods in Natural Language Processing*, 2014.

[28] E. Choi, J. Palomaki, M. Lamm, T. Kwiatkowski, D. Das, and M. Collins. Decontextualization: Making sentences stand-alone. *TACL*, 9:447–461, 2021.

[29] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, P. Schuh, K. Shi, S. Tsvyashchenko, J. Maynez, A. Rao, P. Barnes, Y. Tay, N. M. Shazeer, V. Prabhakaran, E. Reif, N. Du, B. C. Hutchinson, R. Pope, J. Bradbury, J. Austin, M. Isard, G. Gur-Ari, P. Yin, T. Duke, A. Levskaya, S. Ghemawat, S. Dev, H. Michalewski, X. García, V. Misra, K. Robinson, L. Fedus, D. Zhou, D. Ippolito, D. Luan, H. Lim, B. Zoph, A. Spiridonov, R. Sepassi, D. Dohan, S. Agrawal, M. Omernick, A. M. Dai, T. S. Pillai, M. Pellat, A. Lewkowycz, E. Moreira, R. Child, O. Polozov, K. Lee, Z. Zhou, X. Wang, B. Saeta, M. Díaz, O. Firat, M. Catasta, J. Wei, K. S. Meier-Hellstern, D. Eck, J. Dean, S. Petrov, and N. Fiedel. Palm: Scaling language modeling with pathways. *ArXiv*, abs/2204.02311, 2022.

[30] H. W. Chung, L. Hou, S. Longpre, B. Zoph, Y. Tay, W. Fedus, E. Li, X. Wang, M. Dehghani, S. Brahma, A. Webson, S. S. Gu, Z. Dai, M. Suzgun, X. Chen, A. Chowdhery, D. Valter, S. Narang, G. Mishra, A. W. Yu, V. Zhao, Y. Huang, A. M. Dai, H. Yu, S. Petrov, E. H. hsin Chi, J. Dean, J. Devlin, A. Roberts, D. Zhou, Q. Le, and J. Wei. Scaling instruction-finetuned language models. *ArXiv*, abs/2210.11416, 2022.

[31] C. Clark, K. Lee, M.-W. Chang, T. Kwiatkowski, M. Collins, and K. Toutanova. Boolq: Exploring the surprising difficulty of natural yes/no questions. In *NAACL-HLT*, 2019.

[32] W. B. Croft and D. J. Harper. Using probabilistic models of document retrieval without relevance information. *J. Documentation*, 35:285–295, 1979.

[33] Y. Cui, G. Yang, A. Veit, X. Huang, and S. J. Belongie. Learning to evaluate image captioning. In *CVPR*, 2018.

[34] Z. Dai, Z. Yang, Y. Yang, J. Carbonell, Q. Le, and R. Salakhutdinov. Transformer-XL: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, Florence, Italy, July 2019. Association for Computational Linguistics.

[35] P. Dasigi, N. F. Liu, A. Marasović, N. A. Smith, and M. Gardner. Quoref: A reading comprehension dataset with questions requiring coreferential reasoning. In *EMNLP*, 2019.

[36] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT*, 2019.

[37] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *NAACL-HLT*, 2019.

[38] D. Dua, Y. Wang, P. Dasigi, G. Stanovsky, S. Singh, and M. Gardner. Drop: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *NAACL-HLT*, 2019.

[39] N. Dziri, S. Milton, M. Yu, O. Zaiane, and S. Reddy. On the origin of hallucinations in conversational models: Is it the datasets or the models? In *NAACL*, 2022.

[40] S. Edunov, M. Ott, M. Ranzato, and M. Auli. On the evaluation of machine translation systems trained with back-translation. *ArXiv*, abs/1908.05204, 2019.

[41] H. ElSahar, P. Vougiouklis, A. Remaci, C. Gravier, J. Hare, F. Laforest, and E. Simperl. T-rex: A large scale alignment of natural language with knowledge base triples. In *LREC*, 2018.

[42] A. Fisch, A. Talmor, R. Jia, M. Seo, E. Choi, and D. Chen. 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, 2019. Association for Computational Linguistics.

