Bachelorarbeit

Verbal Epistemic Uncertainty Estimation for Numeric Values with GPT-3

Eberhard Karls Universität Tübingen
Mathematisch-Naturwissenschaftliche Fakultät
Wilhelm-Schickard-Institut für Informatik
Methoden des Maschinellen Lernens
Joschka Braun, joschka.braun@student.uni-tuebingen.de

Bearbeitungszeitraum: von 20.08.2022 bis 28.10.2022

Betreuer/Gutachter: Prof. Dr. Philipp Hennig, Universität Tübingen
Betreuer: Marius Hobbhahn, Universität Tübingen

Selbstständigkeitserklärung

Hiermit erkläre ich, dass ich diese schriftliche Abschlussarbeit selbständig verfasst habe, keine anderen als die angegebenen Hilfsmittel und Quellen benutzt habe und alle wörtlich oder sinngemäß aus anderen Werken übernommenen Aussagen als solche gekennzeichnet habe

---

Joschka Braun (Matrikelnummer 4163053), October 30, 2022

Acknowledgments

I am extremely grateful to my supervisors, Prof. Dr. Philipp Hennig and Marius Hobbhahn, for their invaluable feedback and guidance throughout my thesis.

I would also like to thank OpenAI for providing access to their InstructGPT models via their API.

I could not have undertaken this research project without the advice, encouragement, and support of Clara and my parents Christiane and Christof.

I am also grateful for the moral support of Dominik, Moritz, Pauline, and Bjarne during the writing process of my thesis.

Abstract

I study whether the latest versions of the Large Language Model GPT-3 can produce well-calibrated and robust zero-shot verbal epistemic uncertainty estimates about their knowledge. As context for later results, I first estimate each model's knowledge about my dataset, by measuring the accuracy of their numerical point estimates. Secondly, I conduct four experiments with increasingly complex possibilities to express uncertainty about whether a proposed answer is true. Thirdly, I prompt the models to estimate their uncertainty via confidence intervals. Every experiment uses the GPT-3 models Ada, Babbage, Curie, and Davinci, which differ in the number of parameters and allow me to test for changes due to the scaling of language models. To test for robustness, I execute similar experiments, use differently phrased prompts, and create adversarial examples. I find that all models can verbally express uncertainty, but only the largest model returns calibrated epistemic uncertainty estimates for some experiments. Smaller models' epistemic uncertainty estimates are not calibrated for any of the experiments. Overall, larger models are more robust to changes in the prompt and have significantly better question-answering abilities. I hope that these results provide insight into the capabilities of current language models to verbally express epistemic uncertainty. While their verbal epistemic uncertainty estimates might seem plausible at first sight, they are not calibrated, coherent or robust in most cases.

German Abstract

Ich untersuche, ob die neuesten Versionen des Sprachmodells GPT-3 gut kalibrierte und robuste verbale Schätzungen der epistemischen Unsicherheit über ihr eigenes Wissen liefern können. Als Kontext für die späteren Ergebnisse schätze ich zunächst das Wissen der einzelnen Modelle für meinen Datensatz, indem ich die Genauigkeit ihrer Punktschätzungen messe. Zweitens führe ich vier Experimente mit zunehmend komplexeren Möglichkeiten durch, die eigene Unsicherheit darüber auszudrücken, ob eine vorgeschlagene Antwort wahr ist. Drittens lasse ich die Modelle ihre Unsicherheit über Konfidenzintervalle ausdrücken. In jedem Experiment werden die GPT-3 Modelle Ada, Babbage, Curie und Davinci verwendet, die sich in der Anzahl der Parameter unterscheiden und es mir ermöglichen, Veränderungen aufgrund der Skalierung der Sprachmodelle zu testen. Um die Robustheit zu testen, führe ich ähnliche Experimente durch, verwende unterschiedlich formulierte Aufgabenstellungen und erstelle gegnerische Beispiele. Ich stelle fest, dass alle Modelle Unsicherheit verbal ausdrücken können, aber nur das größte Modell für einige Experimente seine epistemische Unsicherheit kalibriert schätzen kann. Kleinere Modelle sind nicht kalibriert im Schätzen ihrer Unsicherheit. Insgesamt sind größere Modelle robuster gegenüber Änderungen in der Aufgabenstellung und haben eine deutlich bessere Fähigkeit, Fragen zu beantworten. Ich hoffe, dass diese Ergebnisse einen Einblick in die Fähigkeiten der aktuellen Sprachmodelle geben, epistemische Unsicherheit verbal auszudrücken. Während ihre verbalen Schätzungen der epistemischen Unsicherheit auf den ersten Blick plausibel erscheinen können, sind sie in den meisten Fällen nicht kalibriert, kohärent oder robust.

Contents

1. Introduction

In recent years, large language models have become increasingly popular in the field of machine learning. These models can learn from large amounts of data and can be used for a variety of natural language processing tasks such as automated question answering, machine translation, and text summarization (Liu et al., 2019; Brown et al., 2020; Rae et al., 2021; Chowdhery et al., 2022; Smith et al., 2022). With the development of the transformer architecture by Vaswani et al. (2017), language models leaped forward in terms of their ability to generate human-like text. Furthermore, the number of parameters and amount of compute used to train such large-scale models has increased very rapidly in recent years (Sevilla et al., 2022). Large-scale transformer-based language models like GPT-3 have shown impressive task-agnostic zero-shot and few-shot capabilities, without any fine-tuning (Brown et al., 2020). This means that they are able to learn new tasks from scratch, simply by receiving instructions or observing a few examples. GPT-3's ability to follow instructions was further improved by Ouyang et al. (2022), through fine-tuning via reinforcement learning from human feedback, and resulted in the improved InstructGPT models. I use four similarly trained Instruct models in my Bachelor's thesis, hereby coined InstructGPT models. Those models will be assessed in zero-shot settings for their verbal epistemic uncertainty estimations. I will evaluate and compare the performance of different epistemic uncertainty estimation experiments, such as estimating the probability that a proposed answer is true or giving confidence intervals for numeric answers. As Kadavath et al. (2022) point out, the ability of AI systems to evaluate their level of confidence in their own knowledge accurately, faithfully, and well-calibrated is a prerequisite for honest, robust, and trustworthy AI systems that can be used safely in applications. A lack of such abilities can have negative consequences in the real world (Hendrycks et al., 2021), as AI systems like GPT-3 are already used in hundreds of applications today (Pilipiszyn, 2021). I hope that the results of my Bachelor's thesis will contribute to a better understanding of the capabilities and limitations of InstructGPT and current language models.

My main findings for the InstructGPT models are as follows:

Larger InstructGPT models have better question-answering capabilities and make smaller errors in their point estimates. The respective mean point estimate accuracies for my dataset are 3% for Ada, 12% for Babbage, 30% for Curie, and 65% for Davinci.

All InstructGPT models are capable of verbally estimating their epistemic uncertainty in a variety of ways. When prompted correctly, every model returned valid estimates for most experiments in more than 90% of cases.

Prompt design has a significant impact on InstructGPT models' answers, and uncertainty estimates between experiments are not internally coherent. Uncertainty estimated are substantially impacted by how the task is explained and in what order possible answers are presented. Different alterations can produce contradicting uncertainty estimates.

Larger InstructGPT models are more reliable to follow instructions than smaller InstructGPT models. The number of ambiguous answers declines and the quality of answers increases with the number of parameters.

Only the biggest InstructGPT model is calibrated for the 'True or False' and the 'True or False or I don't know experiment'. None of the other models show real calibration for any of the experiments.

The calibration of the biggest InstructGPT model for the 'True or False' and the 'True or False or I don't know' experiment is robust and only mildly affected by most adversarial examples. Even when the question is significantly altered, calibration remains as long as the meaning of the question and answer to the question remains the same.

2. Background

2.1. Natural Language Processing

Natural language processing (NLP) has the goal of enabling computers to analyze, understand, and generate human natural language. Typical NLP tasks include machine translation, text classification, language understanding, text summarization, commonsense inference, and question answering. NLP research started in the 1950s focussing on Symbolic NLP, an approach which is based on the idea that language can be represented as a set of symbols and meaning can be derived by formal, rule-based manipulation of those symbols. In the 1990s, statistical NLP became the dominant approach, which follows the idea that natural language can be modeled as a probabilistic process and that the meaning of a sentence can be derived from the statistical properties of the data. Since the 2010s, machine learning methods like deep neural networks became popular in NLP. Recurrent neural networks (RNNs) like long short-term memory (LSTM) (Hochreiter and Schmidhuber, 1997), and gated recurrent units (GRUs) (Chung et al., 2014), as well as convolutional neural networks (CNN) (Wang and Gang, 2018) achieved then state-of-the-art performance in many NLP tasks. However, such models can be difficult to parallelize and there is usually a need for task-specific datasets and fine-tuning. This has so far limited the size and general capabilities of language models.

2.2. Transformer Architecture

In 2017, the transformer architecture was developed by Vaswani et al. (2017) and outperformed previous models in a variety of NLP tasks. Transformers scale with parameters and training data, use efficient parallel training, and capture long-range sequence features of text. Today, the transformer is the most prominent architecture both for natural language understanding and generation (Wolf et al., 2020). In contrast to previous approaches, the transformer architecture is solely based on an attention mechanism, without a need for recurrence and convolutions. Attention captures the relevance of the relationship between tokens in a text and therefore provides context for interpretation. A high attention score means two tokens are relevant to each other, a low score indicates that they are unrelated to one another. A good example for illustration can be found in Vig (2019): "The quick, brown fox jumps over the lazy". The word "brown" is very relevant to “fox”, but irrelevant to "lazy". Therefore, the attention score for "brown" and "fox" should be high and the attention score for "brown" and "lazy" should be low. In contrast to the standard weights of the transformer, which are fixed during runtime, the attention weights change during runtime. GPT-3 and InstructGPT use so-called masked multi-head self-attention, which improves parallel training and can express more high-level attention relationships (Ghojogh and Ghodsi, 2020). Overall, the transformer architecture is particularly suited for pretraining on large text corpora, leading to improvements on many downstream tasks including question answering, text classification, machine translation, sentiment analysis, language understanding, commonsense reasoning and summarization (Wang et al., 2018; Yang et al., 2019; Brown et al., 2020; Chowdhery et al., 2022; Smith et al., 2022).

2.3. GPT-3 and InstructGPT

In 2020, OpenAI released the Generative Pre-trained Transformer 3 (GPT-3), a 175 billion parameter language model, bigger than any previous language model. The scaling-up of the model greatly improved task-agnostic zero-shot and few-shot performance in many NLP tasks (Brown et al., 2020). In the paper, eight models were trained, ranging from 125 million parameters to 175 billion parameters. Four different model sizes are available through the OpenAI API, which allowed me to run the experiments with models of different sizes. All models use the subword tokenization algorithm Byte-Pair Encoding (BPE) by Sennrich et al. (2015). First, a pre-tokenizer splits the training data into words and determines the number of occurrences of each word. Subsequently, BPE creates a vocabulary of a specified size by merging symbols that occur in the set of unique words. This means that the models are working on tokens, which can in some instances be whole words like " you" (including the space), single symbols like "!", or subwords like "lor". Models get trained by predicting the next token, given the previous tokens. All models were trained on around 300 billion of such tokens, which were part of a dataset including English-language Wikipedia (3 billion tokens), Books1 and Books2 (12 and 55 billion tokens respectively), WebText2 (19 billion tokens) and CommonCrawl (filtered to around 410 billion). This means that the models have potentially learned an amplitude of knowledge about the real world that is implicitly contained in the dataset. In 2022, OpenAI improved their previous models' abilities to follow instructions by fine-tuning the models using reinforcement learning from human feedback, which resulted in the improved InstructGPT models (Ouyang et al., 2022). This means that InstructGPT is able to learn new tasks from scratch, simply by receiving instructions or observing a few examples. Precisely because InstructGPT has improved task-agnostic performance, the InstructGPT models are well suited for my experiments. Unfortunately, the specific InstructGPT models published in the paper are not available through the OpenAI API. However, other Instruct models are available, whose training is very similar or the same to the original InstructGPT models. For simplicity's sake, I will refer to those models available to me as InstructGPT models.

