EVALUATING COMMON-SENSE REASONING IN PRETRAINED TRANSFORMER-BASED LANGUAGE MODELS USING ADVERSARIAL SCHEMAS AND CONSISTENCY METRICS

by

Adrian Maler

Submitted in partial fulfillment of the requirements
for the degree of Master of Computer Science

at

Dalhousie University
Halifax, Nova Scotia
April 2023

Table of Contents

List of Tables

List of Figures

Abstract

In artificial intelligence, common sense refers to simple acts of verbal reasoning. The Winograd Schema Challenge (WSC), an important test of common sense, was recently defeated by transformer-based language models. We investigate the implications of that defeat: have language models achieved common sense, or is the challenge flawed? That is, we consider the problem of reevaluating verbal reasoning in language models. We evaluate the accuracy and consistency on Winograd schemas of three important pretrained models: GPT-2, RoBERTa, and T5. We generalize the Winograd schema to a larger class of problems, called adversarial schemas, and propose an evaluation protocol for them that incorporates consistency. We create a new test of common-sense verbal reasoning made up of our adversarial schemas. Each model performs significantly worse on our test than on WSC, and no model exhibits high consistency. We find no convincing evidence of verbal reasoning by language models.

Acknowledgements

This thesis was made possible by the direction of my supervisor, Vlado Keselj. I would also like to thank Darren Abramson, Dirk Arnold, and Sageev Oore.

The experiments reported here were run in a computing environment provided by the Digital Research Alliance of Canada (formerly Compute Canada).

This document was prepared in TeXShop. The graphs were made in MATLAB and the other figures in Pages. The programs were written in Ruby and Python.

Chapter 1

Introduction

In this chapter, we introduce and motivate our research problem, describe our main objectives and contributions, and provide an outline of the chapters to follow.

In Section 1.1, we motivate our research problem with a brief review of selected topics in natural language processing, machine learning, and artificial intelligence, particularly language models for common-sense reasoning, culminating in the defeat of the Winograd Schema Challenge, a common-sense reasoning test, by pretrained transformer-based neural network language models. In Section 1.2, we describe our research problem in light of that background, including a summary of our objectives and contributions. In Section 1.3, we outline the rest of the thesis.

1.1 Motivation

The purpose of this section is to motivate our research problem with a brief, almost minimal review of relevant topics, assuming no familiarity on the part of the reader. Those topics and their dependencies are shown in Figure 1.1.

Chapter 2 does provide more information on some of these topics, particularly the Winograd Schema Challenge, but the interested reader should consult the literature for general information on natural language processing, machine learning, and neural networks. We will suggest some easily accessible sources.

1.1.1 Natural Language Processing

In computer science, the field of *natural language processing* (NLP) is concerned with computer processing of natural language data, typically consisting of written or spoken utterances by human beings. Applications of NLP include spell checking, text-to-speech, machine translation, information retrieval, question answering, and common-sense reasoning, the last of which is discussed in Section 1.1.3.

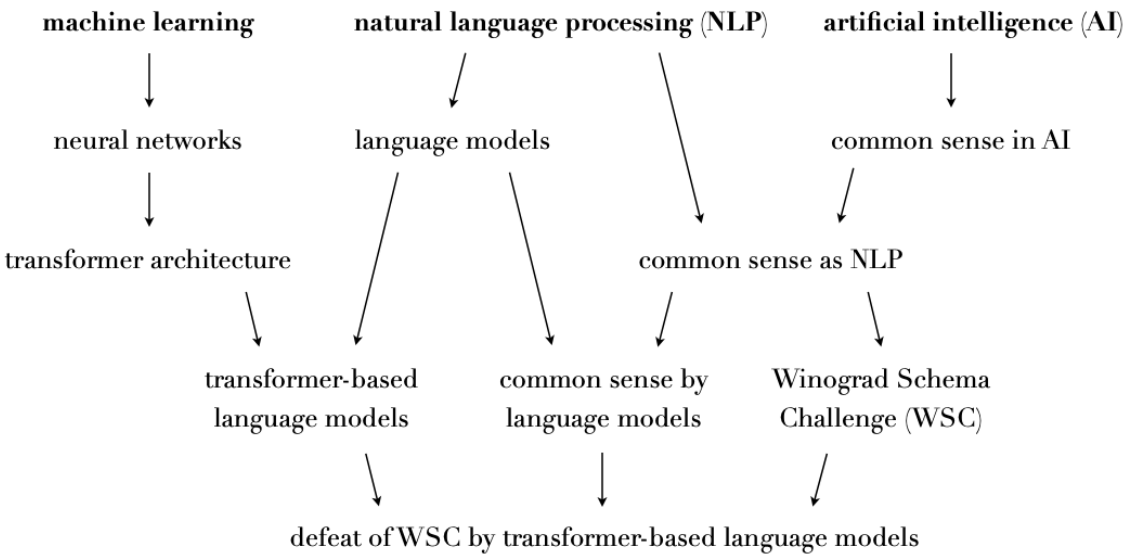

Figure 1.1: Background topics covered in Section 1.1.

For an introduction to NLP as of 2008, the reader may consult Jurafsky and Martin [38], although the field has made rapid advances since that time in specific areas relevant to our research. For example, transformer-based language models, which we will discuss in Section 1.1.2, were introduced as recently as 2017.

In natural language processing, a statistical language model, or *language model* for short, is a way of assigning a numerical score to a finite sequence of tokens from some finite alphabet. Typically, the tokens represent words or parts of words, along with punctuation symbols, drawn from a natural-language corpus. The score assigned to a sequence of tokens is typically taken to represent the "probability" of that sequence of words and punctuation occurring, although no such probability is defined: for one thing, there is no sample space. We can say that a better-scoring sequence is preferred by the model as more likely or plausible, based on the corpus.

Some language models can assign a score to part of a sequence conditional on another part. One important example is scoring the last token in a sequence, say σ_{n+1}, given the preceding tokens, say $\sigma_1 \cdots \sigma_n$. This may be used to predict the token most likely to follow or complete a given prefix. Hence, conditional scoring can be used to perform sentence completion and other fill-in-the-blank tasks.

1.1.2 Machine Learning

A large class of problems in science, engineering, and other quantitative fields can be interpreted, at a high level of abstraction, as the construction of a mathematical model capable of predicting or explaining the behaviour of a complex system. That system is known to us deductively, through general laws that govern many systems, and inductively, through particular sets of data collected from the system in question.

Machine learning may be defined as the study and use of algorithms that either build a model from a set of data or incorporate new data into an existing model with the goal of improving that model's predictive performance or explanatory value. Although the field is relatively new, its literature is vast and rapidly expanding; for a general introduction, the reader may consult Hastie et al. [29] or Murphy [65].

An artificial neural network, or *neural network* for short, is a type of model often used in machine learning. The model is made up of processing units called *neurons* that are connected to one another. The process of improving a neural network's predictive performance on a dataset is called *training* the network on the data. For a general introduction to neural networks, the reader may consult Zhang et al. [107].

The *encoder-decoder* architecture was introduced in 2014 [7, 91]. It was designed for neural networks that process sequential data, where the input is in the form of a sequence, and in particular for sequence-to-sequence modelling, where the output is also in the form of a sequence. Sentence completion is an example of sequence-to-sequence modelling. It is possible to build encoder-only and decoder-only models.

The *transformer* architecture was introduced in 2017 [100]. It optimizes the encoder-decoder architecture by adding an attention mechanism that can selectively focus on particular portions of the input sequence.

Some neural networks are language models (Section 1.1.1). A neural network language model is said to be *pretrained* if it has been trained on a large natural-language corpus, such as English-language Wikipedia articles, to perform a general NLP task, such as sentence completion. In practice, such a model is often *finetuned* later, i.e., trained on other data to perform a more specific *downstream* task.

This thesis investigates the performance of specific pretrained transformer-based neural network language models: GPT-2 [76], RoBERTa [57], and T5 [79]. Each of those models, at some point, achieved state-of-the-art performance on the Winograd

Schema Challenge, a verbal test of common-sense knowledge and reasoning.

1.1.3 Common-Sense Reasoning

Whatever its ordinary meaning, *common sense* is a term of art in the field of artificial intelligence (AI), where it often appears in the form of *common-sense reasoning*, a way of figuring things out, and *common-sense knowledge*, a catalogue of facts. For the purposes of this thesis, all three terms are interchangeable.

As early as 1959, John McCarthy, one of the founders of AI, proposed a project of simulating "certain elementary verbal reasoning processes so simple that they can be carried out by any non-feeble-minded human," in order to produce a "program for solving problems by manipulating sentences in formal languages" [60]. He described the goal of his project as "programs with common sense" (the title of the paper).

So, right from the start of the field, common sense was identified with acts of verbal reasoning: manipulating expressions in a natural or formal language. Natural languages were for human beings, formal languages for machines. The goal was either to simulate the processes by which human beings actually perform those acts of verbal reasoning, or at least to reproduce the end result of those processes by outputting a correct solution to some problem or a valid consequence of some premises.

For McCarthy, at least in 1959, programs with common sense would read and write in formal languages, not natural ones.[1] However, as early as 1970, Terry Winograd recognized important common-sense aspects of natural language processing [103]:

> If we really want computers to understand us, ... they need to have all sorts of knowledge about the subject they are discussing, and they have to use reasoning to combine facts in the right way....

Winograd included one particularly influential example of a pair of interpretation problems where "a knowledge of the meanings of words is not enough." Here, we have annotated a substantially equivalent example from his 1972 follow-up paper [104],

[1] As McCarthy [61] wrote thirty years later: "If a computer is to store facts about the world and reason with them, it needs a precise language.... Therefore, it was natural to try to use mathematical logical languages to express what an intelligent computer program knows...."

because this version is more often cited in the literature:

a. The city councilmen refused the demonstrators a permit because **they** {feared violence}.
b. The city councilmen refused the demonstrators a permit because **they** {advocated revolution}. (1.1)

There are several notable features. First, the sentences differ only by a key phrase (in braces). Second, each sentence includes a grammatically ambiguous pronoun (in bold). Finally, in order to determine which of two referents (underlined) is more likely, which depends on the choice of key phrase, a program "has to have more than the meanings of words. It has to have... information and reasoning power" [103].

That example inspired a test of common-sense knowledge and reasoning.

1.1.4 The Winograd Schema Challenge

In 2011, Hector J. Levesque introduced the *Winograd Schema Challenge* (WSC) based on Winograd's example (1.1) as a test of common-sense reasoning and an alternative to the famous Turing test of artificial intelligence [52]. In the following, we will refer to the revised and expanded 2012 version of that paper: Levesque et al. [53].

The form of the challenge is a multiple-choice test of verbal reasoning in the English language. To pass the test means to achieve "human" or at least "near human" accuracy. At the time, this was "far beyond the current state of the art."

By design, WSC "appeals to world knowledge and default reasoning abilities," i.e., common sense as defined in Section 1.1.3. "In order to pass the [Winograd Schema] Challenge, a system will need to have commonsense knowledge about space, time, physical reasoning, emotions, social constructs, and a wide variety of other domains," as well as "procedures to reason with that knowledge" or bring it to bear.

After its introduction, "the challenge became a focal point of research for both the commonsense reasoning and natural language processing communities" [46].

A dataset for WSC, or *Winograd dataset*, is a specific set of *Winograd schemas* that define the test questions [53]. Each schema includes a sentence with a placeholder and an ambiguous pronoun. For example, with our annotations:

The trophy doesn't fit into the suitcase because **it** is too {[blank]}. (1.2)

A schema also provides two *key words* that can replace the placeholder, yielding alternative sentences. In this example, the key words "large" and "small" yield:

a. The trophy doesn't fit into the suitcase because **it** is too {large}.
b. The trophy doesn't fit into the suitcase because **it** is too {small}. (1.3)

The interpretation of the pronoun must depend on the choice of key word: in this example, the first key word, "large," makes "the trophy" overwhelmingly more likely as the referent, and the second key word, "small," similarly favours "the suitcase."

The point is that each of the schema's alternative sentences, which differ only by a key word, presents a problem to solve: to identify the more likely referent of the ambiguous pronoun, given two answer options that appear in the sentence.

a. The trophy doesn't fit into the suitcase because it is too large.
   What is too large? (i) the trophy ✓ (ii) the suitcase
b. The trophy doesn't fit into the suitcase because it is too small.
   What is too small? (i) the trophy (ii) the suitcase ✓ (1.4)

There are several important Winograd datasets in the literature, of which the most important for our purposes is *WSC273* from 2013 [17]. Due to its popularity and precedence, WSC273 is often referred to simply as "the Winograd Schema Challenge."

1.1.5 Defeat of the Winograd Schema Challenge

In 2018, Trinh and Le [97, 98] described "a simple method" for implementing commonsense reasoning by language models that we call *stochastic fill-in-the-blank*.

Suppose we are given the Winograd problem (1.4a). We need to determine the more likely intended referent of the pronoun "it" from among "the trophy" and "the suitcase." We can try substituting each answer option for the ambiguous pronoun:

The trophy doesn't fit into the suitcase because *the trophy* is too large.
The trophy doesn't fit into the suitcase because *the suitcase* is too large. (1.5)

Having made the substitutions, we use a language model to assign each of the above sentences a score representing its relative likelihood. Whichever candidate yields a better-scoring sentence is taken to be the correct answer to the problem.

The method of stochastic fill-in-the-blank uses a language model as an oracle or black box, and the accuracy of the method depends on the choice of language model. By 2021, the reported accuracy of the best-performing finetuned transformer-based language models, if not also the pretrained ones, was sufficiently close to the untrained human baseline on the most popular Winograd datasets that Kocijan et al. [47] could announce "the defeat of the Winograd Schema Challenge" itself.

1.2 Research Problem and Objectives

The purpose of our research is to investigate the implications of the reported defeat of the Winograd Schema Challenge. To be precise, if a model achieves human-like accuracy on WSC, there are two possibilities. The first is that the model has achieved common sense, because it passes the challenge. The second is that the challenge is flawed, because the model passes it without having achieved common sense.

The problem is to decide experimentally which of those possibilities is more likely. We identified two approaches, both of which are supported by the literature, and both of which amount to reevaluating the common-sense reasoning capabilities of notable high-accuracy language models. First, we apply to the models a new *evaluation protocol* that is more probing than a simple accuracy score. Second, we apply the models to a new test of common-sense verbal reasoning that differs from WSC.

For the first approach, we acknowledge that a number of language models have achieved reasonably high accuracy on WSC273 and other popular datasets. However, reason is general and generalizable, so a model with the ability to reason should exhibit high *consistency* as well: it should answer similar questions in similar ways.

Other authors have proposed evaluation protocols featuring a consistency metric with respect to various inessential alterations or *perturbations* of the dataset. We extended their work by systematically perturbing WSC273 in several new ways and measuring the consistency of three high-accuracy models. In the process, we cleaned up the dataset to create what we call *WSC266*. Due to its standardized format, WSC266 is considerably easier to systematically perturb than the original dataset.

We also introduce new scoring methods, meaning specific implementations of stochastic fill-in-the-blank, some of which are more accurate than published methods.

For the second approach, it is clear that the Winograd Schema Challenge does

not and cannot test every aspect of common-sense verbal reasoning or even every aspect of pronoun disambiguation by common-sense verbal reasoning in the English language. Again, reason is general and generalizable, so a model with the ability to reason should exhibit consistency across challenges as well as within them.

Other authors have proposed alternatives to WSC. We created our own test of common-sense verbal reasoning in the English language, which we call *Reid250*. In the process, we generalized the Winograd schema to what we call the *adversarial schema*, which preserves the key word feature that, in our view, is essential to WSC. The new schema is easier to create, it can test a broader range of common-sense content, and it is equally amenable to solution by stochastic fill-in-the-blank.

Because both of our approaches involve reevaluating the performance of language models, we also developed a new evaluation protocol for Winograd schemas and for adversarial schemas generally, incorporating, from other authors, not only consistency but also a special *control baseline* and the effects of the model's *tokenization*.

Our objectives, collected from both of our approaches to the problem, were as follows: (1a) to clean up the dataset WSC273; (1b) to systematically perturb the clean dataset; (1c) to reevaluate three language models on the clean dataset using a consistency-based evaluation protocol; (2a) to generalize the Winograd schema as a family of problems in natural language processing; (2b) to create a new test of common-sense reasoning out of our new schemas; and (2c) to evaluate the same models on the new dataset, again using a consistency-based evaluation protocol.

Our contributions are a cleaner Winograd dataset in a standardized format; two perturbations of that dataset suitable for measuring consistency; a generalization of the Winograd schema; a new test of common-sense reasoning; performance metrics and evaluation protocols for Winograd and adversarial schemas; and new results on common-sense reasoning in notable pretrained transformer-based language models, including the best scoring methods or implementations of stochastic fill-in-the-blank.

1.3 Outline

For the reader's convenience, we include a chapter-by-chapter outline of the rest of the thesis. Each chapter begins with a section-by-section outline.

Here in Chapter 1, we have introduced and motivated our research problem and

described our main objectives and contributions. In Chapter 2, we review important background material and related work. In Chapter 3, we explain our methodology: how we approached the research problem. In Chapter 4, we describe our experiments and present our results. In Chapter 5, we review our results and contributions and discuss the limitations, generalizations, and possible future directions of our research.

Chapter 2

Background and Related Work

In Chapter 1, we introduced and motivated our research problem and described our main objectives and contributions. For context, we included a brief review of selected topics in natural language processing, machine learning, and artificial intelligence.

In this chapter, we review important background material and related work in more detail, expanding on some of the topics introduced in Chapter 1.

Section 2.1 discusses pretrained transformer-based language models in general. Section 2.2 describes the three models we studied: GPT-2, RoBERTa, and T5.

Section 2.3 is an overview of the Winograd Schema Challenge (WSC). Section 2.4 lists the basic properties of a Winograd schema, and Section 2.5 adds an additional, adversarial constraint on those schemas, with discussion.

Section 2.6 considers WSC in the context of natural language processing, and Section 2.7 describes important testing and training datasets for WSC.

Section 2.8 explains how WSC has been approached and defeated using language models. Section 2.9 discusses evaluation protocols for language models on WSC.

2.1 Pretrained Transformer-Based Language Models

We defined *natural language processing* (NLP) and *language models* in Section 1.1.1. This section expands on material introduced in Section 1.1.2, on machine learning.

2.1.1 Neural Networks

We defined *machine learning* as the study and use of algorithms that either build a model from a set of data or incorporate new data into an existing model with the goal of improving that model's predictive performance or explanatory value. If each data point is labelled with the corresponding behaviour of the system, which is the intended prediction of the model, then we are conducting *supervised learning*.

Machine learning in general often makes use of numerical optimization methods. In particular, supervised learning can be seen as a process of iterated optimization: we define a *loss function* to measure how much the outputs of the model differ from the given labels, then improve the model by minimizing that objective function.

A *neural network* is made up of processing units called *neurons* that are connected to one another. Typically, a neuron receives input from a set of neurons, performs a calculation on its inputs, and sends the resulting value as output to another set of neurons. Special *input neurons* instead receive one feature each from the data and output it unchanged. Designated *output neurons* report their output as a predicted label, and together make up the model's output.

Often, the number of neurons in a network, their positions, and how they connect to one another are fixed. What changes when we improve the model through machine learning is the set of parameters that the neurons use to calculate their outputs. Typically, a neuron calculates the weighted sum of its inputs, subtracts a term called a bias, and finally applies an *activation function* (e.g., tanh). Hence, the parameters of the model are the *weights* of the connections and the *biases* of the neurons.

Some neural networks, including the language models we studied (Section 2.2), have hundreds of millions or even billions of parameters.

The process of improving a neural network's predictive performance by changing its parameters so as to minimize a loss function over some dataset is called *training* the network on the data; it is a special case of supervised learning.

The neurons in a network are often arranged in a series of *layers*, which helps to describe and control how signals propagate through the network. Between the *input layer* that takes the model's input and the *output layer* that gives the model's output may be one or more *hidden layers* whose values are not reported to the user.

In a *feed-forward* neural network, a neuron only ever sends its output to neurons in higher layers, so there are no cycles; a *deep learning network* is a feed-forward neural network with more than one hidden layer. In general, a neural network with cycles is called *recurrent*. However, in the context of NLP, the term *recurrent neural network* is often used to mean one particular type of neural network with cycles that is used for processing sequential data, a topic to which we now turn.

2.1.2 Transformer Architecture

The *encoder-decoder* architecture for deep learning networks was introduced in 2014 and can be attributed to both Sutskever et al. [91] and Cho et al. [7], both of whose work is closely related to Kalchbrenner and Blunsom [39]. The architecture was designed for processing sequential data, and in particular for sequence-to-sequence modelling; for example, predicting the words that follow an incomplete sentence.

Suppose we have a sequence of items $\sigma_1 \cdots \sigma_n$. An *encoder* is a type of neural network that performs a calculation on σ_1 (the first item) to produce an output-in-progress τ_1, then performs a calculation on τ_1 and σ_2 (the next item) to produce an updated output-in-progress τ_2, and so on, until it has processed the entire sequence.

Because of this iterative process, where the output of one iteration becomes the input of the next iteration, encoders are recurrent neural networks; indeed, in some sources, the term *recurrent neural network* is synonymous with this architecture.

Once the encoder has processed the entire input sequence, the complete state of the encoder network, i.e., the current value of each neuron (and not just the output neurons), is passed as input to the *decoder*, which is another neural network. The decoder network is trained to output the next item in the sequence, given this state.

The *transformer* architecture due to Vaswani et al. [100] optimizes the encoder-decoder architecture: instead of passing the complete state of the encoder network to the decoder, we selectively pay attention to different parts of the input sequence.

It is possible to build encoder-only and decoder-only transformer models. Decoder-only models, including GPT-2, are sometimes called *auto-regressive*. Encoder-only models, including RoBERTa and other BERT-based models, are sometimes called *auto-encoding*. Finally, encoder-decoder models, including T5, are sometimes called *sequence-to-sequence*. Those three specific models will be discussed in Section 2.2.

2.1.3 Pretraining, Finetuning, and Task Demonstrations

A neural network language model is said to be *pretrained* if it has been trained on a large natural-language corpus, such as English-language Wikipedia articles, to perform a general NLP task, such as next word prediction or sentence completion.

Often, the corpus or pretraining dataset effectively labels itself: in next word prediction, for example, a partial sequence of words from the corpus can be labelled

automatically with the next word in the sequence. Such a dataset is often called *unsupervised*, and the training process, which is always a case of supervised learning, is often called *unsupervised pretraining* [5, 20, 74, 76, 79, etc.]. To avoid any ambiguity, we will refer to such a dataset or training process as *self-supervised*.

A pretrained model is often further trained on another dataset for a more specific application, such as solving Winograd problems. The follow-up training process is called *finetuning*, and the more specific application is said to be a *downstream* task. The two-stage process of pretraining and finetuning is called *transfer learning* [79].

When we actually run a neural network language model, finetuned or not, on a dataset, we can enhance each data point with one or more demonstrations of the task to be performed [5]. For example, we can prepend a Winograd problem, which is just a string, with a different Winograd problem and its label, the correct answer. This is called *few-shot learning*; with only one demonstration, it is called *one-shot learning*. We can also enhance each data point, not with specific demonstrations, but with a general description of the task in natural language; this is called *zero-shot learning*. We will refer to the use of unenhanced data as *demonstration-free learning*.

The method of stochastic fill-in-the-blank (Section 1.1.5) does not call for the enhancement of the input by any demonstration or description of the task. For that reason, we restrict our attention to demonstration-free learning, with one partial exception: for the T5 model, we consider enhancing the input by prepending a special Winograd task prefix, because the model was trained with one (Section 2.2.3).

Unless otherwise noted, all cited results use demonstration-free learning.

2.2 Notable Transformer Models

Table 2.1 shows some notable pretrained transformer-based language models, all of which are publicly available except for GPT-3.[1] Where applicable, we report the date of the earliest online draft, which may not be the paper's final publication date.

We will use GPT-2, RoBERTa, and T5 in our experiments in Chapter 4, so we include a brief discussion of each model from the standpoint of a user.

Apart from being pretrained transformer-based neural network language models,

[1] We have not included the newly released GPT-4 model because no details are available on it, except that it is a pretrained "Transformer-based" or "Transformer-style" model [70].

Model	Source	Date	Architecture
GPT	Radford et al. [74]	2018-06	decoder only
BERT	Devlin et al. [19, 20]	2018-10	encoder only
GPT-2	Radford et al. [76]	2019-02	decoder only
RoBERTa	Liu et al. [57]	2019-07	encoder only
T5	Raffel et al. [78, 79]	2019-10	encoder-decoder
GPT-3	Brown et al. [5, 6]	2020-05	decoder only

Table 2.1: Notable pretrained transformer-based language models.

the models in question have some points in common. First, they come in different sizes. In each case, we used the largest model, which achieves the best performance using the most parameters: GPT-2, size *xl*, 1.6 billion parameters; RoBERTa, size *large*, 355 million parameters; and T5, size *11b*, 11 billion parameters.

Second, each model deals, not with words from the dictionary, but with short strings called *tokens*: it has a fixed vocabulary, which is a set of those strings. The model splits a given text into a sequence of tokens before processing it further; this is called *tokenizing* the string, and it is sensitive to whitespace and capitalization.

Third, each model has been applied to the Winograd Schema Challenge, which we discussed in Section 1.1.4. Indeed, each model, at some time, achieved state-of-the-art accuracy on the challenge; we will discuss this point again in Section 2.8.

2.2.1 GPT-2

The first model we consider is *GPT-2* from February 2019 [76], an updated version of *GPT* from June 2018 [74]. According to its creators, "GPT-2 is trained with a simple objective: predict the next word, given all of the previous words within some text" [75]. Evidently, this is a case of self-supervised training.

As we mentioned above, the model actually deals with tokens, not words, with a fixed vocabulary $V = \{t_m : 0 \leq m < M\}$ of $M = 50257$ tokens [32].

The training dataset is called *WebText*. The authors describe it as "a new web scrape" that "emphasizes document quality," excludes content from Wikipedia, and consists of "slightly over 8 million documents for a total of 40 GB of text" [76]. As we said, the model was trained on the task of predicting the next token in a text.

As input, the model takes a sequence $\sigma = \sigma_1\sigma_2\cdots\sigma_n$ of $n > 1$ tokens from the

vocabulary. It returns a table of values

$$\{p_{i,m} : 1 \leq i < n,\ 0 \leq m < M\}, \tag{2.1}$$

where $p_{i,m}$ is taken to represent the probability[2] of observing the token t_m in position $i+1$ given that we observe the tokens $\sigma_1 \cdots \sigma_i$ in the previous positions.

More precisely, the model returns the log-odds or *logit* of that probability:

$$\operatorname{logit}(p) = \log\left(\frac{p}{1-p}\right), \qquad 0 < p < 1. \tag{2.2}$$

It is easy to calculate the corresponding probabilities from a table of logits.

In particular, for $i = 1, \ldots, n-1$, we get the probability p_i^* of observing σ_{i+1} given $\sigma_1 \cdots \sigma_i$. To obtain the loss reported by the model on input σ, i.e., the value of the objective function to be minimized, we take the mean of $\log p_i^*$ over all predicted tokens, $i = 1, \ldots, n-1$, then multiply by -1, so a higher probability produces a lower (positive) loss. As the authors put it, "results on language modelling datasets are commonly reported in a quantity which is a scaled or exponentiated version of the average negative log probability per canonical prediction unit" [76].

More generally, by chaining together conditional probabilities, we can calculate, e.g., the probability of observing $\sigma_6\sigma_7\sigma_8$ after $\sigma_1 \cdots \sigma_5$. However, the model does not tell us the probability of any one token, say σ_1, occurring on its own.

We can say more about the vocabulary. It was extracted from the training dataset through a process called *byte pair encoding* [89]. The result is neither minimal nor prefix-free: "t" is a token, and "h" is as well, but "th" and "ht" are also tokens. It is case-sensitive: "Th" and "TH" are included as separate tokens. It is not a complete list of strings up to a certain length: "tH" and "Ht" are not included.

The vocabulary is notably sensitive to preceding whitespace: "␣t" (with a space) is a token, for example, as are "␣th," "␣the," "␣The," and many more. However, "␣ht" is not a token, so the vocabulary is not closed under prepending a space. About 33000 tokens start with a space, out of a total of about 50000 tokens: well over half. The idea is that, for example, "The" at the start of a document, with no preceding space, is treated differently by the model from "␣The" in the middle of a document.

The last token is a special end-of-text token to mark the end of a document.

[2]Strictly speaking, as we said in Section 1.1.1, no such probability is defined.

2.2.2 RoBERTa

The next model is *RoBERTa* from July 2019 [57], which uses the same architecture as the *BERT* model from October 2018 [19, 20], but optimizes the pretraining process.

We will begin with BERT, which was trained on two tasks: *masked language modelling* and *next segment prediction*, both of which are self-supervised.

For masked language modelling, we take a document from the training dataset, replace some of the tokens at random with a special *masking token* not found in the dataset, and train the model to predict the original tokens. Unlike next token prediction, which is the training objective of GPT-2 (Section 2.2.1) and proceeds through a text from left to right, masked language modelling is "bidirectional," as it looks for "context" on both sides of a masked token within a text.

Next segment prediction is a very different task. Given a segment of text from a document, plus another segment of text, the objective is to predict whether the second segment follows the first segment or was taken from another document.[3]

RoBERTa optimizes the BERT pretraining process: it was still trained on masked language modelling, but the authors removed next segment prediction as an objective. Simply put, this choice "slightly improves downstream task performance" [57].

RoBERTa has a fixed vocabulary $V = \{t_m : 0 \leq m < M\}$ of $M = 50265$ tokens [33], which is very similar to the vocabulary of GPT-2.

The training dataset is made up of several smaller datasets, including "an open-source recreation of the WebText corpus" on which GPT-2 was trained [57].

The model takes a sequence $\sigma = \sigma_1\sigma_2\cdots\sigma_n$ of $n > 1$ tokens, where the masking token occurs $N \geq 1$ times at indices $i_1 < \cdots < i_N$. It returns a table of values

$$\{p_{k,m} : 1 \leq k \leq N,\ 0 \leq m < M\}, \tag{2.3}$$

where $p_{k,m}$ is taken to represent the probability of observing the token t_m in the masked position i_k given that we observe the other tokens, σ_i, $i \neq i_k$, in their positions. Note that the other masked tokens, σ_{i_j}, $j \neq k$, are still masked.

In particular, suppose we have candidate tokens $\tau_1, \ldots, \tau_N$ to fill in the masked positions. For $k = 1, \ldots, N$, we get the probability p_k° of observing τ_k in position i_k

[3] In the literature, next segment prediction is often called "next sentence prediction," although, e.g., in BERT pretraining the segments "can each contain multiple natural sentences" [57].

given σ_i, $i \neq i_k$. To obtain the loss reported by the model, we take the mean of $\log p^\circ_k$ over all candidate tokens, $i = 1, \ldots, N$, then multiply by -1.

Before tokenizing any string, RoBERTa places it between a special beginning-of-string token and a special end-of-string token.

Again, the vocabulary was extracted from the training dataset through byte pair encoding. The result is neither minimal nor prefix-free. It is case-sensitive. It is not a complete list of strings up to a certain length. It is sensitive to preceding whitespace, which is represented by a special, more visible character; it is not closed under prepending a space. Well over half the tokens start with a space. Apart from the use of a replacement whitespace character, all of this is consistent with GPT-2.

RoBERTa's vocabulary includes numerous special tokens. Most importantly, t_{50264} is the special masking token used in evaluation as described above.

2.2.3 T5

The final model is *T5* from May 2020 [78, 79]. Its creators take a "unified approach to transfer learning" by training it on a mixture of tasks, each of which is cast as a "text-to-text" problem, i.e., "taking text as input and producing new text as output."

The model's self-supervised training includes masked language modelling similar to RoBERTa. Its supervised training tasks include sentence completion, question answering, natural language inference (Section 2.6.1), and coreference resolution in the form of the Winograd Schema Challenge. Some of the tasks, e.g., summarization, are generative; others, e.g., natural language inference, are classification tasks.

To convert each task into the same form, the authors use *task prefixes*. Consider, for example, the task of translating English into German. The authors assign it the task prefix "translate English to German." Given "translate English to German: That is good," the model is trained to predict the German translation "*Das ist gut.*"

T5 has a fixed vocabulary $V = \{t_m : 0 \leq m < M\}$ of $M = 32100$ tokens [34], which includes several masking tokens.

The training dataset is made up of several task-specific datasets as well as a large self-supervised dataset for masked language modelling. The self-supervised dataset, "consisting of hundreds of gigabytes of clean English text scraped from the web," is called the *Colossal Clean Crawled Corpus*. The task-specific datasets include two

Winograd datasets: DPR and WNLI-2 (Sections 2.7.2 and 2.7.3 respectively).

Our interest is in masked language modelling. The model takes a sequence $\sigma = \sigma_1\sigma_2\cdots\sigma_n$ of $n > 1$ tokens, where we can assume that masking tokens $T_1, \ldots, T_N$ occur at non-consecutive indices $i_1 < \cdots < i_N$ for some $N \geq 1$. Unlike RoBERTa, where a masking token always represents a single token to predict, the masking tokens in T5 stand for a span of one or more tokens to be predicted together.

The model also takes a label: a sequence $\tau = \tau_1\tau_2\cdots\tau_m$ of $m > 1$ tokens, where the masking tokens $T_1, \ldots, T_N, T_{N+1}$ occur. The idea is that the first masking token occurs first, $\tau_1 = T_1$, and the extra masking token occurs last, $\tau_m = T_{N+1}$. In the label sequence τ, we interpret the span of tokens between T_k and T_{k+1} as a candidate to replace T_k in the input sequence σ, for $1 \leq k \leq N$. The model returns

$$\{p^\circ_j : 1 \leq j \leq m\}, \tag{2.4}$$

where p°_j is taken to represent the probability of producing τ_j in position j given σ as input. Note that the masking tokens $T_1, \ldots, T_N, T_{N+1}$ are predicted as well.

To obtain the model's loss, we take the mean of $\log p^\circ_j$ over all masked spans, $j = 1, \ldots, m$, then multiply by -1: another "average negative log probability" [76].

Before tokenizing any string, T5 appends a special end-of-string token.

The vocabulary was extracted from the training dataset through a system called *SentencePiece* [48]. The result is neither minimal nor prefix-free. It is case-sensitive. It is not a complete list of strings up to a certain length. It is sensitive to preceding whitespace, which is represented by a special, more visible character; it is not closed under prepending a space. Well over half the tokens start with a space. Apart from the use of SentencePiece, all of this is consistent with RoBERTa.

T5's vocabulary includes one hundred unique masking tokens.

2.3 Overview of the Winograd Schema Challenge

We saw in Section 1.1.3 that Terry Winograd recognized important common-sense aspects of natural language processing as early as 1970 [103], when he introduced a

version of this influential example of a pair of interpretation problems [104]:

a. The city councilmen refused the demonstrators a permit because **they** {feared violence}.
b. The city councilmen refused the demonstrators a permit because **they** {advocated revolution}. (2.5)

Sentences (2.5a) and (2.5b) differ only by a key phrase (in braces). Each sentence includes a grammatically ambiguous pronoun (in bold). In order to decide which of two referents (underlined) is more likely, which depends on the choice of key phrase, a program, Winograd [103] argued, requires "information and reasoning power."

We saw in Section 1.1.4 that Levesque [52] introduced the *Winograd Schema Challenge* (WSC) based on Winograd's example (2.5) as an alternative to the Turing Test. Again, we will refer to the revised and expanded version of that paper [53].

The form of the challenge is a multiple-choice test of verbal reasoning in the English language. Each problem has two answer options, exactly one of which is correct, so we expect to achieve an accuracy of 50% by guessing at random. The first answer is the correct one in about as many problems as the second answer is, so we expect to achieve about the same 50% accuracy by always choosing the first answer.

To pass the challenge is to achieve "human" or at least "near human" accuracy, which we expect to be close to 100% because the problems are designed so that the correct answers are "obvious to the human reader" [53]. We will discuss human performance in Section 2.5.1. In any case, at the time of the challenge's introduction, human-like performance was "far beyond the current state of the art."

What would it mean to pass the test? WSC "appeals to world knowledge and default reasoning abilities," i.e., common sense (Section 1.1.3). "The claim is that doing better than guessing requires subjects to figure out what is going on."

> You need to have background *knowledge* that is not expressed in the words of the [problem statement] to be able to sort out what is going on.... And it is precisely bringing this background knowledge to bear that we informally call *thinking*. [Emphasis in original.]

In short, Levesque et al. [53] "believe that in order to pass the [Winograd Schema] Challenge, a system will need to have commonsense knowledge about space, time,

physical reasoning, emotions, social constructs, and a wide variety of other domains," as well as "procedures to reason with that knowledge" or bring it to bear.

The authors refer to the common-sense project involving formal languages started by John McCarthy as "the *knowledge-based* approach" (emphasis in original) to AI, or at least to passing a challenge like WSC, and they "believe it remains the most likely path to success." However, they expect and allow for "different approaches" to the challenge, particularly statistical NLP, including language models [53]:

> Statistical approaches toward natural language processing... have become increasingly popular since the 1990s. ... The successes of the last several decades in such NLP tasks as text summarization and question-answering have been based on statistical NLP.
>
> These successes... generally do not extend to the type of deep reasoning that we believe is required to solve the [Winograd Schema Challenge]. But if statistical approaches over large corpora... work better, so be it.

After its introduction, "the challenge attracted a fair amount of favorable interest from both the research community and the popular science press" [47], and "in the years following its publication, the challenge became a focal point of research for both the commonsense reasoning and natural language processing communities" [46].

So what exactly is in the Winograd Schema Challenge?

2.4 The Winograd Schema: Basic Properties

A dataset for WSC, or *Winograd dataset*, consists of a number of *Winograd schemas*, or schemas for short until we introduce other types of schema in Chapter 3. A schema generates a pair of pronoun disambiguation problems, and the problems make up the challenge. We will discuss particular Winograd datasets in Section 2.7.

A Winograd schema has the following properties [53]:

1. The schema includes a sentence with a *placeholder*, { }. For example:

   The trophy doesn't fit into the suitcase because it is too { }. (2.6)

2. The schema identifies two *candidates*, which are noun phrases that appear in the sentence. In this example, the candidates are "the trophy" and "the suitcase."

3. The schema identifies a *target*, which is a pronoun that appears in the sentence and, by its person, number, and gender, could refer to either candidate. In this example, the target is "it" (third person, singular, neuter).

4. The schema provides two *key words* that can replace the placeholder, yielding alternative sentences. In this example, the key words "large" and "small" yield:

   a. The trophy doesn't fit into the suitcase because it is too large.
   b. The trophy doesn't fit into the suitcase because it is too small. (2.7)

5. The interpretation of the ambiguous pronoun depends on the choice of key word: the first key word makes the first candidate overwhelmingly more likely as the referent, and the second key word similarly favours the second candidate.

Annotating the above example with all the relevant information, we can write the complete Winograd schema as follows:

The trophy doesn't fit into the suitcase because **it** is too {large, small}. (2.8)

The point is that each of the schema's alternative sentences, which differ only by a key word, presents a problem to solve: to identify the more likely intended referent of the ambiguous pronoun, given the two candidates as answer options.

a. The trophy doesn't fit into the suitcase because it is too large.
   What is too large? (i) the trophy ✓ (ii) the suitcase
b. The trophy doesn't fit into the suitcase because it is too small.
   What is too small? (i) the trophy (ii) the suitcase ✓ (2.9)

We have formatted the problems as human-readable multiple-choice questions with two natural-language answer options, in the style of Levesque et al. [53].

Thus, a pair of key words yields a pair of problems, almost identical in form, with the same set of answer options. That symmetry is an important feature and justification of the Winograd schema. We will return to this point in Section 2.5.3.

The five properties listed above are necessary but not sufficient: a Winograd schema is supposed to satisfy an additional constraint, to which we now turn.

2.5 An Adversarial Constraint on the Winograd Schema

Winograd schemas are also supposed to satisfy an *adversarial constraint*, and we have placed it here, in a separate section, for two reasons. First, unlike the properties in Section 2.4, which are fairly straightforward, the following constraint is complex and even ambiguous, which necessitates a somewhat lengthy discussion. Second, as we will see in Section 2.7, datasets for the Winograd Schema Challenge in practice either fail to perfectly satisfy the additional constraint or intentionally relax it.

Levesque et al. [53] put the constraint succinctly: a Winograd schema must be "designed so that the correct answer is obvious to the human reader, but cannot easily be found using selectional restrictions or statistical techniques over text corpora."

There are three elements to this explicitly adversarial constraint, and we will discuss each of them below: obvious answers in Section 2.5.1, selectional restrictions in Section 2.5.2, and statistical techniques in Section 2.5.3.

2.5.1 Untrained Subjects and Intelligence Tests

First, the answer to a Winograd problem must be "obvious to the human reader," so that the problem is answerable "immediately" by an "untrained subject" [53]. We already knew that WSC is a test of verbal reasoning in the English language, which makes it an *intelligence test*, at least for English speakers; now we know that it is a relatively easy one, and human-like performance is close to 100% accuracy.

In that respect, the authors are broadly consistent with the literature on common sense in AI: they ask that the problems be easy, but not arbitrarily easy. We saw that McCarthy [60] cites the verbal reasoning processes of "any non-feeble-minded human." Davis [11] refers to knowledge "possessed by every schoolchild."

Similarly, Levesque et al. [53] suggest "your Aunt Edna" as an appropriate test subject. Still, the sample problems they present admittedly "differ on the background knowledge assumed," with some, including (2.9), being suitable for "anyone" ten and up, and others, including (2.5), being "more 'university-level'" in difficulty.

Intelligence tests, i.e., standardized formal tests of human mental ability featuring questions of varying difficulty, have been used to evaluate common sense in computer systems. Ohlsson et al. [68] evaluate one such system—not based on a language

model or a neural network—using a test designed to measure verbal reasoning in young children. Now, Ohlsson et al. [69] argue that WSC, by comparison, does not qualify as an intelligence test "because the questions are designed so that any normally competent speaker of English can easily answer them correctly." On the other hand, their own test includes questions like "What is a house?" Arguably, WSC is indeed an English-language verbal intelligence test, albeit not a carefully calibrated one.

Table 2.2 shows human performance on Winograd datasets, with notes to follow.[4]

Dataset	Source	Accuracy
quasi-WSC285	Bender [3]	0.921
quasi-WSC285	Rudinger et al. [82]	0.865
WSC273	Sakaguchi et al. [85]	0.965
DPR/WSCR	Sakaguchi et al. [85]	0.952
WNLI	Nangia and Bowman [66]	0.959
WinoGrande	Sakaguchi et al. [85]	0.940

Table 2.2: Human performance on the Winograd Schema Challenge.

The sources all employ crowd workers but otherwise differ in their methodology, which is less than detailed; e.g., only Bender [3] reports the number of subjects (407). That paper administers a random sample of problems, 36 on average, presumably from a 2015 version of WSC285; Rudinger et al. [82] presumably use a 2018 version but give no details. We can defer descriptions of the datasets until Section 2.7, because for now it is enough to note that accuracies over 90% are evidently typical, and that the first element of the adversarial constraint can be made reasonably precise.

Incidentally, it is also possible to make a Winograd schema arbitrarily difficult: Cozman and Munhoz [8] encode the digits of a non-computable real number, among other things, as pronoun disambiguation problems. We will not pursue this here.

2.5.2 Selectional Restrictions

The second element of the adversarial constraint is that a Winograd schema "should not be solvable by... selectional restrictions" [53]. From linguistics, a *selectional restriction* is a semantic restriction on how a word can be used in a sentence.

[4]Not included: Davis et al. [16] record 92–93% accuracy on 89 unpublished Winograd schemas.

The authors offer this example of an inappropriate schema:

> The women stopped taking the pills because **they** were {pregnant, carcinogenic}. (2.10)

They argue that because pills cannot be pregnant and women cannot be carcinogenic, a pair of selectional restrictions, both problems can be solved from the candidates and the key word alone, "ignoring the sentence completely."