[43] J. Fu, S.-K. Ng, Z. Jiang, and P. Liu. Gptscore: Evaluate as you desire. *ArXiv*, abs/2302.04166, 2023.

[44] L. Gao, S. R. Biderman, S. Black, L. Golding, T. Hoppe, C. Foster, J. Phang, H. He, A. Thite, N. Nabeshima, S. Presser, and C. Leahy. The pile: An 800gb dataset of diverse text for language modeling. *ArXiv*, abs/2101.00027, 2020.

[45] L. Gao, Z. Dai, P. Pasupat, A. Chen, A. T. Chaganty, Y. Fan, V. Zhao, N. Lao, H. Lee, D.-C. Juan, and K. Guu. Rarr: Researching and revising what language models say, using language models. *ArXiv*, 2022.

[46] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. E. Peters, M. Schmitz, and L. Zettlemoyer. Allennlp: A deep semantic natural language processing platform. *ArXiv*, abs/1803.07640, 2018.

[47] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. H. S. Liu, M. E. Peters, M. Schmitz, and L. Zettlemoyer. A deep semantic natural language processing platform. *ArXiv*, abs/1803.07640, 2017.

[48] Z. Gero, C. Singh, H. Cheng, T. Naumann, M. Galley, J. Gao, and H. Poon. Self-verification improves few-shot clinical information extraction. *ArXiv*, abs/2306.00024, 2023.

[49] M. Geva, Y. Goldberg, and J. Berant. Are we modeling the task or the annotator? an investigation of annotator bias in natural language understanding datasets. In *EMNLP*, 2019.

[50] M. Geva, D. Khashabi, E. Segal, T. Khot, D. Roth, and J. Berant. Did Aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *TACL*, 9:346–361, 2021.

[51] J. Goodman. A bit of progress in language modeling. *ArXiv*, cs.CL/0108005, 2001.

[52] A. Graves, A. rahman Mohamed, and G. E. Hinton. Speech recognition with deep recurrent neural networks. *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 6645–6649, 2013.

[53] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang. Realm: Retrieval-augmented language model pre-training. *ArXiv*, abs/2002.08909, 2020.

[54] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*, 2020.

[55] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M.-W. Chang. REALM: Retrieval-augmented language model pre-training. In *ICML*, 2020.

[56] S. Hochreiter and J. Schmidhuber. Long short-term memory. *Neural Computation*, 9:1735–1780, 1997.

[57] J. Hoffart, M. Yosef, I. Bordino, H. Fürstenau, M. Pinkal, M. Spaniol, B. Taneva, S. Thater, and G. Weikum. Robust disambiguation of named entities in text. In *EMNLP*, 2011.

[58] O. Honovich, R. Aharoni, J. Herzig, H. Taitelbaum, D. Kukliansy, V. Cohen, T. Scialom, I. Szpektor, A. Hassidim, and Y. Matias. TRUE: Re-evaluating factual consistency evaluation. In *Workshop on Document-grounded Dialogue and Conversational Question Answering*, 2022.

[59] O. Honovich, L. Choshen, R. Aharoni, E. Neeman, I. Szpektor, and O. Abend. Q^2: Evaluating factual consistency in knowledge-grounded dialogues via question generation and question answering. In *EMNLP*, 2021.

[60] S. Hooker. The hardware lottery. *Communications of the ACM*, 64:58 – 65, 2020.

[61] L. Huang, R. L. Bras, C. Bhagavatula, and Y. Choi. Cosmos qa: Machine reading comprehension with contextual commonsense reasoning. In *EMNLP*, 2019.

[62] R. Iv, A. Passos, S. Singh, and M.-W. Chang. FRUIT: Faithfully reflecting updated information in text. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 3670–3686, Seattle, United States, July 2022. Association for Computational Linguistics.

[63] G. Izacard and E. Grave. 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, 2021. Association for Computational Linguistics.

[64] G. Izacard and E. Grave. 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, Apr. 2021. Association for Computational Linguistics.