2.4. Ada, Babbage, Curie and Davinci

Through the OpenAI API, four different InstructGPT models are available. In alphabetical and parameter size ascending order, these models are: Ada, named after Ada Lovelace, who is often regarded as the first computer programmer (Fuegi and Francis, 2003); Babbage, named after Charles Babbage, who originated the concept of a digital programmable computer (Copeland, 2020); Curie, named after Marie Curie, the first and only person to win the Nobel Prize in two scientific fields (Calame, 2005); and Davinci, named after Leonardo da Vinci, one of the greatest Renaissance artists (Nicholl, 2004). Although the exact model sizes of Ada, Babbage, Curie and Davinci are not specified by Brown et al. (2020), they can be inferred. According to EleutherAI the Ada, Babbage, Curie and Davinci models' performances line up closely with the 350 million, 1.3 billion, 6.7 billion, and 175 billion models in the OpenAI paper respectively (Gao, 2021). To contextualize these model sizes, I compare the InstructGPT models to other relevant large language models from recent years.

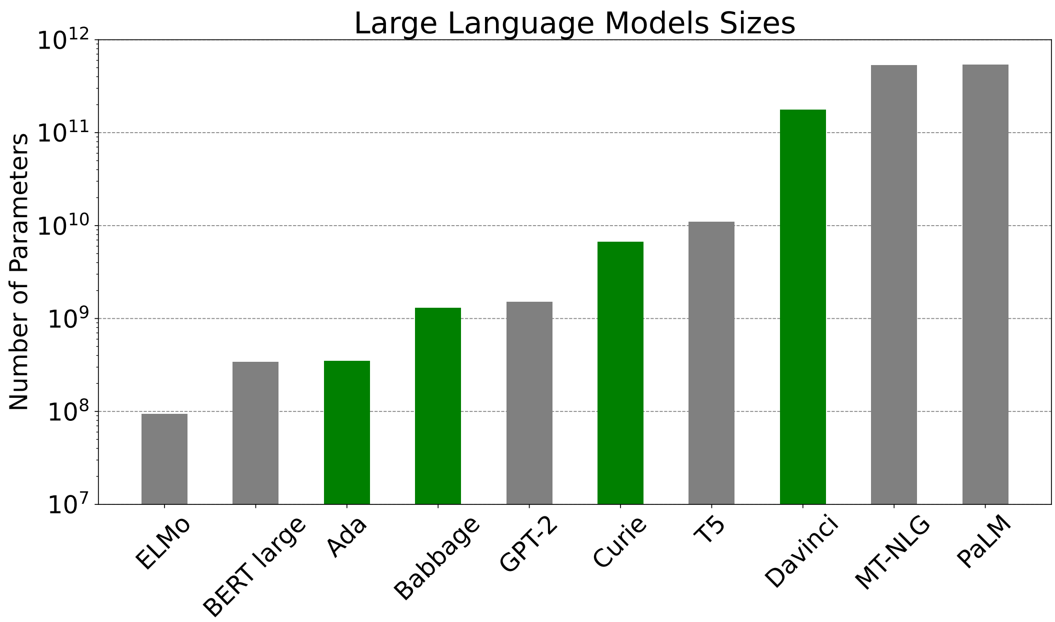

Figure 2.1.: GPT-3 Model Size compared to other LLMs

In 2018, Peters et al. (2018) published their paper "Deep Contextualized Word Representations", in which they present an ensemble of neurally derived representations called ELMo (Embeddings from Language Models). The largest model ELMo$_l$ has around 94 million parameters. Also in 2018, Devlin et al. (2018) propose BERT (Bidirectional Encoder Representations from Transformers), which alleviates the unidirectionality constraint of previous language models by using a "masked language model" (MLM). One year later, Radford et al. (2019) released GPT-2, a 1.5 billion parameters transformer-based language model, the predecessor to GPT-3. In the same year, Raffel et al. (2019) published T5, another transformer-based language model that was pre-trained on about 750 GB of natural English text. T5 also comes in different sizes, the biggest around 11 billion parameters. This year, Smith et al. (2022) trained Megatron-Turing NLG 530B (MT-NLG), a 530 billion parameter transformer-based model, which improved then state-of-the-art performances on several NLP benchmarks for zero-, one-, and few-shot learning. In April this year, a 540-billion parameter transformer language model called Pathways Language Model (PaLM) was published by Chowdhery et al. (2022) from Google Research. The model outperforms average human performance on the BIG-bench benchmark, and often showed steep performance increases with increasing model size.

2.5. Uncertainty Estimates and Calibration

In machine learning, the uncertainty stemming from lack of knowledge is called epistemic uncertainty and the irreducible, statistical uncertainty is called aleatoric uncertainty (Hüllermeier et al., 2022). In my thesis, I want to explore the capability of InstructGPT models to verbally express epistemic uncertainty about the accuracy of their own knowledge. I want to do this by using questions to which a single, unambiguously correct, and known answer exists. Therefore, the expressed uncertainty should only contain uncertainty that the model itself has over its own knowledge, without any irreducible, statistical uncertainty over the answer to the question. If I asked questions, to which no such single, unambiguously correct answer exists, the expressed uncertainty would comprise different forms of uncertainty, and it would be difficult to evaluate answers to such questions adequately. A question with a single, unambiguously correct, and known answer is:

- Who served as the 16th President of the United States of America? - Abraham Lincoln
- How many episodes, in total, does the series How I Met Your Mother have over its nine seasons? - 208

In such cases, it is mostly clear, whether an answer is correct (more on that in the discussion of my dataset 4.6). Examples for questions with no single, unambiguously correct, and known answer are:

- How many people contracted COVID-19 from December 1st, 2019 to September 12th, 2022 at least once? (not known, only estimates exist)
- Is P = NP? (the answer to the P versus NP problem is currently not known)
- Will a fair coin land on heads? (aleatoric uncertainty)
- What is the meaning of life? (unclear if an answer exists)
- What color is the feeling of happiness? (unclear if that question makes sense)

As Lichtenstein and Fischhoff (1980) explain, epistemic uncertainty estimates are expressions of confidence in the state of one's knowledge. A probability statement like "I am 60% certain that Abraham Lincoln was the 16th President of the United States" expresses the internal degree of belief and not a probability in the real world. Epistemic uncertainty estimates should reflect a systematic relationship to the truth of the answer. If numerous uncertainty estimates of 80% are made, around 80% of them should be true. Lichtenstein and Fischhoff (1980) call the formalization of this property calibration. Estimates are well-calibrated if the predicted probabilities are perfectly correlated with the probabilities of correctness. Research in the social sciences has shown, that calibration of humans is usually poor and humans are often overconfident (Lichtenstein et al., 1977; Dunning et al., 1990; Schaefer et al., 2004; Mannes and Moore, 2013). Overconfidence is the judgmental error of overestimating one's own accuracy. It is reasonable to assume that these judgmental errors are ubiquitous in the training dataset of InstructGPT, as the models are trained on human text. As pointed out by Lin et al. (2022) research on the calibration of language models has mostly focused on models' output probabilities, their "logits" (Guo et al., 2017; Jiang et al., 2021). However, such output probability distributions over tokens are not the same as expressing epistemic uncertainty about the claim itself. Instead, I want the model to verbally express its epistemic uncertainty about the claim in natural language, which is called "verbalized probability" by Lin et al. (2022).

2.6. Robustness and Adversarial Examples

Robustness in language models means that the model is able to perform well even when small changes to the textual input are introduced or the textual input is different from the training data. Such changes include paraphrasing the sentence and introducing spelling mistakes or other erroneous symbols. A robust language model should be able to handle such changes without losing accuracy, similar to a human, who would be able to answer a question regardless of small changes to the question. Adversarial examples are purposefully designed to exploit a model's lack of robustness and are therefore used for assessing robustness. Compared to generating adversarial examples in continuous data domains like images, generating natural language adversarial examples that preserve the original meaning is challenging since the text space is discrete and non-differentiable. Textual adversarial attacks are mostly generated through substituting words (Alzantot et al., 2018; Ren et al., 2019; Ebrahimi et al., 2018) or paraphrasing entire sentences (Zhang et al., 2019; Iyyer et al., 2018) to preserve the meaning of a statement or question. While such alterations seem innocuous to humans, research has shown that textual adversarial attacks can successfully deceive large language models like InstructGPT, BERT, and others (Jin et al., 2019; Nie et al., 2019; Wang et al., 2019; Zang et al., 2020; Branch et al., 2022).

2.7. Zero-Short Learning in Language Models

In computer vision research, zero-shot learning aims to classify images that have no training examples (Lampert et al., 2014; Xian et al., 2017). Research on large languagemModels often focuses on zero-shot model adaptations to natural language descriptions of tasks (Puri and Catanzaro, 2019). This allows pretrained language models to perform new tasks when only equipped with instructional prompts (Liu et al., 2021). Results from zero-shot learning are often impressive if the models are prompted appropriately. Kadavath et al. (2022) find that their language models are somewhat calibrated for zero-shot 'True or False' questions. Kojima et al. (2022) show that large language models are decent zero-shot reasoners if prompted correctly. In my experiments, prompts always contain an explicit or implicit natural language instruction of what the model has to do. InstructGPT was not fine-tuned for any of the experiments and prompts do not contain examples, except for the confidence interval experiment.

2.8. Research Question

Is InstructGPT capable of verbally expressing epistemic uncertainty about its knowledge? When prompted to estimate how likely a proposed answer is true, are the returned probabilities between 0% and 100%? When prompted to estimate uncertainty estimates via multiple choice, are answers valid options? When prompted to return a confidence interval for a question with a numeric answer, is InstructGPT able to return a valid confidence interval?

Are InstructGPTs uncertainty estimates coherent and robust? How susceptible are model outputs to changes in the prompt? Are estimates coherent, when changing the order of multiple choice answer options? Do the models adapt its answer to the questions, or are answers mostly static? Do the models return similar uncertainty estimates for different ways to extract uncertainty?

Are InstructGPTs uncertainty estimates calibrated? Are proposed answers, that are far from the correct answer, less likely to be estimated as being true? What are the Brier scores of the confidence intervals? Is InstructGPT overconfident and similarly calibrated, as average humans are? Overall, do InstructGPTs uncertainty estimates make sense?

How does model size change the answer to previous questions? Does robustness or calibration improve with an increase in model size, or do larger models suffer from the same shortcomings? Are there clear trends in performance differences between smaller and larger models?

3. Related Work

Niculescu-Mizil and Caruana (2005) analyze the calibration of classical machine learning algorithms like Naive Bayes, Random Forests, and SVMs but also neural networks. They find that early neural networks are well-calibrated for binary classification tasks. They measure calibration through the model's output probabilities and not through verbalized uncertainty estimates.

Guo et al. (2017) found that calibration for modern neural networks is much worse than for neural networks from a decade earlier. They demonstrate that this deterioration of calibration is due to changes in network architecture and training as well as model size, normalization, and regularization.

Brown et al. (2020) introduce GPT-3 and assess their model's capabilities, including closed-book question answering and general zero-shot and few-shot tasks. They do not test their models for calibration.

Branwen (2020) experiments with GPT-3s ability to express verbalized uncertainty on trivia questions and finds that GPT-3 is incapable of expressing well-calibrated uncertainty estimates and instead often returns "95% confidence" for answers that are wrong and very implausible. This inspired my free uncertainty estimate experiment.

Desai and Durrett (2020) analyze the calibration of BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) for language inference, paraphrase detection, and commonsense reasoning. They find that the "out-of-the-box" models have reasonably well-calibrated output probabilities and that calibration can be improved by post-process temperature scaling, introduced by Guo et al. (2017)

Jiang et al. (2021) find that then state-of-the-art language models like T5 (Raffel et al., 2019) and GPT-3 (Brown et al., 2020) are poorly calibrated for the UnifiedQA dataset by Khashabi et al. (2020). While they use different methods, their research has inspired and informed some of my experiments.

Ouyang et al. (2022) introduce the InstructGPT models. Their approaches informed mine and without access to their InstructGPT models, through the OpenAI API, my research would not have been possible.

Lin et al. (2022) fine-tuned GPT-3 model "Davinci" with supervised learning to express its uncertainty verbally. They compare calibration of uncertainty expressed verbally, to uncertainty extracted from the model output probabilities "logits". Their research inspired my free uncertainty estimate experiment 6.3.4 and my research extends theirs by working with the improved InstructGPT, including "text-davinci-002", and introducing further experiments.