The second element of the constraint can be made reasonably precise if we assume that the designers of Winograd Schema Challenge datasets are fluent speakers of English who can agree on what qualifies as a selectional restriction.

Levesque et al. [53] note that "selectional restrictions... might be learned by sampling a large enough corpus." That leads us to the final element of the constraint.

2.5.3 Associativity and Spurious Correlations

The third and final element of the adversarial constraint is that a Winograd schema "should be Google-proof; that is, there should be no obvious statistical test over text corpora that will reliably disambiguate [the target pronoun] correctly" [53].

The key word feature is a major ingredient in fulfilling that element:

> With [the key word feature], we can see that clever tricks involving word order or other features of words or groups of words will not work. Contexts where [one key word] can appear are statistically quite similar to those where [the other key word] can appear, and yet the answer must change. This helps make the test *Google-proof*: having access to a large corpus of English text would likely not help much.... [Emphasis in original.]

So the key word feature is adversarial by design: it represents an attack on one approach to common-sense tasks, namely statistical NLP.

The authors clearly expect the built-in adversarial feature to make the test more robust, and their reasoning seems valid in general. For example, regarding a different test of verbal reasoning in natural language, Niven and Kao [67] create adversarial versions of the problems and are thereby able to show that one transformer-based model's near-human accuracy "can be entirely accounted for in terms of exploiting spurious statistical cues." We will discuss spurious correlations in more detail below.

However, by itself, the key word feature may not be enough to defeat even obvious statistical tests. We have already seen that a selectional restriction can violate the constraint, as in (2.10). Levesque et al. [53] offer another example:

> The race car zoomed by the school bus because **it** was going so {fast, slow}. (2.11)

The authors argue that, in text corpora, "fast" is much more closely associated with "race car" than with "school bus." They suggest replacing "race car" with "delivery truck," and the revised version appears in the WSC273 dataset (Section 2.7.1).

Recall that the correct answer to a Winograd schema problem depends on the choice of key word. To the extent that the key word, possibly in the context of the sentence, is more strongly associated with the correct answer than with the incorrect answer, however we choose to quantify that association, we call the problem, or the schema that contains it, *associative*, a term due to Trichelair et al. [95]. The third element of the adversarial constraint attempts to exclude associative schemas.

It is not immediately obvious how to quantify the strength of word associations. The term "Google-proof" does suggest a method, of course, and indeed Levesque et al. [53] attempt to identify associative schemas by performing "experiments with searches using Google's count of result pages," while freely acknowledging that search engine result counts are "notoriously unreliable." Despite that unreliability [51], one might expect a modern search engine to reliably solve an example like (2.11).

The type of association exemplified by (2.11) might be called *genuine*, for lack of a better term. Race cars are, in fact, capable of great speed when compared to other land vehicles, including school buses. Therefore, we expect a sufficiently large and representative natural-language corpus to exhibit a relatively high correlation between "race car" and "fast." Therefore, we expect a model trained on such a corpus to achieve better than random accuracy on such a problem without necessarily figuring out anything about the relationship between "going fast" and "zooming by." Therefore, we should exclude such a schema from a test of common sense.

On the other hand, it is not necessarily the case that we should exclude a schema simply because a model solves it. The constraint refers to an "*obvious* statistical test" that solves the problem; a solution that can "*easily* be found using... statistical techniques" (my emphasis). Arguably, a neural network language model with billions

of parameters is not easily trained, and its inner workings are by no means obvious.

Now suppose such a model achieves human-like accuracy on a Winograd dataset that we have checked for associativity. There would be three possible conclusions. First, the model has achieved common sense, because it passes the Winograd Schema Challenge. Second, the Winograd Schema Challenge is flawed, because the model passes it without exhibiting common sense, which we might demonstrate using a different evaluation protocol or a different type of challenge (both of which we will discuss later). Third, the particular problem set we used must be associative in some way that we failed to detect, and therefore not a true Winograd Schema Challenge, because, after all, such a model merely applies statistical tests and techniques. We would really like to exclude the third option as a case of moving the goalposts.

This is essentially the problem of detecting what the literature calls a *spurious correlation*. Spurious correlations can arise in any model that has been trained on a specific dataset: they are "prediction rules that work for the majority" of problems in that dataset "but do not hold in general" [99]. Spurious correlations "can be learned during pretraining or fine-tuning" and, in any case, "may result in successful predictions that do not reflect commonsense reasoning skills" [23].

For example, suppose we crowdsource a training dataset for some NLP task. If the crowd workers generate problems by following a specific protocol, that protocol may introduce *annotation artifacts* to the dataset: "specific linguistic phenomena" that are "highly correlated" with specific labels in that dataset but not in general [27]. A model trained on such a dataset is likely to exhibit spurious correlations.

Elazar et al. [23] attempt to detect and account for spurious correlations. Suppose we have a dataset for the Winograd Schema Challenge. The authors propose two *control baselines*: systematic modifications of the dataset on which a model that is free from spurious correlations is "likely to achieve random performance." Therefore, to the extent that a model performs better than random guessing on the baselines, that model is likely to be exploiting associativity in the dataset. For example, the baseline version of (2.11) is likely to be quite solvable. Conversely, if the model performs no better than random guessing on the baselines, we may be able to exclude associativity as an explanation for the model's success on the original dataset.

We will discuss control baselines in more detail in Section 2.9.4.

There are other ways of accounting for associativity. Sakaguchi et al. [85] attempt to filter annotation artifacts from WinoGrande (Section 2.7.5). They compare their approach to filtering based on pointwise mutual information, a measure of association used by, e.g., Abdou et al. [1] to detect associativity in Winograd datasets.

Filtering annotation artifacts from a dataset is a special case of what Schwartz and Stanovsky [88] call *balancing*: modifying the training data "to prevent models from learning spurious correlations." They argue that, in general, "too much balancing can prevent models from learning valuable knowledge," and, for common-sense reasoning in particular, balancing is actually an "*undesired*" solution (emphasis in original).

Without necessarily accepting their conclusion of a "lost battle against spurious correlations," we have decided not to pursue filtering or other forms of balancing.

With at least one way of accounting for associativity, namely control baselines, the final element of the adversarial constraint can be made reasonably precise.

2.6 The Winograd Schema as Natural Language Processing

With the addition of an adversarial constraint (Section 2.5) to the basic properties listed in Section 2.4, the definition of the Winograd schema is complete. So how does the schema compare to other types of problem in natural language processing?

2.6.1 Coreference Resolution and Related Tasks

Clearly, the task of automatically solving Winograd schema problems belongs to NLP. Figure 2.1 shows how the Winograd schema is related to some other important NLP tasks and problem types. That figure will be updated in Section 3.5.1 (Figure 3.1).

Winograd schemas are based on *pronoun disambiguation*: determining the referent of an ambiguous pronoun. The trophy-suitcase problem (2.9b) is a simple example of pronoun disambiguation, which we reproduce here for convenience:

> The trophy doesn't fit into the suitcase because it is too small.
> What is too small? (i) the trophy (ii) the suitcase ✓ (2.12)

Of course, not every pronoun disambiguation problem belongs to a Winograd schema: for one thing, there must be a key word. Of all the NLP tasks and problem types in this section, only the Winograd schema has such an adversarial feature built in.

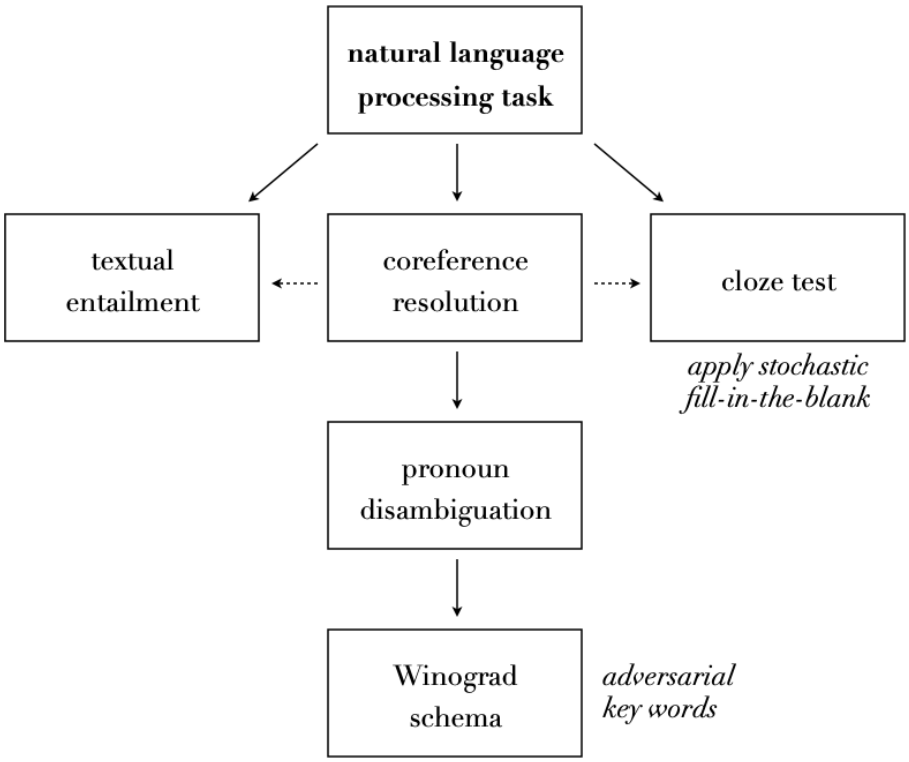

Figure 2.1: Winograd schemas as natural language processing. Solid lines point to more specific tasks; dashed lines indicate possible one-way recasting.

Pronoun disambiguation is a subset of *coreference resolution*: determining the referent of an expression, which may or may not be a pronoun. For example, this adaptation of (2.12) does not involve pronoun disambiguation and in particular does not belong to any Winograd schema, but it is still a case of coreference resolution:

> The trophy doesn't fit into the suitcase because the container is the wrong size. What is the container? (i) the trophy (ii) the suitcase ✓ (2.13)

Another NLP task, which is rather different from coreference resolution, is *textual entailment* [9] or *natural language inference* (NLI) [4]: to determine whether one statement logically, or at least plausibly, implies another statement, contradicts it, or neither. For example, the *Recognizing Textual Entailment* challenge (RTE) conforms to this task [10]. WSC was originally described as a "variant" of RTE [53].

Wang et al. [101] recast Winograd schema problems as textual entailment; the result is the *Winograd Natural Language Inference* dataset (WNLI), which we will revisit in Section 2.7.3. For example, the pronoun disambiguation problem (2.12) can

be recast as two separate natural language inference problems. Here is one:

P. The trophy doesn't fit into the suitcase because it is too small.
Q. The trophy doesn't fit into the suitcase because *the trophy* is too small. Does P imply Q? (i) true (ii) false ✓ (2.14)

The other problem substitutes *the suitcase*, and the correct answer is "true."

Indeed, (2.14) shows that coreference resolution in general can be recast as natural language inference, although it is not at all clear that we ought to do so: Kocijan et al. [44] convert WNLI back into pronoun disambiguation form before tackling it with stochastic fill-in-the-blank, and in T5 pretraining, Raffel et al. [79] convert WNLI into a text-to-text format where the text to be predicted is the referent.

Speaking of filling in the blank, a *cloze test* [92] is another type of NLP problem: basically, any fill-in-the-blank question.[5] An example, based once again on (2.12):

The trophy doesn't fit into the suitcase because it, ⟨ ⟩, is too small.
What is missing? (i) the trophy (ii) the suitcase ✓ (2.15)

If the pronoun is not considered to be meaningful or informative in the context of the entire problem, including the given answer options, then we can drop it:

The trophy doesn't fit into the suitcase because ⟨ ⟩ is too small.
What is missing? (i) the trophy (ii) the suitcase ✓ (2.16)

In any case, (2.15) shows that coreference resolution problems in general, and Winograd schemas in particular, can be recast as cloze tests. Indeed, that is the premise of applying to Winograd schemas the method of Trinh and Le [98] that we call stochastic fill-in-the-blank: that method (Section 1.1.5) applies straightforwardly to any cloze test and therefore to any other problem that can be recast as a cloze test. We will revisit the method of stochastic fill-in-the-blank in Section 2.8.1.

There are many other important tasks in NLP, and some are related to commonsense reasoning. For example, the adversarial problems of Niven and Kao [67] that we mentioned in Section 2.5.3 belong to *reasoning comprehension* [28], which is closely related to textual entailment. But that is outside the scope of our research.

[5]The cloze test as a problem type should not be confused with the *Story Cloze Test* [64], a reading comprehension dataset that has more in common with textual entailment.

2.6.2 Winograd Problems in Datasets

In previous sections, we mentioned a few datasets related to the Winograd Schema Challenge, including WSC273, WinoGrande, and WNLI, without defining them. In Section 2.7, we will discuss Winograd datasets in detail. Before we do, we should add a word to differentiate problems, which we have been presenting in the form of human-readable questions, and data points, which may not take that form.

Each of (2.12), (2.13), (2.15), and (2.16) is a problem suitable for NLP that takes the form of a human-readable multiple-choice question with two natural-language answer options. On the other hand, (2.14) takes the form of a true-or-false question, one of a pair into which (2.12) was recast in a way similar to Wang et al. [101].

Test sets for the Winograd Schema Challenge sometimes do consist of human-readable questions with natural-language answer options. Training datasets for the challenge, on the other hand, typically take a different form.

For some training datasets, a pronoun disambiguation problem is converted into two pairs of plain-text statements. Each pair is treated as a true-or-false problem: does the first statement imply the second? This conversion is accomplished in the same way that (2.12) became (2.14). To highlight the new format:

(2.17)

i. The trophy doesn't fit into the suitcase because it is too small.
   The trophy is too small. Label: 0 (false)
ii. The trophy doesn't fit into the suitcase because it is too small.
   The suitcase is too small. Label: 1 (true)

Thus, a problem generates two data points. Since a Winograd schema already generates two problems because of the key word, a schema may actually be represented in a dataset by as many as four distinct data points.

We now turn to some actual Winograd datasets.

2.7 Notable Winograd Datasets

We know that a testing dataset or test set for the Winograd Schema Challenge consists of a number of data points, derived from pronoun disambiguation problems, derived from Winograd schemas (Sections 2.4 and 2.6.2). We also know that training precedes

deployment (Section 2.1.3): before taking on a Winograd Schema Challenge, a model is either trained on a dataset that involves pronoun disambiguation, or pretrained for a general NLP task on some large natural-language corpus and possibly finetuned for pronoun disambiguation. Table 2.3 shows some notable testing and training datasets. The most important of these for our purposes is WSC273.

Dataset	Source	Testing	Training	Validation
WSC285	Davis [12]	285	–	–
WSC273[a]	Davis [12]	273	–	–
DPR/WSCR	Rahman and Ng [80]	564	1322	–
WNLI[b]	Wang et al. [101]	146	635	71
WSC/WNLI-2[b]	Wang et al. [102]	146	554	104
MaskedWiki	Kocijan et al. [44]	–	2.4M	–
WikiCREM	Kocijan et al. [43]	–	2.4M	–
WinoGrande	Sakaguchi et al. [85]	1767	41K	1267

Table 2.3: Notable datasets for the Winograd Schema Challenge. (a) WSC273 is a subset of WSC285. (b) WNLI and WNLI-2 each consist of true-or-false problems, a pair of which is equivalent to a problem from any other dataset (Section 2.6.2); moreover, the test sets for WNLI and WNLI-2 are identical except for formatting.

Also worth mentioning are Morgenstern et al. [63]'s 2016 *Pronoun Disambiguation Problem* dataset (PDP), Emami et al. [25]'s 2019 *KnowRef* coreference corpus, and Emami et al. [26]'s updated 2020 *KnowRef-60k* corpus. None of them use a key word adversarial feature, and they are outside the scope of our research.

Below, we include a brief history and discussion of the datasets in Table 2.3.

2.7.1 WSC285 and WSC273

In 2011, Levesque [52] presented twelve examples of Winograd schemas in the main text, three of which deliberately violate the adversarial constraint (Section 2.5) and two of which are improved versions of inappropriate schemas. The paper also includes an appendix with ten more examples, for a total of 19 valid schemas and 38 problems.

From 2011 to 2013 [12], Davis et al. [17] added more examples, building up a collection of 142 Winograd schemas and 285 problems called *WSC285*; part of that collection-in-progress was included in Levesque et al. [53]. Some sources report results only on the first 136 schemas and 273 problems, leading later authors to make the same restriction for the sake of consistency: the reduced dataset is called *WSC273*.

Due to its popularity and precedence, WSC273 is often referred to simply as "the Winograd Schema Challenge." Although it is an influential test of common sense, it is still only one test, with only 273 problems. According to Trichelair et al. [96], "the main drawback of the Winograd Schema Challenge [meaning WSC273] is its limited size and the absence of training and validation sets for hyper-parameter tuning."

Unfortunately, generating new Winograd schemas is highly non-trivial: inventing a scenario involving two parties with a pronoun whose interpretation depends on the choice of key word is an exercise in creative writing. Kocijan et al. [47] note that "manually creating a large, diverse collection of high-quality Winograd schemas is inherently difficult." Ruan et al. [81] also mention how difficult it is "to acquire high-quality samples for common-sense reasoning under [WSC] settings." And Trichelair et al. [95] state that "it is difficult and expensive to acquire high-quality datasets for specialized inference tasks" in general, and for WSC in particular.

Illustrating that difficulty, it was later discovered that some of the schemas in WSC273 violate the adversarial constraint (Section 2.5): Trichelair et al. [95], using three human annotators, identify 37 associative problems [93]. For example:

> In the storm, the tree fell down and crashed through the roof of my house. Now, I have to get **it** {removed, repaired}. (2.18)

Repairing a roof is more likely than repairing a tree.

Because the authors use human annotators, any link they report between key words and candidates is expected to be a genuine association (Section 2.5.3). But a model can also learn spurious correlations. Elazar et al. [23], using their method of control baselines (Section 2.9.4), find that one model with high accuracy on WSC273, namely Sakaguchi et al. [85], performs significantly better than random guessing on both baselines, even after removing those 37 associative problems [23]:

> These results indicate that [WSC273] contains many artifacts (over 15 points above random performance), and even after the manual filtering of Trichelair et al. [95] some statistical correlations remain.

Trichelair et al. [95] also note that the problem statements in WSC273 display "predictable structure" or "distinctive regularities" that a computer system "can leverage... in various ways." For example, "for a high number of instances, the [key

word] is the last word, or nearly the last word." Also, "many [WSC273] instances are composed of two clauses connected by a causal discourse connective like *because*" (emphasis in original). A system that is "exploiting these structural regularities" may "become brittle to perturbations that would not affect the judgment of a human."

There are other issues with WSC273, which we will discuss in Section 3.3.

2.7.2 DPR/WSCR

In 2012, Rahman and Ng [80] published a collection of 943 Winograd-like schemas or 1886 problems, written by thirty undergraduate students: the *Definite Pronoun Resolution* dataset (DPR), also known as WSCR [71]. Exact or almost exact copies of three schemas from DPR appear, with attribution, in WSC273 [12].

Rahman and Ng [80] describe DPR as a "relaxed version" of WSC, because they allow associative schemas (Section 2.5.3) by relaxing the adversarial constraint. For example, the following schema from the DPR training set is highly associative:

Lions eat zebras because **they** are {predators, meaty}. (2.19)

The word "predators" is more closely associated with "lions" than with "zebras."

Besides a relaxed constraint on associativity, Opitz and Frank [71] note DPR's "notably... lower quality" overall when compared to WSC273. In particular, DPR "contains sentences with no straightforward resolution." For example:

The bus driver yelled at a kid after **she** {drove her vehicle}.
Who is **she**? (i) the bus driver ✓? (ii) a kid (2.20)

Moreover, although we know that WSC273 problems already display predictable structure, DPR problems are even "less diverse as all consist of exactly one sentence and in every sentence we find at least one discourse connector or a comma connecting a main clause with the [candidates] to a sub-clause that contains the pronoun."

2.7.3 WNLI and WSC/WNLI-2

We saw in Section 2.6.1 that Wang et al. [101] recast Winograd schema problems as textual entailment, more often called natural language inference (NLI), producing the *Winograd Natural Language Inference dataset* (WNLI), which is part of the popular *General Language Understanding Evaluation benchmark* (GLUE) from 2018.

Which specific problems were converted to create WNLI? The test set consists of 146 data points generated from new Winograd-like pronoun disambiguation problems "derived from fiction books" and "shared privately" by the creators of WSC285 [101]. Again, these are true-or-false problems, a pair of which is equivalent to a Winograd problem. The training and validation sets, numbering 635 and 71 respectively, each contain a mixture of new problems, as in the test set, and problems recycled from WSC285. Indeed, between the training and validation sets, WNLI recycles every schema from WSC285 except the last one; in particular, it recycles all of WSC273. Recycled problems make up about 80% of the training and validation sets combined.

Some data points in the WNLI test set belong, in sets of four, to Winograd schemas with a key word; others do not. There are obvious spelling mistakes in the test set; e.g., in various problems, the name "Hermione" is spelled both correctly and as "Hermoine," "tucked" is spelled both correctly and as "tucker," and so on.

Historically, published models that perform well on WSC273 and report results on both datasets perform even better on WNLI [47], making it an easier challenge. We will discuss the performance of models on Winograd datasets in Section 2.8.4.

The updated *SuperGLUE* benchmark from 2019 also includes a Winograd dataset in the form of natural language inference problems [102]. That dataset is called "WSC," but, to avoid ambiguity, we will refer to it as *WNLI-2* instead.

The test set for WNLI-2 is just the test set for WNLI in a different format. On the other hand, the training and validation sets for WNLI-2, numbering 554 and 104 respectively, differ from their counterparts in WNLI, but they each still contain a mixture of new problems and problems recycled from WSC285. Between the training and validation sets, WNLI-2 recycles about half of WSC285 and WSC273. Recycled problems make up about 40% of the training and validation sets combined.

2.7.4 MaskedWiki and WikiCREM

In 2019, Kocijan et al. [44] observed that passing the Winograd Schema Challenge remained "difficult... not only because of the commonsense reasoning challenge, but also due to the small existing datasets making it difficult to train neural networks directly on the task." Kocijan et al. [43], from the same year, attributed the scarcity of "large-scale training sets" to the cost of "manually labelling data." Both papers

then attempt to "address the lack of large training sets for pronoun disambiguation by introducing a large [self-supervised] dataset that can be easily extended" [43].

For each dataset, the authors take sentences from English-language Wikipedia articles and systematically convert them into fill-in-the-blank questions, or cloze tests, inspired by pronoun disambiguation (Section 2.6.2). None of the problems include the key word feature of a Winograd schema, and many of them cannot be recast as pronoun disambiguation. The resulting datasets are *MaskedWiki* [44] and *Wikipedia Co-REferences Masked* (WikiCREM) [43], each with about 2.4 million problems.

MaskedWiki takes a sentence containing multiple occurrences of the same noun, according to an automatic part-of-speech tagger; replaces the second occurrence of that noun with a special token; and for each other noun in the sentence, creates a problem where that other noun is offered as a possible answer. For example [41]:

> This time less damage was done, and the ⟨ ⟩ mainly occurred around the lower right hand ring post. What is missing? (i) damage ✓ (ii) time (2.21)

We can create another version of the problem with "post" as the incorrect answer. It is unlikely that two versions of a given problem appear in the final dataset, as it consists of 2.4 million problems selected at random from an initial set of 130 million.

By inspection of a random sample of 200 problems, the authors find that 8.5% are "unsolvable," as the correct answer "cannot be unambiguously selected with the given context;" 45% are "hard," as "the answer is not trivial to figure out;" 45.5% are "easy," as "the alternative sentence is grammatically incorrect or is very visibly an inferior choice;" and 1% are "noise," as "the example is a result of a parsing error."

Additionally, based on a sample of 100 problems, Kocijan et al. [43] estimate that only 7% of MaskedWiki problems can be recast as pronoun disambiguation by replacing the special token with a pronoun: in the other cases, that method does not produce "a natural-sounding and grammatically correct sentence."

WikiCREM, on the other hand, takes a sentence with more than one "personal name," according to an automated tagger, such that one name is repeated; replaces a later occurrence of the repeated name with a special token; and for each other name,

creates a problem where that other name is a possible answer. For example [42]:

> With Otto she had three sons, Peter I, Amadeus II, and ⟨ ⟩.
> What is missing? (i) Otto ✓ (ii) Peter I (2.22)

In this spurious example, a son shares his father's name.

Based on a random sample of 100 problems, the authors estimate that 18% are "unsolvable," compared to MaskedWiki's 8.5% unsolvable and 1% noise. However, they also find that 63% of the problems in WikiCREM can be recast as pronoun disambiguation (as above), compared to 7% for MaskedWiki.

By inspection, we find numerous defects in both MaskedWiki and WikiCREM. For example, in the former, unreadable byte sequences, which may be encoding artifacts, appear in several problem statements and are even offered as correct answers. And in the latter, several problem statements are sentence fragments ending in "U." because the common abbreviations "U.S." and "U.S.A." were parsed incorrectly.

2.7.5 WinoGrande

In 2019, and in light of recent progress on WSC273 by language models, Sakaguchi et al. [83, 85] questioned whether such models had actually achieved common sense, a topic we will discuss in Section 2.8.4. The authors argued that, in order to measure "the true capabilities of machine commonsense," we need larger datasets that are free from "unwanted biases" or annotation artifacts (Section 2.5.3). Accordingly, they introduced a crowdsourced dataset called *WinoGrande*, as well as a bias reduction algorithm called *AfLite*, which they used to filter WinoGrande.

According to the authors, crowd workers wrote pairs of sentences "that meet the requirements for WSC [pronoun disambiguation] problems." Each pair of sentences was then validated by a team of three crowd workers in "a rigorous process" to ensure that "the two answer options are unambiguous" and "the question cannot be answered simply by word association." Each sentence was then "formatted as a fill-in-the-blank problem," much like MaskedWiki and WikiCREM (Section 2.7.4). The authors also note that "unlike the original WSC [meaning WSC273] problems that were composed by just a few linguistics experts, ... the language used in WinoGrande reflects the more diverse and noisy language used by crowds." Finally, a bias reduction algorithm

was applied to filter out problems that potentially exhibit annotation artifacts.

The end result is a filtered test set of 1767 problems, a filtered validation set of 1267 problems, a filtered training set of 9248 problems, and an unfiltered training set of 40938 problems. Many of the problems in the filtered dataset do not occur in pairs, their counterparts having been filtered out, but all the problems in the unfiltered dataset occur in pairs. WinoGrande has become the standard WSC test for models: Kocijan et al. [47] report that "the majority of the papers published after the release of WinoGrande only evaluated their work on that dataset."

Those authors also note that "many of the sentences in the filtered WinoGrande corpus do not fit the criteria laid out in the WSC, and therefore are not actually Winograd schemas" [47]: some problems are associative, some "contain the answer directly in the sentence," and some are "genuinely hard to understand." Emami et al. [26] further argue that the sentences in WinoGrande, like WSC273 and DPR (Sections 2.7.1 and 2.7.2), display "predictable structure": as usual, "instances are often composed only of two clauses connected by a single causal discourse connective."

For the bias reduction algorithm, Sakaguchi et al. [85] begin by finetuning a model, specifically RoBERTa, on a sample of 6000 WinoGrande-type problems, which were removed from the final dataset. The authors compute the *embedding* in the model of each remaining WinoGrande problem: how the finetuned model represents that input as a vector. Finally, they train simple logistic regression classifier models on random subsets of embeddings; if the simple models are able to predict the correct answer from RoBERTa's embedding, the authors discard the problem. In this way, the algorithm "generalizes human-detectable biases based on word occurrences to machine-detectable biases based on embedding occurrences."

As we stated at the end of Section 2.5.3, we will not pursue filtering here.

2.8 Language Models versus the Winograd Schema Challenge

This section expands on material introduced in Section 1.1.5.

2.8.1 A Simple Method: Stochastic Fill-in-the-Blank

We saw in Section 2.6.1 that coreference resolution problems in general, and Winograd schemas in particular, can be recast as cloze tests (fill-in-the-blank questions), which

is the premise of applying to Winograd datasets the following method.

In 2018, Trinh and Le [97, 98] presented "a simple method," or a family of simple methods, for implementing "commonsense reasoning," and specifically for taking on the Winograd Schema Challenge, with an arbitrary language model. The method is very widely cited in the literature on WSC [1, 23, 25, 43, 44, 76, 81, 85, 95, 96, etc.]. We will refer to it as *stochastic fill-in-the-blank* (our own term).

How does the method work? Suppose we are given the trophy-suitcase problem from (2.9b) or equivalently (2.12). We need to determine the more likely intended referent of the pronoun "it" from among "the trophy" and "the suitcase." We can try substituting each answer option for the ambiguous pronoun:

The trophy doesn't fit into the suitcase because *the trophy* is too small.
The trophy doesn't fit into the suitcase because *the suitcase* is too small. (2.23)

Those substitutions can be performed automatically, provided the schema marks the location of the target pronoun in the problem statement.

Having made the substitutions, we use a language model to assign each of the sentences from (2.23) a score representing its relative likelihood. Whichever candidate yields a better-scoring sentence is taken to be the correct answer to the problem.

Evidently, stochastic fill-in-the-blank applies straightforwardly to any cloze test and therefore to any other problem that can be recast as a cloze test.

The reason we call this a family of methods is that for any given language model, there may be more than one way to assign scores to sentences: Trinh and Le [98] describe "full scoring," "normalized full scoring," and "partial scoring." We will return to this point in Section 3.1 when we implement the method for three models.

There is one other important detail to mention, and it concerns the length of the substituted answers: Trinh and Le [98] assume that each answer is a single word. That is not always the case for Winograd datasets, including WSC273, as we will see in Section 3.3.5. In the example above, the substituted answers, "the trophy" and "the suitcase," have two words each, and the authors are able to reduce them to one word each by fixing "the," which they have in common. However, this method does not work in general, and even when it does work, a model may convert a single word into multiple *tokens* (Section 2.2). We will address this point in Section 3.1 as well.

Elazar et al. [23] argue that the method of stochastic fill-in-the-blank is inherently

"problematic" for a masked language model, such as RoBERTa (Section 2.2.2), if the model converts the substituted answers into different numbers of tokens: for example, if "trophy" is a single token but "suitcase" gets split into two. We will discuss their argument and how we addressed that issue in Section 2.9.3.

In any case, the method of stochastic fill-in-the-blank essentially uses a language model as an oracle or black box, and its accuracy depends on that model.

2.8.2 Stochastic Fill-in-the-Blank with the GPT Series

We described the GPT-2 model and its output in Section 2.2.1. Unlike masked language modelling (e.g., RoBERTa, T5), which seems ideal for stochastic fill-in-the-blank, the GPT series' next token prediction must be adapted to that method.

Radford et al. [74] had already taken on a Winograd Schema Challenge dataset, namely DPR (Section 2.7.2), with the first GPT model. The authors implement a version of stochastic fill-in-the-blank, possibly independently of Trinh and Le [97, 98], whose first draft was published four days earlier. Radford et al. [74] write:

> We replace the definite pronoun with the two [candidates] and predict the resolution [such] that the generative model assigns higher average token log-probability to the rest of the sequence after the substitution.

Presumably, the authors implemented what Trinh and Le [98] call partial scoring: restricting their attention to the part of the sequence that follows the substituted tokens. After all, GPT models are trained to "predict the next word, given all of the previous words" [75], not to predict a previous word given a later word.

In any case, Radford et al. [74] report an accuracy of (we estimate from Figure 2 in that paper) around 55% of the way from "a random guess baseline" (0.5) to "the current state-of-the-art with a single model." The state-of-the-art on DPR in 2018 appears to have been 0.764 [72], in which case GPT achieved roughly 0.65.

Moving on to GPT-2, Radford et al. [76] do cite Trinh and Le [98], and state that "we follow their problem formulation." The authors report accuracies "with both full and partial scoring" on "the Winograd Schema Challenge," a dataset said to contain "273 examples," which is presumably WSC273 (Section 2.7.1). Partial scoring achieves an accuracy of 0.707, which was at the time a new state-of-the-art; full scoring (we estimate from Figure 3 in that paper) achieves around 0.66.

Radford et al. [76] do not describe any preprocessing of the WSC273 dataset, and they give no details of their implementation of either full or partial scoring.

As for GPT-3, Brown et al. [6] use "the same 'partial evaluation' method" as Radford et al. [76], which again is presumably partial scoring. They achieve an accuracy of 0.883 on WSC273 with zero-shot learning, where "the model is only given a natural language instruction describing the task."

We will implement stochastic fill-in-the-blank with GPT-2 in Section 3.1.1.

2.8.3 T5 Pretraining and the Winograd Schema Challenge

We noted in Section 2.2.3 that T5 was trained in part on Winograd training sets, namely DPR and WNLI-2 (Sections 2.7.2 and 2.7.3). Six problems from WSC273 are originally from the DPR training set, and, more importantly, about half of WSC273 appears in the WNLI-2 training set, making up about 40% of that dataset.

For that reason, it may seem problematic to evaluate T5 on WSC273. But T5 was trained to solve Winograd problems, not as cloze tests with masked language modelling (Section 2.2.3) and stochastic fill-in-the-blank (Section 2.8.1), but as text-to-text problems with a special task prefix and the target pronoun highlighted [79]:

"wsc: The trophy doesn't fit into the suitcase because ∗it∗ is too small."
Label: "the suitcase" (2.24)

It is an empirical question whether or to what extent T5 generalizes to stochastic fill-in-the-blank, and we will address that in Chapter 4.

More generally, there may be other ways of taking on the Winograd Schema Challenge with any given language model. However, our interest is in common-sense reasoning by stochastic fill-in-the-blank, so we will not pursue this here.

2.8.4 Progress on the Winograd Schema Challenge

We noted in Section 2.3 that to pass the Winograd Schema Challenge is to achieve near-human accuracy, and we saw in Section 2.5.1 (Table 2.2) that human accuracy can be bounded below at 90%. Here, as usual in the literature, the *accuracy* on a test simply refers to the share of problems that were answered correctly.

Table 2.4 shows recent progress on WSC by language models applying the method of stochastic fill-in-the-blank. Our focus is on the models discussed in Section 2.2, the most popular datasets from Section 2.7, and notable papers we have cited.

For reference, we have also included Emami et al. [24]'s automated framework for information retrieval (IR), which may be the first reported system with better than random accuracy on WSC273. Where applicable, we report the date of the earliest online draft. We indicate any reported finetuning of the model or enhancement of the dataset; otherwise, the result uses a pretrained model and demonstration-free learning. Refer to Table 2.1 for models and Table 2.3 for datasets. Since the test sets for WNLI and WNLI-2 are the same up to formatting, we treat them as equivalent.

Date	Source	Model	WSC273	WNLI	WinoG.
2018-06	Emami et al. [24]	IR	0.542	–	–
2018-06	Trinh and Le [97, 98]	custom	**0.626**	–	–
2019-02	Radford et al. [76]	GPT-2	**0.707**	–	–
2019-04	Ruan et al. [81]	BERT^{d}	**0.711**	–	–
2019-05	Kocijan et al. [44, 45]	BERT$^{d,\ m}$	**0.725**	0.747	–
2019-07	Liu et al. [57]	RoBERTa^{i}	–	**0.890**	–
2019-07	He et al. [30, 31]	BERT-based^{d}	**0.751**	0.836	–
2019-08	Ye et al. [106]	BERT$^{d,\ n}$	**0.755**	0.836	–
2019-09	Lan et al. [49, 50]	BERT-based^{i}	–	**0.918**	–
2019-10	Raffel et al. [78, 79]	T5$^{d,\ ii}$	–	**0.945**	–
2019-11	Sakaguchi et al. [84, 85]	RoBERTa^{w}	**0.901**	0.856	0.791
2020-03	Lin et al. [55]	T5^{w}	–	–	**0.846**
2020-05	Brown et al. [5, 6]	GPT-3^{z}	0.883	0.654	0.702
2020-10	Khashabi et al. [40]	T5$^{w,\ q}$	–	–	**0.903**
2021-03	Lourie et al. [58, 59]	T5$^{w,\ r}$	–	–	**0.913**

Table 2.4: Progress on the Winograd Schema Challenge. "WinoG." is WinoGrande. "WNLI" includes WNLI-2. Boldface indicates a new single-model state-of-the-art. The finetuning datasets are: DPR (d), WNLI (i), WNLI-2 (ii), MaskedWiki (m), WinoGrande (w), ConceptNet [90] (n), RAINBOW [58] (r), and a collection of question-answering datasets [40] (q). GPT-3 applied zero-shot learning (z).

As for the three models we studied: at one time, GPT-2 was the state-of-the-art on WSC273; at various times, RoBERTa was the state-of-the-art on WSC273 and on WNLI, as well as the first reported accuracy on WinoGrande; T5 is the state-of-the-art on both WNLI and WinoGrande, the latter having superseded WSC273.

By 2021, the accuracy of finetuned transformer-based models was sufficiently close

to the untrained human baseline on the WSC273 and WinoGrande datasets that Kocijan et al. [47] could report "the defeat of the Winograd Schema Challenge" itself. Recall that our goal is to investigate the implications of that defeat (Section 1.2).

In Section 2.5.3, we argued that if a model achieves human-like accuracy on a Winograd dataset that satisfies the adversarial constraint, there are two possibilities. First, the model has achieved common sense, because it passes the Winograd Schema Challenge. Second, the Winograd Schema Challenge is flawed, because the model passes it without exhibiting common sense, which we might demonstrate using a different evaluation protocol or a different type of challenge.

The consensus in the literature appears to be that WSC is flawed and that we need new evaluation protocols and new challenges. For the rest of this section, we will review some of the relevant literature in light of the timeline in Table 2.4.

When the state-of-the-art on WSC273 was only 0.637 by Trinh and Le [97] with an ensemble of custom language models, Trichelair et al. [95] argued that "performing at a state-of-the-art level" on that dataset "does not necessarily imply strong commonsense reasoning" because models should also be consistent (Section 2.9.1).

While finetuned BERT-based models, including RoBERTa, were improving the state-of-the-art to 0.751 on WSC273 and 0.890 on the natural language inference test WNLI, McCoy et al. [62] argued that "targeted, challenging datasets... are important for determining whether models," including BERT, "are learning what they are intended to learn" from NLI tasks or just "right for the wrong reasons."

After achieving 0.901 accuracy on WSC273 by finetuning RoBERTa on their new WinoGrande training set, and in light of that result, Sakaguchi et al. [85] argued that "we are likely to be overestimating the true capabilities of machine commonsense" across a variety of benchmarks, and that "we now need AI algorithms to compose challenges that are hard enough for AI, which requires *dynamic* datasets that evolve together with the evolving state-of-the-art" (emphasis in original).

After finetuning T5 on WinoGrande to achieve a state-of-the-art accuracy of 0.846 on that still very recent test set, Lin et al. [55] argued that recent progress on WSC with RoBERTa and T5 "leaves us with two possible explanations: despite careful controls, the WinoGrande challenge *still* contains incidental biases that these more sophisticated models can exploit, or... we are genuinely making at least *some progress*

in commonsense reasoning" (emphasis in original); and that the latter possibility "challenges the notion that commonsense knowledge is (mostly) tacit."

At around the same time, Kocijan et al. [46] argued that models that have passed WSC "have not demonstrated either the ability to perform other natural language understanding tasks, or common sense." Therefore, "the commonsense reasoning and the natural language understanding communities require new tests, more probing than the Winograd Schema Challenge, but still easy to administer and evaluate."

When Brown et al. [6] achieved 0.883 accuracy on WSC273 without finetuning, the authors noted that the model's training data had in fact been "contaminated" by as many as 132 of 273 test questions, but that the model performs almost as well on "a 'clean' version which removes all potentially leaked examples." Still, any confirmed leak of a Winograd test set into a language model's training corpus is concerning.

While T5 models finetuned on WinoGrande and other data were improving the state-of-the-art on WinoGrande to 0.913, so that WSC273, WNLI, and WinoGrande were each above 90% accuracy, Elazar et al. [23] argued that "the apparent progress on [WSC] may not necessarily reflect progress in commonsense reasoning." They identified three possible explanations for that progress that have nothing to do with common sense, including "lax evaluation criteria," associativity and artifacts in the test sets, and "leakage" from supervised finetuning on specialized training sets.

Finally, in light of all the progress cited above, Kocijan et al. [47] argued that WSC is not "an adequate test of commonsense reasoning abilities." Although the challenge "as originally formulated has largely been overcome," common-sense knowledge and reasoning "still stands as one of the major challenges facing AI."

> The fact that large language models with one hundred billion parameters trained on half a trillion words can learn enough linguistic patterns that they can disambiguate the pronouns in the "trophy" sentence does not solve the larger problem reliably; it is not even guaranteed to be progress toward reliably solving the larger problem.

In short, pretrained transformer-based language models have been able to defeat WSC using what Levesque et al. [53] call "clever tricks involving word order or other features of words or groups of words" gleaned from some "large corpus of English text," as opposed to the desired "world knowledge and default reasoning abilities."

We mentioned new evaluation protocols for WSC, a topic to which we now turn.

2.9 Evaluating Models on Winograd Datasets

We mentioned in Section 2.8.4 that for models taking on a Winograd dataset, the basic *performance metric* is accuracy: quite simply, the share of problems answered correctly. The corresponding *evaluation protocol* is simply to measure accuracy.

Different performance metrics and evaluation protocols have been proposed for the Winograd Schema Challenge, not to mention other tests of common-sense knowledge and reasoning and even more general tasks in natural language processing.

2.9.1 Switchable Consistency

In the context of NLP, Elazar et al. [22] define *consistency* as "the ability to make consistent decisions in semantically equivalent contexts, reflecting a systematic ability to generalize in the face of language variability." Put another way, a model should provide the same answer to questions that require substantially the same reasoning, regardless of paraphrase or other inessential alteration. The type of consistency most relevant to WSC is "making consistent assignments in coreference resolution."

We will revisit the concept of consistency in reasoning in Section 2.9.5. For now, Trichelair et al. [95] have proposed a new evaluation protocol for the Winograd Schema Challenge based on a performance metric that we call *switchable consistency*.

A problem is said to be *switchable* if exchanging the candidates "does not obscure the sentence or affect the rationale to make the resolution decision," and moreover "the correct answer changes as well." This problem from WSC273 is switchable:

> Paul tried to call George on the phone, but **he** wasn't {successful}. (2.25)

That is, exchanging the candidates yields an equally valid new problem:

> George tried to call Paul on the phone, but **he** wasn't {successful}. (2.26)

Typically, problems where both candidates are personal names are switchable. This problem from WSC273 is also switchable, despite having no names at all:

> I poured water from the bottle into the cup until **it** was {empty}. (2.27)

On the other hand, the trophy-suitcase problems (2.8) are not switchable: one does not put suitcases into trophies, so this would obscure the sentence.

Using a team of three human annotators, Trichelair et al. [95] identify a total of 131 switchable problems in 65 schemas from WSC273 [94].

As for the performance metric, the authors define the *consistency score* of a model on WSC273 as "the percentage of predictions" by the model "that change (correctly) after candidates in the switchable subset are switched." Again, by definition, the answer to a switchable problem changes when the candidates are exchanged.

According to the new evaluation protocol, we measure a model's accuracy on the switchable subset of WSC273 both before and after transforming the problems by switching the candidates, and also "compute the corresponding consistency score."

Their protocol also measures a model's accuracy on the associative subset of WSC273 identified by the same authors (Section 2.7.1). They argue that *associative accuracy* is informative because "a model can be tailored to use statistical information about the entities themselves but perform poorly when this cannot be exploited."