[65] R. Jia and P. Liang. Adversarial examples for evaluating reading comprehension systems. In *EMNLP*, 2017.

[66] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7:535–547, 2017.

[67] J. Kaplan, S. McCandlish, T. J. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei. Scaling laws for neural language models. *ArXiv*, abs/2001.08361, 2020.

[68] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. 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, Nov. 2020. Association for Computational Linguistics.

[69] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Y. Wu, S. Edunov, D. Chen, and W. tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*, 2020.

[70] V. Karpukhin, B. Oğuz, S. Min, P. Lewis, L. Y. Wu, S. Edunov, D. Chen, and W. tau Yih. Dense passage retrieval for open-domain question answering. In *EMNLP*, 2020.

[71] S. M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Trans. Acoust. Speech Signal Process.*, 35:400–401, 1987.

[72] U. Khandelwal, O. Levy, D. Jurafsky, L. Zettlemoyer, and M. Lewis. Generalization through memorization: Nearest neighbor language models. In *ICLR*, 2020.

[73] D. Khashabi, S. Chaturvedi, M. Roth, S. Upadhyay, and D. Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *NAACL-HLT*, 2018.

[74] O. Khattab and M. A. Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2020.

[75] J. Kim, S. Choi, R. K. Amplayo, and S.-w. Hwang. Retrieval-augmented controllable review generation. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 2284–2295, Barcelona, Spain (Online), Dec. 2020. International Committee on Computational Linguistics.

[76] T. Kociský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette. The narrativeqa reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2017.

[77] P. Koehn, F. J. Och, and D. Marcu. Statistical phrase-based translation. In *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, pages 127–133, 2003.

[78] K. Krippendorff. Computing krippendorff's alpha-reliability, 2011.

[79] K. Krishna, A. Roy, and M. Iyyer. Hurdles to progress in long-form question answering. In *NAACL*, 2021.

[80] K. Krishna, A. Roy, and M. Iyyer. 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, June 2021. Association for Computational Linguistics.

[81] T. Kwiatkowski et al. Natural Questions: A benchmark for question answering research. *TACL*, 7:453–466, 2019.

[82] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. H. Hovy. Race: Large-scale reading comprehension dataset from examinations. In *EMNLP*, 2017.

[83] J. Lee, M. Sung, J. Kang, and D. Chen. Learning dense representations of phrases at scale. 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 6634–6647, Online, Aug. 2021. Association for Computational Linguistics.

[84] K. Lee, M.-W. Chang, and K. Toutanova. Latent retrieval for weakly supervised open domain question answering. In *ACL*, 2019.

[85] V. I. Levenshtein. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics. Doklady*, 10:707–710, 1965.

[86] M. Lewis, M. Ghazvininejad, G. Ghosh, A. Aghajanyan, S. I. Wang, and L. Zettlemoyer. Pre-training via paraphrasing. *ArXiv*, abs/2006.15020, 2020.

[87] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, 2020.

[88] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *NeurIPS*, 2020.

[89] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W. tau Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *NeurIPS*, 2020.

[90] P. Lewis, P. Stenetorp, and S. Riedel. 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, 2021. Association for Computational Linguistics.

[91] P. S. H. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, 2020.

[92] C.-Y. Lin. Rouge: A package for automatic evaluation of summaries. In *ACL*, 2004.

[93] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *ArXiv*, abs/1907.11692, 2019.

[94] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

[95] R. Logan, N. F. Liu, M. E. Peters, M. Gardner, and S. Singh. Barack's wife hillary: Using knowledge graphs for fact-aware language modeling. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5962–5971, Florence, Italy, July 2019. Association for Computational Linguistics.

[96] S. Longpre, Y. Lu, Z. Tu, and C. DuBois. An exploration of data augmentation and sampling techniques for domain-agnostic question answering. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 220–227, Hong Kong, China, 2019. Association for Computational Linguistics.

[97] S. Longpre, K. Perisetla, A. Chen, N. Ramesh, C. DuBois, and S. Singh. Entity-based knowledge conflicts in question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7052–7063, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics.