Mielke et al. (2022) analyze the verbal calibration of BlenderBot by Roller et al. (2020) and find that it is poorly calibrated for verbal uncertainty estimates. They use a different method of expressing linguistic uncertainty, by using four different categories:

- **DK** none: admits not to know
- **LO** low: expresses uncertainty
- **HI** high: confidently answers
- **OT** completely ignores the question

Their paper inspired using multiple choice style questions in Chapter 6.

Kadavath et al. (2022) extensively studied uncertainty estimations and calibration of their own language models. For their experiments, they use four language models ranging in size from 800 million to 52 billion parameters, building on previous research by Bai et al. (2022). They find that after fine-tuning with reinforcement learning from human feedback, their model's logits are well-calibrated on 'True or False' and multiple choice questions. However, their models are poorly calibrated for verbally self-evaluating their epistemic uncertainty. I adopt some of their question formats to make my results comparable to theirs.

The main contribution of my Bachelor thesis is the evaluation of InstructGPT's abilities to verbally estimate epistemic uncertainty about its own knowledge. By using the latest and improved version of GPT-3 I use a state-of-the-art language model, which has only been released a couple of months before writing this thesis. By using a dataset with only numeric values as answers, I can focus on novel experiments like estimating uncertainty through confidence intervals. While my Bachelor thesis is hopefully a small contribution to highlighting the limitations of pretrained transformer-based language models' abilities to self-evaluate their confidence, the more challenging and valuable problem of improving calibration by more than prompt design is, due to limitations in ability, funding, and time, not part of this thesis.

4. Methods

4.1. Models

For my experiments, I used the four latest Instruct models Ada (text-ada-001), Babbage (text-babbage-001), Curie (text-curie-001), and Davinci (text-davinci-002), which all are available via the OpenAI API. This enabled me to evaluate performances across different model sizes to understand possible scaling behavior of large language models. Ouyang et al. (2022) from OpenAI published very similar Instruct models that showed improvements in truthfulness, greater ability to follow instructions, and generally improved alignment with human intent. However, the exact models published in the paper are not available through the OpenAI API. Therefore, I used the aforementioned models, which I will refer to as InstructGPT models.

4.2. OpenAI API

The OpenAI API can be found online at `https://openai.com/api/`, with extensive documentation at `https://beta.openai.com/docs/introduction/overview`. I used the python bindings documented in `https://beta.openai.com/docs/libraries/python-bindings`, which allowed me to fully automate the execution and evaluation of my experiments in Jupyter Notebook. The most relevant details of this method are documented in this chapter. With the provided information, my experiments should be replicable for everyone with a basic understanding of the programming language Python. To quickly check some results manually or play around, the OpenAI "Playground" gives quick access and can be found at `https://beta.openai.com/playground`.

4.3. Prompts

The API provides a way to input text as a prompt, and then use the selected model to generate a text completion that attempts to match the context, instruction, or pattern in the prompt. For example, if one enters the prompt “How are you doing?” into the API, the Davinci model will return “I’m doing well, thank you. How are you?” as a completion. By designing appropriate prompts, the models can be used for different tasks like question answering, summarization, conversation, code generation, and more (Brown et al., 2020). The specific prompt design is critical for generating the

desired responses. Although going into the specifics of good prompt design is out of the scope of this thesis, there are a few basics that are important to understand for the reader. First, the model's response is not "GPT-3 itself". The completion under the settings described under section (4.4) will use greedy decoding, which generally speaking is choosing the token that the model estimates as the most likely continuation of the previous text until an end is most likely or the maximum number of predicted tokens is reached. This means that if the prompt includes real or fictional characters/entities, the most likely continuation of tokens will seem as if GPT-3 takes the role of one or many such characters/entities. For example, to the question "Who are you?" the Davinci model under the standard settings from section 4.4 answers "I am a 20-year-old student at the University of Utah in the United States". Similarly, conversations between people can be prompted and will be continued by the model. Secondly, it is important to note that GPT-3 does not "remember" past questions if they are not part of the prompt. It is a static model that will answer deterministically under specific settings, including the Standard Settings (section 4.4).
For the experiments, every prompt consisted of two parts: a question that is part of the dataset (described in section 4.6), and a context that surrounds the question. The entire prompt, containing both context and question, is given as input to the model. I will give an example in which the entire prompt is printed in `typewriter` font, and additionally, the context is printed in *italic* and the question in **bold**:

`You are the contestant on a game show and the next question is worth the grand prize. You must answer correctly to win.`
`Question: How many episodes, in total, does the series How I Met Your Mother have over its nine seasons?`
`Your answer:`

The resulting completion of the Davinci model would be "There are 208 episodes in total." When running an experiment, the model, request query settings, and context stay unchanged, but the question is changed in each iteration until each question in the dataset was asked once. The prompts for each experiment are found in the respective experiment chapters.

4.4. Standard Request Query

For the ’Point Estimate’ and ’Confidence Interval’ experiments, I used the following settings, if not stated differently:

```
response = openai.Completion.create(
    engine=engine,
    prompt=prompt,
    max_tokens = 100,
    temperature = 0,
    top_p = 1,
    n = 1,
    echo = False,
    frequency_penalty=0,
    presence_penalty=0)
```

As the variable engine, I used any of the models in 4.1 in their most current version (September 2022).

- ’text-ada-001’
- ’text-babbage-001’
- ’text-curie-001’
- ’text-davinci-002’

For the sake of simplicity, I will refer to the models by their respective names Ada, Babbage, Curie and Davinci and not by the "engine" variable string used in the request query. As the variable "prompt", the textual input for the model, I used the respective prompts specified in the experiment chapters. The "temperature" and "top_p" variables were always set to temperature = 0 and top_p = 1, in order to make the response of the models deterministic and the experiments replicable. The meaning of other settings can be looked up in the documentation of OpenAI at https://beta.openai.com/docs/api-reference/completions/create.

4.4.1. Point Estimate Special Settings

In the point estimate experiment, some of the contexts introduce two entities that interact with each other via text (e.g. a host and a contestant in a game show). We call these entities Entity1 and Entity2. In such cases, the standard response setting is adapted in the following way (**changes in bold**):

start_sequence = "\nEntity1:"
restart_sequence = "\nEntity2: "

```
response = openai.Completion.create(
        engine=engine,
        prompt=prompt,
        max_tokens = 100,
        temperature = 0,
        top_p = 1,
        n = 1,
        echo = False,
        frequency_penalty=0,
        presence_penalty=0,
        stop=[" Entity1:", " Entity2:"])
```

This prompts the model to generate text completions by alternating responses from Entity1 and Entity2.

4.5. Extraction of Answers

4.5.1. Extraction of Text Completion

After making a request via the OpenAI API, a response of the type <class 'openai.openai_object.OpenAIObject'> is given. For our standard request query defined in 4.4, engine = 'text-davinci-002' and prompt = 'How many episodes, in total, does the series How I Met Your Mother have over its nine seasons?', we get the following response:


```
{
    "choices": [
        {
            "finish_reason": "stop",
            "index": 0,
            "logprobs": null,
            "text": "\n\nThere are 208 episodes in total."
        }
    ],
    "created": 1661941885,
    "id": "cmpl-5lFIDhkSK1EgzVHuPmStvCRDbbYtC",
    "model": "text-davinci-002",
    "object": "text_completion",
    "usage": {
        "completion_tokens": 9,
        "prompt_tokens": 21,
        "total_tokens": 30
    }
}
```

We can extract the InstructGPTs text completion with response['choices'][0]['text'], as a string. There are many ways in which relevant values can occur in the text completion string.

- Integers, e.g. "There are 208 episodes in total."
- Decimal numbers, e.g. "One World Trade Center is 541.3 meters tall."
- Dates, e.g. "Steamboat Willie was released on November 18th, 1928."
- Word numbers, e.g. "Beethoven composed nine operas."
- Mulitple Integers, e.g. "Bayern Munich scored a total of 91 goals in the Bundesliga Season 2012/2013."
- Multiple Choice Answers, e.g. "(B) 30%"
- Percentages, e.g. "I am 80% certain that the proposed answer is correct"
- Confidence Intervals, e.g. "My confidence Interval is [5, 27]"

Those basic cases can occur in different variations. In most experiments, I proceeded in two steps. The first step was extracting every possibly relevant value from the string. Secondly, I chose the value(s) most likely to be relevant for the experiment. This second step is specific for each experiment and is explained separately in the following subchapters 4.5.2, 4.5.3, 4.5.4 and 4.5.5.

4.5.2. Point Estimate Extraction

As part of the first step of extracting values, the string is split and checked for word numbers (e.g. "five", "twelve") with a library. All word numbers are changed to integers or decimals. After that, a regular expression is used to find all values in the string. I used the re library in Python with: re.findall('\d*\.? \d+', text_completion) to find all integers and decimals and save the identified values in a list. Given the list of extracted values, it is generally difficult to decide which value is the point estimate and which ones are not. The option I chose is to always choose the value which is closest to the correct answer. If the list of extracted values was empty, the point estimate was set to null. More on the limitations and errors in this approach in section 4.5.6

Example Point Estimate Extraction:
answer string: "Bayern Munich scored a total of 91 goals in the Bundesliga Season 2012/2013."
extracted values: 91, 2012, 2013
correct answer: 98
extracted point estimate: 91

4.5.3. Multiple Choice Extraction

For experiments in which a multiple choice formatting was used, the text completion was checked for keywords specific to the available answer options. For example, if the answer option "(C) 50%" existed, the keywords "C)", and "50%" were searched. If at least one keyword for an answer option was identified in the text completion and no contradicting keyword from another answer option was identified, the respective answer was saved. In the case that contradicting keywords were identified in the text completion that belong to different answer options, the answer of the model was saved as 'false'.

Example Multiple Choice Extraction:
answer string: "C) 50%."
extracted values: "C)", "50%"
extracted MC answer: (C)

4.5.4. Free Uncertainty Estimate Extraction

For experiments in which a free uncertainty estimate was used, the given percentage was extracted in the same way as the point estimates were extracted, described in section 4.5.2. If exactly one value was identified, the value was saved. If less or more than one value was identified, null was saved.

Example Free Uncertainty Extraction:
answer string: "95%."
extracted value: 95
extracted uncertainty estimate: 95

4.5.5. Confidence Interval Extraction

Confidence intervals were returned in different styles, but in all valid cases they contained exactly two values.

- from x to y
- from x-y
- x-y
- x:y
- [x, y]
- [x - y]
- between x and y

If exactly two values were extracted, the first was set as the lower bound and the second as the upper bound. Additionally, it was tested whether the lower bound was smaller or equal to the upper bound. If less than two values were extracted, the bounds of the confidence interval were both set to null. In such cases, the confidence interval is ambiguous. If more than two values were extracted, the smallest was chosen as the lower bound and the biggest as the upper bound of the confidence interval. More on the limitations and errors in this approach in section 4.5.6

Example Free Uncertainty Extraction:
answer string: " 1227:1241."
extracted values: 1227, 1241
extracted lower bound: 1227
extracted upper bound: 1241

4.5.6. Errors in Answer Extraction

Empirically, the answer extraction and interpretation methods I used worked extremely well. However, they are not perfect. After spot checking more than a thousand of my automated extractions manually, I discovered a handful of errors where the automated extraction failed. As an example, from the model completion "Pepin the Short, also called Pepin the Younger, died in year 9 of the 9th century." the answer 0 was incorrectly extracted, instead of 909. Nevertheless, for the scope and required accuracy, the automated extraction is still the best option available and sufficiently accurate because:
1. Errors occurred in less than 1% of the spot-checked examples, which were from diverse experiments, models and prompts. Therefore, the empirical error rate is very low.
2. The effect sizes of the experiments are large enough to not be dependent on exact % accuracy.
3. Hand-checking is not a great alternative because it would be prone to some human error and would not be time-efficient.
4. Many other variables influence the results of the experiments more strongly, like prompt design and questions selected for the dataset. Therefore, I rather spent a lot of time on designing a well-balanced dataset and prompt designs.

It is important to note that errors in value extraction are not the same as what I call ambiguous answers. An error in value extraction occurs when the automated extraction does not identify an answer by the system correctly, by overlooking or misinterpreting the answer contained in the text completion. An ambiguous answer, on the other hand, is when the automated extraction identifies that the text completion of the model does not contain a valid answer - for example, if only a single value is returned when a confidence interval is demanded. Depending on the experiment and the model, the number of ambiguous answers can be very high.