A switchable problem cannot be associative (Section 2.5.3); that is, a candidate cannot be associated with the key word. Trichelair et al. [95] argue more generally that their evaluation protocol can help account for spurious correlations:

> A system that relies on the entity itself to make a prediction produces the same answer when the candidates are switched, even though it should not. Thus, a system that correctly resolves both the original and the switched sentence can be said more certainly to reason about the full sentence, instead of exploiting a statistical quirk of the participant entities.

The authors had a clear motivation to invent a new evaluation protocol for the Winograd Schema Challenge: the accuracies of state-of-the-art models at the time could be explained by chance, the null hypothesis. That is, one could argue that any given reported performance was not statistically significant:

> If one were to choose from a set of 10 random, binary classifiers, the best based on its performance on [WSC273], there is more than a 1-in-3 chance of scoring above 55% accuracy with this chosen classifier. As a result, achieving above random accuracy... does not necessarily correspond to capturing common sense; it could be the result of a lucky draw.

Today, statistical significance is less of a concern: we saw in Section 2.8.4 that language models have achieved accuracies over 90% on both WSC273 and the much larger WinoGrande test set (Table 2.4). We will discuss statistical significance in the context of evaluating model accuracy on Winograd datasets in Section 3.8.

Performance metrics based on various definitions of consistency remain a popular tool for reevaluating the performance of high-accuracy models. Emami et al. [25] appear to use the same definition as Trichelair et al. [95]: "We define the *consistency* score as the percentage of predictions that change from the original instances to the switched instances" (emphasis in original). It is easy to generalize the consistency score to transformations other than switching: Abdou et al. [1] use a comparable definition for consistency, which they call "stability," under various transformations, and Zhou et al. [109] also use a comparable definition for consistency under various transformations, which they call "dual test samples," including switching ("swap").

Measuring associative accuracy remains somewhat popular as well [1, 23, 81].

We will discuss the associative and switchable subsets again in Sections 3.4.1 and 3.4.2 respectively, and consistency again in Sections 3.7.2 and 3.7.3.

2.9.2 Group Scoring Metrics

Abdou et al. [1] introduce an evaluation protocol that, in addition to using consistency under various transformations (Section 2.9.1), also uses another performance metric: what they call "pair accuracy" and we call *schema accuracy*. We may refer to the usual accuracy metric as *problem accuracy* to distinguish it from schema accuracy.

Under schema accuracy, each Winograd schema, or pair of problems, "is treated as a single instance." That is, schema accuracy is "the number of pairs for which both examples in the pair are correctly answered divided by the total number of pairs."

Abdou et al. [1] argue that "this is an appropriate standard of evaluation" because, as we know, "WSC examples are constructed as minimally contrastive pairs."

> It is reasonable to suppose that for an answerer which truly "understands," being able to link the concepts [of the first key word and first candidate] in one of the resolutions is closely related and complementary to linking the concepts [of the second key word and second candidate] in the other.

It follows that a "large gap between [schema] accuracy and [problem] accuracy raises some doubts about the performance" of a model.

Ruan et al. [81] introduce another performance metric involving pairs of problems. Like the consistency metric, it is based on switching but can easily be generalized to other transformations. The authors define *consistent accuracy* as "the number of correctly answered pairs (i.e., correctly answered WSC sentences both before and after a switch) divided by the total number of switchable sentences."

Elazar et al. [22] generalize consistent accuracy to multiple transformations: to each problem, we associate a group consisting of its transformations; the consistent accuracy of a model is the share of groups of problems such that the model solves every problem in the group correctly. The authors note that this "combines the requirements" of consistency and accuracy but is "much stricter" than either one.

Elazar et al. [23] generalize schema accuracy and consistent accuracy to what they call "group scoring" and we call *worst-member accuracy*. Suppose we have a set of groups of problems; for example, a schema defines a group of problems, and, as we just explained, a transformation like switching does too. Further suppose we have a way of assigning a score to each problem; for example, we can assign a problem a score of 1 if a given model solves it and 0 otherwise. Under worst-member accuracy, each group is assigned the score of its lowest-scoring problem.

The authors argue that their metric is not only stricter but also "more robust," and that it not only "lowers the probability of random [correct] predictions" but also counters "the use of shallow heuristics." Moreover, they claim, a single low-scoring problem in a group "makes the success on other examples suspicious." However, they acknowledge that worst-member accuracy "does not solve the problem of artifacts" in "cases where all examples in a group can be solved based on [the same] artifact."

By *group scoring* we mean any performance metric based on groups of problems. It can involve groups within a single dataset, across multiple datasets, or both within and across datasets, assuming we conceive of transformations as mappings from one dataset to another. Figure 2.2 illustrates the three types of group scoring.

Consistency is a group scoring metric across datasets. Schema accuracy is a group scoring metric within a dataset. Consistent accuracy is a group scoring metric across datasets. Worst-member accuracy, which generalizes schema accuracy and consistent

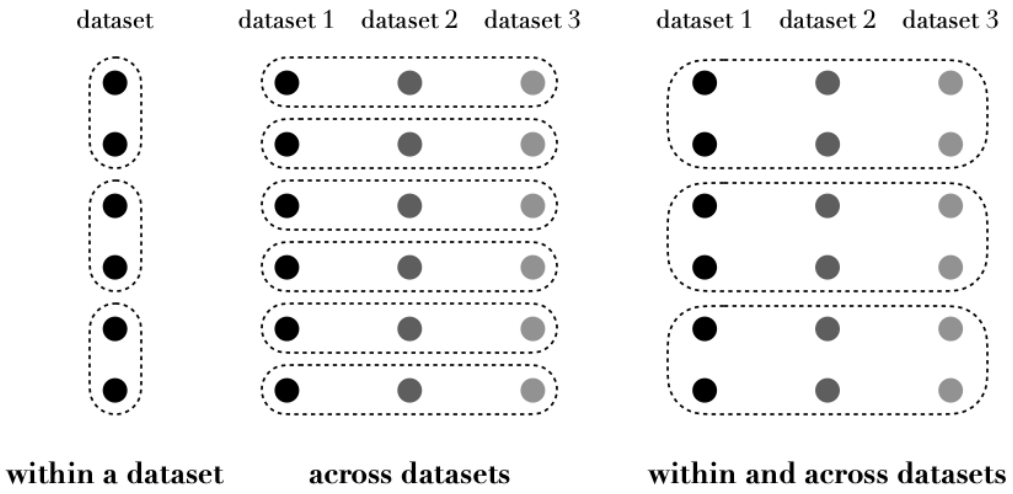

Figure 2.2: Three types of group scoring metric.

accuracy, is a group scoring metric that can be of any type.

We will discuss group scoring again in Sections 3.7.1 and 3.7.2.

2.9.3 Inverted Schemas

As we mentioned in Section 2.8.1, Elazar et al. [23] argue that the method of stochastic fill-in-the-blank is inherently "problematic" for a masked language model, such as RoBERTa, if the model converts the answer options into token sequences of different length: what we call *disparate answer tokenization* for short.

When the answer option token sequences for a problem are equal in length, we say that the problem has *equal-length answers*. In Chapter 4, we will often distinguish between results on the set of *all problems* from a given dataset, i.e., problems with *any-length answers*, and results on the subset of problems with equal-length answers.

For example, in the trophy-suitcase schema (2.8), suppose "trophy" is represented by a single token and "suitcase" by two tokens.[6] Elazar et al. [23] argue:

> In this scenario, the [masked language model] will see a single mask in one case (and estimate the probability of *trophy*), but in the other case, it will see two masks (assigning the *suit* and *case* probabilities). Since the model has access to the number of tokens it has to complete, the

[6]This does not actually occur with GPT-2, RoBERTa, or T5: each admits the token "␣suitcase."

comparison between these two options is flawed. [Emphasis in original.]

Note that the T5 model does not work in exactly this way: as we mentioned in Section 2.2.3, unlike BERT-based models, where a masking token represents a single token, for T5, a masking token stands for a span of one or more tokens.

Even for BERT-based models, it is not clear that disparate answer tokenization is actually "problematic" [23] for stochastic fill-in-the-blank. Of course a model "has access to" the lengths of the answer options in a multiple-choice problem. Even if it is "primed towards a certain answer" by the lengths, it is an empirical question whether or not this affects accuracy, which we will address in Chapter 4.

In any case, Elazar et al. [23] propose an extraordinary and drastic solution to disparate answer tokenization. First, they consider restricting Winograd datasets to problems where each answer option is represented by a single token, but note that this would "result in filtering a great portion of the data." So, they instead direct their attention to the key word and fundamentally transform each schema as follows.

Consider the classic trophy-suitcase schema (2.8) and its problems (2.9), which we reproduce here, with our annotations, for convenience:

(2.28)
a. The trophy doesn't fit into the suitcase because **it** is too {large}.
   What is too large? (i) the trophy ✓ (ii) the suitcase
b. The trophy doesn't fit into the suitcase because **it** is too {small}.
   What is too small? (i) the trophy (ii) the suitcase ✓

Elazar et al. [23] propose *inverting* the schema (our term) by exchanging the roles of the key words and the candidates, so that the resulting problem statements differ only by a *candidate*, and the answer options are the original *key words*. This process produces two new cloze tests (fill-in-the-blank problems). In our example:

(2.29)
a. The trophy doesn't fit into the suitcase because {the trophy} is too ⟨ ⟩. What is missing? (i) large ✓ (ii) small
b. The trophy doesn't fit into the suitcase because {the suitcase} is too ⟨ ⟩. What is missing? (i) large (ii) small ✓

To emphasize the exchanged roles, we may refer to this presentation of a schema as its *solution by key*, as opposed to the usual *solution by answer*.

We offer a few general remarks on inversion. First, the resulting problems do not belong to any Winograd schema: they are not pronoun disambiguation problems, nor can they be recast as such in any obvious way. Although the candidates of (2.28) have become the key words of (2.29), the key words of (2.28), which have become the answers of (2.29), generally do not occur in the problem statement as candidates.

Second, inversion acts only on schemas, not on individual problems: there is no way to meaningfully associate, e.g., (2.28a) with either (2.29a) or (2.29b). Indeed, both problems from (2.29) include an answer, "small," that is nowhere in (2.28a).

As for the argument in Elazar et al. [23], there are a few more points to note. First, inversion does not completely alleviate the supposed problem of disparate answer tokenization: the authors report that "occasionally" a key word "gets tokenized into multiple tokens" anyway, and they have to discard those problems.

Second, it is not clear how inversion affects the accuracy of language models, masked or otherwise. The authors perform some experiments and report "higher performance" by RoBERTa on an inverted WSC273 after finetuning on an inverted WinoGrande, compared to normal WSC273 after finetuning on normal WinoGrande. Again, this is an empirical question, which we will address in Chapter 4.

Third, it is not clear why inversion should result in a more appropriate test of common-sense reasoning. The authors note that the inverted pseudo-schema "is not faithful to the original [Winograd schema], and tests a different mechanism." They do not report the consistency of their models with respect to inversion.

Fourth, the authors seem to imply that every Winograd schema is invertible in this way. We will discuss this topic again in Section 3.4.3.

2.9.4 Control Baselines for Associativity

Recall, from Section 2.5.3, that Elazar et al. [23] propose, for any Winograd dataset, two systematic modifications on which a model that is free from spurious correlations is "likely to achieve random performance." To the extent that a model performs better than random guessing on these transformed *control baselines*, that model is likely to exhibit spurious correlations from associativity and artifacts in the dataset.

We will restrict our attention to the first baseline, as it is the more precise and generalizable of the two. For the *no-candidate baseline*, we remove the candidates

from every problem. For example, the beloved trophy-suitcase schema (2.8) becomes the following more-or-less unsolvable pseudo-schema with the same answers (*sic*):

Doesn't fit into because **it** is too {large, small}.

That is, we expect such a pseudo-schema not to be solvable at a rate very much better than chance by the intended method of common-sense knowledge and reasoning. Granted, "doesn't fit" and "too small" might suggest "the suitcase," as it is more container-like: perhaps this example is slightly associative after all.

On the other hand, the notoriously associative schema (2.11) becomes (*sic*):

Zoomed by because **it** was going so {fast, slow}.

Surely we would not be surprised if a language model achieved better than random accuracy when the key word is "fast" (in which case the answer is "the race car").

2.9.5 Transformations and Perturbations

We described *transformations*, i.e., mappings from one dataset to another, in each of Sections 2.9.1 through 2.9.4. We can think of switching, inversion, and the no-candidate baseline as different transformations of a Winograd dataset or a subset thereof. The result of applying a transformation is a new dataset called a *perturbation*.

Switching and removing candidates act on individual problems; inversion, as we have seen (Section 2.9.3), acts only on schemas. Figure 2.3 illustrates the concept.

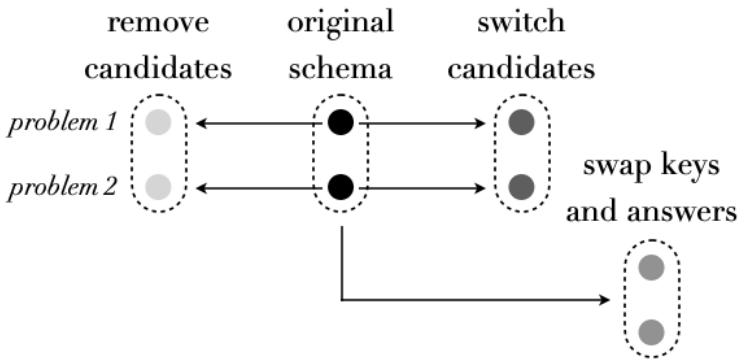

Figure 2.3: Three transformations: switching, inversion, no candidates.

The discussion of consistency at the start of Section 2.9.1 referred to "semantically equivalent contexts" [22] and questions that require substantially the same reasoning.

We will say more about this topic in Section 3.5.1, but for now, we define the *content* of a Winograd schema as the way in which a respondent is intended to solve it: some area of general knowledge or form of default reasoning underlying its problems.

To the extent that a transformation preserves the content of schemas, we call it *content-preserving*, and we call the perturbation *substantially equivalent* to the original dataset. To the extent that a model is consistent, we should expect it to achieve similar accuracies on the original dataset and any substantially equivalent perturbation, as well as a high consistency score between them.

When Trichelair et al. [95] measure the switchable consistency of language models on WSC273, the transformation is clearly designed to be content-preserving. When Abdou et al. [1] test pretrained models for "robustness... to semantic, syntactic, and lexical variation" on WSC285, each of their "variations and perturbations," e.g., changing from past tense to present tense, is designed to be substantially equivalent to the original dataset. When Zhou et al. [109] measure the consistency of pretrained models on WS273, each of their "dual test samples" is designed to "test the same commonsense knowledge" as the original dataset. When Elazar et al. [22] measure the consistency of pretrained models on cloze tests of factual knowledge, their "meaning-preserving alternations" are also designed to preserve the content of the cloze test.

Although Elazar et al. [23] note that the inverted perturbation "tests a different mechanism," we might reasonably expect a model with common-sense knowledge and reasoning to exhibit high consistency with respect to inversion. On the other hand, the no-candidate perturbation is surely not substantially equivalent, and we do not expect high consistency on any dataset with appropriately low associativity.

Measuring the accuracy and consistency of a model under transformations such as switching and inversion could be seen as a form of *adversarial evaluation protocol*. Such protocols "usually focus on a specific trained model, starting from an example that the model classifies correctly, and perturbing it in ways that, under the normative definition of the task, should not affect the [prediction]" [56].

We will discuss transformations and perturbations again in Section 3.7.2, and incorporate schema-based group scoring metrics into that discussion in Section 3.7.3.

2.9.6 Finetuning and Evaluation

In the literature, pretrained language models that are applied to Winograd test sets, with the notable exception of the GPT series [5, 74, 76], are typically finetuned on Winograd training sets and possibly other data (e.g., Table 2.4 in Section 2.8.4).

Still, pretrained language models other than the GPT series have certainly been applied without finetuning: as knowledge bases [73], for example.

The literature on consistency in language models sometimes focuses on pretrained models [22, 109], but it also uses finetuning where appropriate [1, 23].

Elazar et al. [22] argue that "it is important to measure and improve consistency" in pretrained models because much of what they learn "propagates" to every finetuned model. Along the same lines, Li et al. [54] argue that it may be appropriate to "focus on" pretrained models in part because "any deficiencies in their commonsense understanding can... adversely manifest in downstream applications."

To achieve maximal accuracy on a particular test set, finetuning may be the best approach. To measure consistency, to detect and account for associativity and artifacts in a dataset, or indeed to evaluate common-sense reasoning in pretrained language models, it may not be. Moreover, whether or not finetuning is appropriate in general, finetuning for WSC on specialized Winograd datasets is questionable.

Linzen [56] criticizes the dominant pretraining-finetuning-evaluating "paradigm" in several ways, arguing in particular that "we should not fine-tune our models on the evaluation benchmark," and that test sets "should be derived [from] expert-created controlled data sets," not "samples from the same distribution as the fine-tuning set."

Similarly, as we said in Section 2.8.4, McCoy et al. [62] argue that "targeted, challenging datasets... are important for determining whether models are learning what they are intended to learn" from NLI tasks, or just "right for the wrong reasons."

Kocijan et al. [47] note that "there is no point in training an AI program to solve Winograd schemas specifically," as "the point of the Winograd Schema Challenge is to test programs that claim to have solved the problem of pronoun reference resolution."

As the title of this thesis indicates, we decided not to finetune the pretrained language models under consideration. With that decision, our review of background material and related work is complete, and we can move on to methodology.

Chapter 3

Methodology

In Chapter 2, we reviewed the necessary background material and related work on pretrained language models and the Winograd Schema Challenge.

In this chapter, we explain how we approached our research problem and achieved our main objectives, which we stated in Section 1.2. In short, our goal is to investigate the implications of the defeat of the Winograd Schema Challenge. Have language models achieved common sense, or is the challenge flawed in some way? How can we tell? We identified two approaches: to apply to the models a new evaluation protocol based in part on consistency under perturbations, and to apply the models to a new test of common-sense reasoning based on a generalization of the Winograd schema.

In Section 3.1, we implement stochastic fill-in-the-blank for three language models so we can evaluate their common-sense reasoning capabilities on various datasets. In Section 3.2, we describe our preprocessing of WSC273. In Section 3.3, we explain how we modified that dataset to create WSC266, whose schemas all have the same standard format, which makes it easier to systematically perturb the dataset, and in Section 3.4, we describe some important perturbations of the modified dataset.

In Section 3.5, we define adversarial schemas, which generalize Winograd schemas. In Section 3.6, we describe a new test of common-sense reasoning based on adversarial schemas, and in Section 3.7 we discuss evaluation protocols for adversarial schemas.

Finally, in Section 3.8, we end on a brief discussion of statistical significance in the context of evaluating language models on adversarial and Winograd datasets.

3.1 Implementing Stochastic Fill-in-the-Blank

In Section 2.8.1, we described stochastic fill-in-the-blank, a family of methods for implementing common-sense reasoning by language models. The idea is to use the language model as an oracle or black box, but the details of the implementation depend, in general, on how the model delivers its answers.

In Section 2.2, we described the basic capabilities of three language models. Here, we explain how we implemented stochastic fill-in-the-blank for each of them.

3.1.1 Implementation by GPT-2

We described the GPT-2 model from the standpoint of a user in Section 2.2.1, and the literature on its relevant language-modelling capabilities in Section 2.8.2.

Suppose we are given the problem from (2.9b) = (2.12). Our goal is to assign, to each of the following sentences, a score representing its relative likelihood:

$$\begin{array}{l}\text{The trophy doesn't fit into the suitcase because } \textit{the trophy} \text{ is too small.}\\ \text{The trophy doesn't fit into the suitcase because } \textit{the suitcase} \text{ is too small.}\end{array} \tag{3.1}$$

Note that our annotations, e.g., italics, are never part of the model's input.

Consider the first sentence from (3.1). GPT-2 converts the string into a sequence of 15 tokens, where the substituted answer, "the trophy," appears at indices 10–11:

$$\begin{aligned}\sigma &= \sigma_1\sigma_2\cdots(\sigma_{10}\sigma_{11})\sigma_{12}\sigma_{13}\sigma_{14}\sigma_{15}\\ &= t_{464}t_{16383}\cdots(t_{262}t_{16383})t_{318}t_{1165}t_{1402}t_{13}\\ &= \langle \texttt{The}, \texttt{␣trophy}, \ldots, \texttt{␣the}, \texttt{␣trophy}, \texttt{␣is}, \texttt{␣too}, \texttt{␣small}, \texttt{.}\rangle\end{aligned} \tag{3.2}$$

Note that "The" (capitalized) and "␣the" (with a space) are tokenized differently.

From the output of the model on σ, specifically the table of logits it provides, we can calculate (as in Section 2.2.1) the following conditional probability[1]:

$$p_{\text{partial}} = \mathbb{P}(\sigma_{12}\cdots\sigma_{15} = t_{318}\cdots t_{13} \mid \sigma_1\cdots\sigma_{11} = t_{464}\cdots t_{16383}). \tag{3.3}$$

That implements *partial scoring* [98]. We can also calculate

$$p_{2+} = \mathbb{P}(\sigma_2\cdots\sigma_{15} = t_{16383}\cdots t_{13} \mid \sigma_1 = t_{464}). \tag{3.4}$$

If we had $p_1 = \mathbb{P}(\sigma_1 = t_{464})$, then multiplying that by p_{2+} would yield

$$p_{\text{full}} = p_1 p_{2+}, \tag{3.5}$$

the probability of the complete sentence, as required by *full scoring* [98]. But the probability of any single token occurring on its own is not provided by GPT-2.

[1] Once again, strictly speaking, no such probability is defined.

Of course, we only really need to determine which scores higher, x or y, and both sentences start with the same token: multiplying two conditional probabilities of the form (3.4) by the same positive number does not change which is higher.

On the other hand, it is possible, in general, for the first word in a problem statement to be the target pronoun, in which case the sentences to compare might differ in the first index. Then we would need the probability of the first token to apply full scoring. We also need that probability to apply *normalized full scoring* [98].

The obvious solution is to estimate the probability of each token using that token's frequency in the original training dataset for GPT-2, which is called WebText and consists of about eight million "documents" [76] (Section 2.2.1). However, WebText has never been released in its entirety, which is why, for example, Liu et al. [57] had to train their model, RoBERTa, on "an open-source recreation of the WebText corpus," among other training datasets (Section 2.2.2). Fortunately, Radford et al. [77] have released a large subset of WebText consisting of 250 thousand documents.

We calculated the frequency of each token in the subset, after appending to each document the model's special end-of-text token, t_{50256}. That token was reportedly appended in the same way during GPT-2's original training [2].

With an estimate of the probability of each token, we are able to apply, not only partial scoring, but full scoring, even when the token sequences to be compared differ in the first index, and normalized full scoring, about which we have more to say.

For normalized full scoring, we are supposed to divide the full score (3.5) by the frequency, or equivalently the probability, of the substituted "word" [98]. That is, as we explained in Section 2.8.1, Trinh and Le [98] assume that each answer is a single word, which is not always the case for WSC273 (Section 3.3.5). The authors are able to reduce "the trophy" and "the suitcase" to one word each by fixing the word "the," which they have in common, but this method does not work in general, and even when it does work, GPT-2 may convert a single word into multiple tokens.

To continue our example, "the trophy" appears at indices 10–11 as two tokens: $t_{262}t_{16383}$. To implement normalized full scoring, we divide the full score by an estimate of the probability of the entire answer outside the context of the problem:

$$p_{\text{norm}} = \mathbb{P}(\sigma_2 = t_{16383} \mid \sigma_1 = t_{262}) \cdot \mathbb{P}(\sigma_1 = t_{262}), \tag{3.6}$$

where, as above, the conditional probability is provided by the model and the other

factor is estimated using the frequency of t_{262}. That is, we define

$$p_{\text{full norm}} = p_{\text{full}}/p_{\text{norm}} = p_1 p_{2+}/p_{\text{norm}}. \tag{3.7}$$

This method is obvious, but we have not seen it described in the literature.

Recall p_{2+} (3.4), which is conditional on σ_1. The model's *loss* on σ is given by

$$v_{\text{loss}} = -\log p_{2+}/(|\sigma| - 1) \qquad \text{for } |\sigma| > 1, \tag{3.8}$$

which is "the average negative log probability per canonical prediction unit" [76]. A lower-scoring sequence is preferred by the model as more likely or plausible.

For the sake of consistency, and without affecting the results of stochastic fill-in-the-blank, we define the *partial score* of σ, not as p_{partial} (3.3), but as

$$v_{\text{partial}} = -\log p_{\text{partial}}. \tag{3.9}$$

We define the *full score* of σ, not as p_{full} (3.5), but as

$$v_{\text{full}} = -\log p_{\text{full}} = -(\log p_1 + \log p_{2+}). \tag{3.10}$$

We define the *normalized full score* of σ, not as $p_{\text{full norm}}$ (3.7), but as

$$v_{\text{full norm}} = -\log p_{\text{full norm}} = -(\log p_{\text{full}} - \log p_{\text{norm}}). \tag{3.11}$$

In all cases, a lower-scoring sequence is preferred by the model.

We can extend the scoring methods from Trinh and Le [98] in a few ways. First, inspired by the loss function (3.8), which is a mean, we can divide the full score by the number of tokens it predicts, namely $|\sigma|$, to get the *mean full score*,

$$v_{\text{mean full}} = v_{\text{full}}/|\sigma|. \tag{3.12}$$

Next, let $\widehat{\sigma}$ denote the suffix of σ after the last token of the substituted answer. We can divide the partial score by the number of tokens it predicts, namely $|\widehat{\sigma}|$, to get the *mean partial score*,

$$v_{\text{mean partial}} = v_{\text{partial}}/|\widehat{\sigma}|. \tag{3.13}$$

Finally, and somewhat questionably, let $\widetilde{\sigma}$ denote the subsequence of σ that excludes the substituted answer. We can divide the normalized full score by $|\widetilde{\sigma}|$, calling that the number of token it predicts, to get the *mean normalized full score*,

$$v_{\text{mean full norm}} = v_{\text{full norm}}/|\widetilde{\sigma}|. \tag{3.14}$$

That is, we consider only the non-answer tokens to be predicted under this scoring method, since we normalized away all the answer tokens.

If we choose not to estimate the probability p_1 of the first token, we get, instead of the full score, the *all-but-first score*,

$$v_{\text{all but first}} = -\log p_{2+} \qquad \text{for } |\sigma| > 1. \tag{3.15}$$

In light of this, we can refer to the loss (3.8) as the *mean all-but-first score*.

Unless the problem statement happens to start with the target pronoun, full scoring and all-but-first scoring will favour the same answer option, and mean full scoring and loss or mean all-but-first scoring will favour the same answer option.

Trinh and Le [98] found that partial scoring "outperforms" both full scoring and normalized full scoring "by a large margin" for their custom language model on WSC273. Similarly, Radford et al. [76] found that partial scoring is more accurate than full scoring for GPT-2 on WSC273. We would like to replicate their results, but we also have reason to believe that partial scoring is inappropriate for some Winograd schemas. Consider the following schema from WSC273:

> The path to the lake was blocked, so we couldn't {use, reach} **it**. (3.16)

There is only one token after the substituted answer: a period, t_{13}. Therefore, partial scoring would ask the model to estimate the likelihood of a period, conditional on the rest of the sentence. This phenomenon occurs in 20 out of 273 problems.

To address this potential issue, in WSC273 or any other dataset, we propose what we call smart scoring, under which we switch from partial to full scoring if the target pronoun occurs toward the end of the problem statement, which can be verified algorithmically. To be precise, under *smart scoring at limit* n for some $n \geq 0$, we apply partial scoring unless the target pronoun is followed by at most n tokens, in which case we apply full scoring. Every problem statement ends in punctuation, so a limit of 0 is irrelevant. The canonical example is smart scoring at a limit of 1.

Since we have mean versions of both full and partial scoring, (3.12) and (3.13) respectively, we also have a notion of *mean smart scoring* at any limit.

To sum up, with GPT-2, we can implement full scoring, normalized full scoring, partial scoring, all-but-first scoring, and smart scoring at any limit, as well as the means of all of the above, with the model's loss being mean all-but-first scoring.

Of course, it is an empirical question whether or not any of these innovations are worthwhile, and we will address that in Chapter 4.

3.1.2 Implementation by RoBERTa

We described the RoBERTa model from the standpoint of a user in Section 2.2.2.

We mentioned in Section 2.8.2 that RoBERTa's pretraining objective, masked language modelling, seems ideal for stochastic fill-in-the-blank, and we can use loss-based scoring without modification. Recall, though, that a masking token always represents one token to predict: as Elazar et al. [23] note, and as we discussed in Section 2.9.3, "the model has access to the number of tokens it has to complete."

Suppose we are again given the problem from (2.9b) = (2.12). Consider the first sentence from (3.1). RoBERTa can tokenize the following string:

The trophy doesn't fit into the suitcase because [mask] [mask] is too small. (3.17)

Here, we have replaced the target pronoun "␣it," which is represented by one token, by as many copies of the masking token as there are tokens in "␣the trophy," which happens to be two. The model will assign a score to "␣the trophy," or rather to its sequence of tokens, as the predicted label for the input σ.

To be precise, if i_1, i_2 are the positions of the masking tokens in σ_1 and "␣the trophy" tokenizes as $\tau_1\tau_2$, then for $k = 1, 2$, we get the probability p_k° of observing τ_k in position i_k given all the other tokens from σ, including one other masking token. The loss reported by the model is given by the negative mean

$$-(\log p_1^\circ + \log p_2^\circ)/2. \tag{3.18}$$

Similarly, the model will tokenize a version y of the problem statement with enough masking tokens to accommodate "␣the suitcase," which happens to be two again. Then we can compare the scores assigned by the model to the answers.

In general, we have a sequence σ representing a masked problem statement, where the masking token occurs at indices $i_1 < \cdots < i_N$. We also have a sequence τ representing an answer, where $|\tau| = N$. For $k = 1, \ldots, N$, we get the probability p_k° of observing τ_k in position i_k given σ_i, $i \neq i_k$. The model's *loss* on σ is given by

$$v_{\text{loss}} = -(\log p_1^\circ + \cdots + \log p_N^\circ)/N. \tag{3.19}$$

Again, a lower-scoring sequence is preferred by the model.

In imitation of the various scoring methods for GPT-2 (Section 3.1.1), which have mean and non-mean versions, we define the *multi-mask score* of σ as

$$v_{\text{multi-mask}} = -(\log p_1^\circ + \cdots + \log p_N^\circ). \tag{3.20}$$

In light of this, we can refer to the loss (3.19) as the *mean multi-mask score*.

Other scoring methods effectively repurpose a masked language model to assign probabilities to sentences in a way that is similar to GPT-2 but bidirectional. First, Salazar et al. [86, 87] propose "pseudo-log-likelihood scores." Unfortunately, that term actually describes every scoring method in this thesis, for every model, so we will use the term *single-mask statement score* instead, or *statement score* for short.

Take the first sentence from (3.1) as written, without masking:

The trophy doesn't fit into the suitcase because *the trophy* is too small. (3.21)

Suppose it tokenizes as $\sigma = \sigma_1 \cdots \sigma_n$. For $i = 1, \ldots, n$, we replace σ_i by a masking token and have the model provide the probability p_i^* of observing σ_i in position i given every other σ_j, $j \neq i$. The (single-mask) statement score is given by

$$v_{\text{statement}} = -(\log p_1^* + \cdots + \log p_n^*). \tag{3.22}$$

Second, Zhou et al. [108, 109], apparently independently of Salazar et al. [86, 87], propose an unnamed "score" that we will call the *mean statement score*:

$$v_{\text{mean statement}} = v_{\text{statement}}/n. \tag{3.23}$$

We mentioned in Section 2.2.2 that, before tokenizing any string, RoBERTa places it between special beginning-of-string and end-of-string tokens. The special tokens do not contribute to the statement score, since the model assigned them log-probability zero (probability one) in every experiment we ran. However, they do contribute to the total number of tokens n, which affects mean statement scoring. Discounting the special tokens did not seem to improve accuracy, so we kept them in.

Statement scoring requires as many model evaluations per sentence as there are tokens in the sentence. We propose a less expensive version of the statement score, which we will call the *single-mask answer score*, or *answer score* for short.

Again, suppose (3.21) tokenizes as $\sigma = \sigma_1 \cdots \sigma_n$. Suppose the answer option, in this case "the trophy," occurs at indices $i_1, \ldots, i_N$. For $k = 1, \ldots, N$, we replace σ_{i_k} by a masking token and have the model provide the probability $p^*_{i_k}$ of observing σ_{i_k} in position i_k given σ_j, $j \neq i_k$. The (single-mask) answer score is given by

$$v_{\text{masked answer}} = -(\log p^*_{i_1} + \cdots + \log p^*_{i_N}). \tag{3.24}$$

That is, we only mask answer tokens, summing over $i_1, \ldots, i_N$ instead of $1, \ldots, n$.

Of course, the *mean answer score* is given by

$$v_{\text{mean masked answer}} = v_{\text{masked answer}}/N. \tag{3.25}$$

To sum up, with RoBERTa, we can implement multi-mask scoring, (single-mask) statement scoring, and (single-mask) answer scoring, as well as the means of all of the above, with the model's loss being mean multi-mask scoring.

Again, it is an empirical question whether or not any of these scoring methods are accurate or even appropriate, and we will address that in Chapter 4.

3.1.3 Implementation by T5

We described the T5 model from the standpoint of a user in Section 2.2.3.

Again, T5's pretraining objective, masked language modelling, seems ideal for stochastic fill-in-the-blank, and we can use loss-based scoring without modification. For T5, unlike RoBERTa, a masking token stands for a span of one or more tokens.

Once again, take the problem from (2.9b) = (2.12) and the sentences from (3.1). To prepare the model's input, we replace the target pronoun by a single masking token. We can also, optionally, prepend a special task prefix for Winograd problems. That is, we tokenize, say as $\sigma = \sigma_1 \cdots \sigma_n$, either s or "wsc:␣" + s, where

s = "The trophy doesn't fit into the suitcase because [mask$_1$] is too small." (3.26)

To prepare the model's labels, we tokenize "[mask$_1$] the trophy [mask$_2$]" and "[mask$_1$] the suitcase [mask$_2$]." That is, each answer gets sandwiched between a pair of distinct masking tokens. Consider the tokenized first label, say $\tau = \tau_1 \cdots \tau_m$.

For $j = 1, \ldots, m$, we get the probability p°_j of producing τ_j in position j given σ as input. We mentioned in Section 2.2.3 that the two masking tokens are predicted

as well. Also, before tokenizing any string, T5 appends a special end-of-string token. Unlike with RoBERTa, the end-of-string token does contribute to the masked score: T5 typically assigns it a low probability, at least in the experiments we ran.

The *loss* reported by the model for σ and τ is given by the negative mean

$$v_{\text{loss}} = -(\log p_1^\circ + \cdots + \log p_m^\circ)/m. \tag{3.27}$$

As always, a lower-scoring sequence is preferred by the model.

In imitation of the scoring methods for GPT-2 (Section 3.1.1) and RoBERTa (Section 3.1.2), we define the *masked score* as

$$v_{\text{masked}} = -(\log p_1^\circ + \cdots + \log p_m^\circ). \tag{3.28}$$

In light of this, we can refer to the loss (3.27) as the *mean masked score*.

Masked scoring admits other boolean parameters apart from taking or not taking the mean over all predicted tokens. First, as stated above, we can apply or not apply a task prefix. Second, we can count or not count the end-of-string token (EOS) toward the sum of log probabilities and the number m of predicted tokens. That is,

$$v_{\text{masked}}(\text{mean/no mean},\ \text{task/no task},\ \text{EOS/no EOS}) \tag{3.29}$$

is a *parametrized masked score*, with $v_{\text{loss}} = v_{\text{masked}}(\text{mean},\ \cdot,\ \text{EOS})$.

We include the task prefix parameter because, of course, the model was trained with one. We include the end-of-string parameter because T5 assigns that token such a low probability in our experiments that it can profoundly affect the loss. For example, if EOS dominates the sum (3.28) and we take the mean, then a longer sequence will always be preferred: dividing by a larger number yields a lower score.

Once again, we will evaluate masked scoring parameters empirically in Chapter 4.

3.2 Preprocessing WSC273

We have described the manually constructed test set WSC273 (Section 2.7.1) as the most important Winograd dataset for our purposes. Certainly it is the most studied Winograd dataset in the literature, despite the recent dominance of the WinoGrande test set among reported results (e.g., Table 2.4 in Section 2.8.4). Indeed, WSC273 is often referred to simply as "the Winograd Schema Challenge."

Granted, WSC273 is flawed, but so is every other Winograd dataset (Section 2.7). At least WSC273 is amenable to manual review due to its small size, and we will carry out such a review in detail in Section 3.3. To put it simply, if the leading datasets are WSC273 and WinoGrande, we choose the manually constructed test set over the crowdsourced one for the former's manageable size and better-written problems.

Our intention was to use WSC273 exactly as it appears in the original XML-formatted file published by Davis [14]. However, in order to properly apply the method of stochastic fill-in-the-blank as described in Section 3.1, the dataset does require some preprocessing, which could affect accuracy. Typically, papers that report results on WSC273 do not explain whether or how the dataset was preprocessed, but our own methods are given here for completeness. In the following, we will identify any problem from WSC273 by its index, from 1 to 273, in the original file.

Table 3.1 summarizes our preprocessing of WSC273 that can affect tokenization and therefore stochastic fill-in-the-blank, with discussion to follow.

Type of Preprocessing	No. Schemas	No. Problems
remove exterior whitespace	136	273
detect punctuation after pronoun	9	18
remove interior line breaks	32	64
remove interior double spaces	2	4
make answers lowercase	50	100
make answers possessive	13	26

Table 3.1: Preprocessing the WSC273 dataset of 136 schemas and 273 problems.

Each problem includes the following strings: the problem statement before the target pronoun, s_0, and after the target pronoun, s_1, and the target pronoun itself, p. The problem also includes answer strings s_{A} and s_{B}, as well as the identity of the correct answer. For example, s_0 lies between a pair of special XML tags.

By *whitespace* we mean spaces and line breaks. The problems use whitespace very inconsistently. For example, s_0 or s_1 may start or end with a line break, a space, both, or neither; and p may start or end with one space, two spaces, or no spaces, and its whitespace is not always consistent with the whitespace around s_0 and s_1.

Since whitespace can affect the tokenizers we used (Section 2.2), we have removed the exterior whitespace from every string. Of course, if we want to reconstruct the problem statement, we need to separate the pronoun p from s_0 and s_1 with spaces,

except in some problems (e.g., 41), where s_2 starts with punctuation, e.g., a period.

Not counting exterior whitespace, some problems (e.g., 121) contain line breaks, and some (e.g., 115) contain double spaces. We have removed all of these.

The problems capitalize inconsistently, which could affect the tokenizers we used. Based on the position of the pronoun, e.g., following a period, some problems (e.g., 53) should capitalize their answers so that they can be properly substituted for the pronoun, and those problems do so. However, other problems (e.g., 1) incorrectly capitalize one or both of their answers, which should be made lowercase.

Finally, some problems (e.g., 209) have a possessive target pronoun, specifically "his" or "her." Of course, the latter is not necessarily possessive in general, but it happens to always be possessive in WSC273. For those problems, the answers should also be possessive (e.g., "Emma's" rather than "Emma" in 209) so that they can be properly substituted for the pronoun. None of those answers are in that form, and we change them accordingly; of course, this will affect how an answer is tokenized.

3.3 Creating WSC266 from WSC273

The preprocessing described in Section 3.2 is enough to properly apply stochastic fill-in-the-blank to WSC273. However, the schemas still do not all have exactly the same format, which makes it difficult to systematically perturb the dataset.

For example, we know that a Winograd dataset, by definition, consists of a number of schemas and twice as many multiple-choice problems: every schema has two key words and one problem for each. WSC273 consists of 136 schemas and 273 problems. Note that 273 is not divisible by two. Indeed, one schema, or quasi-schema, has three problems. This affects the definitions of schema accuracy and other group scoring metrics (Section 2.9.2) and makes it impossible to invert (Section 2.9.3).

That is, for various reasons, not all the problems in WSC273 fall into Winograd schemas as defined by Levesque et al. [53], though we call them schemas anyway.

We checked every problem in WSC273, making changes where appropriate: the result is a dataset that we call *WSC266*, consisting of 133 schemas—true ones—and 266 problems. Removing three schemas and one additional problem accounts for the seven fewer problems. Table 3.2 summarizes the issues we identified, with sections to follow. We will identify any problem from WSC266 by its index in WSC273.

Issue with Schema	Section	No. Schemas	Change
extra problems	3.3.1	1	remove problem
candidates in key words	3.3.2	2	remove schema
missing candidates	3.3.3	1	remove schema
nested candidates	3.3.4	2	insert candidate
key words are not words	3.3.5	25	none (allow)
answer appearances	3.3.6	59	none (allow)
missing answer articles	3.3.7	2	change answer
spelling and grammar	3.3.8	6	correct problem

Table 3.2: Issues with WSC273 and changes made in WSC266.

First, though, it is worth noting that the original XML file for WSC273 does not explicitly identify schemas, key words, or candidates [14]. Fortunately, the dataset is small, which makes it amenable to manual review and analysis. Otherwise, to identify schemas, we would have to systematically compare sets of consecutive problems for similarities; to identify key words, we would have to systematically compare problem pairs from each schema for disparities; and to identify candidates, we would have to systematically perform coreference resolution (as discussed in Section 3.3.6).

For convenience, in the process of making WSC266, we identified the schemas and all the relevant elements of each problem, and we stored all of that information in a database management system we built for Winograd schemas.

3.3.1 Extra Problems

One schema has three problem statements [23]:

253. George got free tickets to the play, but he gave them to Eric, {even though} **he** was particularly eager to see it.
254. George got free tickets to the play, but he gave them to Eric, {because} **he** was {particularly eager}* to see it.
255. George got free tickets to the play, but he gave them to Eric, because **he** was {not particularly eager}* to see it.

(3.30)

If we intend to follow the original concept of the Winograd Schema Challenge [53], schemas with more than two problems are inappropriate: there are now two pairs of key words, denoted above by { } and { }*, and two out of three problems have the

same correct answer. That is, this seemingly minor issue completely changes the structure of a Winograd schema. We removed problem 255, leaving the other two.

3.3.2 Candidates in Key Words

In two schemas, the key word is or contains one of the candidates. As a result, the multiple-choice answer options differ between problems. For example:

> 266. I put {the butterfly wing} on the table and **it** broke.
> 267. I put {the heavy book} on the table and **it** broke. (3.31)

In the first problem, the candidates and answers are "the butterfly wing" and "the table," whereas in the second they are "the heavy book" and "the table."

Any such schema is disallowed by the definition. Moreover, the different answer options make the problems less comparable: is the butterfly wing as much more breakable by the table (compared to the reverse) as the table is by the book?

This phenomenon occurs twice in WSC273: the other pair of problems is 173, 174 [1]. For the sake of simplicity and consistency, we removed both schemas.

3.3.3 Missing Candidates

In one schema, one of the answers does not appear in the sentence, syntactically speaking. The answers to problems 247 and 248 are given as "Pam's parents" and "Pam and Paul," but the latter noun phrase nowhere to be found:

> Pam's parents came home and found her having sex with her boyfriend, Paul. **They** were {furious, embarrassed} about it. (3.32)

This oddity can make it more difficult to systematically perturb schemas; e.g., by changing the candidates. For the sake of simplicity, we removed the schema.

Granted, we could have rewritten the problems so they include the phrase "Pam and Paul," but it is not totally clear that the new problems would be substantially equivalent to the old ones: as written, the missing candidate can be assembled from other words in the problem, which might be part of the problem-solving process.

3.3.4 Nested Candidates

In problems 51 and 52, one candidate appears only inside the other candidate:

> Joe's uncle can still beat him at tennis, even though **he** is 30 years {younger, older}. (3.33)

Again, this oddity can make it more difficult to systematically perturb schemas. Here, though, an easy fix exists: we replaced "him" with "Joe," and made that the second candidate. The new problems appear to be substantially equivalent to the old ones.