[98] S. Longpre, K. K. Perisetla, A. Chen, N. Ramesh, C. DuBois, and S. Singh. Entity-based knowledge conflicts in question answering. In *EMNLP*, 2021.

[99] S. Longpre, K. K. Perisetla, A. Chen, N. Ramesh, C. DuBois, and S. Singh. Entity-based knowledge conflicts in question answering. *ArXiv*, abs/2109.05052, 2021.

[100] C. Manning, P. Raghavan, and H. Schütze. *Introduction to information retrieval*, volume 1. Cambridge University Press, 2008.

[101] J. Maynez, S. Narayan, B. Bohnet, and R. T. McDonald. On faithfulness and factuality in abstractive summarization. In *ACL*, 2020.

[102] J. Maynez, S. Narayan, B. Bohnet, and R. T. McDonald. On faithfulness and factuality in abstractive summarization. *ArXiv*, abs/2005.00661, 2020.

[103] I. McKenzie, A. Lyzhov, A. Parrish, A. Prabhu, A. Mueller, N. Kim, S. Bowman, and E. Perez. The inverse scaling prize, 2022.

[104] J. Menick, M. Trebacz, V. Mikulik, J. Aslanides, F. Song, M. Chadwick, M. Glaese, S. Young, L. Campbell-Gillingham, G. Irving, and N. McAleese. Teaching language models to support answers with verified quotes. *ArXiv*, abs/2203.11147, 2022.

[105] T. Mikolov, M. Karafiát, L. Burget, J. H. Cernocký, and S. Khudanpur. Recurrent neural network based language model. In *Interspeech*, 2010.

[106] S. Min, E. Wallace, S. Singh, M. Gardner, H. Hajishirzi, and L. Zettlemoyer. Compositional questions do not necessitate multi-hop reasoning. In *ACL*, 2019.

[107] A. Mnih and Y. W. Teh. A fast and simple algorithm for training neural probabilistic language models. In *International Conference on Machine Learning*, 2012.

[108] F. Morin and Y. Bengio. Hierarchical probabilistic neural network language model. In *International Conference on Artificial Intelligence and Statistics*, 2005.

[109] S. Narayan, J. Maynez, R. K. Amplayo, K. Ganchev, A. Louis, F. Huot, D. Das, and M. Lapata. Conditional generation with a question-answering blueprint. *ArXiv*, abs/2207.00397, 2022.

[110] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@NIPS*, 2016.

[111] J. Ni, C. Qu, J. Lu, Z. Dai, G. H. Ábrego, J. Ma, V. Zhao, Y. Luan, K. B. Hall, M.-W. Chang, and Y. Yang. Large dual encoders are generalizable retrievers. *ArXiv*, abs/2112.07899, 2021.

[112] M. Nye, A. J. Andreassen, G. Gur-Ari, H. Michalewski, J. Austin, D. Bieber, D. Dohan, A. Lewkowycz, M. Bosma, D. Luan, C. Sutton, and A. Odena. Show your work: Scratchpads for intermediate computation with language models. *ArXiv*, abs/2112.00114, 2021.

[113] F. J. Och and H. Ney. Improved statistical alignment models. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, pages 440–447, Hong Kong, Oct. 2000. Association for Computational Linguistics.

[114] L. Orr, M. Leszczynski, S. Arora, S. Wu, N. Guha, X. Ling, and C. Ré. Bootleg: Chasing the tail with self-supervised named entity disambiguation. *ArXiv*, abs/2010.10363, 2020.

[115] S. Ostermann, A. Modi, M. Roth, S. Thater, and M. Pinkal. Mcscript: A novel dataset for assessing machine comprehension using script knowledge. In *LREC*, 2018.

[116] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. Bleu: a method for automatic evaluation of machine translation. In *ACL*, 2001.

[117] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala. Pytorch: An imperative style, high-performance deep learning library. *ArXiv*, abs/1912.01703, 2019.