4.6. Dataset

The dataset used in this experiment was created by myself to specifically suit the needs of the experiments for this thesis. It consists of 250 factual questions, each of which has a single integer as a correct answer.

Table 4.1.: Example Questions from the Dataset

question	answer
How many episodes, in total, does the series How I Met Your Mother have over its nine seasons?	208
In which year did Wu Zhao, commonly known as Wu Zetian, the first empress of the Tang dynasty, die?	705
How tall is the Abraj Al-Bait Clock Tower in Mecca in meters?	601
In which year was Charlemagne or Charles the Great crowned King of the Lombards?	774
In which year was the Bank of Saint George, the financial institution of the Republic of Genoa, founded?	1407
How many goals did Bayern Munich score in the Bundesliga Season 2011/2012?	77

The dataset is a mix of questions that I thought up myself and questions that are crawled from the internet. For every question, I checked whether a single correct answer exists and whether the wording is unequivocal. The questions are from a diverse mix of topics ranging from history, architecture, sports to pop culture.

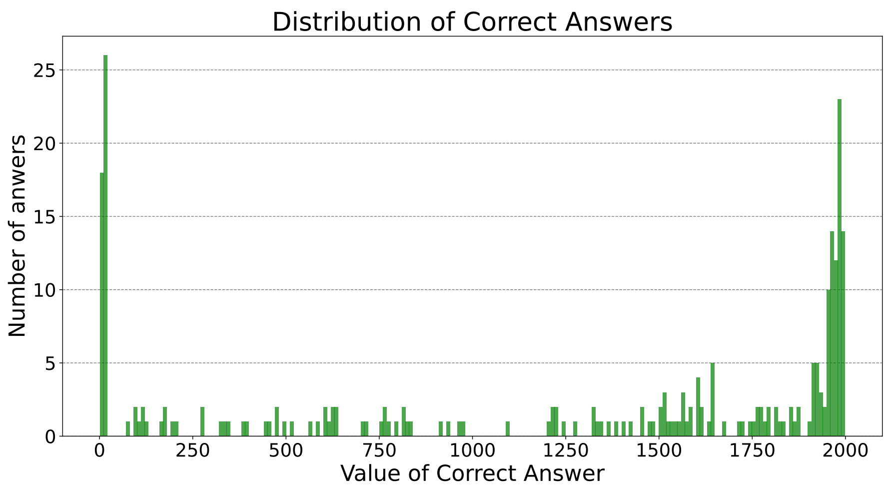

Figure 4.1.: The distribution of correct answers of the dataset is particularly clustered between the values 0 and 50 and between values of 1900 and 2000.

Every question's correct answer is an integer between 1 and 1999. I tried to create a balanced dataset, both in the area and depth of knowledge it requires to answer, as well as in the distribution of answers. However, it is much harder to think of questions whose answers are between 500 and 1500 than questions whose answers are less than 50 or between 1900 and 2000. The reason my dataset only contains 250 questions, instead of multiple thousands of questions, is mostly due to financial constraints of running experiments and the additional time it would take to thoroughly check questions for quality and running times of experiments. Choosing only questions for the dataset, for which a single, known, and unambiguously correct integer exists, has multiple objectives in mind:

Firstly, as I elaborate in section 2.5, my experiments are focused on epistemic uncertainty and not on aleatoric uncertainty.

Secondly, estimating integers makes uncertainty estimates using confidence intervals possible. This is obviously not possible for questions like, "Who was the 16th President of the United States of America?".

Thirdly, one can propose answers that are closer or farther away from the correct answer. The correct answer to the question "How many goals did Bayern Munich score in the Bundesliga Season 2011/2012?" is 77. It is very difficult to distinguish that from a proposed answer of 87, only a very well-informed soccer fan could tell them from another. If we increase the difference between the correct answer and the proposed answer to 100, however, it is possible for most soccer fans who watch the Bundesliga to figure out that 177 goals in a single season of 34 Bundesliga games is highly unlikely. If we increase the proposed answer to 1077, a person who has watched a single soccer match should be able to identify the proposed answer as rather implausible. A model evaluating the probability of such a proposed answer being true should show a clear trend. Creating analogous proposed answers for questions like "Who was the 16th President of the United States of America?" is much more difficult and tedious.

Fourthly, it is much easier to evaluate, whether a given number is correct or not. As described in section 4.5.1, values come in many forms like integers, decimals, word numbers, and dates, but are much more standardized than non-word numbers. If the correct answer to a question is 'Abraham Lincoln', does 'Abe Lincoln', 'Lincoln', 'President Lincoln', 'Uncle Abe' or 'The Great Emancipator' - all of which are common nicknames and abbreviations for Abraham Lincoln - also count as the correct answer? And how would one formalize such a list of possible answers for every question?

Lastly, very few papers have worked with answering questions with numeric answers for LLMs. Mostly, they asked trivia questions with non-numeric answers. Against this backdrop, there is novelty in asking this kind of questions.

5. Point Estimate

5.1. Motivation

As a first experiment, I generated point estimates with the InstructGPT models. A point estimate is the "best guess" of an unknown value. This is important to test if InstructGPT can understand the question and how much knowledge InstructGPT already has about the questions in the dataset. Therefore, this is crucial information for the later experiments because the model's ability to express uncertainty differs if 10% or 90% of questions answers are already known to it.

5.2. Evalutation

5.2.1. Prompts

To test the models for robustness of their predictions under different contexts, I devised 6 different prompts.

1. Standard: Short explanation that the model should return an integer as a most likely point estimate for the question.
2. Human Expert: Model is prompted as a 'World Quizzing Champion'.
3. AI Expert: Model is prompted as an all-knowing AI, that always answers with the correct number.
4. AI Assistant: Model is prompted with OpenAI's 'AI Assistant' prompt.
5. Average Human: Model is prompted as a human with limited general knowledge.
6. Just the question: No context or explanation of the task is given.

The entire "AI Expert" prompt for the first example question in Table 4.1 would look like this (`typewriter` for input, **bold** for possible model output):

```
The following is a conversation with an all-knowing AI. The AI is clever,
knowledgeable and always knows the correct answer.
The AI is asked questions by the Host. Every question has an integer as
the correct answer.
The AI answers only with the correct number.
```

`Host:` *`How many episodes, in total, does the series How I Met Your Mother have over its nine seasons?`*

AI: 208

The exact wordings and details of the six respective settings and prompts can be found in Appendix A.

Correct Answer Percentage

Percentage of questions that are correctly answered by the model. Questions answered ambiguously are counted as incorrect. If n questions are given to the model and c of them get answered correctly, then $\frac{c}{n} = 100 * \frac{c}{n}\%$ is the correct answer percentage.

Combined Best

If the model answers a specific question correctly for any of the six prompts, then this specific question is counted as correctly answered by the combined best of the model. The combined best is therefore at least as good or better than the best prompt and serves an upper limit for the model's knowledge.

Combined Worst

If the model answers a specific question correctly for all six prompts, then this specific question is counted as correctly answered by the combined worst of the model. The combined worst is therefore equal to the worst prompt or worse.

Coefficient of Variation

The coefficient of variation shows the extent of variability in relation to the mean of the population. The higher the coefficient of variation, the greater the dispersion.

$$CV = \frac{Standard\,Deviation}{Mean}$$

The coefficient of variation is useful as it is dimensionless and makes data sets with different units or widely different means comparable (JRC, 2022).

5.3. Results

5.3.1. Correct Answers

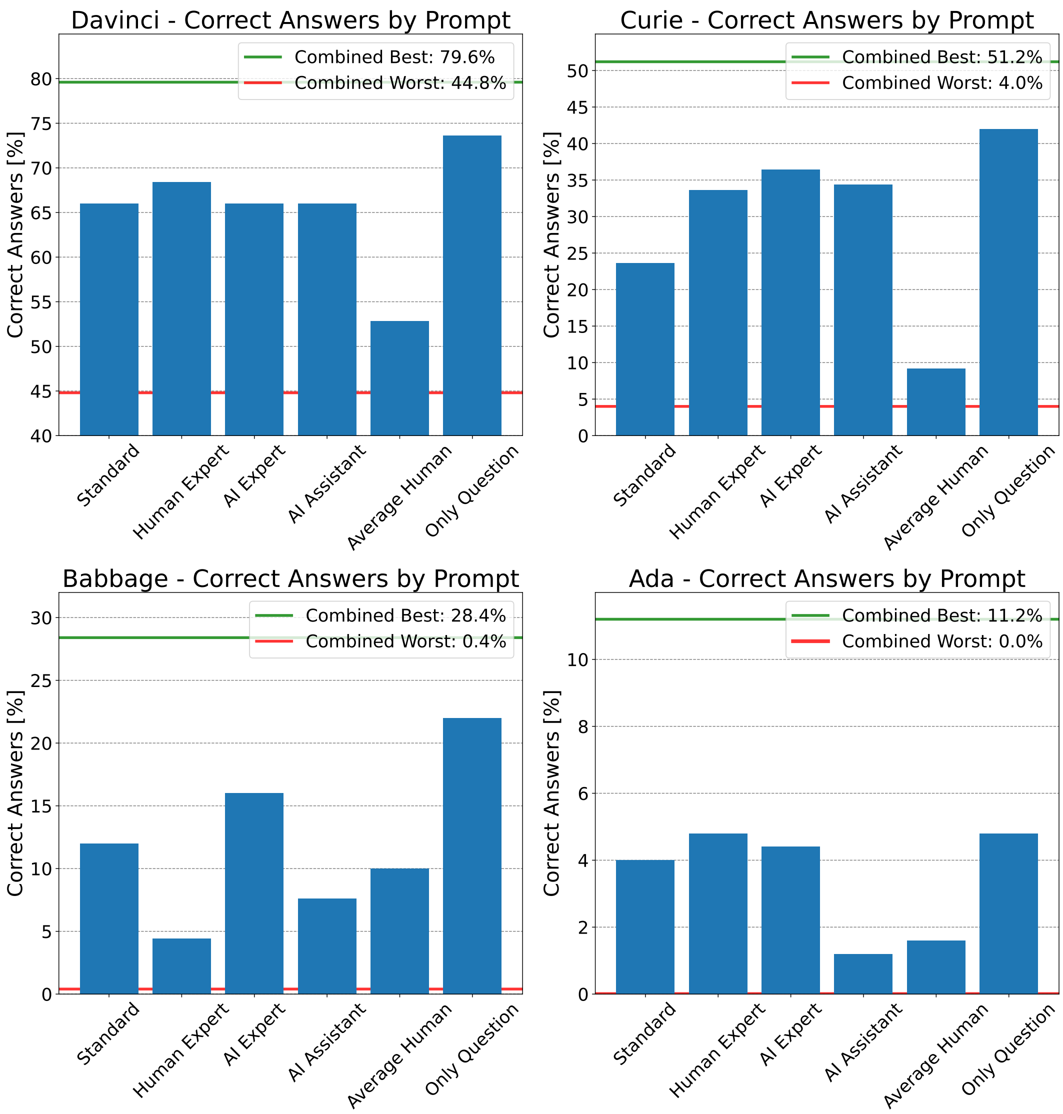

Figure 5.1.: Model size and prompt design have significant impact on Point Estimate accuracy

Five observations stand out, in particular. Firstly, the accuracy of the point estimates is significantly dependent on the context of the prompt in which the question is given to the model. In each of the six different prompt designs, the wording of the question from the dataset is identical. The given answers of the model still vary significantly. Secondly, the "Only Question" prompt is performing best and for all models at least twice to four times as accurately as the "Average Human" prompt, which is performing worst. Thirdly, the "Combined Best" accuracy significantly outperforms all prompts. Even if the model can return the correct answer, even the best prompt used cannot extract the model's best possible knowledge robustly.

	Standard Deviation	**Mean**	**Coefficient of Variation**
Davinci	0.004	0.655	0.006
Curie	0.107	0.299	0.359
Babbage	0.057	0.120	0.478
Ada	0.015	0.035	0.430

Table 5.1.: Coefficient of variation of point estimates for all models

Fourthly, the coefficient of variation, which is a measure of variability, has an inverse relationship to model size: It is bigger in smaller models and smaller in bigger models. Bigger models have more robust point estimate accuracies, as can be seen in Table 5.1. Fifthly, although Davinci's accuracy for the "Average Human" prompt is the worst compared to other prompts, an accuracy of around 53% seems beyond the level of accuracy and general knowledge of most humans. Therefore, Davinci significantly overestimates the accuracy of an average human's general knowledge.