Problems 209 and 210 also feature a nested candidate:

> Emma's mother had died long ago, and **her** {education had been managed, place had been taken} by an excellent woman as governess. (3.34)

We replaced "died" with "left Emma," and made that occurrence of "Emma" the second candidate. Again, the new problems appear to be substantially equivalent.

3.3.5 Key Words Are Not Words

Since, as we mentioned above, WSC273 does not explicitly identify key words, we have to compare sets of two or three consecutive similar problems and figure out how they differ. They do not always differ by one word. For example:

> 203. John hired { } Bill to take care of **him**.
> 204. John hired {himself out to} Bill to take care of **him**. (3.35)

We have annotated the minimal "key words" by which the problem statements differ: one of them is no word at all, an empty list; and the other is a list of three words. Of course, we could add "hired" on the left to both of them, in which case they are no longer minimal, one of them is a word, and the other is a list of four words.

Suppose we always choose the minimal word lists by which problem statements in the same schema differ. In twenty cases, at least one of those "key words" is a list of two or more words; in four cases, it is a list of three or more; in one case (209, 210), it is a list of four. In seven cases, one of the "key words" is an empty list. As for the intersection of those sets, in two cases, including (3.30) above, we have a list of two or more words as well as an empty list (the other case being 264, 265).

We allowed *key phrases*, by which we mean lists of words and punctuation symbols with any number of elements, by which two Winograd problem statements differ.

As minor as this issue may seem, allowing key phrases changes the definition of a Winograd schema from Section 2.4. For the record, Levesque et al. [53] typically refer to "a word" or "the special word" (singular), although they do once mention "sentences that differ only in one or two words," and some of the examples in their appendix use lists of as many as three words. Trinh and Le [98] refer to "a special word" or "the keyword," and they even introduce a method to detect its (single) position. Trichelair et al. [95] refer to the position of "the 'special' word" or "the hinge word." Elazar et al. [23] refer to "a special word," "the word that is different between the twin sentences," although they concede that "occasionally, there is more than one special word." On the other hand, Abdou et al. [1] refer to "the special discriminatory segment," which is a more accurate description.

As we pointed out in Section 2.5.3, the key word feature is a major ingredient in the Winograd schema as a test of common-sense reasoning. Levesque et al. [53] argue that "contexts where [one key word] can appear are statistically quite similar to those where [the other key word] can appear." That statement arguably becomes less plausible, in general, when we replace "key word" with "key phrase": there are just a lot of ways to try to fit a given list of words into different contexts. This is especially true because a Winograd schema does not require its key phrases to be phrases of the same type, or indeed phrases (i.e., grammatical units) at all.

In (3.35), for example, the minimal key phrases are (a) nothing, and (b) {himself out to}. The latter is a reflexive direct object for a verb, plus an adverb modifying the verb, plus a preposition missing its own direct object, which, if it were present, would be the verb's indirect object: that is a fairly specific puzzle piece to fit into a context. On the other hand, {hired} and {hired himself out to} are much more similar phrases—but we have no precise criterion for extending key phrases.

3.3.6 Answer Appearances

In 59 schemas, at least one answer, representing a candidate, does not exactly match the corresponding candidate that actually appears in the problem statement as a noun phrase, even if we ignore differences in capitalization. For example, the actual

trophy-suitcase schema from WSC273, unlike (2.8), modifies the suitcase (3):

> The trophy doesn't fit into the brown suitcase because **it** is too {large}.
> Answers: the trophy, the suitcase* (3.36)

The modifier "brown" does not appear in the second answer: we need to recognize that "the suitcase" refers to "the brown suitcase," which is not particularly difficult for English-speaking respondents. We also need to locate an implicit drawing (27):

> Sam's drawing was hung just above Tina's and **it** did look much better with another one {below} it. Answers: Sam's drawing, Tina's drawing* (3.37)

Here, the present participle "juggling" becomes the noun "juggler" (109):

> John was jogging through the park when he saw a man juggling watermelons. **He** was very {impressed}. Answers: John, the juggler* (3.38)

Sometimes, indefinite articles become definite articles, as is common in English (61):

> There is a pillar between me and the stage, and I can't see {around} **it**.
> Answers: the pillar*, the stage (3.39)

We can accept that the answers do not always exactly match their appearance in the problem statement. In general, this adds to a problem's difficulty, because it may not be trivial to match an answer to a candidate. Indeed, the task in question is coreference resolution (Section 2.6.1), which generalizes pronoun disambiguation.

The answer-candidate coreference resolution problems happen to be very easy, in almost every case, for a human respondent equipped with common sense and a reasonable degree of English fluency, but they cannot strictly be called trivial in the way that matching "the trophy" to "The trophy" (capitalized) is trivial.

We will return to this point briefly in Section 3.5.1.

3.3.7 Missing Answer Articles

We did change the answers of two problem pairs: specifically, by adding articles. The answers to problems 258 and 259 are given as "lemons" and "lemon trees," but they should be "the lemons" and "the lemon trees":

> I tried to paint a picture of an orchard, with lemons in the lemon trees, but **they** came out looking more like {light bulbs, telephone poles}. (3.40)

The same is true of the answer "coats" in problems 147 and 148.

3.3.8 Spelling and Grammar Mistakes

The word "received" is misspelled in problem 5, where it is the key word. The term "Game Boy" is misspelled in problems 229 and 230. The word "Kamchatka," a Russian peninsula, is misspelled in problems 171 and 172, where it is a candidate, though it is spelled correctly as an answer option [1]. We corrected all of the above.

Three problems (217, 243, 245) are missing final punctuation; in each case, the other problem in the schema ends in a period. We added a period to each of those problem statements to make them grammatical.

3.4 Subsets and Perturbations of WSC266

In Section 2.9.5, we mentioned several subsets and perturbations of WSC273. All of them, and a few others, apply at least as well to the standardized WSC266. And our database management system for that standard format (Section 3.3) makes it significantly easier to systematically perturb schemas.

3.4.1 Associative Subset

We mentioned in Section 2.7.1 that Trichelair et al. [95] have identified 37 associative problems in 26 schemas from WSC273 [93].

We consider a schema to be associative if either of its problems is associative. One of the associative schemas (266, 267) was removed from WSC266 because the key phrase contains one of the candidates (Section 3.3.2). That leaves WSC266 with 25 human-detected associative schemas comprising 50 problems.

When we present our results in Chapter 4, we will report separate values for the *associative* and *non-associative* subsets of WSC266 where appropriate, despite the small size of the former, which will affect statistical significance (Section 3.8).

3.4.2 Switchable Subset

In Section 2.9.1, we discussed Trichelair et al. [95]'s switchable subset of WSC273 and the corresponding perturbation: 65 schemas comprising 131 problems [94].

We identified a very slightly different switchable subset of WSC266: 70 schemas comprising 140 problems. Table 3.3 summarizes how our switchable subset and the corresponding perturbation differ from those of Trichelair et al. [95], apart from and in addition to how WSC266 differs from WSC273 in general (Table 3.2). Removing two schemas and adding seven accounts for the change in size.

Issue with Switchable Schema	No. Schemas	Change
correct answer does not change	2	remove schema
candidates broken by switching	3	correct schema
switched schema uncapitalized	118	correct schema
switchable but not included	7	add schema

Table 3.3: Issues with switchable subset of WSC273 and changes made for WSC266.

For example, problem 96 from the switchable subset of WSC273 [95] is not actually switchable, because the correct answer does not change:

> I saw Jim yelling at some guy in a military uniform with a huge red beard. I don't know {who} **he** was, but he looked very unhappy. (3.41)

We know "Jim" (by name, on sight), not "some guy." Therefore, regardless of who was yelling or looking very unhappy, we don't know who *some guy* was. For the same reason, problems 161 and 162 are not actually switchable. We removed those schemas from the switchable subset of WSC266.

In some schemas, at least one candidate appears to have been broken by switching. Clearly, in problem 108, the candidates are "a man" and "John":

> John was doing research in the library when he heard a man humming and whistling. **He** was very {annoying}. (3.42)

However, Trichelair et al. [95] switch it as follows:

> Man [*sic*] was doing research in the library when he heard a john [*sic*] humming and whistling. **He** was very {annoying}. (3.43)

The answers to problems 207 and 208 are given as "Goodman" and "Xenophanes," and we assume the candidates are as follows:

> Sam Goodman's biography of the Spartan general Xenophanes conveys a vivid sense of the difficulties **he** faced in his {research, childhood}. (3.44)

However, Trichelair et al. [95] offer the switched candidates "Sam [X]enophanes" and "the [S]partan general [G]oodman," presumably by switching the answers, which are substrings of the candidates. We think this may obscure the meaning.

Finally, the candidates in problems 15 and 16 are switched inconsistently:

> The man couldn't lift his son because **he** was so {weak, heavy}. (3.45)

We used "his son" and "the man," as given, whereas Trichelair et al. [95] use:

> The son couldn't lift the man because **he** was so {weak}.
> The son couldn't lift his man because **he** was so {heavy}. (3.46)

90% of Trichelair et al. [95]'s switched problems are not capitalized correctly: typically, only the first word is capitalized, regardless of proper nouns or punctuation. This can affect tokenization (Section 2.2), so we corrected this too.

Some schemas that were excluded from the switchable subset of WSC273 by Trichelair et al. [95] nevertheless seem switchable. For example, 127 and 128:

> Sara borrowed the book from the library because she needs it for an article she is working on. She {reads, writes} **it** when she gets home from work. (3.47)

Borrowing an article from a library doesn't seem obscure enough to exclude (e.g., Dalhousie University allows it). Similarly, 141 and 142:

> We went to the lake, because a shark had been seen at the ocean beach, so **it** was a {safer, dangerous} place to swim. (3.48)

Similarly, a shark at a lake doesn't seem obscure enough to exclude.

There is no point leaving out those schemas just to stay consistent with Trichelair et al. [95], as the switchable subsets differ anyway, so we mark them as switchable.

We can concede that foxes attack chickens, but not vice versa (155–158), although there is a news report from 2019 claiming just such an occurrence. We can concede that fish eat worms, but not vice versa (97, 98), although we read on Wikipedia that something called a Bobbit worm can and does. We can even concede that although dogs chase cats, which run up trees, either cats do not chase dogs or dogs do not run up trees (101, 102). Like Trichelair et al. [95], we exclude these schemas.

Other schemas are more debatable. Trichelair et al. [95] switch 231 and 232:

> The man lifted the boy onto **his** {shoulders, bunk bed}. (3.49)

If a boy can lift a man onto the boy's shoulders (say, as a show of strength) or onto the man's bunk bed (say, in an army barracks), then we can switch 85 and 86:

> If the con artist has [*sic*] succeeded in fooling Sam, **he** would have {gotten, lost} a lot of money. (3.50)

We can switch 93 and 94 as well:

> Alice tried frantically to stop her daughter from {chatting, barking} at the party, leaving us to wonder why **she** was behaving so strangely. (3.51)

Arguably, the switched problems would read a little better with a new candidate, "Alice's daughter," rather than "her daughter," but we kept the original candidates (and that one appears as "Alice's daughter" in the answer option anyway).

In the two debatable schemas above, and in three others like them, comprising problems 145, 146, and 167–170, switching the candidates seems to preserve "the rationale to make the resolution decision," and it doesn't appreciably "obscure the sentence," as Trichelair et al. [95] put it: the underlying logic is the same, although the specific factual circumstances may be slightly unusual. Therefore, we included a total of seven additional schemas in the switchable subset of WSC266.

3.4.3 Inverting the Winograd Schema

We discussed the inverted perturbation of WSC273 in Section 2.9.3.

We observed that the problems that result from inverting a Winograd schema, i.e., from exchanging the roles of the key phrases and the candidates, do not belong to any Winograd schema: they are not pronoun disambiguation problems. Nevertheless, the inverted problems do form a pair of cloze tests with their own key phrases, namely the candidates of the original schema.

It may appear that every Winograd schema is invertible. However, it is not necessarily true of an inverted Winograd schema that each key phrase actually makes a different answer far more likely, which is a basic property of the schema (Section 2.4).

To illustrate this unusual possibility, we present an original Winograd schema, though we do not claim it is a very good one for testing common sense:

a. The 2022 turkey cook-off chose a winner. **It** was {fair}.
   What was fair? (i) the cook-off ✓ (ii) the winner
b. The 2022 turkey cook-off chose a winner. **It** was {Julius Caesar}.
   What was Julius Caesar? (i) the cook-off (ii) the winner ✓ (3.52)

This seems to satisfy the basic properties of a Winograd schema: if anything was fair, it seems more likely that it was the competition than the winner; if Julius Caesar was involved, it seems more likely that he was the winner than the competition.

Inverting (3.52) yields, as usual, two cloze tests, but note the answers:

a. The 2022 turkey cook-off chose a winner. {The cook-off} was ⟨ ⟩.
   What is missing? (i) fair ✓ (ii) Julius Caesar
b. The 2022 turkey cook-off chose a winner. {The winner} was ⟨ ⟩.
   What is missing? (i) fair ✓! (ii) Julius Caesar (3.53)

If the winner was anything, it seems more likely that the winner was fair—fair-haired or fair to his competitors—than that the winner was a Roman general who wasn't alive in 2022 and wouldn't know how to cook New World poultry in any case.

The problem with inverting (3.52) is that although the first key phrase makes the first candidate far more likely, and although the second key phrase makes the second candidate far more likely, the second candidate does not make the second key phrase far more likely, because that key phrase is inherently implausible. We simply forced it into the problem statement of (3.52b), asserting: "It was Julius Caesar."

Let C_1 represent the first candidate appearing in the problem statement and C_2 the second candidate appearing there. Let K_1 represent the first key phrase appearing in the problem statement and K_2 the second key phrase appearing there. Assume all of those events have positive probability. Then a Winograd schema requires that

$$\text{(a)}\quad P(C_1|K_1) \gg P(C_2|K_1) \qquad \text{and} \qquad \text{(b)}\quad P(C_2|K_2) \gg P(C_1|K_2). \tag{3.54}$$

That is, each key phrase makes a different candidate far more likely. Now, we would like to assert, of the inverted schema, the corresponding inequalities:

$$\text{(a)}\quad P(K_1|C_1) \gg P(K_2|C_1) \qquad \text{and} \qquad \text{(b)}\quad P(K_2|C_2) \gg P(K_1|C_2). \tag{3.55}$$

Exchanging conditional probabilities is covered by *Bayes' theorem*, which states that

$$P(A|B) = P(B|A)P(A)/P(B) \qquad \text{for } P(B) > 0. \tag{3.56}$$

Unfortunately, (3.54) does not imply (3.55). Indeed, we can choose the factors in Bayes' theorem to satisfy (3.54) but violate, e.g., (3.55b), as we did in (3.52).

Basically, $P(K_2)$, the probability of Julius Caesar showing up at a 2022 turkey cook-off, is very low. It is so low that when we apply Bayes' theorem and write

$$P(K_2|C_2) = P(C_2|K_2)P(K_2)/P(C_2), \tag{3.57}$$

the vanishingly small factor $P(K_2)$ on the right-hand side makes $P(K_2|C_2)$ very small, which violates (3.55b). This is despite the fact that $P(C_2|K_2)$, which also appears on the right-hand side of (3.57), must be a relatively large probability by (3.54b).

Therefore, in theory, the cloze tests from an inverted Winograd schema may not admit definitive answers, and their answers may not depend on the new key phrases. However, it happens that every schema from WSC266 has a reasonably well behaved inversion, so we can measure each model's *invertible consistency* on that dataset.

We will discuss inversion once more in Section 3.5.5.

3.4.4 New Perturbations: Adjectival and Unbalanced

Here, we introduce new subsets and perturbations of WSC266, suitable for comparing accuracies and calculating consistency scores, that we have not seen described in the literature (Section 2.9.5). This is a little easier to do with the standardized format of WSC266, where, e.g., the answer options never differ within a schema (Section 3.3.2).

Recall that the switched perturbation (Section 3.4.2) affects only the candidates and, consequently, the answers, leaving the rest of the problem statement unchanged. Our own perturbations similarly affect only the candidates and answers.

First, we introduce the symmetrical subset and the adjectival perturbation. Recall that the trophy-suitcase schema (2.8) was not switchable because one does not put suitcases into trophies. However, there is nothing implausible about this version:

The red box doesn't fit into the blue box because **it** is too {large, small}. (3.58)

Suppose we replace the candidates in a Winograd schema with noun phrases that differ only by an adjective. If the result is still a Winograd schema that makes sense

and admits definitive answers, we call the schema *symmetrical* and the new version an *adjectival* schema. That is, symmetrical schemas admit adjectival perturbations.

We expect every switchable schema from WSC266 to be symmetrical; on the other hand, as we just saw, there are symmetrical schemas that are not switchable.

We identified 87 symmetrical schemas in WSC266 (compared to 70 switchable schemas) by constructing adjectival versions of them. That is, the symmetrical subset and the adjectival perturbation contain 174 problems each. The switchable schemas are indeed a proper subset of the symmetrical schemas.

The switched and adjectival perturbations are fairly similar. As we have seen, the switchable schemas represent about 80% of the symmetrical schemas. Moreover, the logic of the perturbations is similar: in both, the candidates are considered to be interchangeable. It is an empirical question whether models perform comparably well on these perturbations, and we will address that question in Chapter 4.

Next, we introduce the unbalanced perturbation. Elazar et al. [23] proposed the inverted perturbation as a solution to disparate answer tokenization, in which a model converts the answers into different numbers of tokens (Section 2.9.3). Suppose we want to put a practical upper bound on this phenomenon's effect on accuracy.

Under the *unbalanced* perturbation, we replace one of the candidates with a more complex and implausible version: more words, words that are more improbable, or both. By convention, we always replace the second candidate. Of course, the result needs to still be a Winograd schema that makes sense and admits definitive answers.

For example, the trophy-suitcase schema (2.8) might become:

> The trophy doesn't fit into the oblong burgundy alligator-hide suitcase because **it** is too {large, small}. (3.59)

The new answers to the schema are exactly the new candidates, up to an article: we are mostly spared from having to match answers to candidates (Section 3.3.6).

Every schema in WSC266 has an unbalanced version—infinitely many unbalanced versions, in fact—so our unbalanced perturbation has 266 problems.

The unbalanced perturbation is quite different in concept from either the switched or adjectival perturbation. Again, we will attempt to quantify that in Chapter 4.

3.5 Adversarial Schemas

In light of how we standardized the Winograd Schema Challenge, in the form of WSC273, in Section 3.3, we are now prepared to generalize the Winograd schema.

3.5.1 Beyond the Winograd Schema

We know that the Winograd Schema Challenge has been defeated (Section 2.8.4). The consensus in the literature appears to be that the test is flawed and that we need new common-sense reasoning challenges, although there is no consensus on the type of challenge. Kocijan et al. [46] argue generally that "the commonsense reasoning and the natural language understanding communities require new tests, more probing than the Winograd Schema Challenge, but still easy to administer and evaluate." According to Sakaguchi et al. [85], "we now need AI algorithms to compose challenges that are hard enough for AI, which requires *dynamic* datasets that evolve together with the evolving state-of-the-art" (emphasis in original). On the other hand, McCoy et al. [62] suggest "targeted, challenging datasets," and Linzen [56] calls for "expert-created controlled data sets," in various areas of natural language understanding.

We can add our own short wish list of test attributes: *ease of generation* and *ease of perturbation*. We know from Section 2.7 that generating new Winograd schemas is highly non-trivial; e.g., as Kocijan et al. [47] put it, "manually creating a large, diverse collection of high-quality Winograd schemas is inherently difficult." And we know from Section 2.9 that systematically perturbing schemas is important for evaluating model consistency; e.g., as Trichelair et al. [95] put it, "a system that correctly resolves both the original and the [perturbed] sentence can be said more certainly to reason about the full sentence, instead of exploiting a statistical quirk" of the problem.

Generally, if we want to move away from the Winograd Schema Challenge, we should start with a clear picture of where we are and what direction we want to go. WSC treats common-sense reasoning as natural language processing (Figure 1.1), and we certainly want to remain in the field of NLP while we measure common sense.

Now, Figure 2.1, which shows how the Winograd schema is related to some other important NLP tasks and problem types, does not include a heading for common sense. In NLP, the Winograd schema is a pronoun disambiguation problem that

happens to have a built-in adversarial feature in the form of a key phrase. Pronoun disambiguation, and coreference resolution in general, may partake of common-sense knowledge and reasoning [53, 104], but tests of common sense are by no means bound to remain in that region. The adversarial feature still seems promising, though.

As far as the method of stochastic fill-in-the-blank is concerned (Section 2.8.1), the Winograd schema's defining feature is that it can be recast as a cloze test. Since we intend to keep applying that method, we should keep that property.

At this point, it may be useful to distinguish between three high-level aspects of a Winograd schema: what we call task, content, and format.

The *task* is what a schema asks a respondent to do. Again, by design, the task of any Winograd schema is pronoun disambiguation, a special case of coreference resolution. However, as we saw in Section 3.3.6, general coreference resolution is already implicitly part of the task of many schemas, where at least one answer does not exactly match the corresponding candidate. It is not trivial to determine that "the suitcase" refers to "the brown suitcase" in (3.36), that "Tina's" refers to "Tina's drawing" in (3.37), that "the juggler" refers to "a man juggling watermelons," with no words in common, in (3.38), or even that "the pillar" refers to "a pillar" in (3.39).

Our generalization will not use the same task as the Winograd schema.

The *content* of a schema is how a respondent is intended to complete its task: some particular subset of common sense—an area of general knowledge, a form of default reasoning—that underlies its questions. Any schema can be "solved" by guessing at random, by always choosing the first answer, or indeed by plugging it into a language model, but the intended method is to apply a certain pattern of facts and logic.

Naturally, the content varies from schema to schema. For example, the content of the trophy-suitcase schema (3.36) can be described as the implication of relative size by a spatial relationship of containment, whereas the content of the orchard-painting schema (3.40) is the broad similarity of the shapes of pairs of ordinary objects.

We will discuss the content of common-sense reasoning tests in a bit more detail in Section 3.6.1 when we actually build a test out of our generalized schemas. For now, it is enough to note that the content of WSC273 is broadly faithful to the concept of common-sense reasoning, so our generalization should use similar content.

The *format* of a schema is the way in which the task and content are presented to

the respondent. With the changes described in Section 3.3, every Winograd schema in WSC266 shares the same format: the placeholder, the candidates, the target pronoun, the key phrases; the requirements that the pronoun be ambiguous and that swapping key phrases also swap correct answers; and so on (Sections 2.4 and 2.5).

Not every part of the format can be verified or enforced algorithmically. The format requires that any choice of key phrase make one answer overwhelmingly more plausible or likely as the referent; to verify this requirement is to solve the problem. Similarly, to verify that a pronoun is ambiguous is to determine whether or not a given string admits two or more possible referents, which is another problem in common-sense verbal reasoning. To determine whether or not the answers actually appear in the sentence as candidates is difficult to automate for the same reason (Section 3.3.6). Even to check that a sentence is grammatical is, in general, beyond our means. And verifying or enforcing the format algorithmically becomes drastically more difficult when we include the more complex and empirically-based criteria from Section 2.5.

The format of our generalization should be as precise as that of the Winograd schema; it does not need to be verified or enforced algorithmically, as this appears to be unrealistic; and it should have an adversarial feature similar to the key phrase.

To sum up, our goal is (i) a precisely defined test of verbal reasoning, with (ii) a built-in adversarial feature like a key phrase, (iii) amenable to the method of stochastic fill-in-the-blank, (iv) testing common-sense content similar to that of WSC266, and easier than a Winograd schema to (v) generate and (vi) systematically perturb.

Our generalization, presented in the following sections, is called the *adversarial schema*. Like the Winograd schema, it can be recast as a cloze test. An adversarial schema is either a *substitution schema*, which itself generalizes the Winograd schema, or a *transposition schema*, which does not. Figure 3.1 updates Figure 2.1 accordingly.

Naturally, we extended our database management system for Winograd schemas (Section 3.3) so it could store adversarial schemas too.

3.5.2 The Substitution Schema

A *substitution schema* has the following properties (compare Section 2.4):

1. The schema includes a sentence with two placeholders: a *key placeholder*, { }, and an *answer placeholder*, ⟨ ⟩.

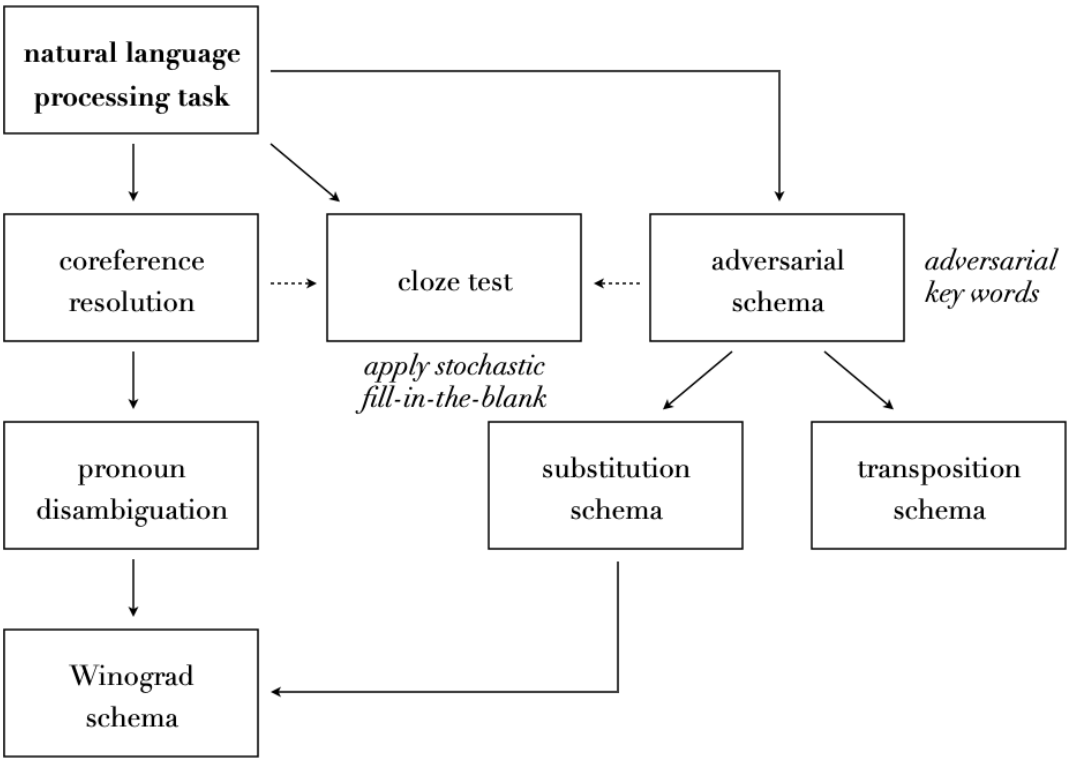

Figure 3.1: Adversarial schemas as natural language processing. Solid lines point to more specific tasks; dashed lines indicate one-way recasting.

2. The schema provides two *key phrases* that can replace the key placeholder and two *answers* that can replace the answer placeholder.

3. Replacing the key placeholder with the first key phrase makes the first answer far more likely as a substitution for the answer placeholder. Similarly, the second key phrase strongly favours the second answer.

Just like the Winograd schema, the point is that each key phrase yields a cloze test: to identify the more likely answer to substitute. Thus, the method of stochastic fill-in-the-blank can be applied without change to substitution schemas.

The following are examples of substitution schemas. For brevity, we write both key phrases inside the key placeholder and both answers inside the answer placeholder:

a. {three, five} objects are ⟨less, more⟩ numerous than four objects
b. a car ⟨can, cannot⟩ overtake a {slower, faster} car (3.60)
c. ⟨all, not all⟩ deserts are {arid, sandy}

(The missing capital letters and periods will be explained in Section 3.5.6.)

Every Winograd schema can be trivially recast as a substitution schema: the original key phrases are still the key phrases, the original answers are still the answers, and the target pronoun is simply replaced by an answer placeholder.

Not every substitution schema is a Winograd schema, however. None of the examples in (3.60) are pronoun disambiguation problems, nor can they be recast as such in any obvious way. So, to summarize the generalizations we have made: instead of an ambiguous pronoun, we have a generic answer placeholder; and we do not require the answers to appear anywhere in the problem statement as candidates.

We can fairly easily add to the list of examples in (3.60):

a. four objects are ⟨more, less⟩ numerous than {three, five} objects
b. a car ⟨cannot, can⟩ be overtaken by a {slower, faster} car (3.61)
c. ⟨no, some⟩ deserts are not {arid, sandy}

Obviously, in (3.61a), we moved the keys to the end of the sentence; in (3.61b), we put the verb in the passive voice; in (3.61c), we negated the keys. The keys themselves are the same in all cases. The answers are in reversed order but otherwise the same, except in (3.61c), where we swapped the quantifiers from universal to existential.

Here are some more easily generated schemas based on (3.60):

a. {sixteen, eighteen} cats are ⟨less, more⟩ numerous than seventeen mice
b. a hovercraft ⟨can, cannot⟩ overtake a {slower, faster} hovercraft (3.62)
c. ⟨all, not all⟩ bananas are {fruit, yellow}

3.5.3 The Transposition Schema

On second thought, there is something slightly unsatisfactory about (3.62c): it isn't a bad schema, by any means, but the content differs significantly between the problems. That is, we conceive of *fruit* as a set or category of which *bananas* form a subset; but we conceive of *yellow* as an attribute or property of which *bananas* may or may not partake—not as the set of all yellow things, which may overlap with *bananas*.

A version of the schema with more consistent content might be:

⟨all, not all⟩ bananas are {fruit, plantains} (3.63)

Or we could try something like:

⟨all, not all⟩ {bananas are fruit, fruit are bananas} (3.64)

It may not be in the spirit of a key phrase to allow (3.64). In any case, it only works because the answer placeholder happens to lie outside what we chose as key phrases.

The above discussion motivates us to add another type of schema, which is disjoint from the substitution schema (and therefore from the Winograd schema).

A *transposition schema* has the following properties (compare Section 3.5.2):

1. The schema includes a sentence with *three* placeholders: two key placeholders, { }, and an answer placeholder, ⟨ ⟩.

2. The schema provides two key phrases that can replace either key placeholder and two answers that can replace the answer placeholder.

3. Replacing the first and second key placeholders with the first and second key phrases, in that order, makes the first answer far more likely as a substitution for the answer placeholder. Similarly, the opposite order of the key phrases strongly favours the second answer.

Each ordering of the key phrases yields a cloze test, and the method of stochastic fill-in-the-blank can again be applied without change to substitution schemas.

The following are examples of transposition schemas. For clarity, we write both key phrases inside each key placeholder, but in opposite orders:

a. the ⟨larval, adult⟩ form of a {butterfly or moth, caterpillar} is a {caterpillar, butterfly or moth}
b. {cats, mice} ⟨do, do not⟩ hunt {mice, cats}
c. ⟨all, not all⟩ {bananas, fruit} are {fruit, bananas}

(3.65)

Evidently, (3.65c) is equivalent to (3.64), without the awkwardly complex key phrases. (3.65a) is a similar example in that it could also be achieved somewhat awkwardly with a substitution schema, because the answer placeholder again happens to lie outside the key phrases. However, (3.65b) has no directly corresponding substitution schema, because the answer placeholder has to lie between the key placeholders.

Substitution schemas and transposition schemas form disjoint sets: count the key placeholders. In particular, no Winograd schema is a transposition schema: there is no place for a second key phrase in a standard pronoun disambiguation problem.

We can fairly easily add to (3.65) in the style of (3.61):

a. a {butterfly or moth, caterpillar} is the ⟨adult, larval⟩ form of a {caterpillar, butterfly or moth}
b. {cats, mice} ⟨do not, do⟩ get hunted by {mice, cats}
c. ⟨no, some⟩ {bananas, fruit} are not {fruit, bananas}

(3.66)

Obviously, in (3.66a), we moved the first key to the start of the sentence; in (3.66b), we put the verb in the passive voice; in (3.66c), we negated the second key. The keys themselves are the same and in the same order in all cases. The answers are in reversed order but otherwise the same, except in (3.66c), where we swapped the quantifiers from universal to existential. All of the above is similar to (3.61).

Here are some more easily generated schemas based on (3.65) in the style of (3.62):

a. the ⟨juvenile, adult⟩ form of a {ferret, kit} is a {kit, ferret}
b. {shrikes, bees} ⟨do, do not⟩ hunt {bees, shrikes}
c. ⟨all, not all⟩ {chestnuts, nuts} are {nuts, chestnuts}

(3.67)

Note that if a dataset includes the transposition schema (3.65c) regarding banana inclusion, it should probably not also include either substitution schema (3.62c) or (3.63) regarding banana classification, because they all share a problem:

⟨all, not all⟩ bananas are fruit (3.68)

To avoid overlap, (3.62c) and (3.63) can be replaced by, respectively:

a. ⟨all, not all⟩ radishes are {vegetables, red}
b. ⟨all, not all⟩ oranges are {fruit, mandarins}

(3.69)

3.5.4 The Adversarial Schema

An *adversarial schema* is either a substitution schema or a transposition schema (Sections 3.5.2 and 3.5.3 and Figure 3.1). We motivated its design in Section 3.5.1.

Adversarial schemas make use of a built-in adversarial feature (hence the name) in the form of a key phrase or a pair of key phrases. They are amenable to solution by stochastic fill-in-the-blank. All Winograd schemas are substitution schemas and therefore adversarial schemas. Arguably, adversarial schemas are easier to generate than Winograd schemas in general, being less of an exercise in creative writing.

The task of an adversarial schema is, in general, multiple-choice fill-in-the-blank question answering. The content varies from schema to schema, but it can certainly replicate the content of any Winograd schema. The two types of adversarial schema, substitution and transposition, have slightly different but equally precise formats.

We were motivated to introduce the transposition schema in order to express the content of certain substitution schemas in a more natural or elegant way, but the transposition schema can also help us avoid associativity (Section 2.5.3). For example, each of the following schemas tests content very similar to (3.37) from WSC266. The first is a substitution schema and the second a transposition schema:

a. if a drawing is {above, below} a painting, the drawing is ⟨higher, lower⟩ than the painting
b. if a {drawing, painting} is above a {painting, drawing}, the drawing is ⟨higher, lower⟩ than the painting

(3.70)

To the extent that "above" is more strongly associated with "higher" in a corpus, or "below" with "lower," (3.70a) is associative. A model that simply matches words accordingly will perform well on that and similar schemas. Of course, that would be a spurious correlation: in general, the word "above" does not imply that anything in particular is "higher" than anything else, as we can see from (3.70b). Indeed, we expect a model that simply matches words to perform no better than chance on that and similar schemas, which is the point of a key phrase in general.

As a standalone schema, (3.70b) is to be preferred over (3.70a). On the other hand, suppose we perturb (3.70a) as follows, and include both versions in a dataset:

if a drawing is {above, below} a painting, the painting is ⟨lower, higher⟩ than the drawing

(3.71)

A model that simply matches words is expected to perform no better than chance on the four problems from (3.70a) and (3.71) combined. This may be a countermeasure

to associativity. However, in general, the unit of our new test of common-sense reasoning (Section 3.6) will be the adversarial schema, not a pair of schemas specially designed to counteract associativity, so we will not pursue this further.

There is a major difference between substitution and transposition schemas that pertains to inversion, which we will discuss in the next section.

3.5.5 Inverting the Adversarial Schema

Recall that a Winograd schema is *inverted* by exchanging the roles of the key phrases and candidates (Section 2.9.3). Inversion generalizes to substitution schemas: we exchange the roles of the keys and answers. For example, (3.60c) becomes:

$$\{\text{all, not all}\}\ \text{deserts are}\ \langle\text{arid, sandy}\rangle \tag{3.72}$$

The result is a new substitution schema. The only difference from the original schema is that the key placeholder { } and the answer placeholder ⟨ ⟩ have swapped places.

Recall that it is not necessarily true of an inverted Winograd schema that each key phrase makes a different answer far more likely (Section 3.4.3). The same holds of an adversarial schema: we need to check that the inverted schema still admits definitive answers, which in the case of (3.72) it surely does.

In short, not every substitution schema is invertible, but if a substitution schema is invertible, its inversion is another substitution schema. Clearly, inverting the inverted schema gets us back to the original schema. Every Winograd schema can be treated as a substitution schema; as we know, not every Winograd schema is invertible, but if a Winograd schema is invertible, its inversion is *never* a Winograd schema.

Transposition schemas are not invertible in the same way as substitution schemas: the result of applying an equivalent operation will not be a transposition schema, because there are two key placeholders to swap with only one answer placeholder. For example, the following inversion of (3.65b) requires the respondent to answer with an ordered pair, as "cats/mice" or "mice/cats":

$$\langle\text{cats, mice}\rangle\ \{\text{do, do not}\}\ \text{hunt}\ \langle\text{mice, cats}\rangle \tag{3.73}$$

Although (3.73) is not an adversarial schema, it is still amenable to solution by stochastic fill-in-the-blank in the obvious way, albeit with two blanks. Therefore, if

the result of inverting a transposition schema still admits definitive answers, we are willing to include it in an inverted perturbation of an adversarial dataset.

Figure 3.2 illustrates the concept of inverting adversarial schemas.

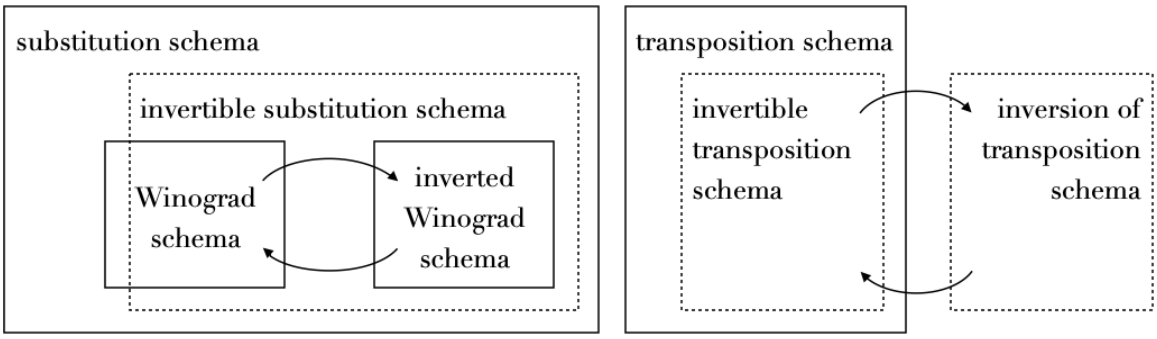

Figure 3.2: Inverting adversarial schemas, including Winograd schemas.

To sum up, given an adversarial schema, we can fix a key phrase (or an ordering of key phrases, for a transposition schema), thereby defining a problem, then select an answer to solve it. Given an invertible adversarial schema, we can fix an answer, thereby defining a problem, then select a key phrase (or an ordering of key phrases) to solve it. Table 3.4 shows the two modes of solution for adversarial schemas. What we call solution by key is the same as solution by answer of an inverted schema.

Define Problem	Guess Solution	Applicable Schemas
fix key phrase	by answer	any
fix answer	by key	invertible

Table 3.4: Modes of solution for adversarial schemas.

3.5.6 Adversarial Schema Dyads

Any schema that is grammatically similar to the ones in (3.60) and (3.65) can be converted systematically into a pair of schemas by substituting it into what we call a *boolean wrapper*, meaning an expression of one of these forms or similar:

a. It is ‖true, false‖ that [original schema].
b. That [original schema] is ‖true, false‖. (3.74)

Each choice from ‖true, false‖ yields a schema with different answers. For example,

the substitution schema (3.60c) can become the following pair of schemas:

$$\begin{aligned} &\text{i.} && \text{It is true that } \langle \text{all, not all} \rangle \text{ deserts are } \{\text{arid, sandy}\}. \\ &\text{ii.} && \text{It is false that } \langle \text{not all, all} \rangle \text{ deserts are } \{\text{arid, sandy}\}. \end{aligned} \tag{3.75}$$

Similarly, the transposition schema (3.65b) can become:

$$\begin{aligned} &\text{i.} && \text{It is true that } \{\text{cats, mice}\}\ \langle \text{do, do not} \rangle \text{ hunt } \{\text{mice, cats}\}. \\ &\text{ii.} && \text{It is false that } \{\text{cats, mice}\}\ \langle \text{do not, do} \rangle \text{ hunt } \{\text{mice, cats}\}. \end{aligned} \tag{3.76}$$

Note that in (3.75ii) and (3.76ii), the order of the answers has been reversed.

We call a pair of schemas of the form (3.75) or (3.76) an *adversarial dyad*, a *schema dyad*, or just a *dyad* for short. Each dyad generates four problems sharing the same set of two answers. Imitating (3.74), we can write a dyad more compactly:

$$\begin{aligned} &\text{It is } \|\text{true, false}\| \text{ that } \langle \text{all, not all} \rangle \text{ deserts are } \{\text{arid, sandy}\}. \\ &\text{It is } \|\text{true, false}\| \text{ that } \{\text{cats, mice}\}\ \langle \text{do not, do} \rangle \text{ hunt } \{\text{mice, cats}\}. \end{aligned} \tag{3.77}$$

In order, we (i) fix a truth value, which defines a schema, (ii) fix a key phrase, which defines a problem; and (iii) guess an answer, which solves the problem.

A dyad is said to be *invertible* if each of its schemas is invertible.

In (3.74), we made boolean wrappers out of the pair ‖true, false‖. We call this the *truth pair* for both the wrapper and the dyad, and its elements the *truth values*. There are many alternative options for the truth pair, such as ‖generally true, not generally true‖, ‖probable, improbable‖, and ‖plausible, implausible‖. All of those examples can be said to soften the original dyad, as they effectively allow for exceptions to the rule; e.g., the favoured answer need only hold "generally" or "probably."

We can apply the method of stochastic fill-in-the-blank to adversarial dyads without change: after all, they are merely pairs of adversarial schemas.

If we systematically apply a boolean wrapper and a truth pair to an adversarial dataset to generate dyads, we get a *dyadic version* of the dataset. We can think of each dyadic version—each choice of boolean wrapper and truth pair—as a perturbation.

Recall, from Table 3.4, the modes of solution for adversarial schemas. Adversarial dyads permit three modes of solution: by answer, by key, and by boolean. That is, given an invertible dyad, we can fix any one of the following, thereby defining a

schema: a truth value, a key phrase, or an answer. Then we can fix any one of the other two, thereby defining a problem with two answer options.

Table 3.5 shows the three modes of solution for adversarial dyads.

Define Schema	Define Problem	Guess Solution	Applicable Dyads
fix truth value fix key phrase	fix key phrase fix truth value	by answer	any
fix truth value fix answer	fix answer fix truth value	by key	invertible
fix key phrase fix answer	fix answer fix key phrase	by boolean	any

Table 3.5: Modes of solution for adversarial dyads. Recall Table 3.4.

Whether we fix a truth value and then a key phrase, or vice versa, we get the same problem in the end. However, the order does affect the schema we produce. For example, in (3.75), if we fix ‖true‖ and then a key phrase, we get the schema

$$\text{It is true that } \langle \text{all, not all} \rangle \text{ deserts are } \{\text{arid, sandy}\}. \tag{3.78}$$

But if we fix {arid} and then a truth value, we get the schema

$$\text{It is } \|\text{true, false}\| \text{ that } \langle \text{all, not all} \rangle \text{ deserts are arid.} \tag{3.79}$$

Suppose we wanted to compare solution by answer and solution by key on the same dataset using a schema-based metric like the ones we will introduce in Section 3.7.3. (3.78) can be solved by answer or by key, so it is an appropriate way to make a schema from a dyad for that comparison. (3.79), on the other hand, cannot be solved by key—its key is fixed—so it is not an appropriate schema for that comparison; it would be appropriate for comparing solution by answer and solution by boolean.

We will not explicitly address this point again, but the reader should assume that whenever we deal with adversarial dyads, we are distributing problems into schemas in an appropriate way for whatever comparison we are making.