[118] F. Petroni, A. Piktus, A. Fan, P. Lewis, M. Yazdani, N. D. Cao, J. Thorne, Y. Jernite, V. Plachouras, T. Rocktaschel, and S. Riedel. Kilt: a benchmark for knowledge intensive language tasks. In *NAACL-HLT*, 2020.

[119] F. Petroni, T. Rocktäschel, S. Riedel, P. Lewis, A. Bakhtin, Y. Wu, and A. Miller. Language models as knowledge bases? 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 2463–2473, Hong Kong, China, 2019. Association for Computational Linguistics.

[120] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, and I. Sutskever. Language models are unsupervised multitask learners, 2019.

[121] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv e-prints*, 2019.

[122] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21:1–67, 2020.

[123] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.

[124] C. Raffel, N. M. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *JMLR*, 21, 2020.

[125] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. Squad: 100, 000+ questions for machine comprehension of text. In *EMNLP*, 2016.

[126] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 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, Nov. 2016. Association for Computational Linguistics.

[127] J. Ramos. Using tf-idf to determine word relevance in document queries, 1999.

[128] D. Rao, P. McNamee, and M. Dredze. Entity linking: Finding extracted entities in a knowledge base. In *Multi-source, Multilingual Information Extraction and Summarization*, 2013.

[129] H. Rashkin, V. Nikolaev, M. Lamm, L. Aroyo, M. Collins, D. Das, S. Petrov, G. S. Tomar, I. Turc, and D. Reitter. Measuring attribution in natural language generation models. *ArXiv*, abs/2112.12870, 2021.

[130] H. Rashkin, V. Nikolaev, M. Lamm, M. Collins, D. Das, S. Petrov, G. S. Tomar, I. Turc, and D. Reitter. Measuring attribution in natural language generation models. *ArXiv*, abs/2112.12870, 2021.

[131] V. Raunak, A. Menezes, and M. Junczys-Dowmunt. The curious case of hallucinations in neural machine translation. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1172–1183, Online, June 2021. Association for Computational Linguistics.

[132] M. Richardson, C. J. C. Burges, and E. Renshaw. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, 2013.

[133] A. Roberts, C. Raffel, and N. Shazeer. How much knowledge can you pack into the parameters of a language model? In *EMNLP*, 2020.

[134] A. Roberts, C. Raffel, and N. M. Shazeer. How much knowledge can you pack into the parameters of a language model? In *EMNLP*, 2020.

[135] S. E. Robertson, C. J. van Rijsbergen, and M. F. Porter. Probabilistic models of indexing and searching. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1980.

[136] S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. In *Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 1994.

[137] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja, M. Dey, M. S. Bari, C. Xu, U. Thakker, S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. V. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Févry, J. A. Fries, R. Teehan, S. R. Biderman, L. Gao, T. Bers, T. Wolf, and A. M. Rush. Multitask prompted training enables zero-shot task generalization. *ArXiv*, abs/2110.08207, 2021.

[138] M. Sap, H. Rashkin, D. Chen, R. L. Bras, and Y. Choi. Social iqa: Commonsense reasoning about social interactions. In *EMNLP*, 2019.

[139] T. Schick, J. Dwivedi-Yu, Z. Jiang, F. Petroni, P. Lewis, G. Izacard, Q. You, C. Nalmpantis, E. Grave, and S. Riedel. Peer: A collaborative language model. *ArXiv*, abs/2208.11663, 2022.

[140] T. Schuster, A. Fisch, and R. Barzilay. 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, 2021. Association for Computational Linguistics.

[141] T. Schuster, A. Fisch, and R. Barzilay. Get your vitamin C! robust fact verification with contrastive evidence. In *NAACL*, 2021.

[142] T. Sellam, D. Das, and A. P. Parikh. Bleurt: Learning robust metrics for text generation. In *ACL*, 2020.

[143] R. Sennrich, B. Haddow, and A. Birch. Improving neural machine translation models with monolingual data. *ArXiv*, abs/1511.06709, 2016.

[144] O. Sevgili, A. Shelmanov, M. V. Arkhipov, A. Panchenko, and C. Biemann. Neural entity linking: A survey of models based on deep learning. *ArXiv*, abs/2006.00575, 2020.