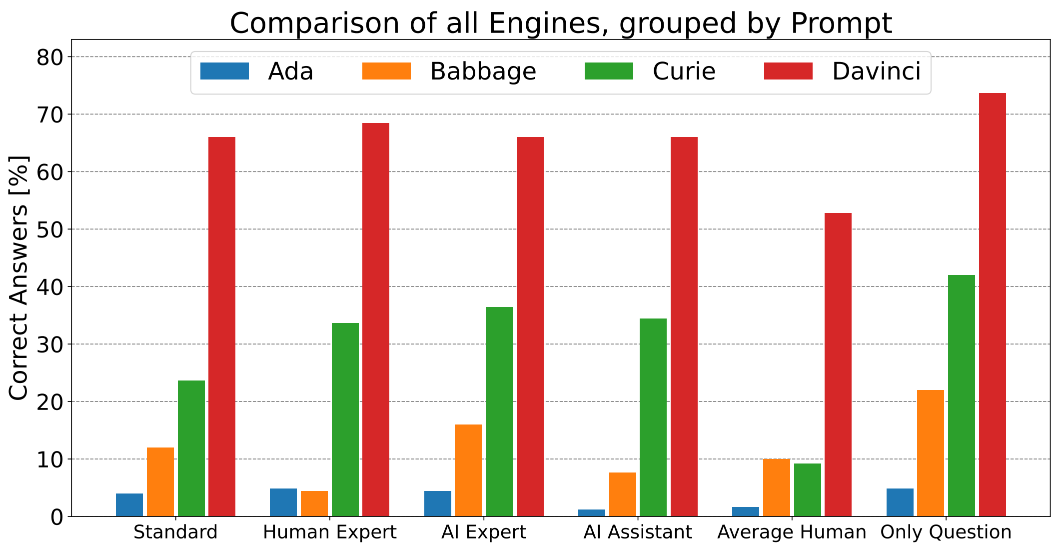

Figure 5.2.: Model size greatly improves point estimate accuracy

Comparing the models point estimates directly to each other makes two further observations easily visible. Firstly, model size increases the accuracy of point estimates enormously. The respective mean over all prompts and questions for the models Ada, Babbage, Curie, Davinci are rounded to 3%, 12%, 30% and 65% respectively. Secondly, in two cases can a smaller model outperform a bigger model in one specific prompt.

Namely, Ada outperforms Babbage's accuracy for the "Human Expert" prompt and Babbage outperforms Curie's accuracy for the"Average Human" prompt.

5.3.2. Distribution of Errors

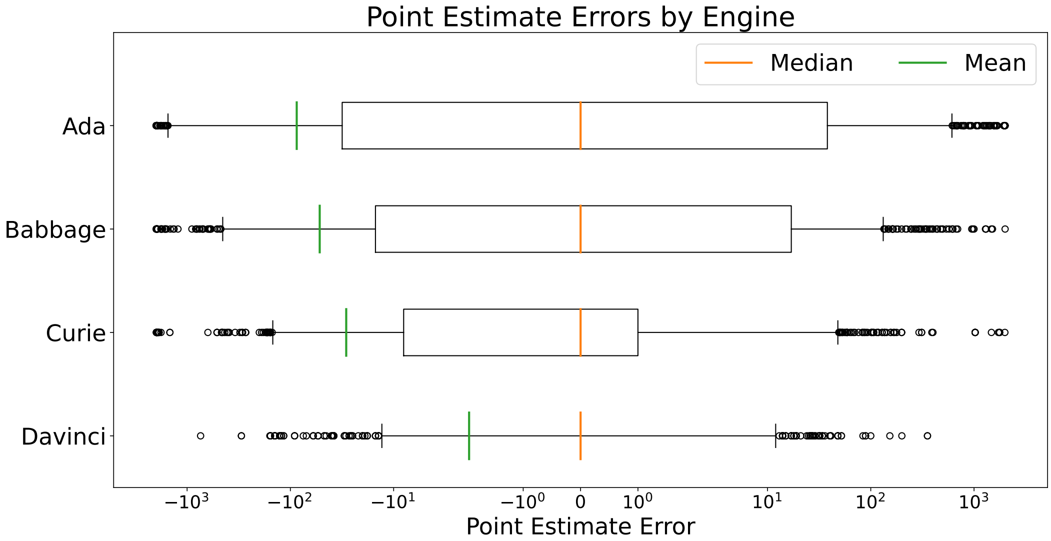

Figure 5.3.: Point Estimate Errors are smaller for bigger models

In Figure 5.3, the model responses for all six prompts were combined. Ambiguous responses were not counted because they do not have a valid point estimate error. Two observations about the point estimate prediction errors stand out. Firstly, bigger model size not only improves the percentage of correct answers as seen in Figure 5.2, but also minimizes the size of prediction errors for false answers. It is also observed that the interquartile range, the 5th and 95th percentiles (whiskers), and the extreme values are closer to the true value when the model size is bigger. Secondly, while the median is always 0, the mean is negative for all models. This means that there is a bias in the point estimates for all models, which are on average too low. The bias is bigger for smaller models. Overall, the average share of ambiguous answers for the Davinci, Curie, Babbage, and Ada model is 1.9%, 13.8%, 18.2%, and 19.2% respectively.

6. Uncertainty Estimates

6.1. Motivation

In this first set of experiments, we ask the models to estimate how likely a proposed answer is true. In every experiment, the model makes estimates for four different proposed answers separately. The four proposed answers are the correct answer $n_{correct}$, the correct answer + 10, $n_{10} = n_{correct} + 10$, the correct answer + 100, $n_{100} = n_{correct} + 100$ and the correct answer + 1000, $n_{1000} = n_{correct} + 1000$. We have previously evaluated each model's accuracy in the 'Point Estimate' Experiments from section 5. From a calibrated model, we would expect that the estimated probability that $n_{correct}$ is, in fact, correct reflects the knowledge of the respective model. So, Davinci should be more certain that $n_{correct}$ is correct than Ada. And we would expect, that the bigger the difference between the proposed answer and $n_{correct}$ is, the less likely each model evaluates that the proposed answer is true because the models do not only get the answer right sometimes but also made point estimates, that scatter around the correct answer, as seen in Figure 5.3. Each experiment is increasingly demanding, as I start out with a simple 'True or False' multiple choice question, and end with an open question about the uncertainty estimate of the model. To test the models for robustness of their predictions, I changed the order of the returned answers or did experiments that are similar to each other.

6.2. Evaluation

6.2.1. Prompts

For all experiments, each model answered every question in the dataset for four different proposed answers. In order to make my results comparable to previous research, I used the same multiple choice formatting as used by Rae et al. (2021) and Kadavath et al. (2022).

The entire prompts for the first example question in Table 4.1 would look like this (`typewriter` for entire prompt, *italic* for variable question and **bold** for variable proposed answer):

True or False

```
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Proposed Answer: 208
Is the proposed answer:
  (A) True
  (B) False
The proposed answer is:
```

To test for robustness, I also evaluated the same 'True or False' prompt with 'True' and 'False' interchanged

```
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Proposed Answer: 208
Is the proposed answer:
  (A) False
  (B) True
The proposed answer is:
```

True or False or I don't know

```
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Proposed Answer: 208
Is the proposed answer:
  (A) True
  (B) False
  (C) I don't know
The proposed answer is:
```

Multiple Choice Uncertainty Estimates

To increase the likelihood that the model will return one of the possible multiple choice answers, I included the "(" in the prompt.

```
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Proposed Answer: 208
How likely is it that the proposed answer is true?
   (A) 10%
   (B) 30%
   (C) 50%
   (D) 70%
   (E) 90%
The likelihood that the proposed answer is correct is (
```

To test for robustness, I also evaluated different percentages as multiple choice answers.

```
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Proposed Answer: 208
How likely is it that the proposed answer is true?
   (A) 20%
   (B) 40%
   (C) 60%
   (D) 80%
The likelihood that the proposed answer is correct is (
```

Free Uncertainty Estimates

```
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Proposed Answer: 208
How likely is it that the proposed answer is true?
The likelihood in percent that the proposed answer is true is
```

6.3. Results

6.3.1. True or False

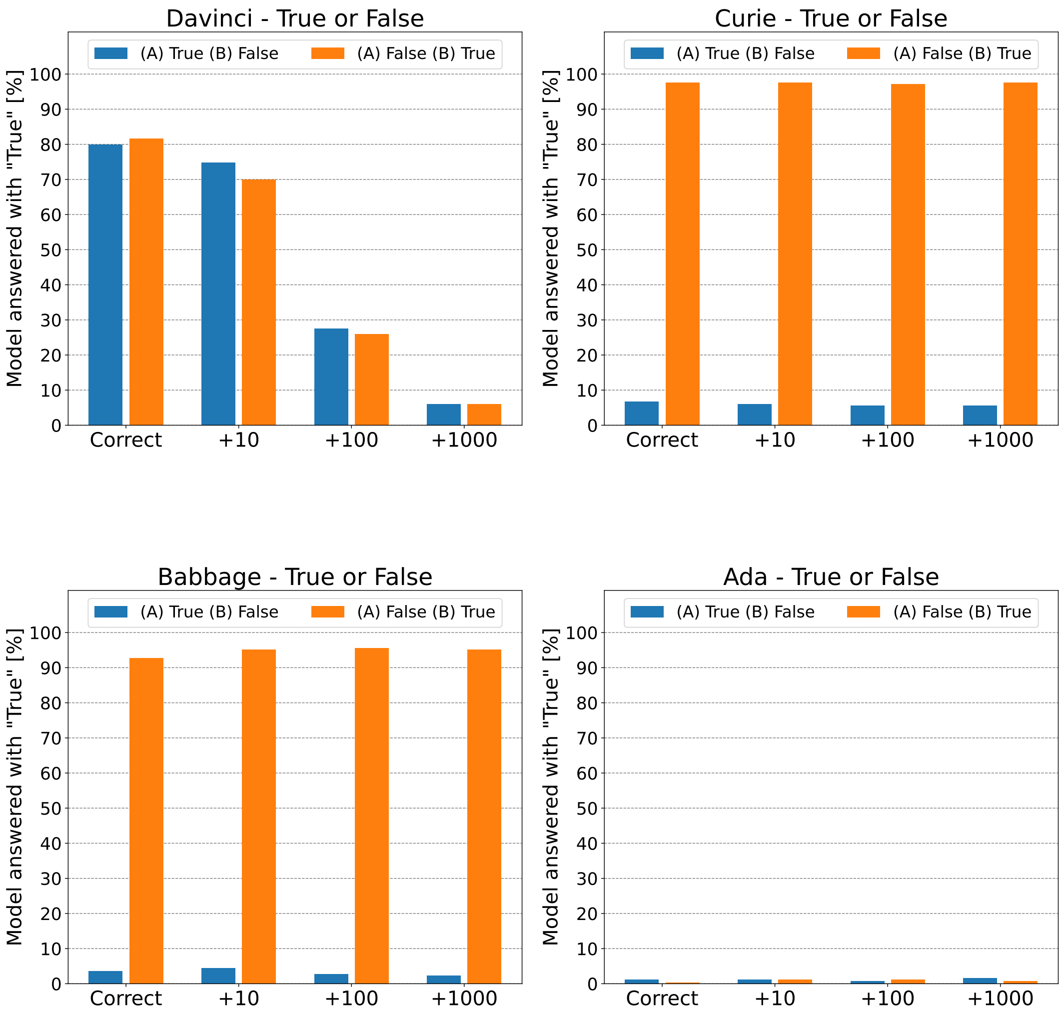

Figure 6.1.: Models answer to True or False questions. Davinci is roughly calibrated, but overestimates the chance of a proposed answer being true. Curie and Babbage are both not calibrated and not robust. They almost always choose the most recent Multiple Choice option. Ada is also not calibrated but consistent in choosing "False" as the answer in around 99% of the cases

The Davinci Model shows robustness for interchanging the answer options 'True' and 'False'. However, there is an average 3.3% increase in probability to answer 'True' for the order (A) False (B) True. The probability of identifying the correct answer as true is above 80%, which is higher than any prompt in the 'Point Estimate' experiment in section (5.2) as well as the combined best values of 79.6% for the Davinci model. There is a clear trend of identifying an incorrect proposed answer as incorrect when the proposed answer is worse. While only 25% - 30% of proposed answers that are +10 of the correct answe, are labeled as incorrect, for proposed answers that are +1000 away from the correct answer, more than 90% are identified as false. Overall, the Davinci model shows calibration because the more "off" a proposed answer is, the more likely Davinci is to identify it as false. However, the Davinci model has a bias toward labeling proposed answers as true. The Curie and Babbage model show an almost identical pattern. They both answer with false for the (A) True (B) False prompt and True for the (A) False (B) True prompt more than 90% of the time - independently of whether the answer is correct, slightly wrong, or entirely wrong. Therefore, they show no robustness for switching answers and show no difference in identifying proposed answers as true or false, depending on whether they are true or false. The Ada Model always returns 'False' as an answer, regardless of the correctness of the proposed answer, or the order of the choices. In that way, it is more "robust" than the Curie and Babbage model, but its answers are still not even rudimentary calibrated. The share of ambiguous answers for the Davinci, Curie, Babbage, and Ada model are 0.7%, 0.1%, 1.3%, and 1.9% respectively.