3.6 A New Test of Common-Sense Reasoning

In Section 3.5, we introduced the adversarial schema. In this section, we construct a test of common-sense reasoning made up of those schemas.

3.6.1 Common-Sense Content

We introduced common-sense knowledge and reasoning as a term of art in the field of AI in Section 1.1.3, mentioning early work by McCarthy [60] and Winograd [103, 104]. For example, Winograd [103] argued that "if we really want computers to understand us, ... they need to have *all sorts of knowledge* about the subject they are discussing" (my emphasis). Here, we review more of the literature to make sure our new challenge, presented in the next section, is testing appropriate content (Section 3.5.1).

For McCarthy [61], common sense "includes the basic facts about events (including actions) and their effects, facts about knowledge and how it is obtained, facts about beliefs and desires," and "the basic facts about material objects and their properties."

Levesque et al. [53] refer to "commonsense knowledge about space, time, physical reasoning, emotions, social constructs, and a wide variety of other domains."

According to Davis [11], common-sense verbal reasoning requires and encompasses specific "relevant knowledge" from "domains" or conceptual categories that include "quantity, space, time, physics, goals, plans, needs, and communication."

Davis [13] presents common-sense problems in natural language understanding that "draw on basic concepts and presume knowledge of basic facts from a variety of familiar domains," including "physics" and "social institutions," e.g., respectively, "a river can change course" and "a museum can own a painting." Other examples of domains include the relations of parts and inclusion, e.g., respectively, "the Battle of Gettysburg was part of the Civil War" and "'Moby Dick' is a book."

In "a sampling of commonsense reasoning problems," Davis and Morgenstern [15] include "the egg-cracking domain," in which "the problem... is to characterize the correct procedure of cracking an egg and transferring its contents to a bowl."

> The problem itself is quite a complex one, since cracking an egg involves reasoning about so many domains of physical reasoning: containment and parts, materials, collisions, liquids, and vessels.

In truth, the literature gives common-sense content a practically unlimited scope. For example, Davis [11] argues that "the kinds of reasoning involved in common sense include, in simple form, most if not all of the kinds of reasoning that are consciously usable by human intelligence," and they apply to "most types of intelligent activities,

such as... planning, learning, high-level vision, and expert-level reasoning."

It is one thing to test a program's ability to simulate verbal reasoning, and quite another to test its ability to plan or learn in general, let alone its capacity for carrying out general human-like intelligence. Suffice it to say that, by this definition, no language model can exhibit common sense, even if we exclude "high-level vision."

The content of published Winograd schemas may be quite broad, though perhaps not as broad as human reason. For example, we said in Section 3.5.1 that the content of the trophy-suitcase schema (3.36) can be described as the implication of relative size by a spatial relationship of containment, but of course we need to know English grammar first. The content of the orchard-painting schema (3.40) was described as the broad similarity of the shapes of pairs of ordinary objects, but of course we need to understand the intended and attempted matching of a painting to real life.

We should add a word on the *difficulty* of the test (Section 2.5.1). McCarthy [60]'s "programs with common sense" would simulate the verbal reasoning processes of "any non-feeble-minded human," making it sound easy, but Winograd [104] notes that language itself "is one of the most complex and unique of human activities," which puts a daunting lower bound on the difficulty of any such test.

Levesque et al. [53] require that the answer to a test question be "obvious to the human reader," so the problem is answerable "immediately" by an "untrained subject." However, the sample questions they present "differ on the background knowledge assumed," with some being "more 'university-level'" in difficulty.

Davis [11] refers to the knowledge "possessed by every schoolchild," but Davis and Morgenstern [15] note that common-sense problems can be "quite... complex," and among Davis [13]'s "basic concepts" and "basic facts" from "familiar domains" are some that seem university-level in difficulty; e.g., "a word [namely *allopatric*] in one language [English] can be formed out of two words in a different language [Greek]."

When it comes to difficulty, again the literature gives common sense a practically unlimited scope. It seems appropriate to allow test questions like the adversarial dyads (3.75) and (3.76), whether or not "your Aunt Edna" [53] can solve them.

Reviewing the content of our own examples, we find that (3.60a) concerns basic numeracy, specifically the relative size of small whole numbers; (3.60b) connects speed to time and distance, roughly speaking, and covers the same content as the Winograd

schema (2.11) without being obviously associative; (3.60c) pertains to set inclusion or the properties of objects, based on the definitions of words. The variant schemas in (3.61) and (3.62) cover the same type of content.

By design, (3.63) is broadly similar in content to (3.62c). (3.65a) and (3.65b) both test basic knowledge of the life cycles and behaviour, specifically predation, of well-known animals; (3.65c) is again broadly similar to (3.62c). The variant schemas in (3.66) and (3.67) cover the same type of content as (3.65). By design, the replacement schemas (3.69a) and (3.69b) are very similar to (3.62c) and (3.63), respectively.

Finally, as we said, the content of (3.70) and (3.71) is very similar to (3.37) from WSC266, although (3.70a) may be highly associative.

3.6.2 Reid250, a Common-Sense Challenge

> ... withdraw this penurious and malignant ray: I despise Philosophy, and renounce its guidance: let my soul dwell with Common Sense.
>
> *Thomas Reid*

We introduce a new test of common-sense knowledge and reasoning[2] made up of 125 adversarial schemas comprising 250 fill-in-the-blank problems. We call this test *Reid250* after Thomas Reid, the author of *An Inquiry into the Human Mind on the Principles of Common Sense* (1764), which we quoted above, a little out of context.

The test contains 75 substitution schemas and 50 transposition schemas, for a ratio of three to two. The content tested by those schemas, particularly the difficulty of that content, is broadly consistent with the discussion in Section 3.6.1.

Reid250 is obviously comparable in size to WSC273 and WSC266.

Given a boolean wrapper, e.g., (3.74a), and a truth pair, e.g., ‖true, false‖, we can systematically generate 125 adversarial dyads (Section 3.5.6) and a dyadic dataset that we call *Reid250×2*, consisting of 250 schemas comprising 500 problems.

Reid250 is invertible: each schema admits solution by answer, i.e., the usual way, and by key, i.e., inverted (Table 3.4). Moreover, any dyadic version of Reid250 admits solution by answer, by key, and by boolean (Table 3.5).

[2]Available here: `https://github.com/adrianmaler85/evaluating-common-sense`.

3.7 Evaluating Models on Adversarial Datasets

We discussed evaluation protocols for language models on the Winograd Schema Challenge, i.e., on Winograd schemas, in Section 2.9. In this section, we generalize that discussion and extend it to datasets made up of adversarial schemas.

3.7.1 Problem Accuracy and Schema Accuracy

Some terminology: If a model answers both of a schema's problems correctly, we say that the model *solves* the schema. If a model answers exactly one of a schema's problems correctly, we say that the model *half-solves* the schema; in that case, the model must have given the same answer to both problems. If a model answers both of a schema's problems incorrectly, we say that the model *anti-solves* the schema.

Recall, from Section 2.9.2, *problem accuracy* (the usual accuracy metric) and *schema accuracy* (a group scoring metric). We expect to achieve a schema accuracy of around 0.25 by guessing at random, and 0 by always choosing the first answer, whereas the expected problem accuracy for both strategies is 0.5 (Section 2.3).

Suppose we evaluate a model on a dataset. If the shares of solved, half-solved, and anti-solved schemas are denoted by A, A', and A'', so that A is the schema accuracy and $A + A' + A'' = 1$, then the model's problem accuracy is given by

$$a = A + A'/2. \tag{3.80}$$

We will see in Section 4.2 that anti-solved schemas are rare with the models and datasets under consideration. If we set $A'' = 0$, so that $A + A' = 1$, then (3.80) reduces to $a = (1 + A)/2$. For example, schema accuracy 0.2, which is worse than chance, leads to problem accuracy 0.6, which is better than chance, simply because, by hypothesis, the model never anti-solves schemas. In Chapter 4, we will note these unusual, perhaps even paradoxical results whenever they occur.

In general, (3.80) can be rewritten as $a = (1 + A - A'')/2$. Figure 3.3 shows problem accuracy a as a function of schema accuracy A for a few small values of A''.

3.7.2 Transformations, Perturbations, and Consistency

Consistency and other group scoring metrics for Winograd schemas (Sections 2.9.1 and 2.9.2) can be generalized to adversarial schemas in a straightforward way.

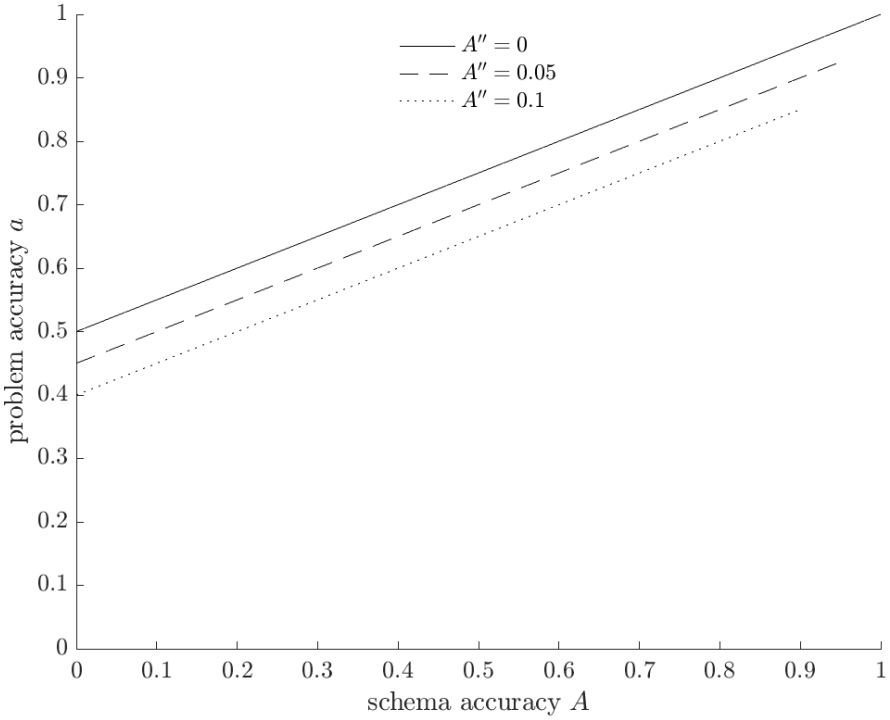

Figure 3.3: Problem accuracy a as a function of schema accuracy A for a fixed rate A'' of anti-solved schemas, where $A'' = 0$ (top), 0.05 (middle), and 0.1 (bottom).

Let X and Y be sets of problems derived from adversarial schemas. Let

$$f : A \subseteq X \to Y$$

be a function. Suppose f preserves schemas: that is, $f(x_1)$ and $f(x_2)$ belong to the same schema in Y whenever x_1 and x_2 belong to the same schema in X. In that case, we call f a *transformation* of X, and $f(A) \subseteq Y$ the corresponding *perturbation*.

Although we have distinguished between a transformation and the perturbation it defines, for simplicity, we may identify a transformation with its perturbation.

Given a model M, we can measure its accuracies on both the *transformable* subset A and the perturbation or *transformed* subset $f(A)$. We can also measure the model's consistency under the transformation f, which can be defined in a few ways.

In general, a model will solve some problems from the transformable subset and some problems from the transformed subset. That leads us to define the *consistency*

of M under the transformation f as $c = c(M, f)$, where

$$c = \frac{|\{x \in A : M \text{ solves } x \text{ and } f(x), \text{ or neither } x \text{ nor } f(x)\}|}{|A|}. \tag{3.81}$$

Consistency generalizes Trichelair et al. [95]'s switchable consistency, which was also used by Emami et al. [25] and Abdou et al. [1].

We define the *consistent accuracy* of M under f as

$$c_a = \frac{|\{x \in A : M \text{ solves } x \text{ and } f(x)\}|}{|A|}. \tag{3.82}$$

Consistent accuracy generalizes Ruan et al. [81]'s consistent accuracy, and it can easily be generalized to multiple transformations as in Elazar et al. [22].

We define the *preserved accuracies* of M under f as follows:

$$c_p = \frac{|\{x \in A : M \text{ solves } x \text{ and } f(x)\}|}{|\{x \in A : M \text{ solves } x\}|} \tag{3.83}$$

$$\widehat{c_p} = \frac{|\{x \in A : M \text{ solves } x \text{ and } f(x)\}|}{|\{x \in A : M \text{ solves } f(x)\}|} \tag{3.84}$$

We have not seen either preserved accuracy used in the literature.

c, c_a, c_p, and $\widehat{c_p}$ are group scoring metrics across datasets (Section 2.9.2).

Clearly, $c, c_p, \widehat{c_p} \geq c_a$, making consistent accuracy stricter in general than either consistency or the preserved accuracies. We can say that consistency c does not concern itself with accuracy; that consistent accuracy c_a does not value consistently incorrect answers; and that the preserved accuracies c_p and $\widehat{c_p}$ only look at problems that were correct before and after the transformation, respectively.

The preserved accuracies are asymmetrical: that is, consistency c and consistent accuracy c_a stay the same when we invert the transformation,

$$f^{-1} : f(A) \subseteq Y \to X,$$

but the preserved accuracies c_p and $\widehat{c_p}$ are exchanged and, in general, will not stay the same. c_p may be especially appropriate when the transformable subset A is the original dataset, which we are comparing to several perturbations, and when we expect or observe that f increases accuracy, as we might like it to preserve correctness in the process; $\widehat{c_p}$ may be especially appropriate when f decreases accuracy.

We can generalize one more time: we defined consistency under transformations, but we can also measure a model's consistency with respect to a change of scoring method (Section 2.2), which we can think of as a sort of transformation.

3.7.3 Schema Consistency

Recall that inversion (Section 2.9.3), which we called a transformation in Section 2.9.5, actually acts on schemas, not problems: there is no way to meaningfully associate either of a schema's problems with either of the inverted schema's problems.

Let W and Z be sets of adversarial schemas. Let

$$F : A \subseteq W \to Z$$

be a function. In that case, we call F a *schema transformation* of W, and $F(A) \subseteq Z$ the corresponding *schema perturbation*. We may refer to the usual transformations as *problem transformations* to distinguish them from schema transformations.

Every problem transformation f defines a schema transformation $F = \widehat{f}$, where

$$\widehat{f}(w) = \langle f(x_1), f(x_2) \rangle \qquad \text{for each schema } w = \langle x_1, x_2 \rangle. \tag{3.85}$$

In that case, we say that M solves $w \in A$ and $F(w)$ *identically* if

$$\forall\, x \in w \ (M \text{ solves } x \text{ and } f(x), \text{ or } M \text{ solves neither } x \text{ nor } f(x)). \tag{3.86}$$

Just as we have notions of problem accuracy and schema accuracy (Section 3.7.1), in addition to our consistency metric c (3.81), which we may now want to refer to as *problem consistency*, we also have a notion of schema consistency that applies to schema transformations; indeed, we have more than one notion.

Each schema $w = \langle x_1, x_2 \rangle$ is made up of problems that can be solved or not solved, so, as we know, the schemas themselves can be solved, half-solved, or anti-solved.

We define the *weak schema consistency* of a model M under F as

$$\widetilde{C} = \frac{|\{w \in A : M \text{ solves } w \text{ and } f(w), \text{ or } M \text{ solves neither } w \text{ nor } F(w)\}|}{|A|}. \tag{3.87}$$

We define the *schema consistency* of M under F as

$$C = \frac{|\{w \in A : M \text{ solves equally many problems from } w \text{ and } F(w)\}|}{|A|}. \tag{3.88}$$

Now suppose $F = \widetilde{f}$ is defined by a problem transformation as in (3.85). We define the *strict schema consistency* of M under F as

$$C^* = \frac{|\{w \in A : M \text{ solves } w \text{ and } F(w) \text{ identically}\}|}{|A|}. \tag{3.89}$$

$\widetilde{C}$, C, and C^* are group scoring metrics within and across datasets (Section 2.9.2).

We can say that weak schema consistency $\widetilde{C}$ does not distinguish between half-solved and anti-solved schemas; that schema consistency C does not care how many transformed problems are answered differently, so long as the number of correctly answered problems stays the same; and that strict schema consistency C^* applies problem-wise to each schema, assuming it applies at all.

Clearly, $\widetilde{C} \geq C$, and where C^* is defined, we also have $C \geq C^*$. That is, if a model solves two problem pairs identically, then it solves equally many problems from each pair, which implies that it solves either both pairs or neither pair. We also have $c \geq C^*$, where c is problem consistency (3.81).

There is no such simple relationship between c and C or $\widetilde{C}$. We can have, e.g., $c = C^* = 0$ and $C = \widetilde{C} = 1$, if the model half-solves every schema before and after the transformation, but the transformation changes which problem within each schema is solved. Or we can have, e.g., $c = 0.5$ and $C^* = C = \widetilde{C} = 0$, if the model solves every schema before the transformation and half-solves every schema after.

At least we have justified the terms "weak" and "strict." Strict schema consistency C^* is an appropriate schema consistency metric wherever it applies: the only reason we define C and $\widetilde{C}$ is for schema perturbations where C^* does not apply.

We find no support in the literature for any particular definition of schema-based consistency [1, 22, 23, 25, 44, 81, 95, 96, 109]. That said, on problem transformations, we can already apply consistency c. On schema transformations, on the other hand, neither c nor strict schema consistency C^* applies in general. We should report c and C^* wherever they apply, i.e., on problem transformations, and we should report either schema consistency C or weak schema consistency $\widetilde{C}$ wherever c and C^* do not apply, so we have some way of measuring consistency under, e.g., inversion. We chose to report schema consistency C, simply because it is the stricter of the two.

Very briefly, we will extend consistent accuracy c_a (3.82) and preserved accuracy c_p (3.83) and $\widehat{c}_p$ (3.84) to schema-based metrics as well.

We define the *consistent schema accuracy* of M under F as

$$C_a = \frac{|\{w \in A : M \text{ solves } w \text{ and } F(w)\}|}{|A|}. \tag{3.90}$$

We define the *preserved schema accuracies* of M under F as follows:

$$C_p = \frac{|\{w \in A : M \text{ solves } w \text{ and } F(w)\}|}{|\{x \in A : M \text{ solves } w\}|} \tag{3.91}$$

$$\widehat{C}_p = \frac{|\{w \in A : M \text{ solves } w \text{ and } F(w)\}|}{|\{x \in A : M \text{ solves } F(w)\}|}. \tag{3.92}$$

Of course, C_a, C_p, and $\widehat{C}_p$ are group scoring metrics within and across datasets. Clearly, $C, C_p, \widehat{C}_p \geq C_a$, and where C^* is defined, we also have $C_a \geq C^*$.

Table 3.6 shows our definitions of consistency and where they apply.

Consistency Metric	Symbol	Definition	Type of Transformation
problem consistency	c	(3.81)	problem
consistent accuracy	c_a	(3.82)	problem
preserved consistency	c_p, $\widehat{c}_p$	(3.83), (3.84)	problem
weak schema consistency	$\widetilde{C}$	(3.87)	problem or schema
schema consistency	C	(3.88)	problem or schema
strict schema consistency	C^*	(3.89)	problem
consistent schema accuracy	C_a	(3.90)	problem or schema
preserved schema accuracy	C_p, $\widehat{C}_p$	(3.91), (3.92)	problem or schema

Table 3.6: Consistency metrics and the type of transformation to which they apply.

3.7.4 Disparate Answer Tokenization

In Section 2.9.3, we introduced the term *disparate answer tokenization*: when a model converts the answer options for a problem into token sequences of different length. We described problems with *any-length answers* and *equal-length answers*. We also mentioned that we intend to measure the phenomenon's effect on model accuracy.

The unbalanced perturbation (Section 3.4.4) is designed in part to put a practical upper bound on that effect. In general, for any dataset, we can measure the effect by reporting separate results on the subset of problems with equal-length answers.

Detecting disparate answer tokenization is not entirely trivial. Tokenization is sensitive to both capitalization and preceding whitespace (Section 2.2), and of course stochastic fill-in-the-blank involves substituting answers into problems (Section 2.8.1). Therefore, how an answer tokenizes will depend, in general, on whether or not we put it in the context of the problem statement, which, e.g., may capitalize it.

In preprocessing WSC273 (Section 3.2), we capitalized the answers according to their appearance in the context of the problem: it seemed reasonable to prepare the dataset for solution by stochastic fill-in-the-blank in advance, given that the alternative would be to capitalize answers on the fly, detecting the start of English sentences in the middle of running a language model.

We also capitalized answers according to their appearance in the context of the problem when we created WSC266 (Section 3.3) and Reid250 (Section 3.6.2). This choice is appropriate for detecting disparate answer tokenization in any dataset.

It also seems appropriate to apply or not apply preceding whitespace to each answer according to its position in the problem statement, and we did so. In fact, this choice can greatly affect the share of problems with equal-length answers.

As a preview of the experimental setup in Section 4.1, Table 3.7 shows the share of problems with equal-length answers, with and without the context of the problem, for various datasets and perturbations. For T5, we prepended the model's special WSC task prefix to each Winograd problem, but this has no effect on the share.

In every case, with or without the context of the problem, the share of problems with equal-length answers is identical for GPT-2 and RoBERTa. The share for T5 is sometimes higher than the value for the other models but more often lower.

In the unbalanced perturbation, no problems have equal-length answers. When solving Reid250×2 by boolean with the truth pair ‖true, false‖, all problems have equal-length answers, namely *true* and *false* (one token each). For an unequally tokenized truth pair, e.g., ‖true, not true‖, the share would of course drop to zero.

Putting the answers in the context of the problem never changes the share of equal-length answers for T5: an attribute of that model's tokenizer. For GPT-2 and RoBERTa, it does change the share in general. The change is largest for WSC266 switched and inverted, Reid250, and Reid250×2. The change is zero for WSC266 adjectival, Reid250 by key, and of course for WSC266 unbalanced and Reid250×2 by boolean where the shares are identically zero and one respectively.

Inverted transposition schemas almost always have equal-length answers, because the answers are orderings of the same two key phrases (Section 3.5.5).

Dataset	Perturbation 1	Perturbation 2	Context	GPT-2 and RoBERTa	T5
WSC273	original	none	none	0.615	0.593
			problem	0.681	0.593
WSC266	original	none	none	0.624	0.609
			problem	0.684	0.609
		inverted	none	0.511	0.669
			problem	0.789	0.669
	switched	none	none	0.671	0.700
			problem	0.829	0.700
	adjectival	none	none	0.897	0.862
			problem	0.897	0.862
	unbalanced	none	none	0.000	0.000
			problem	0.000	0.000
Reid250	original	none	none	0.568	0.704
			problem	0.752	0.704
		inverted	none	0.688	0.656
			problem	0.688	0.656
Reid250×2	(3.74a) true, false	none	none	0.568	0.704
			problem	0.752	0.704
		by key	none	0.736	0.728
			problem	0.776	0.728
		by boolean	none	1.000	1.000
			problem	1.000	1.000

Table 3.7: Share of problems with equal-length answers.

3.7.5 Answer Placeholder Position

In Section 3.1.1, we argued that partial scoring for GPT-2 may be inappropriate for some Winograd schemas: specifically, where the target pronoun occurs toward the end of the problem statement. For adversarial schemas in general, the answer placeholder or, in the case of an inverted transposition schema, the second of two placeholders may occur toward the end of the problem statement and create the same issue.

As a supplement to the experimental setup in Section 4.1, Table 3.8 shows the share of problems with an answer placeholder in the second-last position for various datasets, subsets of datasets, and perturbations, with notes to follow.

Dataset	Perturbation 1	Subset	Perturbation 2	Size	Target at End
WSC273	original	all	none	273	0.073
WSC266	original	all	none	266	0.068
			no candidates	266	0.068
			inverted	266	0.489
		associative	none	50	0.120
			no candidates	50	0.120
		non-assoc.	none	216	0.056
			no candidates	216	0.056
	switched	all	none	140	0.043
	adjectival	all	none	174	0.057
	unbalanced	all	none	266	0.068
Reid250	original	all	none	250	0.112
			inverted	250	0.448
Reid250×2	(3.74a) true, false	all	none	500	0.112
			by key	500	0.448
			by boolean	500	0.000

Table 3.8: Share of problems with an answer placeholder second-last.

Note that the answer placeholder occurs in the second-last position for almost half the problems in WSC266 inverted, Reid250 inverted, and Reid250×2 by key, and no more than 12% of the problems in any other dataset, subset, or perturbation.

Incidentally, it is possible for an answer placeholder to occur in the very last position for WSC273, because some of those problems are missing final punctuation (Section 3.3.8), but this does not actually occur in the dataset.

3.7.6 An Evaluation Protocol for Adversarial Schemas

We propose an evaluation protocol for adversarial schemas. To be precise, suppose we have a language model, a dataset made up of adversarial schemas, and possibly some perturbations of that dataset.

We treat the following generalized perturbations separately: the no-candidate perturbation, which can be applied to any Winograd schema; inversion or solution by key, which can be applied to any adversarial schema; and solution by boolean, which can be applied to any adversarial dyad.

For each of the model's scoring methods, we calculate the problem accuracy and schema accuracy, and compare them both to chance. We repeat the calculations on the subset of problems with equal-length answers.

On a Winograd dataset, e.g., WSC266, we form the no-candidate perturbation and calculate the model's accuracies on that perturbation. We compare that to the original dataset to quantify spurious correlations in the model. If a so-called associative subset of the dataset has been identified, say by human annotators, we also report separate comparisons for that subset and its complement.

We calculate the model's accuracies on each perturbation, as well as the model's problem consistency and schema consistency with respect to it. We also invert the invertible subset of the dataset and report the model's accuracies and consistencies with respect to inversion, i.e., solution by key.

On a dataset that admits adversarial dyads, e.g., Reid250, we select a boolean wrapper and a truth pair, and form a dataset, twice the size, made up of those dyads. We calculate the model's accuracies under each mode of solution: by answer, i.e., the usual way; by key, i.e., inverted; and by boolean, which is specific to dyads.

One last note: whenever we calculate schema accuracy or any other schema-based group scoring metric on WSC273, we have to deal with two minor issues.

First, one schema has three problems (Section 3.3.1), which changes every metric; e.g., for this one schema, we expect to achieve a schema accuracy of 0.125 by guessing at random, which is half the usual value. Not wanting to ignore any problems from WSC273, for the sake of simplicity and consistency, we treated the first and second problems as one schema, and the second and third problems as a separate schema. We did not count the second problem twice when calculating problem-based metrics.

Second, if we restrict WSC273 to a subset of problems, some schemas may have only one problem left. For example, we want to study problems with equal-length answers. In WSC273, the answer options can differ between problems in the same schema (Section 3.3.2). As a result, under each of GPT-2, RoBERTa, and T5, when we restrict to problems with equal-length answers, we end up with a single degenerate one-problem schema, for which problem accuracy must equal schema accuracy. Under each model, we did not count that schema toward any schema-based metrics.

3.8 Statistical Significance

Before we move on to our results, we should add a remark on statistical significance in the context of evaluating model accuracy on Winograd datasets.

3.8.1 Binomial Statistical Significance

Suppose we have a model M and some large space S of problems to solve with it, where $N = |S| \gg 1$. Suppose the true accuracy of the model—the share of problems it solves correctly—is a, where $0 < a < 1$.

Now suppose we choose a random sample of problems, or in other words a test set: $T \subseteq S$ where $n = |T| \ll N$. Then the accuracy a_T of the model on the test set is a random variable. To be precise, let X be the number of correctly solved test problems, so that $0 \leq X \leq n$ and $a_T = X/n$. If we can assume that the model's success or failure on any given test problem is independent of its performance on the other problems, then X has a *binomial distribution* with parameters n and a:

$$\mathbb{P}(X = k) = \binom{n}{k} a^k (1-a)^{n-k}, \qquad k = 0, \ldots, n. \tag{3.93}$$

The mean is $\mu = \mathbb{E}[X] = na$, which corresponds to a sample accuracy of $a_T = a$: we expect the model's accuracy on a random sample to be its true accuracy. The *confidence intervals* for the sample accuracy may be estimated by the Wald method:

$$a_T \pm z\sqrt{\frac{a_T(1-a_T)}{n}} \tag{3.94}$$

where z is a parameter derived from the standard normal distribution $N(0,1)$. For a 95% confidence interval, set $z = 1.96$; that is, the observed sample accuracy a_T lies in the interval given by (3.94) with probability about 0.95 for that choice of z.

Suppose we believe that the true accuracy is a_0: the *null hypothesis*. If we observe a sample accuracy a_T such that $a_T \neq a_0$, can we, with any confidence, reject the null hypothesis and conclude that the true accuracy is not a_0? Our confidence depends on two things: the size of the sample and the size of the gap between observed and expected accuracies. Larger samples and larger gaps inspire greater confidence.

To be precise, let $m_1 = a_0 n$ be the expected number of correct answers and $m_2 = (1 - a_0)n$ the expected number of incorrect answers. Let $n_1 = a_T n$ be the observed number of correct answers and $n_2 = (1 - a_T)n$ the observed number of incorrect answers. The test statistic[3] is a χ^2 test with one degree of freedom:

$$t = \frac{(n_1 - m_1)^2}{m_1} + \frac{(n_2 - m_2)^2}{m_2}. \tag{3.95}$$

With this random variable, we can apply an upper one-tailed test by looking up a standard table of critical values (thresholds for significance) for the χ^2 distribution.

For example, if the test statistic t is greater than about 3.84, a critical value, then our observed accuracy a_T is fairly far from the expected value a_0, given the size of the sample: to be precise, observed accuracies on a sample of that size that are at least that far from a_0 occur with a probability of about $p < .05$ when the true accuracy is a_0. Therefore, based on our observation, we would reject the null hypothesis and conclude that $a \neq a_0$ with some confidence, reporting $p < .05$ as our *p-value*.

3.8.2 Statistical Significance and Schemas

Now suppose we have n schemas of two problems each. It seems plausible that the model's success or failure on one problem from a schema will not be independent of its performance on the other problem. If we treat each schema as a single entity, the analysis in Section 3.8.1 applies; on the other hand, it would not be appropriate to report confidence intervals and p-values for problem-based metrics.

Let z be a random schema. Let z_1, z_2 be its problems. Let a_i be the ith-problem

[3]Of course, the test statistic is not a test in the same sense as a test set or a test problem.

accuracy, $a_i = \mathbb{P}(M \text{ solves } z_i)$, for $i = 1, 2$. Define four conditional probabilities:

$$\begin{aligned} u &= \mathbb{P}(M \text{ solves } z_2 \mid M \text{ solves } z_1), \\ v &= \mathbb{P}(M \text{ solves } z_2 \mid M \text{ does not solve } z_1), \\ u' &= \mathbb{P}(M \text{ solves } z_1 \mid M \text{ solves } z_2), \\ v' &= \mathbb{P}(M \text{ solves } z_1 \mid M \text{ does not solve } z_2). \end{aligned} \tag{3.96}$$

If the first and second problems are independent, then $u = v = a_2$ and $u' = v' = a_1$.

As a preview of the results in Section 4.2, Table 3.9 shows the observed values of a, a_1, a_2, u, v, u', and v' for GPT-2, RoBERTa, and T5 on WSC266, with each model using its most accurate scoring method (Section 4.2.4).

Problems	Condition	Metric	GPT-2	RoBERTa	T5
all	–	a	0.744	0.786	0.868
second only	none	a_2	0.797	0.865	0.940
	first problem solved	u	0.717	0.809	0.925
	first problem failed	v	0.976	1.000	1.000
first only	none	a_1	0.692	0.707	0.797
	second problem solved	u'	0.623	0.661	0.784
	second problem failed	v'	0.963	1.000	1.000

Table 3.9: Problem accuracy statistics for GPT-2, RoBERTa, and T5 on WSC266.

One important fact is that v and v' are consistently over 0.9: the models rarely anti-solve these schemas, as we mentioned in Section 3.7.1, so if, say, the first problem is answered incorrectly, the second is almost guaranteed to be answered correctly.

We find no support for the statistical independence of the problems in a schema. By the methods of Section 3.8.1, under the most accurate scoring methods (Table 3.9), we reject the null hypothesis that $v = a_2$ for GPT-2, $p < .01$, and RoBERTa, $p < .025$. For T5, we are unable to reject the null hypothesis, $p < .25$: the model's accuracies are so high, it is difficult to tell them apart. However, under model loss scoring (mean masked scoring) with a task prefix, we are able to reject $v = a_2$ for T5, $p < .05$.

That brings us to a method of calculating statistical significance that works in spite of problem pair correlations, which is *bootstrapping*, specifically by the *Monte Carlo* method [18]. Let the null hypothesis include values for a_1, u, and v, so that

$$a = a_1 u + (1 - a_1)v/2 + a_1(1 - u)/2. \tag{3.97}$$

If we observe, on a sample of n schemas, a problem accuracy $a_T \neq a$, then we need to find the probability of observing, in a sample of that size, an accuracy that is at least as far from a as a_T is, assuming a_1, u, and v are accurate. We estimate that probability p using a Monte Carlo simulation with ten thousand trials.

For example, when we restrict WSC266 to problems with answers of equal length (Section 2.9.3), then the accuracy of GPT-2 becomes 0.720, compared to 0.744 on all problems (Section 4.2). Is the change in accuracy significant?

The reduced dataset has $n = 91$ schemas with equal-length answers, and we get the values of a_1, u, and v from Table 3.9. Through bootstrapping, we find that a difference in accuracy of at least that magnitude, $|\Delta a| \geq 0.025$, occurs in $k = 4140$ of $R = 10^4$ trials, yielding the Monte Carlo approximation

$$p \approx p_{\text{mc}} = \frac{k+1}{R+1} \approx 0.414, \tag{3.98}$$

so we can report $p = .4$ in the traditional style, with exactly one non-zero digit. With as many as $R = 10^6$ trials, we get $k = 415108$, so we can report $p = .4$ again.

In short, we are unable to reject the null hypothesis that restricting to answers of equal length has no effect on problem accuracy for GPT-2 on WSC266, $p = .4$.

Bootstrapping provides confidence intervals as well, which are either *empirical* or *percentile* intervals. The end results were indistinguishable in our experiments, so we will report percentile intervals exclusively.

3.8.3 Statistical Significance and Schema Dyads

For adversarial dyads (Section 3.5.6), we cannot even treat each schema as a single entity, as the schemas occur in pairs. Presumably, the model's success or failure on the problems from one schema in a dyad will not be independent of its performance on the problems from the other schema.

Extending the analysis in Section 3.8.2 to account for schema pair correlations is probably overkill, given the actual results we will report in Section 4.7, particularly Figure 4.12. Simply put, all the models perform very poorly, almost identically so, on Reid250×2, with a problem accuracy very close to chance and a schema accuracy well below chance. The statistical significance of the outcome is hardly questionable, so we decided to report approximate bootstrapped p-values and confidence intervals for adversarial dyads, treating the schemas in each dyad as if they were independent.

3.8.4 Statistical Significance and Winograd Datasets

Is this at all relevant to the Winograd Schema Challenge or tests of common-sense reasoning in general? Is there even such a thing as a space of problems?

Some Winograd datasets, including DPR, WNLI, MaskedWiki, WikiCREM, and Winogrande, are or can be split into training and test sets (Section 2.7). Arguably, we can interpret the test set as a sample from a larger space of problems; if the sampling is random, we can even treat the problems as independent. On the other hand, if, say, we insist on including in the sample either both problems from a schema or neither of them, we can at least treat the schemas, thought not the problems, as independent.

Crowdsourced Winograd datasets, e.g., DPR and WinoGrande (Sections 2.7.2 and 2.7.5 respectively), generally provide their crowd workers with a specific protocol for generating problems; e.g., Sakaguchi et al. [85] give a fairly detailed description of the protocol they used to generate WinoGrande. In theory, we can generate more problems using the same protocol, so we have a space of problems. However, the statistical independence of those problems is questionable: both datasets display predictable structure, and in general we expect a protocol to introduce annotation artifacts, hence spurious correlations (Section 2.5.3). Indeed, Elazar et al. [23] report that Sakaguchi et al. [85]'s model exhibits spurious correlations.

Self-supervised datasets such as MaskedWiki and WikiCREM (Section 2.7.4) are based on large text corpora. The world has no shortage of digital text, so in theory, we can always generate more problems: again, we have a space of problems, and again, the statistical independence of those problems is questionable. The contents of the English-language Wikipedia articles used to create both of the above-mentioned datasets no doubt include various annotation artifacts.

Of course, the most influential Winograd dataset, WSC273 (Section 2.7.1), was constructed manually and has only 273 problems, with no training set.

In the context of evaluating model accuracy on Winograd datasets, the confidence intervals and p-values from Sections 3.8.1 and 3.8.2 are thought experiments. We can imagine a large space of problems, which may not actually exist, of which our test set is a random sample, though it may not actually be random. We can imagine that the model treats each schema independently, and we can work around the lack of independence of problems within a schema by bootstrapping.

If a difference in accuracy between models or datasets or scoring methods is not "statistically significant," we may be justified in treating those models or datasets or scoring methods as equivalent; on the other hand, if the difference is "statistically significant," that may not be enough to conclude that they are actually not equivalent.

When we present our results in Chapter 4, we will include bootstrapped p-values and confidence intervals for problems and for schemas.

Chapter 4

Experiments and Results

In Chapter 3, we explained how we approached the problem of evaluating common-sense reasoning in language models. In particular, we described a new test of common sense based on adversarial schemas, which generalize Winograd schemas, and we presented a new evaluation protocol based in part on consistency metrics.

In this chapter, we describe a series of experiments that implement our protocol to evaluate common-sense reasoning in three pretrained transformer-based language models, along with our results and some discussion.

In Section 4.1, we explain how we set up our experiments, including the models, datasets, and subsets and perturbations of datasets we used.

In Section 4.2, we measure accuracy on WSC266 and WSC273, and identify the best scoring method for each model. In Section 4.3, we measure consistency on perturbations of WSC266. In Section 4.4, we quantify associativity on WSC266.

In Section 4.5, we measure accuracy on Reid250, and identify the new best scoring method for each model. In Section 4.6, we measure invertible consistency on Reid250. In Section 4.7, we measure accuracy and consistency on the dyads of Reid250×2.

4.1 Experimental Setup

Table 4.1 shows the models we used, each of which is the largest model available in its family, and each of which has been implemented in Python using a library called Transformers [105]. Refer to Section 2.2 for more details on the models and Section 3.1 for our implementations of stochastic fill-in-the-blank for each one.

Family	Size	No. Parameters	Implementation
GPT-2	*xl*	1.6 billion	Hugging Face [36]
RoBERTa	*large*	355 million	Hugging Face [35]
T5	*11b*	11 billion	Hugging Face [37]

Table 4.1: Models used in our experiments.

As stated in Section 2.9.6, we did not finetune the models on a downstream task. However, as we noted in Section 2.8.3, the T5 model's supervised pretraining includes two Winograd datasets, though not in a fill-in-the-blank format. Also, as stated in Section 2.8.4, by 2020, at least one important pretraining dataset, that of GPT-3, had been contaminated by about half the test questions from WSC273.

We ran the models on a server, called Cedar, maintained by the Digital Research Alliance of Canada (formerly Compute Canada). The largest model, T5, is too big to run on a typical personal computer or even academic computing environment.

Table 4.2 shows the datasets we used, with notes to follow.

Dataset	Perturbation 1	Subset	Perturbation 2	Size	Equal Length
WSC273	original	all	none	273	0.681, 0.593
WSC266	original	all	none	266	0.684, 0.609
			no candidates	266	0.684, 0.609
			inverted	266	0.789, 0.669
		associative	none	50	0.560, 0.600
			no candidates	50	0.560, 0.600
		non-assoc.	none	216	0.713, 0.611
			no candidates	216	0.713, 0.611
	switched	all	none	140	0.829, 0.700
	adjectival	all	none	174	0.897, 0.862
	unbalanced	all	none	266	0.000, 0.000
Reid250	original	all	none	250	0.752, 0.704
			inverted	250	0.688, 0.656
Reid250×2	(3.74a) true, false	all	none	500	0.752, 0.704
			by key	500	0.776, 0.728
			by boolean	500	1.000, 1.000

Table 4.2: Datasets used in our experiments. The last column is the share of problems with equal-length answers under GPT-2/RoBERTa (same value) and under T5.

The table puts the following generalized perturbations in a separate column: the no-candidate perturbation, which can be applied to any Winograd schema; inversion or solution by key, which can be applied to any adversarial schema; and solution by boolean, which can be applied to any adversarial dyad.

The table also includes the share of problems with equal-length answers under GPT-2/RoBERTa (same value) and under T5 (Section 3.7.4). In every experiment, we calculated separate results for that subset, whether we report them or not.

For the implementation of stochastic fill-in-the-blank, including scoring methods,

see Section 3.1.1 for GPT-2, Section 3.1.2 for RoBERTa, and Section 3.1.3 for T5.

For the WSC273 dataset, see Sections 2.7.1 and 3.2. For WSC266, see Section 3.3.

For Reid250 and Reid250×2, see Section 3.6.2.

For adversarial dyads in general, e.g., Reid250×2, see Section 3.5.6.

For the associative subset of WSC266, see Sections 2.7.1 and 3.4.1.

For the switchable subset and switched perturbation of WSC266, see Sections 2.9.1 and 3.4.2. For the inverted perturbation of WSC266, see Sections 2.9.3 and 3.4.3. For the adjectival and unbalanced perturbations of WSC266, see Section 3.4.4. For the no-candidate perturbation of a Winograd dataset, e.g., WSC266, see Section 2.9.4.

For inversion or solution by key of adversarial schemas, see Section 3.5.5. For solution by key and by boolean of adversarial dyads, see Section 3.5.6.

For the evaluation protocol, see Section 3.7.6. For the definitions of our problem-based and schema-based consistency metrics, see Sections 3.7.2 and 3.7.3 respectively.

We report statistical significance where appropriate, as discussed in Section 3.8.4.

In the figures in this chapter, if the colours are unavailable or illegible, note that in each bar graph, the items in the legend, from top to bottom and left to right, correspond to the individual bars, from left to right, in each group of bars.

4.2 Accuracy on Winograd Schemas

Following our evaluation protocol, we begin by calculating accuracy on WSC266. We evaluate scoring methods for GPT-2 in Section 4.2.1, for RoBERTa in Section 4.2.2, and for T5 in Section 4.2.3, and we compare the best accuracies in Section 4.2.4.

Our goal is to answer four questions. First, which scoring methods are the most accurate? Second, how accurate are the models at best? Third, did our changes to WSC273 to create WSC266 affect model accuracy? Fourth, are the models more accurate on problems where the answer options have the same number of tokens?

We also try out our consistency metrics, measuring the consistency of each model as we switch between some of the most accurate scoring methods.

Where relevant, we mention how our results compare to the published performance of these and other models on WSC273 (e.g., Table 2.4 in Section 2.8.4).

4.2.1 Accuracy of GPT-2 Scoring Methods

Table 4.3 shows problem accuracy a and schema accuracy A for GPT-2 on WSC266, under ten scoring methods, on problems with any-length answers and equal-length answers. Smart scoring at limits 2 and 3 was less accurate than at limit 1, so we do not report results for limits $n > 1$. Model loss is mean all-but-one scoring ($*$).

Scoring Method	Take Mean?	a, Any Length	a, Equal Length	A, Any Length	A, Equal Length
smart scoring	no	0.744	0.720	0.496	0.440
at limit 1	yes	0.744	0.720	0.496	0.440
partial	no	0.726	0.698	0.474	0.418
scoring	yes	0.726	0.698	0.474	0.418
all-but-one	no	0.650	0.632	0.308	0.264
scoring	yes*	0.635	0.632	0.278	0.264
full scoring	no	0.650	0.632	0.308	0.264
	yes	0.620	0.632	0.241	0.264
normalized	no	0.613	0.621	0.233	0.253
full scoring	yes	0.613	0.621	0.233	0.253

Table 4.3: Accuracy of scoring methods for GPT-2 on WSC266.