[145] R. Shin, C. H. Lin, S. Thomson, C. Chen, S. Roy, E. A. Platanios, A. Pauls, D. Klein, J. Eisner, and B. Van Durme. Constrained language models yield few-shot semantic parsers. In *EMNLP*, 2021.

[146] V. Shwartz, R. Rudinger, and O. Tafjord. "you are grounded!": Latent name artifacts in pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6850–6861, Online, Nov. 2020. Association for Computational Linguistics.

[147] C. Si, Z. Gan, Z. Yang, S. Wang, J. Wang, J. L. Boyd-Graber, and L. Wang. Prompting gpt-3 to be reliable. *ArXiv*, abs/2210.09150, 2022.

[148] S. Stanton, P. Izmailov, P. Kirichenko, A. A. Alemi, and A. G. Wilson. Does knowledge distillation really work? *ArXiv*, abs/2106.05945, 2021.

[149] A. Talmor, J. Herzig, N. Lourie, and J. Berant. Commonsenseqa: A question answering challenge targeting commonsense knowledge. In *NAACL-HLT*, 2018.

[150] R. Thoppilan et al. LaMDA: Language models for dialog applications. *ArXiv*, abs/2201.08239, 2022.

[151] J. Thorne and A. Vlachos. Evidence-based factual error correction. In *Annual Meeting of the Association for Computational Linguistics*, 2020.

[152] J. Thorne and A. Vlachos. Evidence-based factual error correction. In *ACL*, 2021.

[153] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal. Fever: a large-scale dataset for fact extraction and verification. In *NAACL-HLT*, 2018.

[154] J. Thorne, A. Vlachos, C. Christodoulopoulos, and A. Mittal. FEVER: a large-scale dataset for fact extraction and verification. In *NAACL*, 2018.

[155] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, A. Rodriguez, A. Joulin, E. Grave, and G. Lample. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971, 2023.

[156] A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada, 2017. Association for Computational Linguistics.

[157] A. Vaswani, N. M. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin. Attention is all you need. In *NIPS*, 2017.

[158] D. Vrandecic and M. Krötzsch. Wikidata: a free collaborative knowledgebase. *Commun. ACM*, 57:78–85, 2014.

[159] A. Wang, K. Cho, and M. Lewis. Asking and answering questions to evaluate the factual consistency of summaries. In *ACL*, 2020.

[160] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In *Neural Information Processing Systems*, 2019.

[161] J. Wei, M. Bosma, V. Zhao, K. Guu, A. W. Yu, B. Lester, N. Du, A. M. Dai, and Q. V. Le. Finetuned language models are zero-shot learners. *ArXiv*, abs/2109.01652, 2021.

[162] J. Wei, Y. Tay, R. Bommasani, C. Raffel, B. Zoph, S. Borgeaud, D. Yogatama, M. Bosma, D. Zhou, D. Metzler, E. H. hsin Chi, T. Hashimoto, O. Vinyals, P. Liang, J. Dean, and W. Fedus. Emergent abilities of large language models. *Trans. Mach. Learn. Res.*, 2022, 2022.

[163] J. Wei, X. Wang, D. Schuurmans, M. Bosma, E. Chi, Q. Le, and D. Zhou. Chain of thought prompting elicits reasoning in large language models. *ArXiv*, abs/2201.11903, 2022.

[164] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, and J. Brew. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv*, abs/1910.03771, 2019.

[165] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T. Le Scao, S. Gugger, M. Drame, Q. Lhoest, and A. Rush. 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, Oct. 2020. Association for Computational Linguistics.

[166] L. Y. Wu, F. Petroni, M. Josifoski, S. Riedel, and L. Zettlemoyer. Zero-shot entity linking with dense entity retrieval. In *EMNLP*, 2020.

[167] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert. In *ICLR*, 2019.

[168] W. Zhou, S. Zhang, H. Poon, and M. Chen. Context-faithful prompting for large language models. *ArXiv*, abs/2303.11315, 2023.