6.3.2. True or False or I don't know

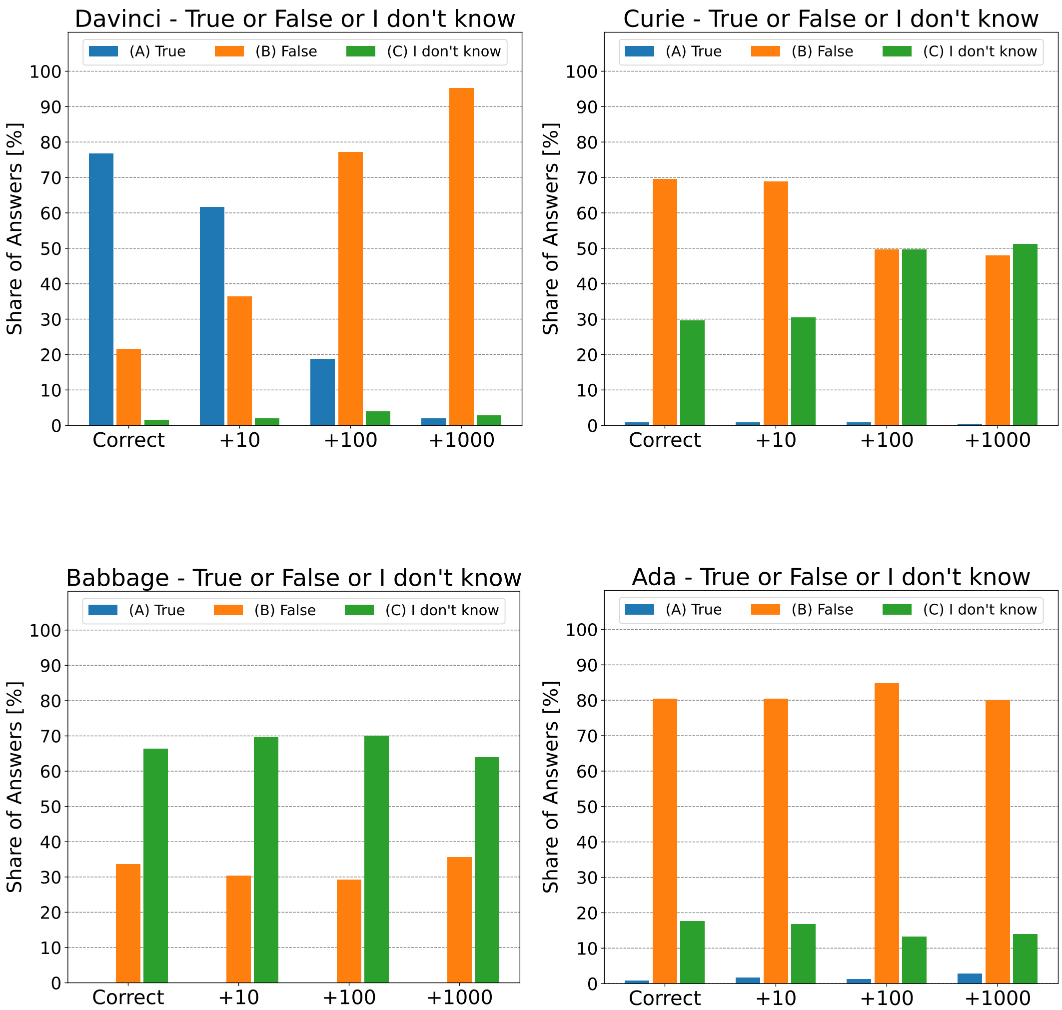

Figure 6.2.: Models answers to True or False or I don't know Multiple Choice question. Davinci is roughly calibrated, but overconfident in its answers. Curie, Babbage and Ada are all not calibrated. Babbage, to a lesser degree Curie and to an even lesser degree Ada acknowledge their lack of knowledge. All three smaller models answer with "True" less than 2% on average.

The Davinci model identifies 75% of correct answers as true and identifies 95% of proposed answers that differ by 1000 from the correct answer as false. Overall, it is calibrated because the more the proposed answer differs from the actual correct answer, the lesser Davinci estimates the answer to be correct. However, the Davinci model is overconfident in its ability to differentiate correct from incorrect answers, and only answers with "I don't know" in 2.6% of cases. The Curie, Babbage, and Ada Model are much more willing to answer with "I don't know" with average probabilities of 40%, 68%, and 15% respectively. All three models hardly ever respond with (A) True with 0.7%, 0.0%, and 1.6% respectively. The Curie Model is much more likely to select 'False' if the proposed answer is correct or off by 10, than if a proposed answer is off by 100 or 1000 and the Ada model selects 'True' more often when the proposed answer differs more from the correct answer, with probabilities for 'True' of 0.8%, 1.6%, 1.2%, and 2.8%, respectively. This is exactly inverse to the relationship we would expect from a calibrated model. The Babbage and Ada models' responses are constant, irrespective of the proposed answer. Therefore, they too are not calibrated. The share of ambiguous answers for the Davinci, Curie, Babbage, and Ada model are 0.0%, 0.1%, 0.3%, and 1.6% respectively, which is a similar but lower share than for the 'True or False' experiment.

6.3.3. Multiple Choice Uncertainty Estimates

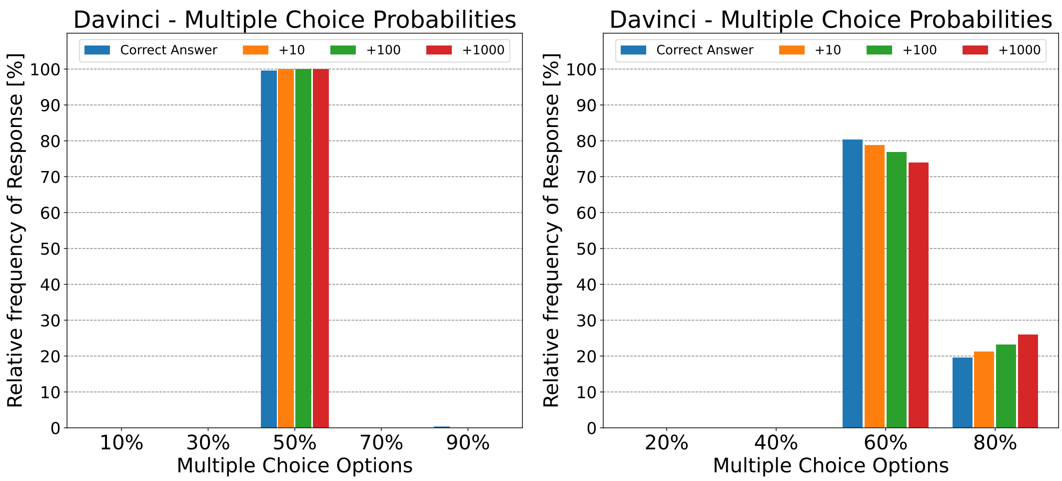

Figure 6.3.: The Davinci model is not calibrated, its estimates are relatively static for different proposed answers and inconsistent between the prompts

The Davinci model is not calibrated and uncertainty estimates are either static, irrespective of whether the proposed answer is correct or incorrect, or inversely to what one would expect. Furthermore, Davinci is inconsistent with itself, as it almost always assigns a 50% chance for the first prompt and a mix between 60% and 80% for the second prompt. The Davinci model does not return any ambiguous answers for this experiment.

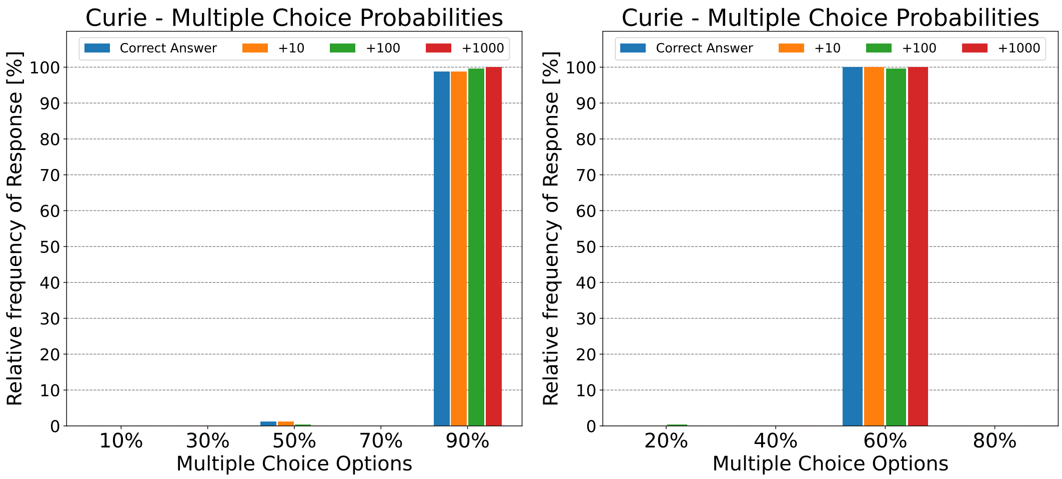

Figure 6.4.: The Curie model is not calibrated, its estimates are static for different proposed answers and inconsistent between the prompts

The Curie model is not calibrated and answers completely static. Furthermore, Curie is inconsistent with itself, as it almost always assigns a 90% chance for the first prompt and 60% for the second prompt. The Curie model does not return any ambiguous answers for this experiment.

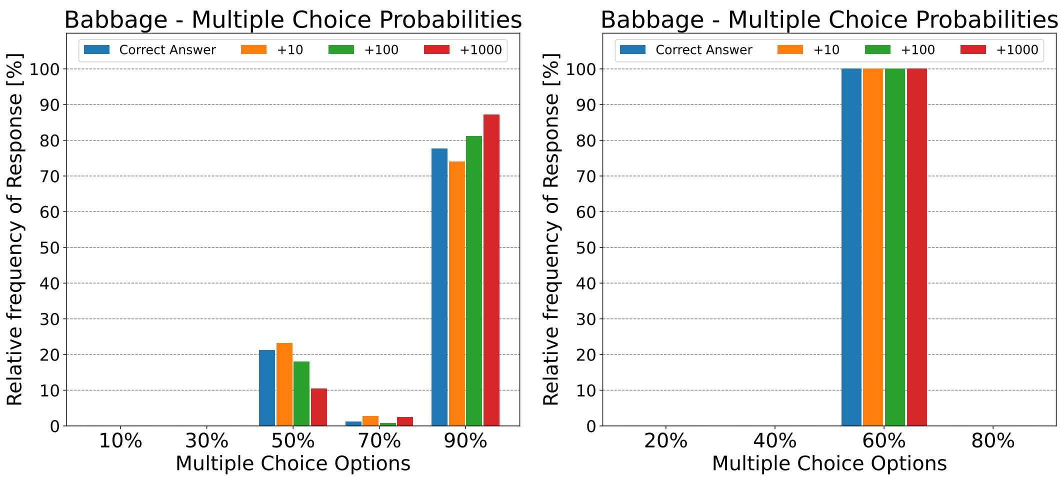

Figure 6.5.: The Babbage model is not calibrated and inconsistent between the prompts. Its estimates show some variability for the first prompt but are completely static for the second prompt

The Babbage model is not calibrated and answers inversely to what one would expect for the first prompt, assigning 90% most often for proposed answers that are off by 1000. For the second prompt, Babbage always selects 60%, which is inconsistent with its answer for the first prompt. Although there is some variability in the answers for the first prompt, overall, the chosen answers are quite static and do not reflect adequate change with different proposed answers. The Babbage model does not return any ambiguous answers for this experiment.