Taking the mean never improves the accuracy of a scoring method, and it has no effect at all on smart scoring (at limit 1), partial scoring, or normalized full scoring, so we restrict our attention to non-mean scoring. Of course, as a mathematical certainty, taking the mean has no effect on problems with equal-length answers.

In WSC266, no problem statement starts with the target pronoun, so full scoring and all-but-one scoring give identical results, since we are not taking the mean.

The ranking of scoring methods is straightforward: without mentioning statistical significance yet, from most to least accurate, we have smart scoring, partial scoring, full scoring tied with all-but-one scoring, and normalized full scoring.

We identify all-but-one scoring with full scoring, and we reject normalized full scoring as less accurate than full scoring. Since we already decided not to take the mean, that leaves three scoring methods: full, partial, and smart.

Figure 4.1 displays the key results from Table 4.3 with 95% confidence intervals.

Moving on to our second question, evidently, the highest accuracies are achieved under smart scoring: $a = 0.744$ and $A = 0.496$. Table 4.4 shows the differences in

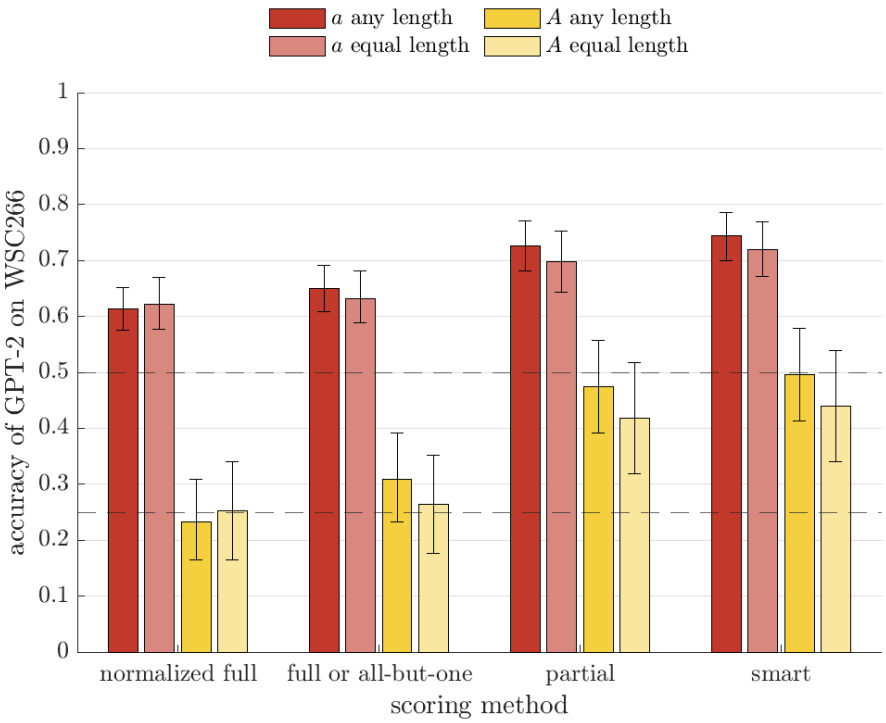

Figure 4.1: Accuracy of scoring methods for GPT-2 on WSC266.

accuracy between full, partial, and smart scoring for GPT-2 on WSC266. It also includes the statistical significance, i.e., the p-value, of those differences.

Metric	Answer Length	Partial − Full	p	Smart − Full	p	Smart − Partial	p
a	any	+0.075	$< .001$	+0.094	$< .001$	+0.019	.5
	equal	+0.066	.006	+0.088	$< .001$	+0.022	.4
A	any	+0.165	$< .001$	+0.188	$< .001$	+0.023	.7
	equal	+0.154	$< .001$	+0.176	$< .001$	+0.022	.8

Table 4.4: Difference in scoring methods for GPT-2 on WSC266.

Clearly, partial scoring and smart scoring are not significantly different, $p \geq .4$, and both are significantly more accurate than full scoring, $p \leq .006$.

For the record, all twenty problem accuracies in Table 4.3 are significantly better than chance (0.5), $p \leq .001$. The schema accuracies that are significantly better than chance (0.25) are those achieved under partial, mean partial, smart, and mean smart scoring, $p < .001$. Under the other six scoring methods, including full scoring, the schema accuracies are not significantly different from chance, $p \geq .1$.

Differences in schema accuracy are less significant than differences in problem accuracy mainly because there are always half as many schemas as problems.

We saw in Section 3.7.1 that a model can have better than random problem accuracy and worse than random schema accuracy ($a > 0.5$ and $A < 0.25$) for a sufficiently low rate of anti-solved schemas, $A'' \approx 0$. That did occur in this experiment: under mean full and mean normalized full scoring, any-length answers. A'' was never higher than 0.023, which was its value under partial and mean partial scoring.

It is clear that, under several scoring methods, GPT-2 is able to achieve much better than random accuracy on the Winograd Schema Challenge.

Moving on to our third question, we compare model accuracy across datasets. Table 4.5 shows problem accuracy a and schema accuracy A for GPT-2, under full, partial, and smart scoring, on WSC273 and WSC266, on problems with any-length answers and equal-length answers. It also includes the differences in accuracy between the datasets and the statistical significance of those differences.

Scoring Method	Metric	Answer Length	WSC273	WSC266	WSC266 − WSC273	p
smart scoring	a	any	0.736	0.744	+0.008	.7
		equal	0.710	0.720	+0.010	.8
	A	any	0.482	0.496	+0.014	.8
		equal	0.419	0.440	+0.020	.7
partial scoring	a	any	0.725	0.726	0.000	$> .9$
		equal	0.699	0.698	−0.001	$> .9$
	A	any	0.474	0.474	−0.001	$> .9$
		equal	0.419	0.418	−0.002	$> .9$
full scoring	a	any	0.648	0.650	+0.002	.9
		equal	0.634	0.632	−0.002	$> .9$
	A	any	0.307	0.308	+0.002	$> .9$
		equal	0.269	0.264	−0.005	$> .9$

Table 4.5: Comparing accuracy of GPT-2 on WSC266 and WSC273.

Clearly, accuracies on WSC266 and WSC273 are not significantly different, $p \geq .7$. Our changes to WSC273 have not affected GPT-2's ability to solve the dataset.

There are some relevant results in the literature. Comparing full scoring to partial scoring on WSC273, any-length answers, we find a significant increase in both problem and schema accuracy: $\Delta a = 0.077$, $p < .001$, and $\Delta A = 0.168$, $p < .001$. Our results are consistent with Trinh and Le [98], who found that partial scoring "outperforms"

full scoring "by a large margin" for their custom language model, and with Radford et al. [76], who found that partial scoring is more accurate than full scoring for GPT-2.

The accuracy we obtained on WSC273 with partial scoring, 0.725, is higher than the value of 0.7070 reported by Radford et al. [76] with the same model: $\Delta a = -0.018$, $p = .5$. The difference is not significant, but our revised GPT-2 accuracy is identical to a later state-of-the-art reported by Kocijan et al. [44], who finetuned BERT on Winograd training datasets. It is not clear why GPT-2 is more accurate now. Our preprocessing of the dataset (Section 3.2) seems like a more likely culprit than our implementation of partial scoring (Section 3.1.1). In any case, this may illustrate the value of stating one's methodology and reporting statistical significance.

Moving on to our fourth question, we consider the effect of answer length on accuracy. For WSC266, referring back to Table 4.3, we compare each accuracy on all problems to the corresponding accuracy on problems with equal-length answers, twenty comparisons in all, and find no significant differences, $p \geq .3$. For WSC273, referring back to Table 4.5, we make the same comparisons, six in all, and again find no significant differences, $p \geq .3$. On both WSC266 and WSC273, GPT-2 appears to be fairly robust with respect to the number of tokens in the answer options.

Finally, we try out our consistency metrics. Again, our three scoring methods are full, partial, and smart scoring. Recall that smart scoring is identical to partial scoring unless the target pronoun is at the end of the problem statement, which occurs in only 6.8% of problems in WSC266 and 7.3% in WSC273. Therefore, we expect the consistency between partial and smart scoring to be very high, and we report it only for comparison to the consistency between partial and full scoring.

Table 4.6 shows consistency c, consistent accuracy c_a, preserved accuracy c_p, schema consistency C, strict schema consistency C^*, consistent schema accuracy C_a, and preserved schema accuracy C_p for GPT-2 on WSC273 and WSC266 when we change from full scoring to partial scoring and, for comparison, from partial scoring to smart scoring, both of which changes increase accuracy.

As expected, the consistencies from partial to smart scoring are very high.

The consistencies from full to partial scoring are not high. On WSC266, the change increased problem accuracy by 7.5 percentage points and schema accuracy by 16.5 (Δa, ΔA). It did so by changing the predicted answers on 31.6% of problems

Dataset	Answer Length	c	c_a	c_p	C	C^*	C_a	C_p
From Partial Scoring to Smart Scoring								
WSC266	any	0.974	0.722	0.995	0.970	0.962	0.474	1.000
	equal	0.967	0.692	0.992	0.967	0.956	0.418	1.000
WSC273	any	0.967	0.714	0.985	0.956	0.949	0.460	0.969
	equal	0.957	0.683	0.977	0.946	0.935	0.398	0.949
From Full Scoring to Partial Scoring								
WSC266	any	0.684	0.530	0.815	0.617	0.496	0.203	0.659
	equal	0.637	0.484	0.765	0.593	0.440	0.143	0.542
WSC273	any	0.689	0.531	0.819	0.620	0.504	0.204	0.667
	equal	0.645	0.489	0.771	0.602	0.452	0.151	0.560

Table 4.6: Consistency of GPT-2 with respect to scoring method.

and 50.4% of schemas ($1 - c$, $1 - C^*$). 18.5% of problems and 34.1% of schemas that were solved under full scoring were not solved under partial scoring ($1 - c_p$, $1 - C_p$).

To go into a little more detail, 27.1% of schemas gained accuracy from the change, 11.3% lost accuracy, and 61.7% (C) kept the same accuracy. But 19.5% of the last group, making up 12.0% of all schemas, kept the same accuracy while changing the predicted answers. In short, 23.3% of schemas lost or only maintained accuracy while changing answers, 27.1% gained accuracy, and 49.6% (C^*) were unchanged.

As Table 4.6 shows, the consistency values for WSC273 are almost identical to the corresponding values for WSC266 across the board.

4.2.2 Accuracy of RoBERTa Scoring Methods

Moving on to RoBERTa, Table 4.7 shows problem accuracy a and schema accuracy A for that model on WSC266, under six scoring methods, on problems with any-length answers and equal-length answers. Model loss is mean multi-mask scoring ($*$).

Taking the mean improves the accuracy of both statement scoring and multi-mask scoring, but it worsens the accuracy of answer scoring. We restrict our attention to mean statement scoring, mean multi-mask scoring, and (non-mean) answer scoring.

Ranking the scoring methods from most to least accurate, we have mean statement scoring, mean multi-mask scoring, and answer scoring.

We reject answer scoring for being both less accurate and more computationally intensive than multi-mask scoring. Statement scoring is far more computationally

Scoring Method	Take Mean?	a, Any Length	a, Equal Length	A, Any Length	A, Equal Length
statement	no	0.778	0.791	0.556	0.582
scoring	yes	0.786	0.791	0.571	0.582
multi-mask	no	0.744	0.775	0.489	0.549
scoring	yes*	0.759	0.775	0.519	0.549
answer	no	0.718	0.758	0.444	0.527
scoring	yes	0.699	0.758	0.406	0.527

Table 4.7: Accuracy of scoring methods for RoBERTa on WSC266.

intensive than either of the other two scoring methods, but at least it yields higher accuracy. Since we already decided how to handle means, that leaves two scoring methods: mean multi-mask or loss scoring and mean statement scoring. We may still report some results for answer scoring for purposes of comparison.

Figure 4.2 displays the key results from Table 4.7 with 95% confidence intervals.

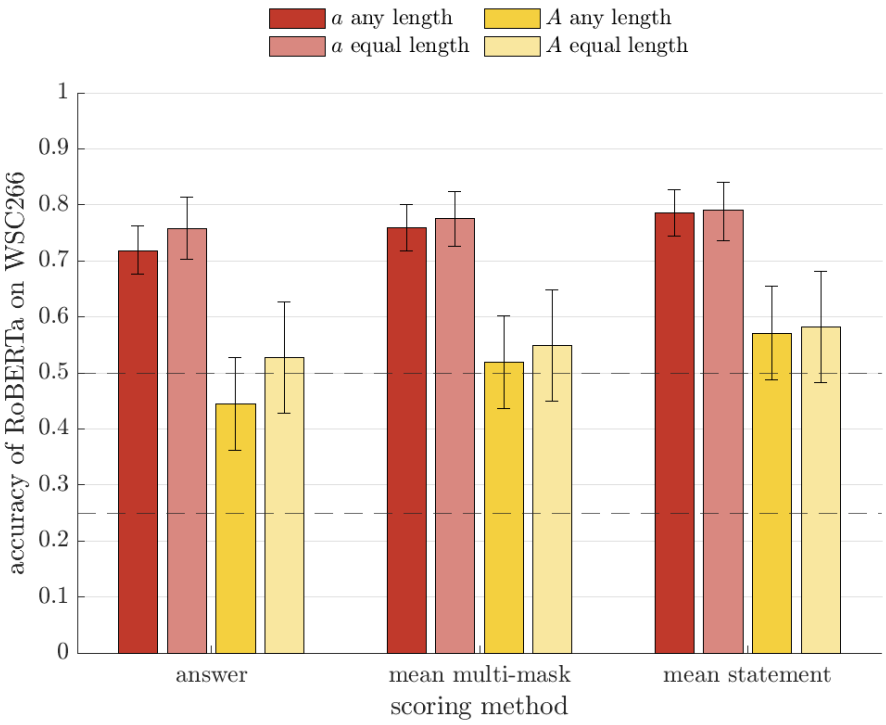

Figure 4.2: Accuracy of scoring methods for RoBERTa on WSC266.

Moving on to our second question, evidently, the highest accuracies are achieved under mean statement scoring: $a = 0.778$ and $A = 0.556$. Table 4.8 shows the

differences in accuracy between answer scoring, mean multi-mask scoring, and mean statement scoring for RoBERTa on WSC266, with statistical significance.

Metric	Answer Length	Multi-Mask − Answer	p	Statement − Answer	p	Statement − Multi-Mask	p
a	any	+0.041	.07	+0.068	.002	+0.026	.3
	equal	+0.016	.5	+0.033	.3	+0.016	.6
A	any	+0.075	.08	+0.128	.003	+0.053	.3
	equal	+0.022	.8	+0.055	.4	+0.033	.6

Table 4.8: Difference in scoring methods for RoBERTa on WSC266

Mean statement scoring, despite its relatively high cost, is not significantly more accurate than mean multi-mask scoring, $p \geq .3$. On all problems from WSC266, mean statement scoring is more accurate than answer scoring, $p \leq .003$, and mean multi-mask scoring is marginally more accurate, $p \leq .08$. However, when we restrict to problems with equal-length answers, neither is significantly more accurate, $p \geq .3$.

For the record, all twenty-four accuracies in Table 4.7 are significantly better than chance, $p < .001$. In particular, RoBERTa never has worse than random schema accuracy, although the rate A'' of anti-solved schemas was never higher than 0.008, which was its value under answer scoring and mean answer scoring.

It is clear that, under several scoring methods, RoBERTa is able to achieve much better than random accuracy on the Winograd Schema Challenge.

Moving on to our third question, we compare model accuracy across datasets. Table 4.9 shows problem accuracy a and schema accuracy A for RoBERTa, under answer scoring, mean multi-mask scoring, and mean statement scoring, on WSC273 and WSC266, on problems with any-length answers and equal-length answers. It also includes the differences in accuracy and their statistical significance.

Clearly, accuracies on WSC266 and WSC273 are not significantly different, $p \geq .8$. Our changes to WSC273 have not affected RoBERTa's ability to solve the dataset.

The accuracy we obtained on WSC273 with mean multi-mask scoring, which is model loss, 0.755, is identical to the 2019 state-of-the-art by Ye et al. [106], who finetuned BERT on a Winograd training set, among other data. As we already noted, the accuracy we obtained with mean statement scoring, 0.780, is higher than that value: $\Delta a = -0.026$, $p = .3$. Indeed, that new retroactive state-of-the-art would not be surpassed until Sakaguchi et al. [84, 85] finetuned RoBERTa on WinoGrande

Scoring Method	Metric	Answer Length	WSC273	WSC266	WSC266 − WSC273	p
mean statement	a	any	0.780	0.786	+0.004	.9
		equal	0.796	0.791	+0.006	.8
	A	any	0.562	0.571	+0.006	.9
		equal	0.591	0.582	+0.011	.8
mean multi-masked	a	any	0.755	0.759	+0.005	.9
		equal	0.774	0.775	+0.001	$> .9$
	A	any	0.511	0.519	+0.008	.9
		equal	0.548	0.549	+0.001	$> .9$
answer	a	any	0.714	0.718	+0.005	.9
		equal	0.753	0.758	−0.004	.9
	A	any	0.438	0.444	+0.009	.9
		equal	0.516	0.527	−0.009	.9

Table 4.9: Comparing accuracy of RoBERTa on WSC266 and WSC273.

to obtain an accuracy of 0.901 on WSC273: as expected, their value is significantly higher than the pretrained model's best accuracy, $\Delta a = +0.121$, $p < .001$.

The accuracy on WSC273 with our implementation of mean statement scoring, 0.780, is significantly higher than the value of 0.694 reported by Zhou et al. [109] on the same model with, apparently, the same scoring method, $p < .001$. On the other hand, the accuracy we obtained with our method of mean answer scoring, 0.692, is almost identical to their value, $p = .9$. The cause of the disparity is unknown.

Moving on to our fourth question, we consider the effect of answer length on accuracy. For WSC266, referring back to Table 4.7, we compare each accuracy on all problems to the corresponding accuracy on problems with equal-length answers, twelve comparisons in all, and find exactly one case of significant differences: mean answer scoring, which we already rejected, $\Delta a = +0.059$, $p = .03$ and $\Delta A = 0.121$, $p = .02$. Otherwise, the differences are insignificant, $p \geq .1$, and indeed $p \geq .6$ for our chosen methods of mean multi-mask scoring and mean statement scoring.

For completeness, we make the same comparisons for WSC273, twelve in all, and again find exactly one case of significant differences: again, mean answer scoring, $\Delta a = +0.060$, $p = .03$ and $\Delta A = 0.122$, $p = .02$. Otherwise, the differences are insignificant, $p \geq .1$, and indeed $p \geq .5$ for our chosen scoring methods.

Since the outlier, mean answer scoring, is less accurate than answer scoring, which is less accurate than either of our chosen scoring methods, it seems fair to say that

on both WSC266 and WSC273, RoBERTa is robust with respect to the number of tokens in the answer options for the best scoring methods.

Finally, we try out our consistency metrics. Again, our chosen scoring methods are mean multi-mask scoring and mean statement scoring.

Table 4.10 shows consistency c, consistent accuracy c_a, preserved accuracy c_p, schema consistency C, strict schema consistency C^*, consistent schema accuracy C_a, and preserved schema accuracy C_p for RoBERTa on WSC273 and WSC266 when we change from mean multi-mask scoring to the more accurate mean statement scoring.

Dataset	Answer Length	c	c_a	c_p	C	C^*	C_a	C_p
From Mean Multi-Mask Scoring to Mean Statement Scoring								
WSC266	any	0.801	0.673	0.886	0.707	0.654	0.398	0.768
	equal	0.863	0.714	0.922	0.747	0.736	0.440	0.800
WSC273	any	0.806	0.670	0.888	0.715	0.664	0.394	0.771
	equal	0.849	0.710	0.917	0.742	0.720	0.441	0.804

Table 4.10: Consistency of RoBERTa with respect to scoring method.

The consistencies are mediocre. On WSC266, problem accuracy increased by 2.6 percentage points and schema accuracy by 5.3 (Δa, ΔA) as a result of changing the predicted answers on 19.9% of problems and 34.6% of schemas ($1-c$, $1-C^*$). 11.4% of problems and 23.2% of schemas that were solved under mean multi-mask scoring were no longer solved under mean statement scoring ($1-c_p$, $1-C_p$).

To go into a little more detail again, 17.3% of schemas gained accuracy, 12.0% lost accuracy, and 70.7% (C) kept the same accuracy. Only 7.4% of the last group, making up 5.3% of all schemas, kept the same accuracy while changing the predicted answers. In short, 17.3% of schemas lost or only maintained accuracy while changing answers, another 17.3% gained accuracy, and 65.4% (C^*) were unchanged.

As Table 4.10 shows, the consistency values for WSC273 are almost identical to the corresponding values for WSC266 across the board.

4.2.3 Accuracy of T5 Scoring Methods

Moving on to T5, its parametrized scoring (3.29) admits eight variants: mean or no mean; task prefix or no prefix; and EOS, counting the end-of-string token, or no EOS, discounting it. Model loss is mean EOS scoring, and it can use a prefix or not.

To avoid a large, unilluminating table of accuracies, we collect the following general facts about problem accuracy on all problems from WSC266. First, a prefix always increases accuracy, regardless of the other parameters, $+0.015 \leq \Delta a \leq +0.045$.

Second, discounting EOS increases accuracy by a relatively large amount if we also take the mean, $+0.064 \leq \Delta a \leq +0.094$, whereas discounting EOS barely changes accuracy if we do not also take the mean, $-0.004 \leq \Delta a \leq +0.023$.

Third, taking the mean barely increases accuracy if we discount EOS, $\Delta a = +0.015$, and it decreases accuracy if we count EOS, $-0.056 \leq \Delta a \leq -0.053$.

Our goal is not to adjust the parameters of masked scoring to maximize accuracy on one dataset. The parameters were justified in Section 3.1.3. We included the prefix parameter because the model was trained with prefixes. The fact that accuracy increases with prefixes may indicate that the model's text-to-text pretraining on DPR and WNLI-2 is generalizing to stochastic fill-in-the-blank (Section 2.8.3). For that reason, we will continue to report results both with and without a prefix.

We included the end-of-string token parameter to prevent one apparently spurious token from dominating the mean. The fact that, basically, discounting EOS increases accuracy only when we take the mean, and taking the mean increases accuracy only when we discount EOS, may indicate that we were successful.

For example, the highest accuracy is achieved under mean prefix no-EOS masked scoring. Compared to that, not taking the mean yields $\Delta a = -0.015$, and counting EOS yields $\Delta a = -0.064$. But flipping both parameters yields $\Delta a = -0.011$: with no mean, we no longer need to prevent any one token from dominating the mean.

In light of the above discussion, we will discount EOS if and only if we take the mean: unless otherwise stated, all our scoring methods follow this rule.

We decided to report accuracies for the following scoring methods: *mean prefix scoring*, the most accurate method; *prefix scoring* with no mean, the second most accurate method, as well as the most accurate without a mean and the most accurate that counts EOS; *mean scoring* with no prefix, the most accurate method without a prefix; and *basic scoring* with no mean or prefix, to round out the list.

Model loss did not make the list. But, for comparison, we will report *prefix loss scoring* (mean, EOS) and *loss scoring* with no prefix (also mean, EOS).

Table 4.11 shows problem accuracy a and schema accuracy A for T5 on WSC266,

under various scoring methods, on problems with any-length answers and equal-length answers. Model loss is, of course, represented by loss and prefix loss ($*$).

Scoring Method	a, Any Length	a, Equal Length	A, Any Length	A, Equal Length
mean prefix	0.868	0.870	0.737	0.741
prefix	0.857	0.864†	0.714	0.728†
mean	0.853	0.840	0.707	0.679
basic	0.816	0.809‡	0.632	0.617‡
prefix loss*	0.805	0.864†	0.609	0.728†
loss*	0.759	0.809‡	0.519	0.617‡

Table 4.11: Accuracy of scoring methods for T5 on WSC266.

Once again, as a mathematical certainty, the mean has no effect on problems with equal-length answers. For that reason, on that subset, prefix loss scoring is identical to prefix scoring (†), and loss scoring is identical to basic scoring (‡).

Apart from that, ranking the scoring methods from most to least accurate, we have mean prefix, prefix, mean, basic (i.e., no mean, no prefix), prefix loss, and loss.

Figure 4.3 displays the results from Table 4.11 with 95% confidence intervals.

Moving on to our second question, evidently, the highest accuracies are achieved under mean prefix scoring, and they are very good: $a = 0.868$ and $A = 0.737$. Table 4.12 shows the differences in accuracy, for T5 on WSC266, between prefix loss scoring, a sort of default; mean scoring, which properly discounts EOS without bothering with a prefix; and mean prefix scoring, which is the most accurate.

Metric	Answer Length	Mean − Prefix Loss	p	Mean Prefix − Prefix Loss	p	Mean Prefix − Mean	p
a	any	+0.049	.03	+0.064	.003	+0.015	.5
	equal	−0.025	.4	+0.006	.8	+0.031	.5
A	any	+0.098	.02	+0.128	.003	+0.030	.2
	equal	−0.049	.4	+0.012	.9	+0.062	.3

Table 4.12: Difference in scoring methods for T5 on WSC266.

Clearly, mean and mean prefix scoring are not significantly different, $p \geq .2$, and compared to prefix loss scoring, both are significantly more accurate on problems with any-length answers, $p \leq .03$, but not on problems with equal-length answers, $p \geq .4$.

For the record, all twenty-four accuracies in Table 4.11 are significantly better than

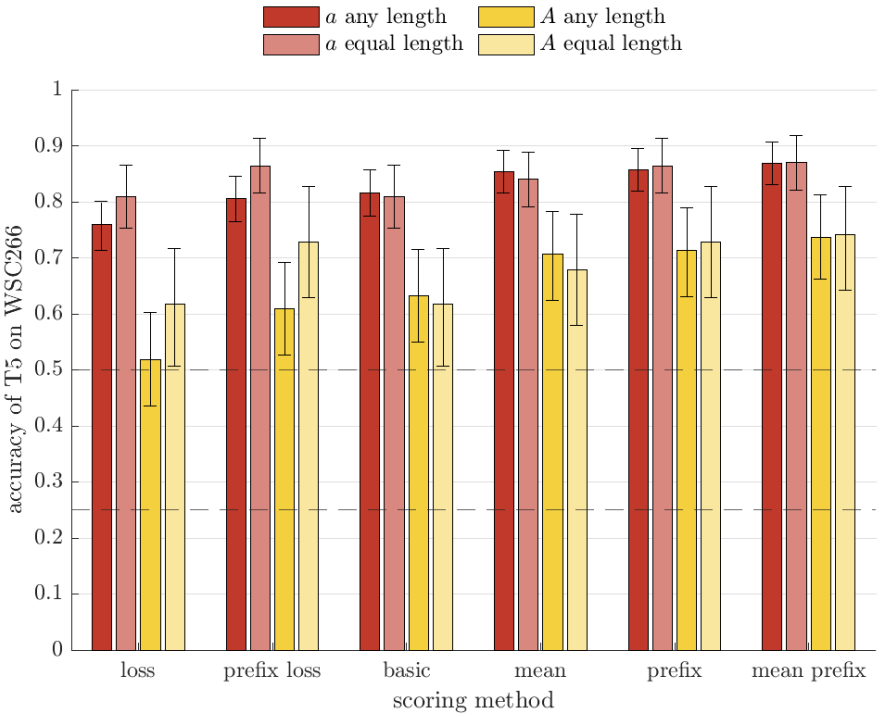

Figure 4.3: Accuracy of scoring methods for T5 on WSC266.

chance, $p < .001$. In particular, T5 never has worse than random schema accuracy, although the rate A'' of anti-solved schemas was identically 0.

It is abundantly clear that, under several scoring methods, T5 is able to achieve much better than random accuracy on the Winograd Schema Challenge.

Moving on to our third question, we compare model accuracy across datasets. Table 4.13 shows problem accuracy a and schema accuracy A for T5, under the scoring methods from Table 4.12, on WSC273 and WSC266, on problems with any-length answers and equal-length answers, with changes and statistical significance.

Clearly, for these scoring methods, accuracies on WSC266 and WSC273 are not significantly different, $p \geq .7$. The three scoring methods shown are representative of all six. Our changes to WSC273 have not affected T5's ability to solve the dataset.

The highest accuracy we obtained on WSC273, 0.861, which was with mean prefix scoring, is lower than the 2019 state-of-the-art of 0.901 by Sakaguchi et al. [84, 85], who finetuned RoBERTa on WinoGrande: $\Delta a = -0.040$, $p = .04$.

Our highest accuracy is not significantly lower than the zero-shot value of 0.883

Scoring Method	Metric	Answer Length	WSC273	WSC266	WSC266 − WSC273	p
mean prefix	a	any	0.861	0.868	+0.008	.7
		equal	0.870	0.870	0.000	$> .9$
	A	any	0.723	0.737	+0.014	.8
		equal	0.741	0.741	0.000	$> .9$
mean	a	any	0.850	0.853	+0.004	.9
		equal	0.840	0.840	0.000	$> .9$
	A	any	0.701	0.707	+0.006	.9
		equal	0.679	0.679	0.000	$> .9$
prefix loss	a	any	0.802	0.805	+0.002	.9
		equal	0.864	0.864	0.000	$> .9$
	A	any	0.606	0.609	+0.003	$> .9$
		equal	0.728	0.728	0.000	$> .9$

Table 4.13: Comparing accuracy of T5 on WSC266 and WSC273.

reported for GPT-3 on WSC273 by Brown et al. [5, 6]: $\Delta a = -0.022$, $p = .3$.

Moving on to our fourth question, we consider the effect of answer length on accuracy. For WSC266, referring back to Table 4.11, we compare each accuracy on all problems to the corresponding accuracy on problems with equal-length answers, twelve comparisons in all, and find exactly one case of significant differences: prefix loss scoring, $\Delta a = +0.060$, $p = .03$ and $\Delta A = 0.119$, $p = .03$. Otherwise, the differences are insignificant, $p \geq .09$, and indeed $p \geq .6$ apart from loss scoring.

For completeness, we make the same comparisons for WSC273, twelve in all, and again find exactly one case of significant differences: again, prefix loss scoring, $\Delta a = +0.062$, $p = .02$ and $\Delta A = 0.123$, $p = .03$. Otherwise, the differences are insignificant, $p \geq .1$, and indeed $p \geq .7$ apart from loss scoring.

Since loss and prefix loss scoring are less accurate than our chosen scoring methods, it seems fair to say that on both WSC266 and WSC273, T5 is robust with respect to the number of tokens in the answer options for the best scoring methods.

Finally, we try out our consistency metrics. Again, we take prefix loss scoring as a sort of default and compare it to mean prefix scoring, which is the most accurate.

Table 4.14 shows consistency c, consistent accuracy c_a, preserved accuracy c_p, schema consistency C, strict schema consistency C^*, consistent schema accuracy C_a, and preserved schema accuracy C_p for T5 on WSC273 and WSC266 when we change from prefix loss scoring to the more accurate mean prefix scoring.

Dataset	Answer Length	c	c_a	c_p	C	C^*	C_a	C_p
From Prefix Loss Scoring to Mean Prefix Scoring								
WSC266	any	0.914	0.793	0.986	0.842	0.835	0.594	0.975
	equal	0.994	0.864	1.000	0.988	0.988	0.728	1.000
WSC273	any	0.912	0.788	0.982	0.839	0.832	0.584	0.964
	equal	0.994	0.864	1.000	0.988	0.988	0.728	1.000

Table 4.14: Consistency of T5 with respect to scoring method.

The consistencies are quite high. On WSC266, problem accuracy increased by 6.4 percentage points and schema accuracy by 12.8 (Δa, ΔA) as a result of changing the predicted answers on 8.6% of problems and 16.5% of schemas ($1 - c$, $1 - C^*$). Only 1.4% of problems (three) and 2.5% of schemas (two) that were solved under prefix loss scoring were no longer solved under mean prefix scoring ($1 - c_p$, $1 - C_p$).

To go into a little more detail once again, 14.3% of schemas gained accuracy, 1.5% lost accuracy, and 84.2% (C) kept the same accuracy. Only 0.9% of the last group (one schema), making up 0.8% of all schemas, kept the same accuracy while changing the predicted answers. In short, 2.3% of schemas (three) lost or only maintained accuracy while changing answers, 14.3% gained accuracy, and 83.5% (C^*) were unchanged.

As Table 4.14 shows, the consistency values for WSC273 are almost identical to the corresponding values for WSC266 across the board.

4.2.4 Model Comparison: Best Scoring Methods

Based on the results and discussion in Sections 4.2.1 through 4.2.3, it is clear that each of GPT-2, RoBERTa, and T5 is able to achieve significantly better than random accuracy on the Winograd Schema Challenge; that the choice of scoring method, in general, significantly affects each model's accuracy; that our changes to WSC273 to create WSC266 do not affect any model's ability to solve the dataset under any of the best scoring methods; and that on both datasets, each model is fairly robust with respect to answer option token counts under any of the best scoring methods.

Therefore, on Winograd datasets, we will report results for all three models, but only under smart scoring for GPT-2, mean statement scoring for RoBERTa, and mean prefix scoring for T5, which we call the *standard setup*. Also, since we based our perturbations on WSC266, we will report results on that dataset rather than on

WSC273. On the other hand, where appropriate, we will continue to report separate results for problems with any-length answers and equal-length answers.

Figure 4.4 displays the models' best accuracies on WSC266, meaning under the standard setup, on problems with any-length answers and equal-length answers.

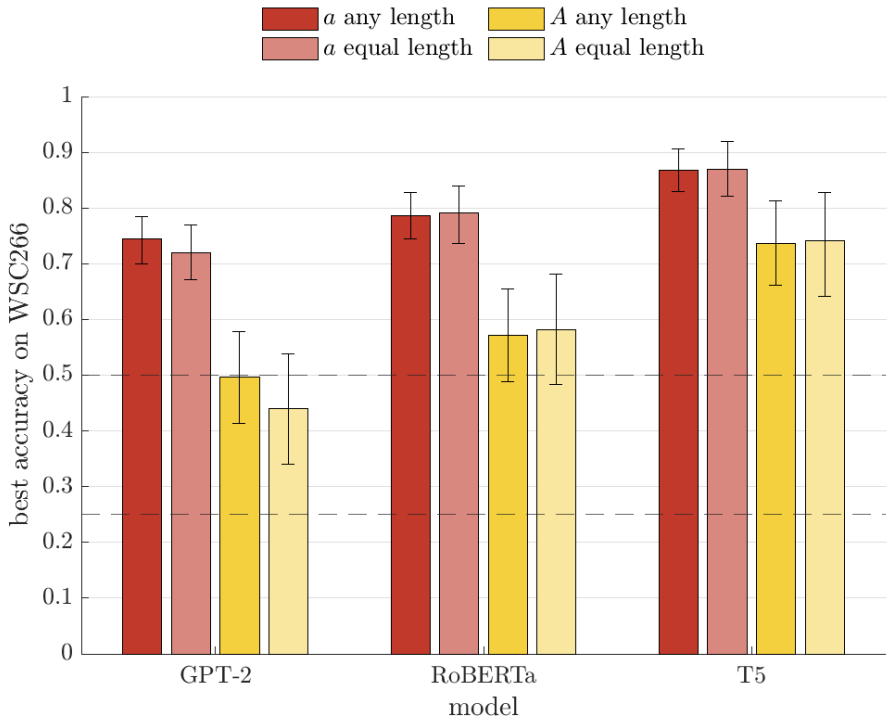

Figure 4.4: Best model accuracies on WSC266 under the standard setup.

Table 4.15 shows the differences in best accuracy between models. Clearly, T5 is more accurate than RoBERTa or GPT-2, $p \leq .005$. And RoBERTa is more accurate than GPT-2, albeit marginally on problems with any-length answers, $.08 \leq p \leq .1$, but significantly on problems with equal-length answers, $.005 \leq p \leq .009$.

Metric	Answer Length	RoBERTa − GPT-2	p	T5 − RoBERTa	p	T5 − GPT-2	p
a	any	+0.041	.08	+0.083	$< .001$	+0.124	$< .001$
	equal	+0.071	.009	+0.079	.004	+0.151	$< .001$
A	any	+0.075	.1	+0.165	$< .001$	+0.241	$< .001$
	equal	+0.143	.005	+0.158	.005	+0.301	$< .001$

Table 4.15: Comparison of best model accuracies on WSC266.

Bear in mind that the subset of WSC266 with equal-length answers differs between GPT-2 or RoBERTa and T5 (Table 4.2).

At this point, we have reported enough results that it may be helpful to formally collect them: by the *performance profile* of a model, a dataset, or a perturbation, we mean the complete set of reported accuracies and consistencies for that model, dataset, or perturbation, particularly under the best scoring methods or at least our chosen scoring methods. Currently, our profiles are limited to WSC266 and WSC273, but in the next section we extend them to perturbations of WSC266.

4.3 Perturbations of Winograd Schemas

Continuing our evaluation protocol, we calculate the accuracy and consistency of each model on perturbations of WSC266. As stated in Section 4.2.4, we report results only under the standard setup and only on perturbations of WSC266, not WSC273.

Our perturbations are as follows: switched, adjectival, unbalanced, and inverted. The first three are problem transformations, and each has a strong claim to being substantially equivalent. The fourth is a schema transformation, and whether or not it substantially preserves the content of a schema is an open question.

The no-candidate perturbation, which is certainly not content-preserving, will be treated separately when we quantify associativity on WSC266 in Section 4.4.

Our goal is to answer three questions. First, how do the perturbations affect model accuracy? Second, how consistent are the models on the perturbations? Third, is the inverted dataset substantially equivalent to its original formulation?

We address the first two questions, on perturbations in general, in Section 4.3.1. Then we focus exclusively on the third question, about inversion, in Section 4.3.2.

We will not report separate results for the problems with equal-length answers because that property varies so much between perturbations (Table 4.2).

Where relevant, we mention how our results compare to the published performance of these and other models, but there are very few directly comparable results.

4.3.1 Consistency on Perturbations

Table 4.16 shows problem accuracy a and schema accuracy A for the usual models, under the standard setup, on WSC266 and four perturbations thereof. It also includes

the differences between each perturbation and the corresponding subset of the original dataset, and the statistical significance, i.e., p-value, of those differences.

Model	Perturbation	Metric	Value	Perturbation − Original Subset	p
GPT-2	original	a	0.744	–	–
		A	0.496	–	–
	switched	a	0.693	−0.014	.6
		A	0.414	0.000	–
	adjectival	a	0.701	−0.017	.6
		A	0.448	+0.011	.9
	unbalanced	a	0.654	−0.090	$< .001$
		A	0.376	−0.120	.006
	inverted	a	0.628	−0.117	$< .001$
		A	0.278	−0.218	$< .001$
RoBERTa	original	a	0.786	–	–
		A	0.571	–	–
	switched	a	0.707	−0.079	.009
		A	0.443	−0.129	.04
	adjectival	a	0.753	−0.023	.4
		A	0.506	−0.046	.5
	unbalanced	a	0.541	−0.244	$< .001$
		A	0.083	−0.489	$< .001$
	inverted	a	0.808	+0.023	.3
		A	0.617	+0.045	.3
T5	original	a	0.868	–	–
		A	0.737	–	–
	switched	a	0.779	−0.057	.05
		A	0.571	−0.100	.09
	adjectival	a	0.816	−0.017	.5
		A	0.644	−0.023	.7
	unbalanced	a	0.722	−0.147	$< .001$
		A	0.466	−0.271	$< .001$
	inverted	a	0.805	−0.064	$< .001$
		A	0.609	−0.128	.001

Table 4.16: Accuracy on perturbations of WSC266.

Figure 4.5 displays the accuracies from Table 4.16 with 95% confidence intervals.

The most significant changes in problem accuracy, all of which are decreases, are as follows: the switched perturbation for RoBERTa, $p \leq .009$, and T5, $p \leq .05$; the unbalanced perturbation for every model, $p < .001$; and the inverted perturbation for GPT-2 and T5, $p < .001$. We get the same list from the schema accuracies.

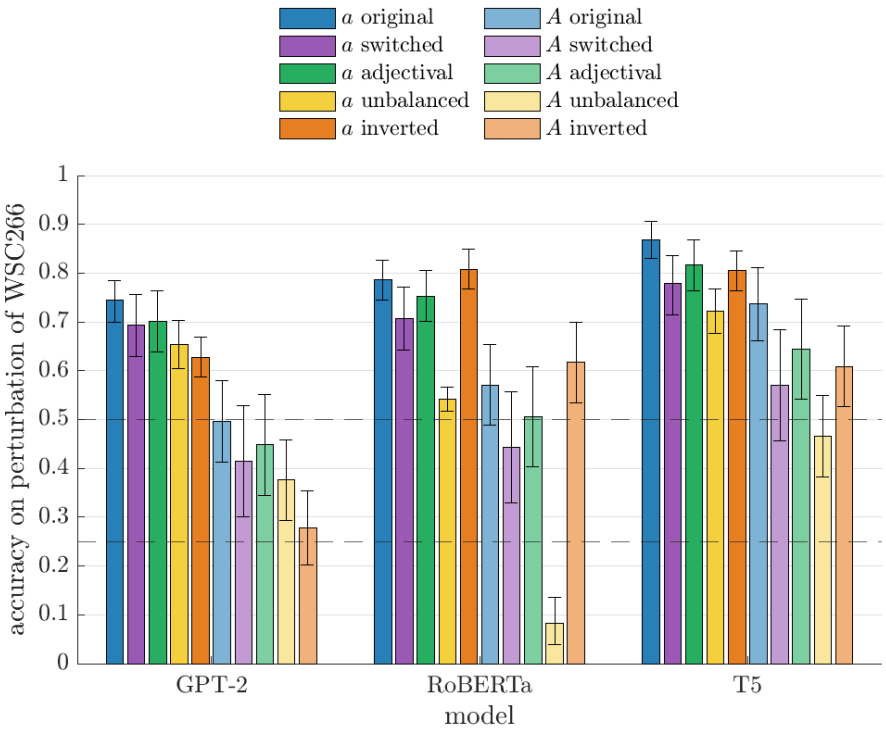

Figure 4.5: Accuracy on perturbations of WSC266.

The three largest absolute changes in accuracy are also the three largest relative changes, expressed as a fraction of the model's accuracy on the original dataset. They are as follows, $p < .001$ in all cases: GPT-2 on the inverted perturbation, $\Delta a = -0.117$ (-15.7%), $\Delta A = -0.218$ (-43.9%); RoBERTa on the unbalanced perturbation, $\Delta a = -0.244$ (-31.1%), $\Delta A = -0.489$ (-85.5%); and T5 on the unbalanced perturbation, $\Delta a = -0.147$ (-16.9%), $\Delta A = -0.271$ (-36.7%).

Indeed, on the inverted perturbation, GPT-2's schema accuracy is only marginally better than chance, $p = .5$, although the model's problem accuracy is significantly better than chance, $p < .001$. On the unbalanced perturbation, RoBERTa's schema accuracy is worse than chance, $p < .001$, and the model's problem accuracy is only marginally better than chance, $p = .2$. Other than that, every model achieves better than random problem and schema accuracy on every perturbation, $p \leq .001$.

We are now able to characterize a model by its performance profile with respect to perturbations. For example, only GPT-2 is not significantly worse on the switched perturbation; only RoBERTa is not significantly worse on the inverted perturbation.

However, the simplest way to characterize a model, as our discussion has indicated, is with respect to the unbalanced and inverted perturbations. Table 4.17 shows the relative change in accuracy between the original dataset and each of those. Note the model order, which would be the same if we used the absolute change instead.

Model	Metric	Unbalanced − Original	Inverted − Original
GPT-2	a	−12.1%	−15.7%
	A	−24.2%	−43.9%
T5	a	−16.9%	− 7.4%
	A	−36.7%	−17.3%
RoBERTa	a	−31.1%	+ 2.9%
	A	−85.5%	+ 7.9%

Table 4.17: Ranking models by change in accuracy on perturbations of WSC266.

That is, the models can be ranked according to their change in accuracy on those perturbations: for the unbalanced perturbation, RoBERTa loses the most accuracy and GPT-2 the least; for the inverted perturbation, the order is reversed.

We argued in Section 3.4.4 that the switched and adjectival perturbations are fairly similar, at least conceptually, and these results somewhat support that claim. Table 4.18 shows the differences in accuracy between the switched perturbation and the corresponding subset of the adjectival perturbation, with p-values.

Model	Metric	Switched − Adjectival	p
GPT-2	a	−0.007	.9
	A	−0.029	.7
RoBERTa	a	−0.050	.1
	A	−0.071	.3
T5	a	−0.050	.1
	A	−0.100	.09

Table 4.18: Comparing WSC266 adjectival and switched.

None of the differences are very significant, $p \geq .09$. For that reason, we can treat the switched and adjectival perturbations as a group: call it the *interchangeable family* of perturbations. Because the symmetrical subset of WSC266, which becomes the adjectival perturbation, contains the entire switchable subset, we will take the larger adjectival perturbation as the representative of the interchangeable family.

We also argued in Section 3.4.4 that the unbalanced perturbation is quite different in concept from the interchangeable family, and these results seem to support that claim as well. We will discuss the inverted perturbation in detail in Section 4.3.2, but for now, our results also indicate that it differs from every other perturbation.

Table 4.19 shows the differences in accuracy between the adjectival perturbation, as a representative of the interchangeable family, and the corresponding subsets of the unbalanced and inverted perturbations, with p-values. Note the model order.

Model	Metric	Unbalanced − Adjectival	p	Inverted − Adjectival	p
GPT-2	a	-0.086	.006	-0.103	.001
	A	-0.138	.009	-0.241	$< .001$
T5	a	-0.109	$< .001$	-0.052	.07
	A	-0.195	$< .001$	-0.115	.03
RoBERTa	a	-0.207	$< .001$	$+0.023$	.5
	A	-0.414	$< .001$	$+0.046$	.5

Table 4.19: Comparing WSC266 adjectival, unbalanced, and inverted.

The differences between the adjectival and unbalanced perturbations vary greatly between models; indeed, we already ranked the models according to their performance on the unbalanced perturbation (Table 4.17). The differences between the adjectival and inverted perturbations also vary greatly, but our ranking of models is reversed.

Moving on to our second question, Table 4.20 shows each applicable metric of consistency c, consistent accuracy c_a, preserved accuracy c_p, schema consistency C, strict schema consistency C^*, consistent schema accuracy C_a, and preserved schema accuracy C_p for each model on each perturbation of WSC266.

Recall that the switched and adjectival perturbations both have fewer than 266 problems, because they are transformations of proper subsets of the original dataset: the switchable and symmetrical subsets, respectively. According to our consistency metrics, we compare a perturbation only to the corresponding subset of WSC266.

Figure 4.6 displays the consistency scores C and C_a from Table 4.20.

In Table 4.17, we noted that GPT-2 performs particularly poorly on the inverted perturbation, and RoBERTa on the unbalanced perturbation. Indeed, the consistent accuracies are the lowest in exactly those cases: $C_a = 0.188$ for GPT-2 on WSC266 inverted, and $c_a = 0.481$ and $C_a = 0.053$ for RoBERTa on WSC266 unbalanced.

Model	Perturbation	c	c_a	c_p	C	C^*	C_a	C_p
GPT-2	switched	0.686	0.543	0.768	0.686	0.529	0.271	0.655
	adjectival	0.718	0.569	0.792	0.644	0.540	0.287	0.658
	unbalanced	0.737	0.568	0.763	0.654	0.571	0.286	0.576
	inverted	–	–	–	0.579	–	0.188	0.379
RoBERTa	switched	0.807	0.650	0.827	0.743	0.686	0.386	0.675
	adjectival	0.770	0.649	0.837	0.701	0.621	0.379	0.688
	unbalanced	0.635	0.481	0.612	0.451	0.361	0.053	0.092
	inverted	–	–	–	0.699	–	0.444	0.776
T5	switched	0.800	0.707	0.846	0.700	0.657	0.471	0.702
	adjectival	0.856	0.753	0.903	0.770	0.747	0.540	0.810
	unbalanced	0.718	0.654	0.753	0.564	0.504	0.391	0.531
	inverted	–	–	–	0.662	–	0.504	0.684

Table 4.20: Consistency on perturbations of WSC266.

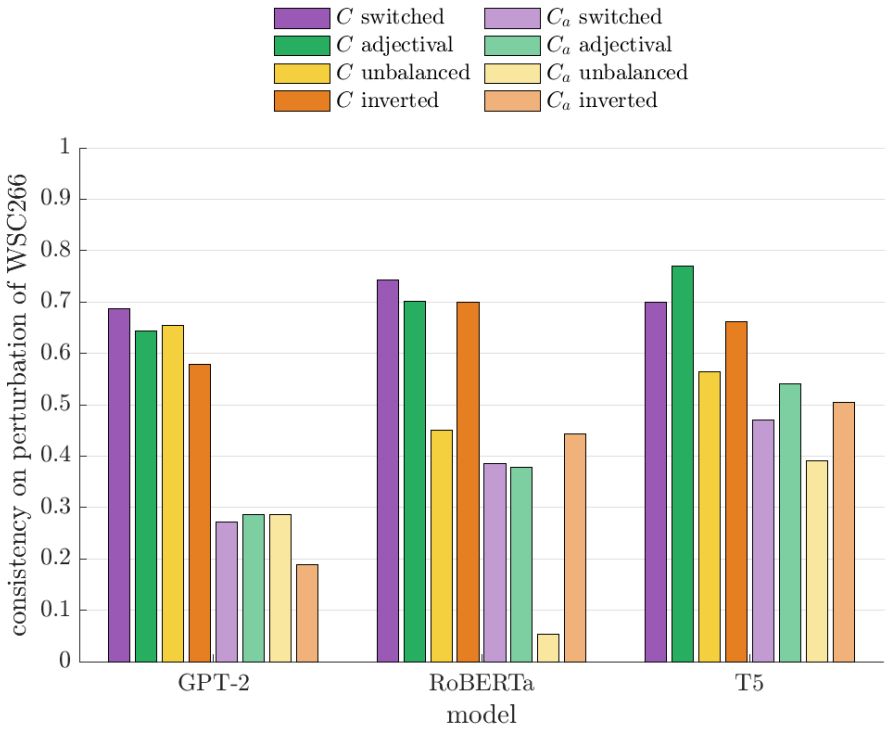

Figure 4.6: Consistency on perturbations of WSC266.

The perturbation consistencies may be compared to the scoring consistencies of the models on all problems from WS266 in Tables 4.6, 4.10, and 4.14. For example, $c = 0.684$ for GPT-2 when we change from full scoring to partial scoring; $c = 0.801$ for RoBERTa when we change from mean multi-mask scoring to mean statement scoring; and $c = 0.914$ for T5 when we change from prefix loss scoring to mean prefix scoring.

We highlight the highest consistency: T5 on the adjectival perturbation. The model's problem accuracy decreased by 1.7 percentage points (three problems) and its schema accuracy by 2.3 (two schemas) as a result of changing the predicted answers on 14.4% of problems and 25.3% of schemas. 9.7% of problems and 19.0% of schemas that were solved in the original dataset were not solved in the perturbation.

10.3% of schemas gained accuracy from the perturbation, 12.6% lost accuracy, and 77.0% kept the same accuracy. Only 3.0% of the last group (two schemas), making up 2.3% of all schemas, kept the same accuracy while changing the predicted answers. In short, 14.9% of schemas lost or only maintained accuracy while changing answers, 10.3% gained accuracy, and 74.7% were unchanged.

We would like to compare our results on WSC266 to published results, but very few of those are directly comparable. We begin with the invertible perturbation, for which there is one arguably relevant result, then move on to the switched perturbation.

Elazar et al. [23] evaluate RoBERTa on an inverted perturbation of WSC273 [21] consisting of 226 problems reportedly with answers of length one. Not all the problems were correctly inverted, and the dataset is not well preprocessed. We found that 196 inverted problems from WSC266 have unit-length answers, and indeed at most 196 of their 226 problems actually have unit-length answers (in the context of the problem).

The authors report accuracies of $a = 0.7391$ and $A = 0.4783$. Our values of $a = 0.808$ and $A = 0.617$ on the inverted perturbation of WSC266 under mean statement scoring are significantly higher: $\Delta a = +0.069$, $p = .002$ and $\Delta A = +0.138$, $p = .004$. Restricting to problems with unit-length answers, we get $a = 0.816$ and $A = 0.633$, which are higher still: $\Delta a = +0.077$, $p < .001$ and $\Delta A = +0.154$, $p < .001$.

Moving on to the switched perturbation, for which there are several arguably relevant results, we note that those are typically for WSC273, rarely for WSC285, and, of course, never for WSC266. Moreover, they typically use the switchable subset identified by Trichelair et al. [95] and, of course, never our modified version.

Trichelair et al. [95] report the switchable consistency of Trinh and Le [97]'s custom language model and Emami et al. [24]'s information retrieval system, which are not relevant models for this thesis. Emami et al. [25] report the switchable consistency of BERT on the KnowRef coreference corpus, which is not a relevant dataset.

Kocijan et al. [44] report the switchable consistency on WSC273 of GPT, not GPT-2, and BERT, under partial scoring and mean multi-mask scoring respectively, but these are still not directly comparable models. With the pretrained models, they report a consistency of $c = 0.466$ for GPT, much lower than our value of $c = 0.693$ for GPT-2 under partial scoring or $c = 0.686$ under smart scoring; and they report $c = 0.458$ for BERT, much lower than our value of $c = 0.743$ for RoBERTa under mean multi-mask scoring or $c = 0.807$ under mean statement scoring.

Abdou et al. [1] report the consistency of RoBERTa on various perturbations of WSC285, none of which we studied. Also, they describe a scoring method which is apparently a variant of mean multi-mask scoring that averages the token probabilities rather than log probabilities. It appears to be less accurate: $a = 0.6982$ on WSC285, compared to our value of $a = 0.755$ on WSC273 with mean multi-mask scoring. With the pretrained model, they report a consistency of $c = 0.6877$ on the closest match to switching: a perturbation that substitutes each candidate with "an appropriate synonym" or, if it is a personal name, "a random name of the same gender." That is considerably lower than our value of $c = 0.743$ for RoBERTa's switchable consistency under mean multi-mask scoring, or $c = 0.807$ under mean statement scoring.

Zhou et al. [109] report the consistency of GPT-2 *medium*, not *xl*, and RoBERTa on various perturbations of WSC273 including switching, which they call "swap." Apparently, they use mean full scoring for GPT-2 and mean statement scoring for RoBERTa.[1] The RoBERTa result sounds promising, except that they use their own switchable subset of 75 problems, rather than Trichelair et al. [95]'s 131. With the pretrained models, they report a consistency of $c = 0.52$ for GPT-2 *medium*, much lower than our value of $c = 0.671$ for the *xl* model under mean full scoring, or $c = 0.686$ under smart scoring; and they report $c = 0.56$ for RoBERTa, much lower than our value of $c = 0.807$ for RoBERTa under mean statement scoring.

Ruan et al. [81] report the switchable consistent accuracy c_a, not the consistency

[1] In Section 4.2.2, we were unable to replicate their accuracy for RoBERTa on WSC273.

c, of BERT on WSC273. Not only is this not a directly comparable model, but the authors solve Winograd problems using next sentence prediction, not stochastic fill-in-the-blank. With the pretrained model, they report a consistent accuracy of $c_a = 0.227$ for BERT, much lower than our value of $c_a = 0.650$ for RoBERTa under mean statement scoring, or 0.600 under mean multi-mask scoring.

Trichelair et al. [96] report the switchable consistency of GPT-2 *large*, not *xl*, on WSC273 under full and partial scoring. With the pretrained model, they report a consistency of $c = 0.4504$ under full scoring, much lower than our value of $c = 0.721$ for the *xl* model under full scoring; and they report $c = 0.6335$ under partial scoring, considerably lower than our value of $c = 0.693$ under partial scoring.

Although we did change the dataset and the switchable subset, and in some cases the model or the size of the model, making direct comparisons questionable, it does appear that more accurate scoring methods can lead to much higher consistencies.

4.3.2 Inverted Schemas and Scoring Methods

That brings us to our third question: is the inverted dataset substantially equivalent to its original formulation? That is, does it preserve the content of the challenge?

In Section 4.3.1, we showed that, under the standard setup, the performance profile of the inverted perturbation differs significantly from that of the original dataset and the switched, adjectival, and unbalanced perturbations, and that we can characterize and rank the models according to their performance on WSC266 inverted.

All of that was under the standard setup. Here, we add some context by comparing scoring methods for WSC266 inverted as we did for the original dataset in Section 4.2. These results include and extend the results for WSC inverted in Table 4.16.

Table 4.21 shows problem accuracy a and schema accuracy A for GPT-2 on WSC266 inverted, under each scoring method from Table 4.3. It also includes the difference between the original dataset and its inversion, with statistical significance.

Taking the mean has almost no effect on accuracy—at most, one problem and one schema—so we restrict our attention to non-mean scoring, as we did in Section 4.2.1.

In the inverted dataset, exactly two problem statements start with the target pronoun, so full scoring and all-but-one scoring give almost identical results.

Inverting the schemas decreases the accuracy of the former best scoring methods,

Scoring Method	Take Mean?	a	Inverted − Original	p	A	Inverted − Original	p
smart	no	0.628	−0.117	$< .001$	0.278	−0.218	$< .001$
scoring	yes	0.628	−0.117	$< .001$	0.278	−0.218	$< .001$
partial	no	0.579	−0.147	$< .001$	0.188	−0.286	$< .001$
scoring	yes	0.579	−0.147	$< .001$	0.188	−0.286	$< .001$
all-but-one	no	0.628	−0.023	.3	0.278	−0.030	.5
scoring	yes	0.632	−0.004	.9	0.286	+0.008	.9
full scoring	no	0.628	−0.023	.3	0.278	−0.030	.5
	yes	0.628	+0.008	.7	0.286	+0.045	.3
normalized	no	0.650	+0.038	.06	0.308	+0.075	.05
full scoring	yes	0.650	+0.038	.06	0.308	+0.075	.05

Table 4.21: Accuracy of scoring methods for GPT-2 on WSC266 inverted.

smart scoring and partial scoring, $p < .001$, and increases the accuracy of the former worst scoring method, normalized full scoring, $p \leq .06$.

The ranking of scoring methods on WSC266 inverted looks very different from the ranking on WSC266 (which is the order of Table 4.21): from most to least accurate, we have normalized full scoring, a three-way tie between smart scoring, all-but-one scoring, and full scoring, and partial scoring last. Actually, though, all the scoring methods except partial scoring are similar in accuracy now. For example, the change from smart scoring to normalized full scoring is not significant: $\Delta a = +0.023$, $p = .3$ and $\Delta A = +0.030$, $p = .5$. However, the change from smart scoring to partial scoring is significant: $\Delta a = -0.049$, $p = .02$ and $\Delta A = -0.090$, $p = .03$.

The relatively low accuracy of partial scoring may be due to the relatively high number of problems in WSC266 inverted for which the answer placeholder occurs in the second-last position: 48.9%, versus 6.8% in the original dataset (Table 3.8).

All the problem accuracies are significantly better than chance, $p \leq .01$, and indeed $p < .001$ except for the outlier of partial scoring. However, none of the schema accuracies are significantly different from chance, $p \geq .1$.

Table 4.22 shows problem accuracy a and schema accuracy A for RoBERTa on WSC266 inverted, under each scoring method from Table 4.7. It also includes the difference between the original dataset and its inversion, with statistical significance.

Taking the mean still improves the accuracy of both statement scoring and multi-mask scoring, as in Section 4.2.2, but now it also improves the accuracy of answer

Scoring Method	Take Mean?	a	Inverted − Original	p	A	Inverted − Original	p
statement	no	0.797	+0.019	.4	0.594	+0.038	.4
scoring	yes	0.808	+0.023	.3	0.617	+0.045	.3
multi-mask	no	0.714	−0.030	.2	0.436	−0.053	.3
scoring	yes	0.729	−0.030	.2	0.466	−0.053	.2
answer	no	0.729	+0.011	.7	0.459	+0.015	.8
scoring	yes	0.733	+0.034	.1	0.466	+0.060	.2

Table 4.22: Accuracy of scoring methods for RoBERTa on WSC266 inverted.

scoring. However, the improvement to answer scoring is not even close to significant, $p = .9$, so we restrict our attention to mean statement scoring, mean multi-mask scoring, and (non-mean) answer scoring, just as we did in Section 4.2.2. Using mean answer scoring instead does not appreciably affect any of the following comparisons.

Inverting the schemas has no significant effect on accuracy, $p \geq .1$.

The ranking of scoring methods on WSC266 inverted differs from the ranking on WSC266 (which is the order of Table 4.22): from most to least accurate, we have mean statement scoring, answer scoring, and mean multi-mask scoring. Actually, though, the only difference now is that answer scoring has moved up to match multi-mask scoring. That is, the change from mean multi-mask scoring to answer scoring is not significant: $\Delta a = +0.000$ (no p) and $\Delta A = -0.008$, $p = .9$. However, the change from mean statement scoring to answer scoring, the new second-best method, is still significant: $\Delta a = -0.079$, $p < .001$ and $\Delta A = -0.158$, $p < .001$.

All the accuracies are significantly better than chance, $p < .001$.

Table 4.23 shows problem accuracy a and schema accuracy A for T5 on WSC266 inverted, under each scoring method from Table 4.11. It also includes the difference between the original dataset and its inversion, with statistical significance.

The Winograd task prefix still always increases accuracy, albeit insignificantly, regardless of the other parameters, although inverted Winograd schemas are not Winograd schemas or even pronoun disambiguation tasks.

Inverting the schemas decreases the accuracy of the former best scoring methods, mean prefix scoring, prefix scoring, and mean scoring, $p \leq .006$, and marginally increases the accuracy of the former worst scoring method, loss scoring, $p \leq .1$.

The ranking of scoring methods on WSC266 inverted looks very different from the

Scoring Method	a	Inverted − Original	p	A	Inverted − Original	p
mean prefix	0.805	−0.064	$< .001$	0.609	−0.128	.001
prefix	0.797	−0.060	.002	0.594	−0.120	.003
mean	0.797	−0.056	.004	0.594	−0.113	.006
basic	0.812	−0.004	.9	0.624	−0.008	.9
prefix loss	0.812	+0.008	.8	0.624	+0.015	.8
loss	0.797	+0.038	.1	0.609	+0.090	.04

Table 4.23: Accuracy of scoring methods for T5 on WSC266 inverted.

ranking on WSC266 (which is the order of Table 4.23): from most to least accurate, we have a tie between basic scoring and prefix loss scoring, mean prefix scoring, loss scoring, and a tie between prefix scoring and mean scoring. Actually, though, all the scoring methods are similar in accuracy now. For example, the largest change, from prefix scoring or mean scoring to basic scoring or prefix loss scoring, is not significant: $\Delta a = +0.015$, $p = .5$ and $\Delta A = +0.030$, $p = .5$.

All the accuracies are significantly better than chance, $p < .001$.

Looking at all the models and accounting for statistical significance or lack thereof, the best scoring methods for WSC266 tend to still be the best scoring methods for WSC266 inverted, although two models, GPT-2 and T5, are significantly less accurate on the inverted perturbation under the best scoring methods.

Table 4.24 shows the applicable metrics of schema consistency C, consistent schema accuracy C_a, and preserved schema accuracy C_p on the inverted perturbation of WSC266, under each relevant scoring method for each model, discussed above.

Note that, e.g., consistency c and strict schema consistency C^* are not applicable to WSC266 inverted, because inversion is not a problem transformation.

The consistency metrics are not difficult to interpret. For example, smart scoring was the best scoring method on WSC266, but it lost accuracy under inversion, so it has relatively high consistent accuracies and relatively low preserved accuracies. Partial scoring, on the other hand, was less accurate to begin with, and it lost even more accuracy, so its consistent accuracies and preserved accuracies are the lowest.

For RoBERTa and T5, the ranking of scoring methods by consistency score is generally consistent, so to speak, with their ranking by accuracy on the original dataset (which is the order in the table). GPT-2 is less consistent in that regard.

Model	Scoring Method	C	C_a	C_p
GPT-2	smart	0.579	0.188	0.379
	partial	0.511	0.105	0.222
	all-but-one	0.699	0.158	0.512
	full	0.699	0.158	0.512
	normalized full	0.714	0.128	0.548
RoBERTa	mean statement	0.701	0.444	0.688
	mean multi-mask	0.624	0.308	0.594
	answer	0.617	0.263	0.593
T5	mean prefix	0.662	0.504	0.684
	prefix	0.699	0.504	0.705
	mean	0.647	0.474	0.670
	basic	0.647	0.451	0.714
	prefix loss	0.579	0.406	0.667
	loss	0.579	0.361	0.696

Table 4.24: Consistency on WSC266 inverted: expanded.

At this point, collecting all the results reported here and in Section 4.3.1, we still cannot conclusively say whether or not an inverted schema is substantially equivalent to its original formulation, because the models are not consistent enough in general. We note that each of the other perturbations has a much stronger or at least much more obvious claim to being substantially equivalent than the inverted perturbation, but we find no clear empirical evidence that the models are more consistent on them.

For example, from Table 4.20, the most accurate and consistent model, T5, achieves consistency scores of $C = 0.700$ on the switched perturbation, $C = 0.770$ on the adjectival, $C = 0.564$ on the unbalanced, and $C = 0.662$ on the inverted, which is barely below the switched perturbation and actually above the unbalanced. Granted, the unbalanced perturbation is an outlier, so consider the most consistent model on that perturbation, GPT-2: it achieves consistency scores of $C = 0.686$ on the switched perturbation, $C = 0.644$ on the adjectival, $C = 0.654$ on the unbalanced, and $C = 0.579$ on the inverted. Now the inverted perturbation is less consistent than the others; on the other hand, the adjectival perturbation is much less consistent.

At least we can say that every model is less consistent under inversion than under the switched and adjectival perturbations. And although we showed in this section that scoring methods may perform differently on WSC266 inverted than on WSC266, particularly for GPT-2, the extended accuracies in Tables 4.21, 4.22, and 4.23, and

the extended consistencies in Table 4.24, do not appreciably change our findings.

4.4 Associativity of Winograd Schemas

Continuing our evaluation protocol, we use the no-candidate perturbation to quantify spurious correlations in each model, and associativity in the dataset itself. As usual, we report results only under the standard setup and only on WSC266.

Our goal is to answer two questions. First, can the no-candidate perturbation confirm that the so-called associative subset of WSC266 is indeed associative? Second, how can we correct model accuracies to account for spurious correlations?

First, we need to address the size of the associative subset of WSC266: because it consists of only 50 problems, the statistical significance of our results is likely to be low. Moreover, the associative subset arguably cannot even be interpreted as a sample from a larger space of problems (Section 3.8.4), as it consists of specific schemas identified by a specific team of annotators. Nevertheless, we will continue to report p-values and confidence intervals. However, we will entirely omit results for the subset of problems with equal-length answers because the associative subset of that subset consists of only 30 problems, which we consider too few to study.

Table 4.25 shows problem accuracy a and schema accuracy A for each model, under the standard setup, on the associative and non-associative subsets of WSC266, before and after applying the no-candidate perturbation. It also includes the differences in accuracy between the associative and non-associative subsets, as well as the statistical significance of those differences and of each accuracy versus chance.

Figure 4.7 displays the accuracies from Table 4.25 with 95% confidence intervals.

Every model is more accurate on the associative subset of WSC266 than the non-associative subset, though the significance of this result varies: $p \leq .008$ for GPT-2, $p = .03$ for RoBERTa, and $.07 \leq p \leq .08$ for T5, which is very accurate on both.

After applying the no-candidate perturbation, every model is more accurate on the associative subset of WSC266 than on the non-associative subset, $p < .001$.

On the original dataset, as expected, all the accuracies were better than chance, $p < .001$. After the perturbation, on the non-associative subset, problem accuracies are only marginally better than chance, $.2 \leq p \leq .9$, and schema accuracies are actually worse than chance, $.001 \leq p \leq .04$. On the associative subset, however, all

Model	Metric	Non-Assoc.	p	Assoc.	p	Assoc. − Non-Assoc.	p
Original Problems							
GPT-2	a	0.718	$< .001$	0.860	$< .001$	+0.142	.008
	A	0.444	$< .001$	0.720	$< .001$	+0.276	.008
RoBERTa	a	0.764	$< .001$	0.880	$< .001$	+0.116	.03
	A	0.528	$< .001$	0.760	$< .001$	+0.232	.03
T5	a	0.852	$< .001$	0.940	$< .001$	+0.088	.08
	A	0.704	$< .001$	0.880	$< .001$	+0.176	.07
No-Candidate Perturbation							
GPT-2	a	0.505	.9	0.760	$< .001$	+0.255	$< .001$
	A	0.167	.04	0.520	.003	+0.353	$< .001$
RoBERTa	a	0.551	.2	0.780	$< .001$	+0.229	$< .001$
	A	0.176	.009	0.560	$< .001$	+0.384	$< .001$
T5	a	0.546	.2	0.760	$< .001$	+0.214	$< .001$
	A	0.139	.001	0.520	.003	+0.381	$< .001$

Table 4.25: Accuracy on associative and non-associative subsets of WSC266.

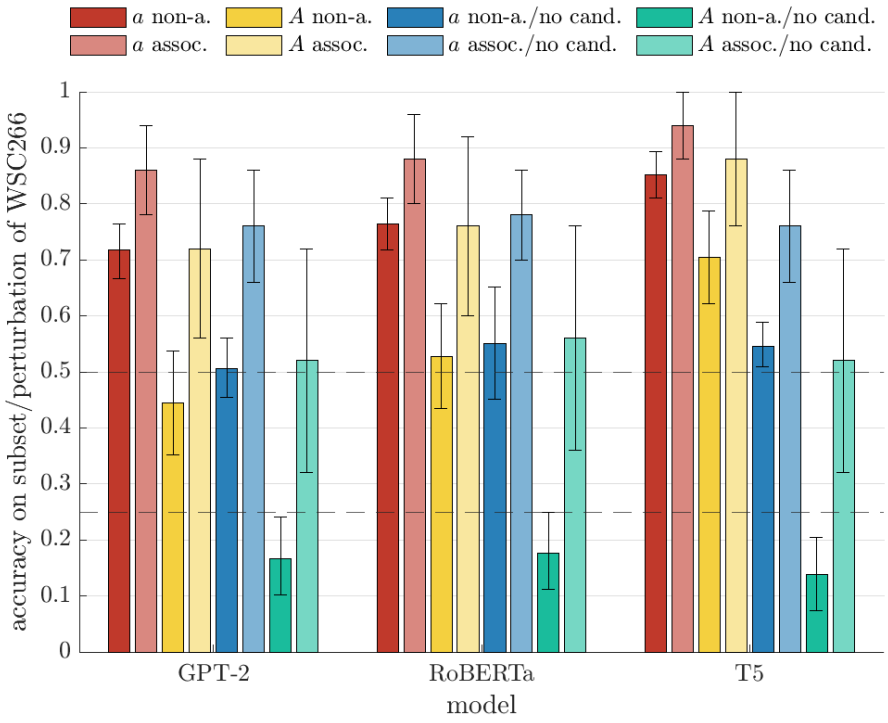

Figure 4.7: Accuracy on associative and non-associative subsets of WSC266, before and after applying no-candidate perturbation.

the accuracies are much better than chance, $p \leq .003$.

After the no-candidate perturbation, the gaps in accuracy between the associative and non-associative subsets (second-last column) increase in size for every model.

It already seems fairly clear that the so-called associative subset is indeed far more associative than its complement. These results may be broadly consistent with Elazar et al. [23], who finetuned RoBERTa on WinoGrande, in that their accuracies on the no-candidate perturbation were much lower than on the original set but still better than chance, and their accuracies on the no-candidate non-associative subset were slightly lower than on the no-candidate associative subset.

Table 4.26 rearranges Table 4.25 to show how accuracy on the associative and non-associative subsets of WSC266 changes after applying the no-candidate perturbation.

Model	Metric	Subset	Original	No Cand.	No Cand. − Original	p
GPT-2	a	non-assoc.	0.718	0.505	−0.213	$< .001$
		assoc.	0.860	0.760	−0.100	.04
	A	non-assoc.	0.444	0.167	−0.278	$< .001$
		assoc.	0.720	0.520	−0.200	.04
RoBERTa	a	non-assoc.	0.764	0.551	−0.213	$< .001$
		assoc.	0.880	0.780	−0.100	.03
	A	non-assoc.	0.528	0.176	−0.352	$< .001$
		assoc.	0.760	0.560	−0.200	.03
T5	a	non-assoc.	0.852	0.546	−0.306	$< .001$
		assoc.	0.940	0.760	−0.180	$< .001$
	A	non-assoc.	0.704	0.139	−0.565	$< .001$
		assoc.	0.880	0.520	−0.360	$< .001$

Table 4.26: Effect of no-candidate perturbation on accuracy, compared to original problems, on associative and non-associative subsets of WSC266.

Figure 4.8 displays the losses in accuracy from Table 4.26.

As expected, every model is significantly more accurate on an original subset of WSC266 than on its no-candidate perturbation, $p \leq .04$. Also as expected, comparing original subsets to their no-candidate perturbations, every model loses much more accuracy on the non-associative subset than on the associative subset.

What about the relative change in accuracy? Let r denote an accuracy, problem or schema, on a subset of WSC266, and let r_B denote the corresponding accuracy on its no-candidate perturbation. Then of course $r - r_B$ is the accuracy loss from

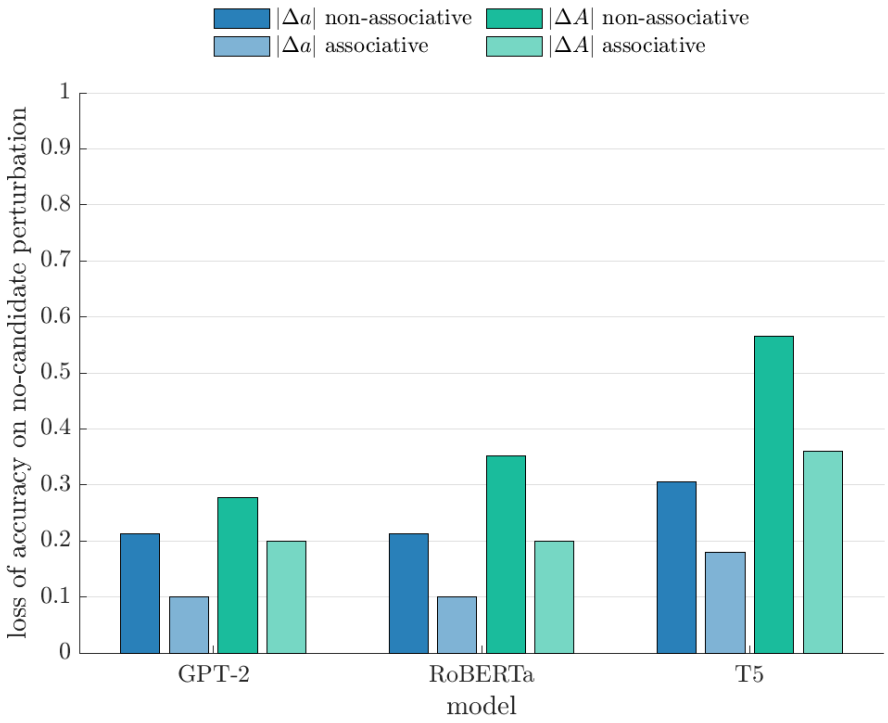

Figure 4.8: Loss of accuracy from applying the no-candidate perturbation to the associative and non-associative subsets of WSC266.

Figure 4.8, or equivalently the accuracy gain we obtain by undoing the transformation and putting candidates back into the mostly unsolvable no-candidate problems.

The accuracy lost due to the transformation can be expressed as a fraction of the original accuracy: define the *relative loss* as

$$r_1 = (r - r_B)/r \qquad \text{for } r > 0. \tag{4.1}$$

If we undo the transformation, the accuracy gained can be expressed as a fraction of the unsolved share of the no-candidate perturbation: define the *relative gain* as

$$r_2 = (r - r_B)/(1 - r_B) \qquad \text{for } r_B < 1. \tag{4.2}$$

Table 4.27 shows the relative loss and gain of accuracy under the no-candidate perturbation on the associative and non-associative subsets of WSC266.

Model	Metric	Subset	$r - r_B$	$\frac{r-r_B}{r}$	$\frac{r-r_B}{1-r_B}$
GPT-2	a	non-assoc.	0.213	0.297	0.430
		associative	0.100	0.116	0.278
	A	non-assoc.	0.278	0.625	0.333
		associative	0.200	0.417	0.417
RoBERTa	a	non-assoc.	0.213	0.279	0.474
		associative	0.100	0.114	0.455
	A	non-assoc.	0.352	0.667	0.427
		associative	0.200	0.263	0.455
T5	a	non-assoc.	0.306	0.359	0.673
		associative	0.180	0.191	0.409
	A	non-assoc.	0.565	0.803	0.656
		associative	0.360	0.409	0.750

Table 4.27: Relative loss of accuracy from applying the no-candidate perturbation to the associative and non-associative subsets of WSC266.

Every model loses much more accuracy on the non-associative subset than on the associative subset according to relative loss, though not always with relative gain.

That leaves us with our second question: how can we correct model accuracies to account for spurious correlations? The simplest solution might be to subtract a model's accuracy on the no-candidate perturbation from its accuracy on the original dataset: $r - r_B$, as above. We could also express the difference in accuracy as a fraction of the unsolved share of the no-candidate perturbation, as in (4.2).

Model	Metric	r	$r - r_B$	$\frac{r-r_B}{1-r_B}$
GPT-2	a	0.744	0.192	0.429
	A	0.496	0.263	0.343
RoBERTa	a	0.786	0.192	0.472
	A	0.571	0.323	0.430
T5	a	0.868	0.282	0.682
	A	0.737	0.526	0.667

Table 4.28: Accuracies on WSC266 adjusted for spurious correlations.

Table 4.28 shows the adjusted accuracies from Figure 4.4.

This method essentially gives a model no credit for solving associative problems. Also, it does not take into account consistency: just because a model solves fewer problems from the no-candidate perturbation does not mean that it solves exactly those problems in the original dataset. We will not pursue this here, but we are referring to the method more modestly as adjusting, not correcting, the accuracies.

Probably the safest way of accounting for spurious correlations is to report a full set of comparisons as we have done, in accordance with our evaluation protocol.

4.5 Accuracy on Adversarial Schemas

Continuing our evaluation protocol, we calculate accuracy on Reid250, extending the models' performance profiles to that dataset (Section 4.2.4). We evaluate scoring methods for GPT-2 in Section 4.5.1, for RoBERTa in Section 4.5.2, and for T5 in Section 4.5.3, and we compare the best accuracies in Section 4.5.4.

Our goal is to answer three questions. First, which scoring methods are the most accurate on Reid250? Second, how accurate are the models on Reid250 at best? Third, are the models, at best, more or less accurate on Reid250 than on WSC266?

4.5.1 Accuracy of GPT-2 Scoring Methods

Table 4.29 shows problem accuracy a and schema accuracy A for GPT-2 on Reid250, under the ten scoring methods from Table 4.3, on problems with any-length answers and equal-length answers. Model loss is mean all-but-one scoring $(*)$.

As with WSC266, taking the mean never improves the accuracy of a scoring method, so we restrict our attention to non-mean scoring.

Scoring Method	Take Mean?	a, Any Length	a, Equal Length	A, Any Length	A, Equal Length
smart scoring	no	0.552	0.548	0.168	0.181
at limit 1	yes	0.548	0.548	0.160	0.181
partial	no	0.540	0.543	0.152	0.170
scoring	yes	0.540	0.543	0.152	0.170
all-but-one	no	0.552	0.543	0.136	0.128
scoring	yes*	0.536	0.543	0.112	0.128
full scoring	no	0.552	0.543	0.136	0.128
	yes	0.536	0.543	0.104	0.128
normalized	no	0.556	0.559	0.152	0.170
full scoring	yes	0.556	0.559	0.152	0.170

Table 4.29: Accuracy of scoring methods for GPT-2 on Reid250.

In Reid250, 14.4% of problem statements start with the target pronoun, but full scoring and all-but-one scoring still give identical results.

Ranking the scoring methods is not entirely straightforward, because problem accuracy and schema accuracy give different rankings. We decided to prioritize the stricter, schema-based metric: from most to least accurate, we have smart scoring, normalized full scoring, partial scoring, and full scoring tied with all-but-one scoring. Compared to WSC266, normalized full scoring has moved up to second best.

Actually, none of the scoring methods are significantly different. Comparing the highest schema accuracy, smart scoring, to the lowest, full or all-but-one scoring, on problems with any-length answers, we get $\Delta a = +0.000$ (no p) and $\Delta A = -0.032$, $p = .4$. Comparing the highest problem accuracy, normalized full scoring, to the lowest, partial scoring, we get $\Delta a = -0.016$, $p = .4$ and $\Delta A = +0.000$ (no p).

We decided to use smart scoring again: it was the most accurate on WSC266, it has the highest schema accuracy and the second-highest problem accuracy on Reid250, and none of the differences on Reid250 are significant anyway.

All the problem accuracies in Table 4.29 are marginally better than chance, $.08 \leq p \leq .3$. The schema accuracies are significantly worse than chance, $p \leq .04$, except smart scoring, normalized full scoring, and partial scoring, mean or no mean, on problems with equal-length answers, which are marginally worse, $.07 \leq p \leq .1$.

Across the board, GPT-2 has better than random problem accuracy and worse than random schema accuracy on Reid250 ($a > 0.5$ and $A < 0.25$). A'' was never

higher than 0.072, which was its value under partial and mean partial scoring.

4.5.2 Accuracy of RoBERTa Scoring Methods

Table 4.30 shows problem accuracy a and schema accuracy A for RoBERTa on Reid250, under the six scoring methods from Table 4.7, on problems with any-length answers and equal-length answers. Model loss is mean multi-mask scoring ($*$).

Scoring Method	Take Mean?	a, Any Length	a, Equal Length	A, Any Length	A, Equal Length
statement	no	0.600	0.617	0.272	0.298
scoring	yes	0.600	0.617	0.256	0.298
multi-mask	no	0.580	0.596	0.176	0.202
scoring	yes*	0.592	0.596	0.208	0.202
answer	no	0.580	0.596	0.168	0.202
scoring	yes	0.584	0.596	0.176	0.202

Table 4.30: Accuracy of scoring methods for RoBERTa on Reid250.

As with WSC266, taking the mean has the largest effect on the accuracy of multi-mask scoring, and it is still an improvement, but now it worsens statement scoring and improves answer scoring. However, the differences between mean and non-mean statement and answer scoring are marginal, $p \geq .7$, so we restrict our attention to mean statement scoring, mean multi-mask scoring, and (non-mean) answer scoring, just as we did in Section 4.2.2. Using (non-mean) statement scoring or mean answer scoring instead does not appreciably affect any of the following comparisons.

Ranking the scoring methods from most to least accurate, we have mean statement scoring, mean multi-mask scoring, and answer scoring, the same as with WSC266.

Actually, none of the scoring methods differ significantly in problem accuracy: $p = .3$ at best, comparing the highest value, mean statement scoring, to the lowest, answer scoring. For schema accuracy, mean statement scoring is marginally higher than mean multi-mask scoring, $\Delta A = +0.048$, $p = .2$; mean multi-mask scoring is marginally higher than answer scoring, $\Delta A = +0.040$, $p = .3$; and mean statement scoring is significantly higher than answer scoring, $\Delta A = +0.088$, $p = .01$.

We decided to use mean statement scoring again for obvious reasons: it was the most accurate on WSC266, and it is the most accurate (left) on Reid250.

All the problem accuracies in Table 4.30 are significantly better than chance, $.002 \leq p \leq .01$. None of the schema accuracies are, though: on problems with any-length answers, statement and mean statement scoring are marginally better, $.6 \leq p \leq .9$, and the other scoring methods are at least marginally worse, $.04 \leq p \leq .3$.

It follows that RoBERTa has better than random problem accuracy and worse than random schema accuracy under four out of six scoring methods. A'' was never higher than 0.072, which was its value under statement scoring.

4.5.3 Accuracy of T5 Scoring Methods

Table 4.31 shows problem accuracy a and schema accuracy A for T5 on Reid250, under the six scoring methods from Table 4.11, on problems with any-length answers and equal-length answers. Model loss is represented by loss and prefix loss ($*$).

Scoring Method	a, Any Length	a, Equal Length	A, Any Length	A, Equal Length
mean prefix	0.656	0.676	0.336	0.375
prefix	0.644	$0.676^{\dagger}$	0.320	$0.375^{\dagger}$
mean	0.668	0.688	0.360	0.375
basic	0.656	$0.676^{\ddagger}$	0.336	$0.364^{\ddagger}$
prefix loss*	0.644	$0.676^{\dagger}$	0.304	$0.375^{\dagger}$
loss*	0.632	$0.676^{\ddagger}$	0.288	$0.364^{\ddagger}$

Table 4.31: Accuracy of scoring methods for T5 on Reid250.

As we already noted, on problems with equal-length answers, prefix loss scoring is necessarily identical to prefix scoring (†), and loss scoring to basic scoring (‡).

Ranking the scoring methods from most to least accurate, we have mean, basic tied with mean prefix, prefix, prefix loss, and loss. Compared to WSC266, mean and basic scoring have moved up, higher than or equal to mean prefix and prefix scoring.

It would make sense that the Winograd task prefix is no longer reliably increasing accuracy, since adversarial schemas are not Winograd schemas, in general. Actually, though, none of the scoring methods are significantly different. Comparing the most accurate method, mean scoring, to the least accurate, loss scoring, on problems with any-length answers, we get $\Delta a = +0.036$, $p = .1$, and $\Delta A = +0.072$, $p = .07$.

We decided to use mean scoring for T5 on Reid250: it is the most accurate on that dataset, and although mean prefix scoring was marginally more accurate on WSC266,

this is not a Winograd dataset, and Winograd task prefixes are not appropriate.

All the problem accuracies in Table 4.29 are significantly better than chance, $p < .001$. The schema accuracies are significantly better than chance too, $p \leq .03$, except prefix, prefix loss, and loss scoring on problems with any-length answers, which are only marginally better, $.08 \leq p \leq .4$.

In particular, T5 never has better than random problem accuracy and worse than random schema accuracy on Reid250. A'' was never higher than 0.032, which was its value under prefix scoring.

4.5.4 Model Comparison: Best Scoring Methods

On Reid250, we will of course report results for all three models, but only under smart scoring for GPT-2, mean statement scoring for RoBERTa, and mean scoring for T5, which we call the *standard setup for Reid250*.

Table 4.32 shows the differences in best accuracy between models. Clearly, T5 is more accurate than GPT-2, $p < .001$; and RoBERTa, $.004 \leq p \leq .02$, except for schema accuracy on problems with equal-length answers, where it is marginally more accurate, $p = .1$. And RoBERTa is more accurate than GPT-2, $.004 \leq p \leq .003$.

Metric	Answer Length	RoBERTa − GPT-2	p	T5 − RoBERTa	p	T5 − GPT-2	p
a	any	+0.048	.03	+0.068	.004	+0.116	$< .001$
	equal	+0.069	.009	+0.070	.02	+0.140	$< .001$
A	any	+0.088	.01	+0.104	.01	+0.192	$< .001$
	equal	+0.117	.004	+0.077	.1	+0.194	$< .001$

Table 4.32: Comparison of best model accuracies on Reid250.

Bear in mind that the subset of Reid250 with equal-length answers differs between GPT-2 or RoBERTa and T5 (Table 4.2).