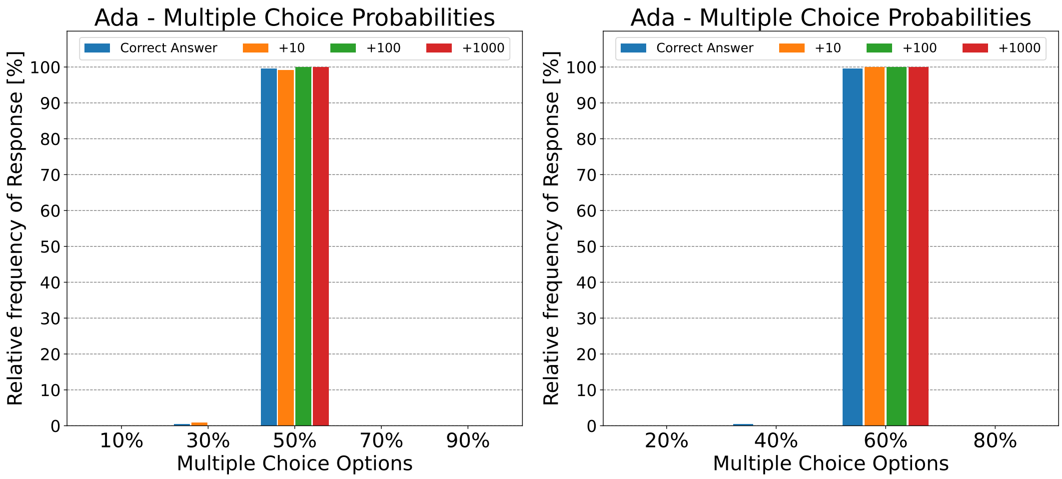

Figure 6.6.: The Ada model is not calibrated, static for each prompt and inconsistent between the prompts

The Ada model is not calibrated and answers identically, irrespective of whether the proposed answer is correct or incorrect. Furthermore, Ada is inconsistent with itself, as it almost always assigns a 90% chance for the first prompt and 60% for the second prompt. The share of ambiguous answers for the Ada model is 10.9% and 4.9% for the two different prompts respectively.

Overall, the models are not calibrated for multiple choice uncertainty estimates. The proposed answers do not have any, or an inverse effect, of how calibrated models would respond to answers with varying plausibility. For the first prompt, the models selected (C) 50% more than half the time, and for the second prompt, selected (C) 60% around 90% on average over all models. This suggests that none of the models understood the assigned task in a zero-shot setting.

6.3.4. Free Uncertainty Estimates

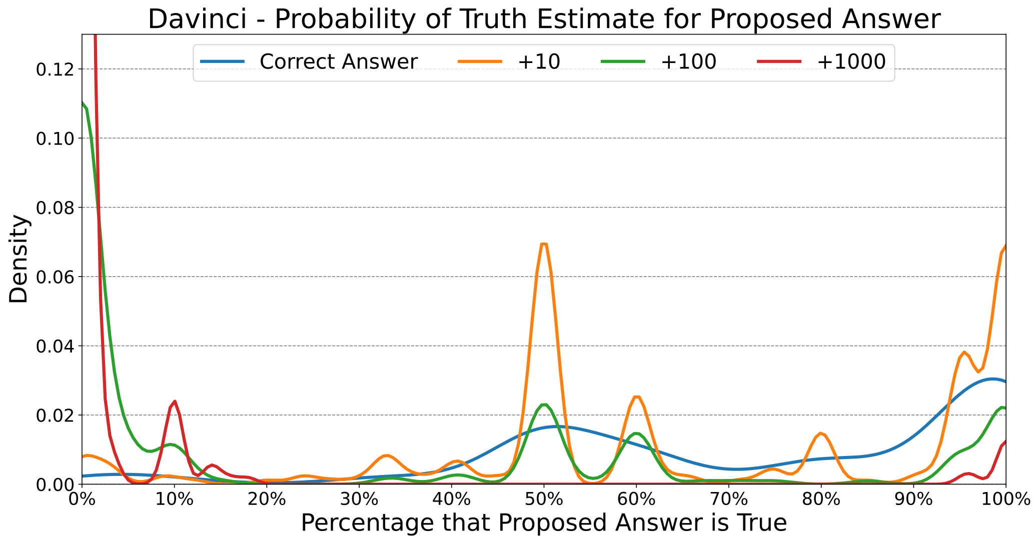

Figure 6.7.: Davinci shows a weak but recognizable calibration for uncertainty estimates. Overall, answers are variable, and the distribution is most dense in the extremes. Proposed answers that are correct or differ by 10 mostly have an estimated likelihood of 50% and 100% being correct, while proposed answers that differ by 100 or 1000 mostly have an estimated likelihood of 10% and 0% being correct. 4.7% of answers are ambiguous.

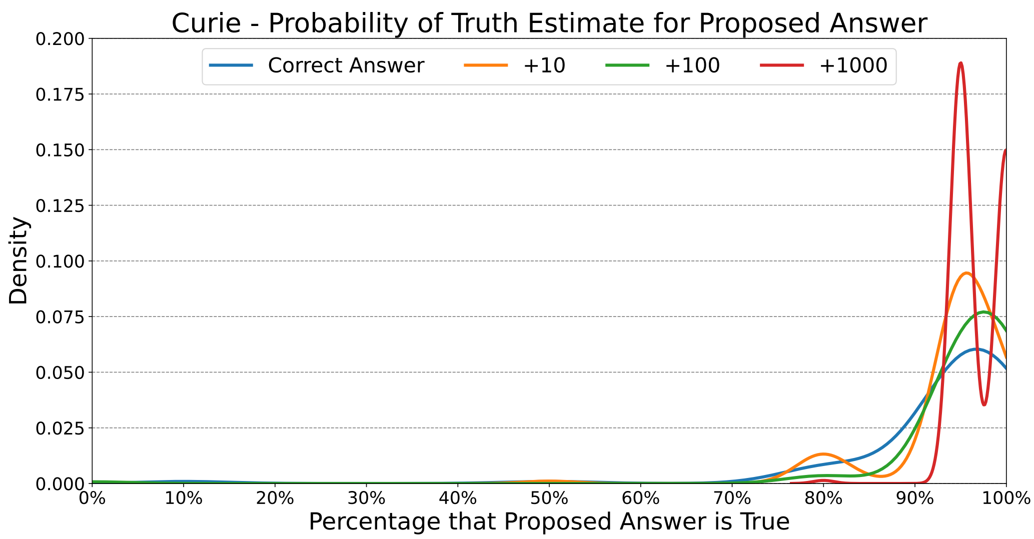

Figure 6.8.: Curie is not calibrated for free uncertainty estimates. Most probabilities are between 90% and 100%, irrespective of the proposed answer. 0.5% of answers are ambiguous.

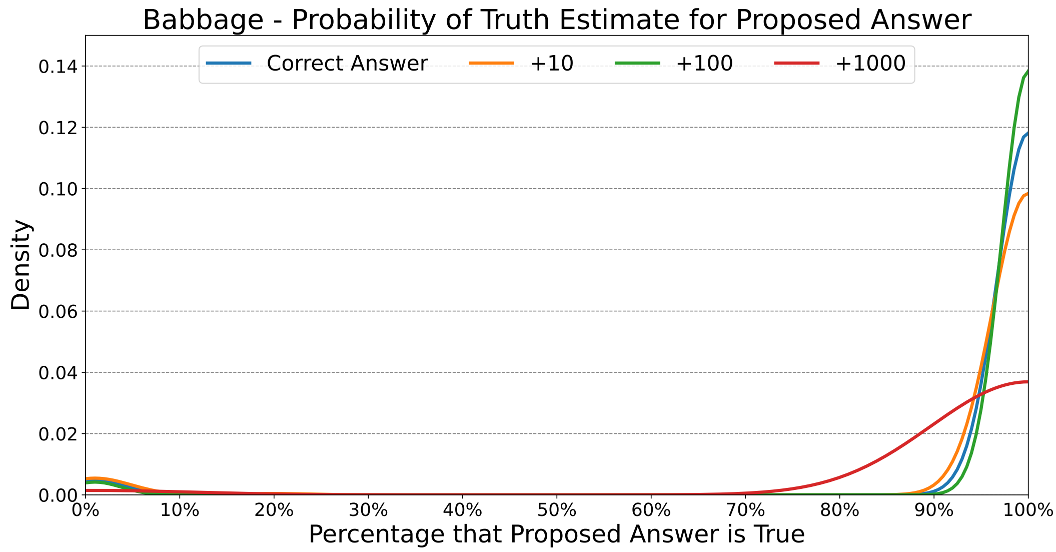

Figure 6.9.: Babbage is not calibrated for free uncertainty estimates. Most probabilities are between 95% and 100%, except if the proposed answers differs by 1000 from the correct answer. In very few cases, Babbage estimates 0%. 6.3% of answers are ambiguous.

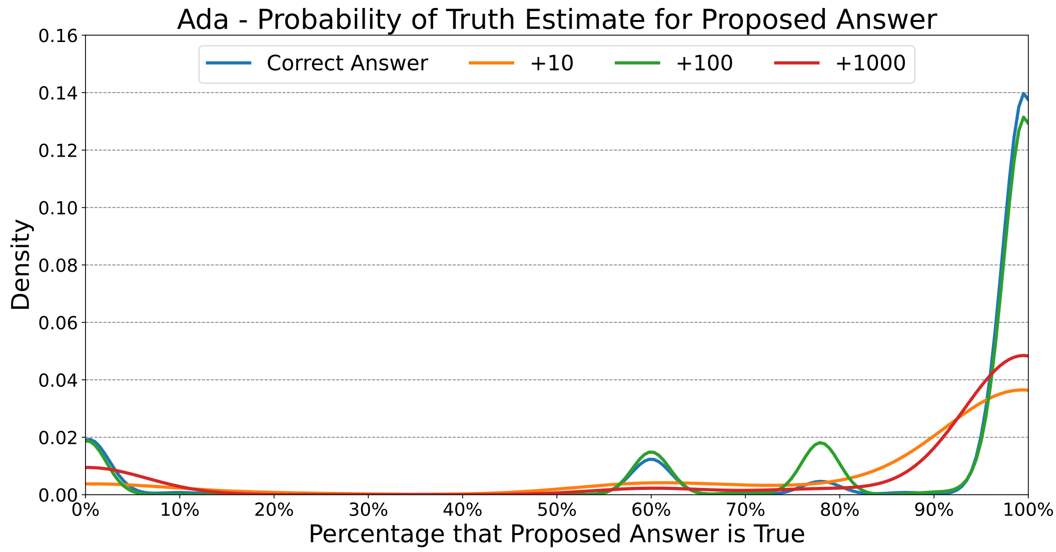

Figure 6.10.: Ada is not calibrated for free uncertainty estimates because the model always estimates a high probability for the proposed answers being true. The variability of its answers are higher than for the Babbage and Curie model, but lower than the Davinci model. The results for the correct answer and +100, and the pair of +10 and +1000 are similar. 4.5% of answers are ambiguous.

Overall, the probability distributions are most dense between 95% and 100%. While the uncertainty estimates are variable for the Davinci model, the other models only return a very limited number of values including 0%, 60% and values around 80% and above. This tendency to return few different values is similar to the results of the 'Multiple Choice Uncertainty Estimates' experiment, where the models also return very few different values, including the Davinci model. However, for the free uncertainty estimates, the returned probabilities are mostly between 95% and 100%. On the other hand, for the multiple choice uncertainty estimates, the majority split between 50%, 60%, and 90% - with 60% being the most common. Therefore, while the models tend to return the same uncertainty estimates regardless of whether the proposed answer is plausible or not in both experiments, the extent of reported uncertainty differs across experiments.

7. Confidence Interval

A confidence interval (CI) is a range of estimates for an unknown value. It is used in frequentist statistics to express uncertainty in a standardized way. A well-calibrated X% CI should contain the unknown value X% of the time in expectation.

7.1. Motivation

Confidence intervals are a common way to express uncertainty and should be well represented in the training data of InstructGPT. If a model has a good understanding of the epistemic uncertainty of its own knowledge, it should be able to return well-calibrated CIs. Calibrated confidence intervals should both increase in accuracy and size with increasing X%.