Moving on to our third question, Table 4.33 shows the problem accuracy a and schema accuracy A for GPT-2, RoBERTa, and T5, under the standard setup for Reid250, on problems with any-length answers and equal-length answers, as well as the differences between those accuracies and the best accuracies on WSC266 from Figure 4.4. It also includes the statistical significance of those differences.

Model	Metric	Answer Length	Reid250	Reid250 − WSC266	p
GPT-2	a	any	0.552	−0.192	$< .001$
		equal	0.548	−0.172	$< .001$
	A	any	0.168	−0.328	$< .001$
		equal	0.181	−0.259	$< .001$
RoBERTa	a	any	0.600	−0.186	$< .001$
		equal	0.617	−0.174	$< .001$
	A	any	0.256	−0.315	$< .001$
		equal	0.298	−0.285	$< .001$
T5	a	any	0.668	−0.200	$< .001$
		equal	0.688	−0.183	$< .001$
	A	any	0.360	−0.377	$< .001$
		equal	0.375	−0.366	$< .001$

Table 4.33: Comparing best model accuracies on WSC266 and Reid250.

Note that we are comparing mean scoring for T5 on Reid250, with no Winograd task prefix, to mean prefix scoring for T5 on WSC266.

Clearly, every model is less accurate on Reid250 than on WSC266, $p < .001$. The loss of accuracy from WSC266 to Reid250 is roughly comparable for each model.

It seems reasonable to say that, for at least these three pretrained language models, Reid250 is a more challenging test of common-sense reasoning than WSC266.

Figure 4.9 displays the accuracies on Reid250 from Table 4.33, plus the accuracies on WSC266 from Figure 4.4 for comparison, with 95% confidence intervals.

We should add a note on the effect of answer length on accuracy for Reid250. We compare each Reid250 accuracy from Table 4.33 on all problems to the corresponding accuracy on problems with equal-length answers, six comparisons in all, and find no significant differences, $p \geq .4$. On Reid250, as with WSC266, each model appears to be robust with respect to the number of tokens in the answer options.

4.6 Inversion of Adversarial Schemas

Continuing our evaluation protocol, we calculate the accuracy and consistency of each model on our only perturbation of Reid250, which is the inverted perturbation. As stated in Section 4.5.4, we report results only under the standard setup.

Inversion is a schema transformation, and whether or not it substantially preserves

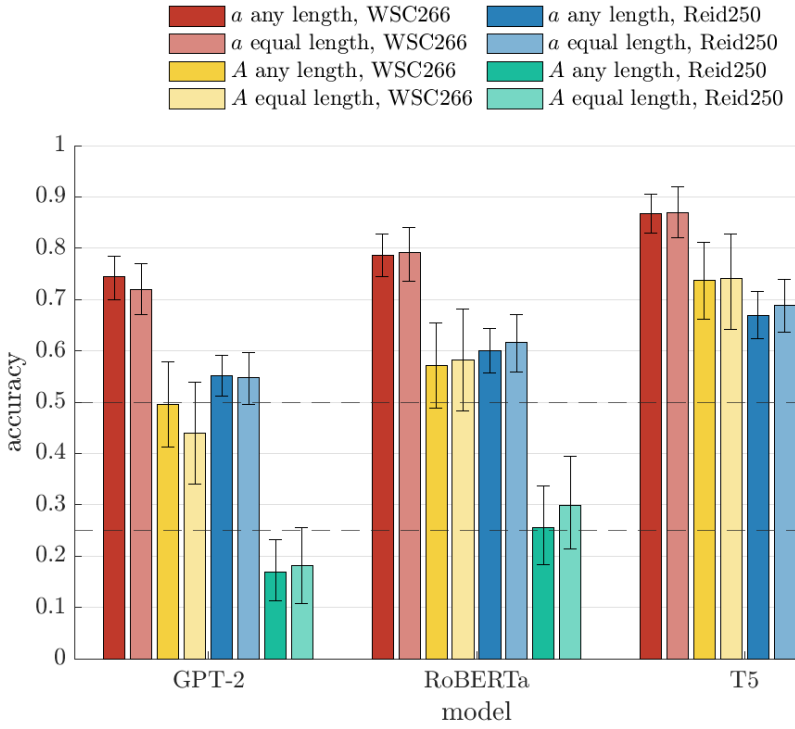

Figure 4.9: Comparing best model accuracies on WSC266 and Reid250.

the content of a schema is still an open question (Section 4.3.2). Note that the switched, adjectival, and unbalanced perturbations do not apply to Reid250.

Our goal is to answer two questions. First, how does inversion affect the accuracy of the models on Reid250? Second, how consistent are the models under inversion?

We begin by very briefly comparing scoring methods for Reid250 inverted, as we did for Reid250 in Section 4.5 and WSC266 inverted in Section 4.3.2.

Tables 4.34, 4.35, and 4.36 show problem accuracy a and schema accuracy A on the inverted perturbation of Reid250 for, respectively, GPT-2 under each scoring method from Table 4.29, RoBERTa under each scoring method from Table 4.30, and T5 under each scoring method from Table 4.31.

Although some scoring methods are marginally more accurate on Reid250 inverted than the designated scoring methods from the standard setup for that dataset, the differences are not large enough to justify changing any of the scoring methods.

The differences in accuracy between Reid250 and its inverted perturbation are not even close to significant for the standard setup, $.6 \leq p \leq .8$, except for T5, where the

Scoring Method	Take Mean?	a	Inverted − Original	p	A	Inverted − Original	p
smart	no	0.564	+0.012	.6	0.160	−0.008	.8
scoring	yes	0.560	+0.012	.6	0.136	−0.024	.6
partial	no	0.544	+0.004	.9	0.152	0.000	–
scoring	yes	0.544	+0.004	.9	0.152	0.000	–
all-but-one	no	0.584	+0.032	.09	0.200	+0.064	.05
scoring	yes	0.568	+0.032	.08	0.160	+0.048	.1
full scoring	no	0.576	+0.024	.2	0.192	+0.056	.09
	yes	0.580	+0.044	.009	0.176	+0.072	.01
normalized	no	0.572	+0.016	.5	0.152	0.000	–
full scoring	yes	0.568	+0.012	.6	0.136	−0.016	.7

Table 4.34: Accuracy of scoring methods for GPT-2 on Reid250 inverted.

Scoring Method	Take Mean?	a	Inverted − Original	p	A	Inverted − Original	p
statement	no	0.612	+0.012	.7	0.256	−0.016	.7
scoring	yes	0.616	+0.016	.5	0.256	0.000	–
multi-mask	no	0.572	−0.008	.7	0.152	−0.003	.6
scoring	yes	0.568	−0.024	.3	0.168	−0.040	.3
answer	no	0.568	−0.012	.6	0.144	−0.024	.5
scoring	yes	0.576	−0.008	.7	0.152	−0.024	.6

Table 4.35: Accuracy of scoring methods for RoBERTa on Reid250 inverted.

Scoring Method	a	Inverted − Original	p	A	Inverted − Original	p
mean prefix	0.648	−0.008	.8	0.312	−0.024	.6
prefix	0.616	−0.028	.3	0.280	−0.040	.4
mean	0.624	−0.044	.06	0.272	−0.088	.05
basic	0.640	−0.016	.5	0.320	−0.016	.7
prefix loss	0.616	−0.028	.2	0.264	−0.040	.4
loss	0.636	+0.004	.9	0.288	0.000	–

Table 4.36: Accuracy of scoring methods for T5 on Reid250 inverted.

differences are still marginal: $\Delta a = -0.044$, $p = .06$ and $\Delta A = -0.088$, $p = .05$.

Figure 4.10 displays the key results from Tables 4.34, 4.35, and 4.36, namely the accuracies under the standard setup, with 95% confidence intervals.

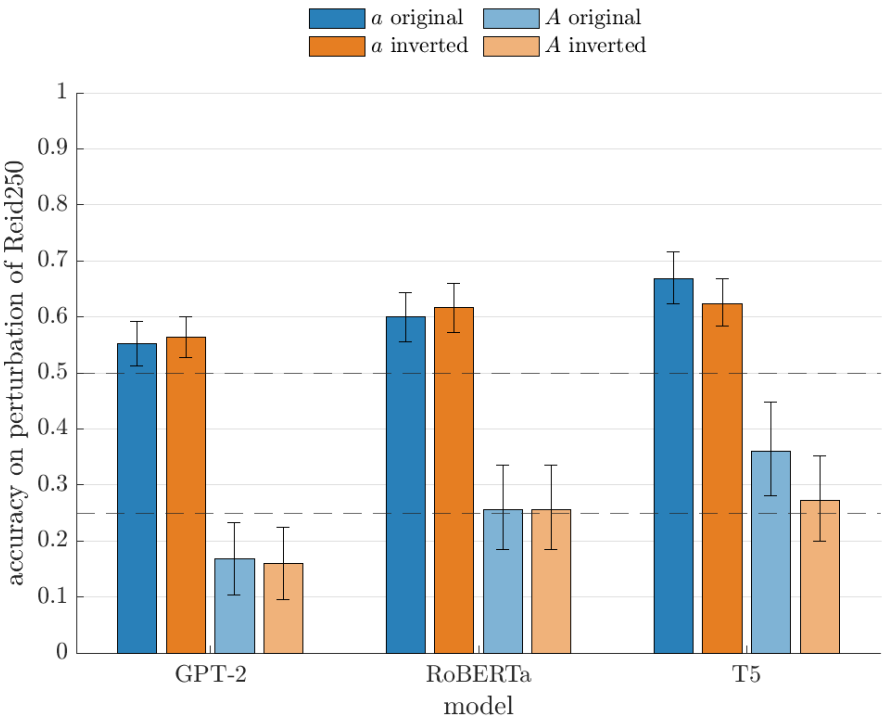

Figure 4.10: Accuracy on Reid250 and its inversion.

Moving on to our second question, Table 4.37 shows the applicable metrics of schema consistency C, consistent schema accuracy C_a, and preserved schema accuracy C_p on Reid250 inverted under the standard setup.

Model	C	C_a	C_p
GPT-2	0.688	0.056	0.333
RoBERTa	0.680	0.128	0.500
T5	0.584	0.120	0.333

Table 4.37: Consistency on the inverted perturbation of Reid250.

Again, note that, e.g., consistency c and strict schema consistency C^* are not applicable to Reid250 inverted, because inversion is not a problem transformation.

Figure 4.11 displays the consistency scores C and C_a from Table 4.37, plus the inversion consistencies on WSC266 from Figure 4.6 for comparison.

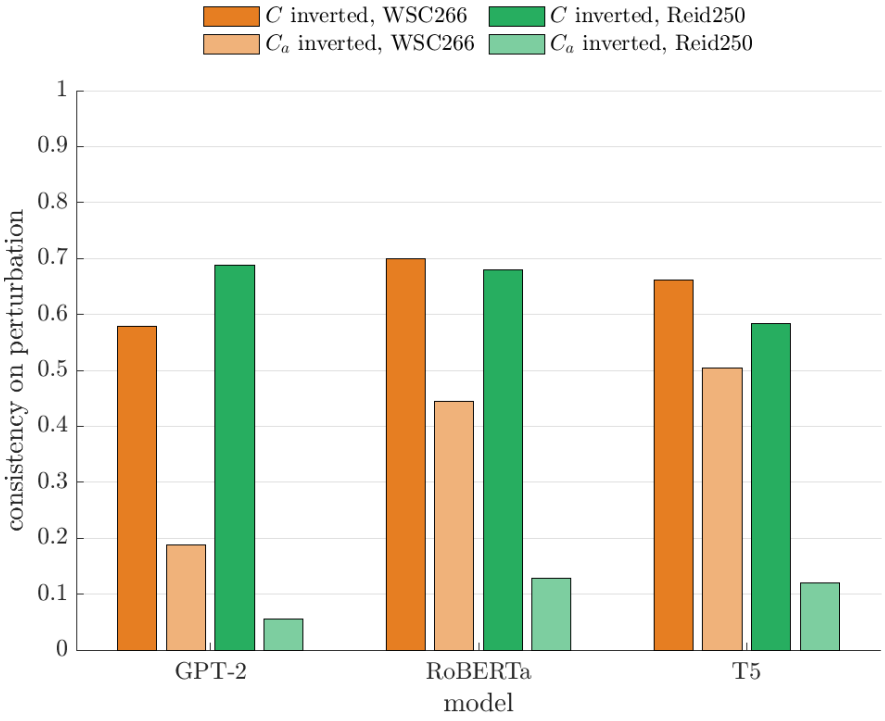

Figure 4.11: Consistency on inverted perturbation of Reid250.

The inversion consistencies are fairly consistent, so to speak, between WSC266 and Reid250. In particular, the consistencies on Reid250 are about equally mediocre. Also, consistencies on Reid250, unlike WSC266, are fairly consistent across models.

4.7 Accuracy on Adversarial Dyads

Completing our evaluation protocol, we calculate the accuracy of each model on the adversarial dyads of Reid250×2 and its inversions: by key and by boolean. Specifically, we apply the boolean wrapper (3.74a) and the truth pair ‖true, false‖:

It is ‖true, false‖ that [original schema].

As usual, we report results only under the standard setup for Reid250.

Our goal is to answer two questions. First, how accurate are the models on Reid250×2? Second, how do both types of inversion affect the accuracy of the models?

Table 4.38 shows problem accuracy a and schema accuracy A for each model on Reid250×2 and its inversions, under the standard setup for Reid250. It also includes

the differences between each perturbation and the original dataset, as well as the statistical significance of those differences and of each accuracy versus chance.

Model	Perturbation	Metric	Value	p	Perturbation − Original	p
GPT-2	by answer	a	0.524	.3	–	–
		A	0.168	.003	–	–
	by key	a	0.502	$> .9$	−0.022	.2
		A	0.108	$< .001$	−0.060	.01
	by boolean	a	0.518	.4	−0.006	.7
		A	0.176	.008	+0.076	$< .001$
RoBERTa	by answer	a	0.526	.3	–	–
		A	0.152	$< .001$	–	–
	by key	a	0.518	.4	−0.008	.7
		A	0.148	$< .001$	−0.004	.9
	by boolean	a	0.504	.9	−0.022	.1
		A	0.020	$< .001$	−0.100	$< .001$
T5	by answer	a	0.524	.3	–	–
		A	0.176	.008	–	–
	by key	a	0.502	$> .9$	−0.022	.2
		A	0.128	$< .001$	−0.048	.06
	by boolean	a	0.500	–	−0.024	.04
		A	0.000	$< .001$	−0.088	$< .001$

Table 4.38: Accuracy on Reid250×2 and its inversions: by key and by boolean.

Regarding the schema-based metric A: we have distributed problems into schemas in accordance with the discussion in Section 3.5.6.

Figure 4.12 displays the accuracies from Table 4.38 with 95% confidence intervals.

In all cases, problem accuracy is marginally better than chance, $p \geq .3$, and schema accuracy is significantly worse than chance, $p \leq .008$.

Solving by boolean, T5 answered *true* to every problem, RoBERTa answered *true* to 98.4% of them, and GPT-2 answered *true* to 54.6%.

To put it simply, the models make no progress at all on Reid250×2. It appears to be a far more challenging test of common-sense reasoning than WSC266.

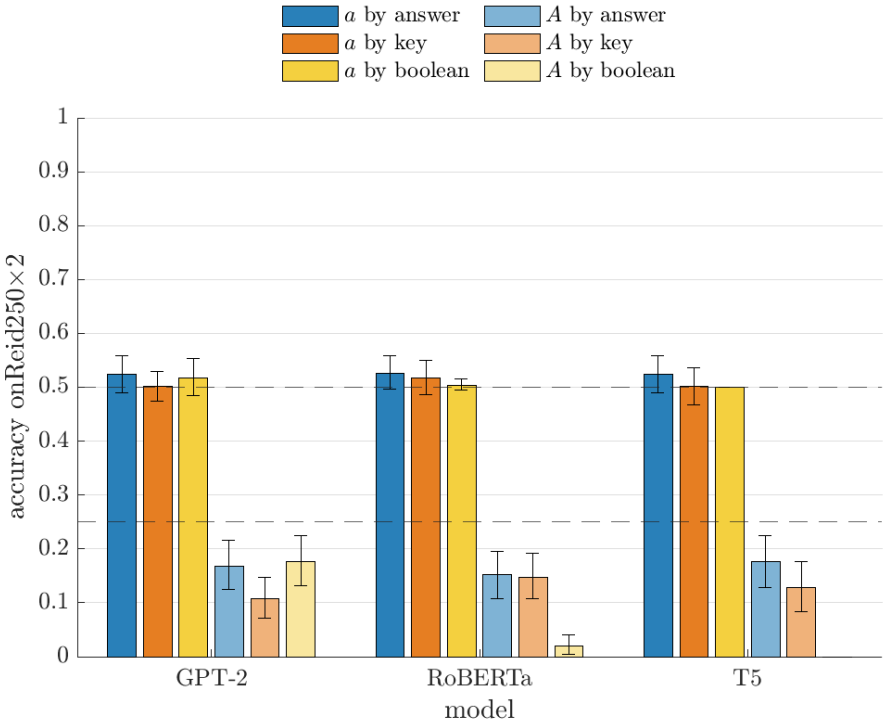

Figure 4.12: Accuracy on Reid250×2 and its inversions: by key and by boolean.

Chapter 5

Conclusion

In Chapter 4, we stated and discussed the results of experiments designed to evaluate common-sense reasoning in pretrained transformer-based language models.

In this chapter, we review the progress we made on our research problem, including results and contributions (Section 5.1), then discuss the limitations, generalizations, and possible future directions of our research (Section 5.2).

5.1 Results and Contributions

We described our research problem and objectives in Section 1.2.

At this point, we have completed all our objectives: to clean up the WSC273 test set; to systematically perturb the clean dataset; to reevaluate three language models on the clean dataset using a consistency-based evaluation protocol; to generalize the Winograd schema as a family of problems in natural language processing; to create a new test of common-sense reasoning out of our new schemas; and to evaluate the same models on the new dataset, again using a consistency-based evaluation protocol.

With those objectives complete, our main contributions are as follows: a cleaner Winograd test set in a standardized format; two perturbations of that dataset suitable for measuring consistency; a generalization of the Winograd schema; a new common-sense reasoning challenge; an evaluation protocol for generalized Winograd schemas; and new results on common-sense reasoning in notable pretrained language models, including the best scoring methods or implementations of stochastic fill-in-the-blank.

Here, we review those results, without repeating the analysis in Chapter 4.

In Section 4.2, we showed that each of the pretrained models GPT-2, RoBERTa, and T5 can achieve better than random accuracy on WSC273, and that the choice of scoring method, in general, can significantly affect each model's accuracy. For GPT-2, the most accurate scoring method is smart scoring, which we introduced. For RoBERTa, the most accurate scoring method is mean statement scoring, which

is computationally intensive but only marginally more accurate than mean multi-mask scoring. For T5, the most accurate scoring method is mean prefix scoring, which we also introduced. Under the best scoring methods, T5 is more accurate than RoBERTa, which is marginally more accurate than GPT-2. Each model is robust with respect to the length in tokens of the answer options. Our changes to WSC273 to create WSC266 do not significantly affect accuracy of any model.

In Section 4.3, we showed how each model's accuracy changes on the switched, adjectival, unbalanced, and inverted perturbations. We were able to characterize each model according to its performance profile on perturbations; for example, on the unbalanced perturbation, RoBERTa loses the most accuracy and GPT-2 the least, whereas on the inverted perturbation, the order is reversed. Conversely, we can characterize each perturbation according to the performance of the models on it. We obtained much higher consistency scores than those reported in the literature.

In Section 4.4, we showed that the associative subset of WSC266 is indeed far more associative than its complement: under the no-candidate perturbation, designed to detect spurious correlations from associativity, the models perform no better than chance on the non-associative subset, but fairly well on the associative subset.

In Section 4.5, we showed that Reid250 is a more challenging test of common-sense reasoning than WSC266, as the performance of every model plummets on it. T5 is still more accurate than RoBERTa, which is still slightly more accurate than GPT-2. Every model is still robust with respect to answer option token counts.

In Section 4.6, we showed that the models are no more accurate on the inverted perturbation of Reid250 than on the original dataset, and that model consistencies with respect to inversion are mediocre. There is little evidence from either WSC266 or Reid250 that an inverted schema is substantially equivalent to its original formulation. This lack of evidence may reflect the inconsistency of the models more than it does any fundamental difference in the task or content underlying the inverted schema.

In Section 4.7, we put the final nail in the coffin of model performance on Reid250: on the adversarial dyads that make up Reid250×2, every model exhibits random problem accuracy and worse than random schema accuracy, whether solving them by answer, by key, or by boolean. That is, wrapping an adversarial schema, which is a simple multiple-choice fill-in-the-blank test of common-sense verbal reasoning, in a

phrase of the form "it is [true, false] that" appears to have a devastating effect on the ability of pretrained language models to consistently imitate reasoning.

Based on each model's performance on the new challenge, as well as its consistency on both WSC266 and Reid250, we argue that there is no convincing evidence of any verbal reasoning process in these pretrained transformer-based language models.

5.2 Limitations, Generalizations, and Future Directions

Our research has several notable limitations. On the other hand, that means that it can be generalized in several ways, and indeed we see a few directions for possible future research in this area, based on the literature and our own findings.

Some of the limitations are relatively minor. By convention, in our unbalanced perturbation, we always replaced the second of two candidates. We do not know what effect this choice had, because we have not studied the effect of replacing the first candidate. But we can straightforwardly create a version of the perturbation that targets the first candidate in each schema. And although we mentioned correcting model accuracies to account for spurious correlations, we did not take consistency into account in any of our adjustments: this is certainly a possible area of improvement.

Of course, we did not finetune any models to improve performance. Now, we did argue that it is important to assess, if not actually improve, common-sense reasoning in the pretrained models on which finetuned models are based. Still, we could finetune a model on a Winograd dataset and see how that affects its performance on Reid250. We could also try to finetune a model on Reid250 or another new adversarial dataset and see how that affects its performance on Winograd schemas.

Elazar et al. [22] have noted that "the typical training procedure" of a language model "does not encourage consistency." Apart from finetuning models on Winograd datasets or adversarial datasets, we could try to train a model for consistency. We already have a few Winograd perturbations that could serve the purpose. If we could systematically perturb Reid250 as well, we would have the raw material for training language models to minimize some kind of consistency-based loss function.

Our common-sense reasoning challenge, Reid250, has of course never been studied before, and we cannot really claim that it was thoroughly tested to be well balanced by type of content or by difficulty. We did make an effort to include roughly the same

number of problems from several broad areas of common-sense verbal reasoning, but our process was far from rigorous. Indeed, we decided not to pursue filtering or any other algorithmic form of balancing. Moreover, the challenge is made up of two different types of schema, substitution and transposition, and we have not studied the differential performance of models on the two types. Each of these limitations is also a possible generalization. And, of course, we can always expand Reid250.

Along those lines, given that we characterized models and perturbations according to the performance of the former on the latter, we may also be able to empirically categorize schemas from particular Winograd or adversarial datasets according to the performance of various models on those schemas and various perturbations thereof.

But is any one test enough? Sakaguchi et al. [85] have argued that "we now need AI algorithms to compose challenges that are hard enough for AI, which requires *dynamic* datasets that evolve together with the evolving state-of-the-art."

That is, even with a new, more challenging test of common-sense reasoning, it should be possible to train a sufficiently large transformer-based language model to solve it, if only by overfitting. We might be able to defeat such tricks by systematically generating new challenges. Then again, maybe all our tests will be defeated, in which case we would again be confronted by two possibilities: either common-sense reasoning has been achieved this time, or the entire challenge-generating process is flawed.

Before getting ahead of ourselves, can we even generate adversarial schemas, and Reid datasets, algorithmically? It seems possible, given that we built a somewhat challenging common-sense dataset, in part, out of mundane facts about bananas and such. Schemas like (3.63) may not be search-proof, but they seem to be difficult for language models to solve, especially inside a boolean wrapper. Indeed, the fact that we can search for facts about bananas is what makes them easy to generate.

The ultimate goal of such a research project would be a challenge-generating process such that even if a language model has been finetuned to solve one of the challenges, it will perform no better than chance on the next challenge.

Bibliography

[1] Mostafa Abdou, Vinit Ravishankar, Maria Barrett, Yonatan Belinkov, Desmond Elliott, and Anders Søgaard. The sensitivity of language models and humans to Winograd Schema perturbations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7590–7604, July 2020.

[2] He Bai, Peng Shi, Jimmy Lin, Luchen Tan, Kun Xiong, Wen Gao, Jie Liu, and Ming Li. Semantics of the unwritten: the effect of end of paragraph and sequence tokens on text generation with GPT2. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing: Student Research Workshop*, pages 148–162, 2021.

[3] David Bender. Establishing a human baseline for the Winograd Schema Challenge. In *Proceedings of the 26th Modern AI and Cognitive Science Conference*, pages 39–45, Greensboro, USA, 2015.

[4] Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal, 2015.

[5] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, volume 33, pages 1877–1901, December 2020.

[6] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. *arXiv*, May 2020. URL `https://arxiv.org/abs/2005.14165v1`.

[7] Kyunghyun Cho, Bart van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*, pages 1724–1734, Doha, Qatar, October 2014.

[8] Fábio Gagliardi Cozman and Hugo Neri Munhoz. The Winograd schemas from hell. In *Anais do XVII Encontro Nacional de Inteligência Artificial e Computacional*, pages 531–542, Porto Alegre, Brazil, October 2020.

[9] Ido Dagan and Oren Glickman. Probabilistic textual entailment: generic applied modeling of language variability. In *PASCAL Workshop on Learning Methods for Text Understanding and Mining*, Grenoble, France, 2004.

[10] Ido Dagan, Oren Glickman, and Bernardo Magnini. The PASCAL Recognising Textual Entailment Challenge. In *Proceedings of the 1st International Conference on Machine Learning Challenges*, pages 177–190, Southampton, UK, 2005.

[11] Ernest Davis. *Representations of Commonsense Knowledge*. Morgan Kaufmann, San Mateo, USA, 1990.

[12] Ernest Davis. Previous versions of the Winograd Schema collection. August 2013. URL `https://cs.nyu.edu/~davise/papers/WSOldVersions.html`.

[13] Ernest Davis. Logical formalizations of commonsense reasoning: a survey. *Journal of Artificial Intelligence Research*, 59:651–723, 2017.

[14] Ernest Davis. Collection of Winograd schemas in XML. May 2018. URL `https://cs.nyu.edu/~davise/papers/WinogradSchemas/WSCollection.xml`.

[15] Ernest Davis and Leora Morgenstern. Introduction: progress in formal commonsense reasoning. *Artificial Intelligence*, 153:1–12, 2004.

[16] Ernest Davis, Leora Morgenstern, and Charles L. Ortiz Jr. Human tests of materials for the Winograd Schema Challenge 2016. 2016. URL `https://cs.nyu.edu/~davise/papers/WS2016SubjectTests.pdf`.

[17] Ernest Davis, Leora Morgenstern, and Charles L. Ortiz Jr. The Winograd Schema Challenge. November 2022. URL `https://cs.nyu.edu/~davise/papers/WinogradSchemas/WS.html`.

[18] A. C. Davison and D. V. Hinkley. *Bootstrap Methods and Their Application*. Cambridge University Press, 1997.

[19] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *arXiv*, October 2018. URL `https://arxiv.org/abs/1810.04805v1`.

[20] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 4171–4186, Minneapolis, USA, June 2019.

[21] Yanai Elazar. Winograd schema: back to square one. *GitHub*, October 2021. URL `https://github.com/yanaiela/winograd_square_one`.

[22] Yanai Elazar, Nora Kassner, Shauli Ravfogel, Abhilasha Ravichander, Eduard Hovy, Hinrich Schütze, and Yoav Goldberg. Measuring and improving consistency in pretrained language models. *Transactions of the Association for Computational Linguistics*, 9:1012–1031, September 2021.

[23] Yanai Elazar, Hongming Zhang, Yoav Goldberg, and Dan Roth. Back to square one: bias detection, training and commonsense disentanglement in the Winograd schema. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 10486–10500, Punta Cana, Dominican Republic, November 2021.

[24] Ali Emami, Adam Trischler, Kaheer Suleman, and Jackie Chi Kit Cheung. A generalized knowledge hunting framework for the Winograd Schema Challenge. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Student Research Workshop*, pages 25–31, New Orleans, USA, 2–4 June 2018.

[25] Ali Emami, Paul Trichelair, Adam Trischler, Kaheer Suleman, Hannes Schulz, and Jackie Chi Kit Cheung. The KnowRef coreference corpus: removing gender and number cues for difficult pronominal anaphora resolution. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3952–3961, Florence, Italy, July 2019.

[26] Ali Emami, Adam Trischler, Kaheer Suleman, and Jackie Chi Kit Cheung. An analysis of dataset overlap on Winograd-style tasks. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 5855–5865, Barcelona, Spain, December 2020.

[27] Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R. Bowman, and Noah A. Smith. Annotation artifacts in natural language inference data. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 107–112, New Orleans, USA, 2018.

[28] Ivan Habernal, Henning Wachsmuth, Iryna Gurevych, and Benno Stein. The Argument Reasoning Comprehension Task: identification and reconstruction of implicit warrants. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 1, pages 1930–1940, New Orleans, USA, 2018.

[29] Trevor J. Hastie, Robert J. Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer-Verlag, 2nd edition, 2009.

[30] Pengcheng He, Xiaodong Liu, Weizhu Chen, and Jianfeng Gao. A hybrid neural network model for commonsense reasoning. In *Proceedings of the 1st Workshop on Commonsense Inference in Natural Language Processing*, pages 13–21, Hong Kong, China, November 2019.

[31] Pengcheng He, Xiaodong Liu, Weizhu Chen, and Jianfeng Gao. A hybrid neural network model for commonsense reasoning. *arXiv*, 27 July 2019. URL `https://arxiv.org/abs/1907.11983v1`.

[32] Hugging Face. GPT-2 vocabulary file, vocab.json. *GitHub*, February 2019. URL `https://huggingface.co/gpt2/blob/main/vocab.json`.

[33] Hugging Face. RoBERTa vocabulary file, vocab.json. *GitHub*, August 2019. URL `https://huggingface.co/roberta-large/blob/main/vocab.json`.

[34] Hugging Face. T5 tokenizer file, tokenizer.json. *GitHub*, October 2020. URL `https://huggingface.co/t5-11b/blob/main/tokenizer.json`.

[35] Hugging Face. Model card for RoBERTa large model. *GitHub*, September 2022. URL `https://huggingface.co/roberta-large`.

[36] Hugging Face. Model card for GPT-2 XL. *GitHub*, January 2023. URL `https://huggingface.co/gpt2-xl`.

[37] Hugging Face. Model card for T5 11B. *GitHub*, January 2023. URL `https://huggingface.co/t5-11b`.

[38] Dan Jurafsky and James H. Martin. *Speech and Language Processing*. Prentice-Hall, 2nd edition, 2008.

[39] Nal Kalchbrenner and Phil Blunsom. Recurrent continuous translation models. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1700–1709, Seattle, USA, 2013.

[40] Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. UNIFIEDQA: crossing format boundaries with a single QA system. *arXiv*, October 2020. URL `https://arxiv.org/abs/2005.00700v2`.

[41] Vid Kocijan. Dataset: MaskedWiki. *Oxford University Research Archive*, 2019. URL `https://ora.ox.ac.uk/objects/uuid:9b34602b-c982-4b49-b4f4-6555b5a82c3d`.

[42] Vid Kocijan. Dataset: WikiCREM. *Oxford University Research Archive*, 2019. URL https://ora.ox.ac.uk/objects/uuid:c83e94bb-7584-41a1-aef9-85b0e764d9e3.

[43] Vid Kocijan, Oana-Maria Camburu, Ana-Maria Creţu, Yordan Yordanov, Phil Blunsom, and Thomas Lukasiewicz. WikiCREM: a large unsupervised corpus for coreference resolution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 4303–4312, Hong Kong, China, November 2019.

[44] Vid Kocijan, Ana-Maria Creţu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. A surprisingly robust trick for the Winograd Schema Challenge. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4837–4842, Florence, Italy, July 2019.

[45] Vid Kocijan, Ana-Maria Creţu, Oana-Maria Camburu, Yordan Yordanov, and Thomas Lukasiewicz. A surprisingly robust trick for the Winograd Schema Challenge. *arXiv*, May 2019. URL https://arxiv.org/abs/1905.06290v1.

[46] Vid Kocijan, Thomas Lukasiewicz, Ernest Davis, Gary Marcus, and Leora Morgenstern. A review of Winograd Schema Challenge datasets and approaches. *arXiv*, April 2020. URL https://arxiv.org/abs/2004.13831v1.

[47] Vid Kocijan, Ernest Davis, Thomas Lukasiewicz, Gary Marcus, and Leora Morgenstern. The defeat of the Winograd Schema Challenge. *arXiv*, January 2023. URL https://arxiv.org/abs/2201.02387v3.

[48] Taku Kudo and John Richardson. SentencePiece: a simple and language independent subword tokenizer and detokenizer for Neural Text Processing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 66–71, Brussels, Belgium, 2018.

[49] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: a lite BERT for self-supervised learning of language representations. *arXiv*, September 2019. URL https://arxiv.org/abs/1909.11942v1.

[50] Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: a lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*, April 2020.

[51] Mirelle Lapata and Frank Keller. Web-based models for natural language processing. *ACM Transactions on Speech and Language Processing*, 2(1):1–31, 2005.

[52] Hector J. Levesque. The Winograd Schema Challenge. In *10th International Symposium on Logical Formalizations of Commonsense Reasoning*, Stanford, USA, 2011.

[53] Hector J. Levesque, Ernest Davis, and Leora Morgenstern. The Winograd Schema Challenge. In *Proceedings of the 13th International Conference on Principles of Knowledge Representation and Reasoning*, pages 552–561, Rome, Italy, 2012.

[54] Xiang Lorraine Li, Adhiguna Kuncoro, Jordan Hoffmann, Cyprien de Masson d'Autume, Phil Blunsom, and Aida Nematzadeh. A systematic investigation of commonsense knowledge in large language models. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 11838–11855, Abu Dhabi, United Arab Emirates, December 2022.

[55] Sheng-Chieh Lin, Jheng-Hong Yang, Rodrigo Nogueira, Ming-Feng Tsai, Chuan-Ju Wang, and Jimmy Lin. TTTTTackling WinoGrande schemas. *arXiv*, March 2020. URL `https://arxiv.org/abs/2003.08380v1`.

[56] Tal Linzen. How can we accelerate progress towards human-like linguistic generalization? In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5210–5217, July 2020.

[57] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. RoBERTa: a robustly optimized BERT pretraining approach. *arXiv*, 26 July 2019. URL `https://arxiv.org/abs/1907.11692v1`.

[58] Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. UNICORN on RAINBOW: a universal commonsense reasoning model on a new multitask benchmark. In *Proceedings of the 35th AAAI Conference on Artificial Intelligence: AAAI-21 Technical Tracks 15*, pages 13480–13488, May 2021.

[59] Nicholas Lourie, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. UNICORN on RAINBOW: a universal commonsense reasoning model on a new multitask benchmark. *arXiv*, March 2021. URL `https://arxiv.org/abs/2103.13009v1`.

[60] John McCarthy. *Programs with Common Sense*. National Physical Laboratory, 1959.

[61] John McCarthy. Artificial intelligence, logic and formalizing common sense. In Richmond H. Thomason, editor, *Philosophical Logic and Artificial Intelligence*, pages 161–190. Springer, Dordrecht, Netherlands, 1989.

[62] R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3428–3448, Florence, Italy, July 2019.

[63] Leora Morgenstern, Ernest Davis, and Charles L. Ortiz Jr. Planning, executing, and evaluating the Winograd Schema Challenge. *AI Magazine*, 37(1):50–54, 2016.

[64] Nasrin Mostafazadeh, Michael Roth, Annie Louis, Nathanael Chambers, and James Allen. LSDSem 2017 shared task: the Story Cloze Test. In *Proceedings of the 2nd Workshop on Linking Models of Lexical, Sentential and Discourse-Level Semantics*, pages 46–51, Valencia, Spain, 2017.

[65] Kevin Patrick Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[66] Nikita Nangia and Samuel R. Bowman. Human vs. muppet: a conservative estimate of human performance on the GLUE benchmark. *arXiv*, June 2019. URL `https://arxiv.org/abs/1905.10425v3`.

[67] Timothy Niven and Hung-Yu Kao. Probing neural network comprehension of natural language arguments. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4658–4664, Florence, Italy, July 2019.

[68] Stellan Ohlsson, Robert H. Sloan, György Turán, Daniel Uber, and Aaron Urasky. An approach to evaluate AI commonsense reasoning systems. In *Proceedings of the 25th International Florida Artificial Intelligence Research Society Conference*, pages 371–374, Marco Island, USA, 2012.

[69] Stellan Ohlsson, Robert H. Sloan, György Turán, and Aaron Urasky. Measuring an artificial intelligence system's performance on a verbal IQ test for young children. *Journal of Experimental and Theoretical Artificial Intelligence*, 29(4): 679–693, 2016.

[70] OpenAI. GPT-4 technical report. *arXiv*, March 2023. URL `https://arxiv.org/abs/2303.08774v2`.

[71] Juri Opitz and Anette Frank. Addressing the Winograd Schema Challenge as a sequence ranking task. In *Proceedings of the 1st International Workshop on Language Cognition and Computational Models*, pages 41–52, Santa Fe, USA, 2018.

[72] Haoruo Peng, Daniel Khashabi, and Dan Roth. Solving hard coreference problems. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 809–819, Denver, USA, June 2015.

[73] Fabio Petroni, Tim Rocktäschel, Sebastian Riedel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, and Alexander 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*, pages 2463–2473, Hong Kong, China, 2019.

[74] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. Technical report, OpenAI, 11 June 2018. URL `https://openai.com/research/language-unsupervised`.

[75] Alec Radford, Jeffrey Wu, Dario Amodei, Daniela Amodei, Jack Clark, Miles Brundage, and Ilya Sutskever. Better language models and their implications. *OpenAI Blog*, February 2019. URL `https://openai.com/blog/better-language-models/`.

[76] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. Technical report, OpenAI, February 2019. URL `https://openai.com/blog/better-language-models/`.

[77] Alec Radford, Jong Wook Kim, and Jeffrey Wu. GPT-2 output dataset. *GitHub*, February 2021. URL `https://github.com/openai/gpt-2-output-dataset`.

[78] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv*, October 2019. URL `https://arxiv.org/abs/1910.10683v1`.

[79] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, January 2020.

[80] Altaf Rahman and Vincent Ng. Resolving complex cases of definite pronouns: the Winograd Schema Challenge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 777–789, Jeju Island, Korea, 2012.

[81] Yu-Ping Ruan, Xiaodan Zhu, Zhen-Hua Ling, Zhan Shi, Quan Liu, and Si Wei. Exploring unsupervised pretraining and sentence structure modelling for Winograd Schema Challenge. *arXiv*, April 2019. URL `https://arxiv.org/abs/1904.09705v1`.

[82] Rachel Rudinger, Jason Naradowsky, Brian Leonard, and Benjamin Van Durme. Gender bias in coreference resolution. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, volume 2, pages 8–14, New Orleans, USA, 2018.

[83] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: an adversarial Winograd Schema Challenge at scale. *arXiv*, July 2019. URL `https://arxiv.org/abs/1907.10641v1`.

[84] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: an adversarial Winograd Schema Challenge at scale. *arXiv*, November 2019. URL `https://arxiv.org/abs/1907.10641v2`.

[85] Keisuke Sakaguchi, Ronan Le Bras, Chandra Bhagavatula, and Yejin Choi. WinoGrande: an adversarial Winograd Schema Challenge at scale. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence*, pages 8732–8740, New York, USA, February 2020.

[86] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Pseudo-likelihood reranking with masked language models. *arXiv*, October 2019. URL `https://arxiv.org/abs/1910.14659v1`.

[87] Julian Salazar, Davis Liang, Toan Q. Nguyen, and Katrin Kirchhoff. Masked language model scoring. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2699–2712, 2020.

[88] Roy Schwartz and Gabriel Stanovsky. On the limitations of dataset balancing: the lost battle against spurious correlations. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 2182–2194, Seattle, USA, July 2022. Association for Computational Linguistics.

[89] Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*, volume 1, pages 1715–1725, Berlin, Germany, August 2016. Association for Computational Linguistics.

[90] Robyn Speer, Joshua Chin, and Catherine Havasi. ConceptNet 5.5: an open multilingual graph of general knowledge. In *Proceedings of the 31st AAAI Conference on Artificial Intelligence*, pages 4444–4451, San Francisco, USA, 2017.

[91] Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, volume 2, pages 3104–3112, Montreal, Canada, December 2014.

[92] Wilson L. Taylor. "Cloze procedure": a new tool for measuring readability. *Journalism Quarterly*, 30(4):415–433, 1953.

[93] Paul Trichelair. Winograd dataset, WSC_associative_label.json. *GitHub*, August 2019. URL `https://github.com/ptrichel/How-Reasonable-are-Common-Sense-Reasoning-Tasks/blob/master/WSC_associative_label.json`.

[94] Paul Trichelair. Winograd dataset, WSC_switched_label.json. *GitHub*, August 2019. URL `https://github.com/ptrichel/How-Reasonable-are-Common-Sense-Reasoning-Tasks/blob/master/WSC_switched_label.json`.

[95] Paul Trichelair, Ali Emami, Jackie Chi Kit Cheung, Adam Trischler, Kaheer Suleman, and Fernando Diaz. On the evaluation of common-sense reasoning in natural language understanding. In *NeurIPS 2018 Workshop: Critiquing and Correcting Trends in Machine Learning*, Montreal, Canada, December 2018.

[96] Paul Trichelair, Ali Emami, Adam Trischler, Kaheer Suleman, and Jackie Chi Kit Cheung. How reasonable are common-sense reasoning tasks: a case-study on the Winograd Schema Challenge and SWAG. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, pages 3382–3387, Hong Kong, China, November 2019.

[97] Trieu H. Trinh and Quoc V. Le. A simple method for commonsense reasoning. *arXiv*, 7 June 2018. URL `https://arxiv.org/abs/1806.02847v1`.

[98] Trieu H. Trinh and Quoc V. Le. A simple method for commonsense reasoning. *arXiv*, September 2019. URL `https://arxiv.org/abs/1806.02847v2`.

[99] Lifu Tu, Garima Lalwani, Spandana Gella, and He He. An empirical study on robustness to spurious correlations using pre-trained language models. *Transactions of the Association for Computational Linguistics*, 8:621–633, 2020.

[100] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6000–6010, Long Beach, USA, December 2017.

[101] Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. GLUE: a multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, November 2018.

[102] Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. SuperGLUE: a stickier benchmark for general-purpose language understanding systems. In *Proceedings of the 33rd International Conference on Neural Information Processing Systems*, pages 3266–3280, Vancouver, Canada, December 2019.

[103] Terry Winograd. *Procedures as a representation for data in a computer program for understanding natural language*. PhD thesis, Massachusetts Institute of Technology, 1970.

[104] Terry Winograd. *Understanding Natural Language*. Academic Press, 1972.

[105] Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 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, October 2020.

[106] Zhi-Xiu Ye, Qian Chen, Wen Wang, and Zhen-Hua Ling. Align, mask and select: a simple method for incorporating commonsense knowledge into language representation models. *arXiv*, August 2019. URL `https://arxiv.org/abs/1908.06725v1`.

[107] Aston Zhang, Zachary C. Lipton, Mu Li, and Alexander J. Smola. Dive into deep learning. *arXiv*, 2023. URL `https://arxiv.org/abs/2106.11342v4`.

[108] Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. Evaluating commonsense in pre-trained language models. *arXiv*, November 2019. URL `https://arxiv.org/abs/1911.11931v1`.

[109] Xuhui Zhou, Yue Zhang, Leyang Cui, and Dandan Huang. Evaluating commonsense in pre-trained language models. In *Proceedings of the 34th AAAI Conference on Artificial Intelligence: AAAI-20 Technical Tracks 5*, pages 9733–9740, New York, USA, 2020.