7.2. Evalutation

For the 'Confidence Interval' Experiment, all four models are prompted to return an X% CI for every question in the dataset. Although the most used CIs in statistics are usually 90% or 95% CIs, it is common in forecasting and calibration training to use a wide variety of CI percentages. In this experiment, the models are asked for their 0% all the way to their 100% CIs. In total 4 (models) * 3 (prompts) * 11 (X%) * 250 (questions) = 33000 different prompts were evaluated.

7.2.1. Prompts

For the 'Confidence Interval' experiment, I devised 3 different prompts of different lengths.

1. **Short**: Very short instructions to give a confidence interval
2. **Medium**: Wikipedia (2022) definition of confidence interval with small changes and short instructions.
3. **Long**: Open Philanthropy Project Calibrate your Judgement (2018) definition of confidence intervals with small changes. Instructions are already included in the definition.

Every prompt consists of three parts: a question from the dataset in section (4.6), a probability X%, and a context that surrounds the question. The entire prompt, containing question, probability, and context, is given as input to the model. The wording of the prompts can be found in Appendix B.

Presumed Knowledge

In the 'Point Estimate' experiment, I introduced the "combined best" (5.2.1) as an upper bound of the model's knowledge, where at least for a single of the six prompts the model answered a specific question correctly. As a lower bound, I will use the mean accuracy over all six prompts. The mean point estimate accuracy is found in Table 5.1. When I refer to a model's knowledge about the dataset, I mean this definition of presumed knowledge. Of course, the question of what a transformer model "knows" is very difficult to answer, but I believe that for the purpose and scope of this thesis, my definition of (presumed) knowledge is reasonable and sufficient to provide a reference for the accuracy of the confidence intervals.

Ambiguous answers

Whenever a model's completion text contains less than two values that can get extracted, this answer is counted a ambiguous. Either, the model's answer does not contain two answers, or in very rare cases, the extraction function is unable to recognize the values. In the plots, I will include the ambiguous answers by extending the accuracy bar to a lighter color.

Scoring Rules and Calibration

Scoring rules are cost or loss functions for evaluating predictions. So-called proper scoring rules are widely used, and aiming to minimize these loss functions leads to more calibrated uncertainty estimates. Additional to the plots, they can be a simple but precise summary of a model's overall calibration (Gneiting and Raftery, 2007) and (Carvalho, 2016). As a scoring rule, I will use the Brier score.

7.2.2. Brier Score

The Brier score by Brier et al. (1950) is a popular metric for calibration and measures the mean squared prediction error. Therefore, the lower the Brier score, the better the prediction. Brier scores range between 0 (perfect score) and 1 (worst score).

$$Brier\,Score\,(BS) = \frac{1}{N}\sum_{i=1}^{N}(f_i - o_i)^2$$

f_i being the forecasted probability (e.g. % of Confidence Interval), and o_i whether the forecast is true (e.g. 1 if the confidence interval contains the correct answer, 0 if it does not).

Examples:
A model makes predictions for different confidence intervals. The confidence interval either contains the correct answer or it does not. For the 90% confidence interval, the model returns a confidence interval of [50, 2000], the correct answer is 1987 and therefore, the Brier score is $(f_t - o_t) = (0.9 - 1.0)^2 = 0.01$ an excellent score.

For the 50% confidence interval, the model returns a confidence interval of [1962, 1987], the correct answer is 1992 and therefore, the Brier score is $(f_t - o_t) = (0.5 - 0.0)^2 = 0.25$.

Note that although 0.0 is the perfect score, a perfectly calibrated model does not have a Brier score of 0.0 in most cases. If a perfectly calibrated model returns perfectly calibrated 80% confidence intervals that contain the true answer exactly 80% of the time, the Brier score is not 0.0.

For our dataset, 200 out of the 250 80% Confidence Intervals would contain the true answers, and 50 out of the 250 80% CIs would not contain the true answer. This would result in a Brier score of

$$\frac{1}{N}\sum_{i=1}^{N}(f_i - o_i)^2 = \frac{1}{250}(\sum_{i=1}^{200}(0.8-1)^2 + \sum_{i=201}^{250}(0.8-0)^2) = \frac{1}{250}(200 * 0.04 + 50 * 0.64) = 0.16$$

To provide reference values to compare the Brier scores of the Instruct models to, we look at four examples with different degrees of calibration.

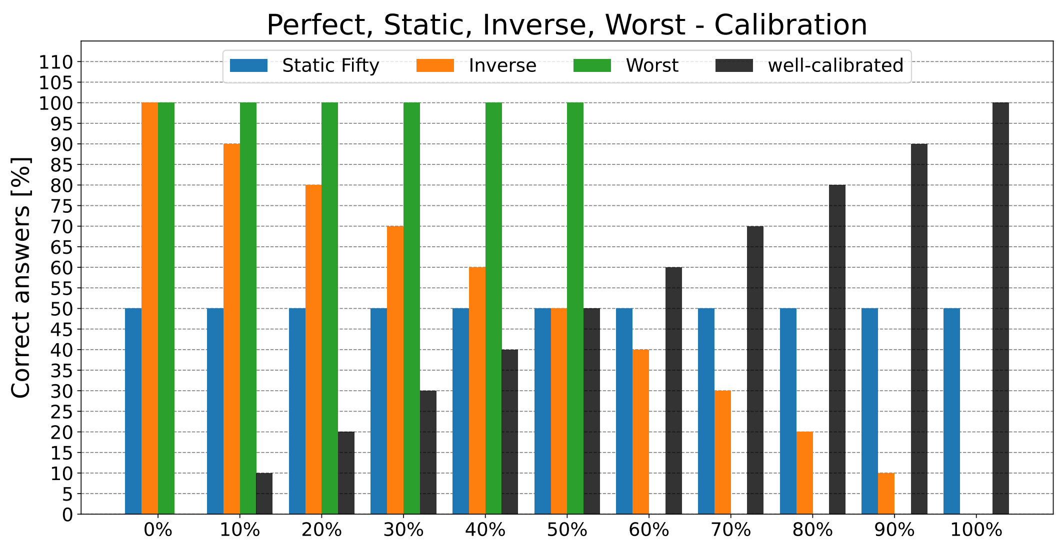

Figure 7.1.: Well-Calibrated, Static, Inverse and Worst Calibration

	Static Fifty	Inverse	Worst	Perfect
Average Brier Score from 0% - 100%	0.35	0.55	0.623	0.15
Average Brier Score from 10% - 90%	0.317	0.45	0.539	0.183

Table 7.1.: Brier Scores of Perfect, Static, Inverse and Worst Calibration

As you can see in Figure 7.1 the static, inverse and worst accuracies are in no way calibrated and do not match the expected accuracy. In our specific experiment, the Brier scores can broadly be classified like this:

- < 0.15: Impossible
- 0.15 - 0.20: Great calibration
- 0.20- 0.25: Good calibration
- 0.25 - 0.30: Medium calibration
- 0.30 - 0.35: Bad calibration
- > 0.35: Awful calibration

7.3. Results

7.3.1. Davinci

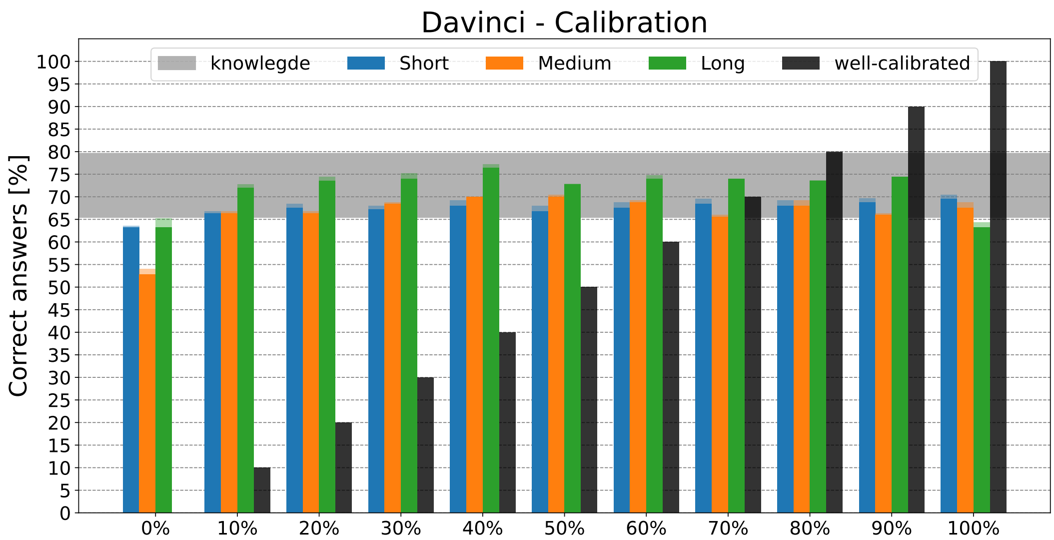

Figure 7.2.: Confidence intervals of Davinci are not calibrated

The accuracy of the confidence intervals for all three prompts are approximately constant, with slightly worse accuracy for the 0% and 100% confidence intervals. Therefore, it is clear that the Davinci Model is not calibrated for any of the three prompts used in the experiment. This is further supported by Table 7.2, where we can see that all three prompts are equally badly calibrated and broadly fall into the "Bad Calibration" category with Brier scores above 0.30. The accuracy of all prompts for all confidence intervals is similar to the presumed "knowledge" of the Davinci model. Although the Davinci model's point estimate errors are often small as seen in section 5.3, none of the confidence intervals outperforms the combined best of all point estimate prompts. Only 0.7% of answers by the Davinci model are ambiguous. The Davinci model returns very similar results and very few ambiguous answers over the three prompts, which makes the model robust under different prompts and returning valid answers. However, the returned confidence intervals are not calibrated according to the given percentages and therefore, fail to meet the demands of the task.

	Short	Medium	Long	Perfect
Average Brier score from 0% - 100%	0.342	0.337	0.349	0.150
Average Brier score from 10% - 90%	0.314	0.318	0.315	0.183

Table 7.2.: Brier scores of Davinci's CI's for short, medium and long Prompt

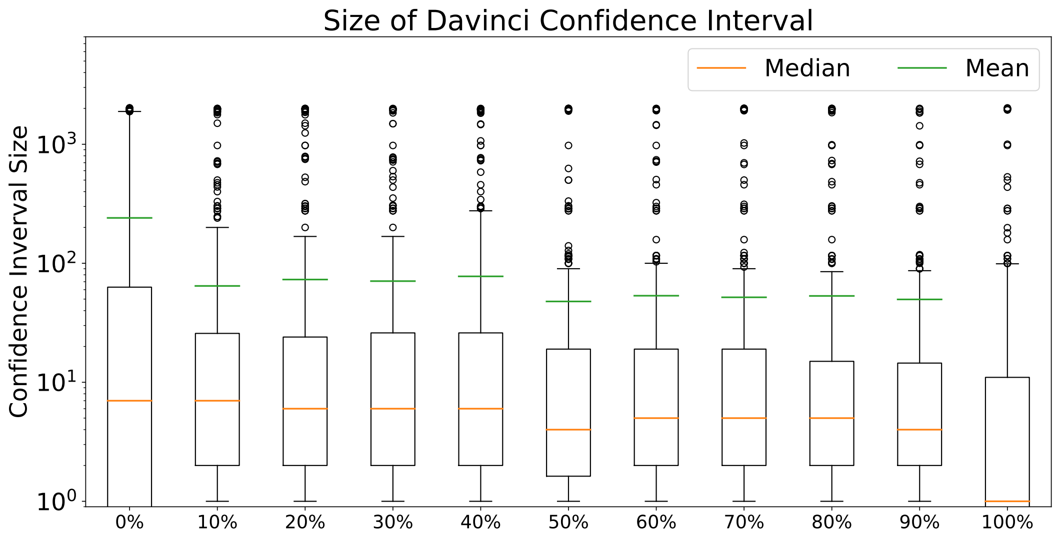

Figure 7.3.: Confidence interval sizes of Davinci are steady

Having a closer look at the average sizes of different X% confidence intervals, we can generally see that there are only small differences in Median, Mean and general distribution between the 10% and 90% confidence intervals. Higher X% have even slightly smaller confidence interval sizes, the opposite of what would be expected for calibrated confidence intervals. Most confidence intervals have sizes bigger than 1 and smaller than 100 and the biggest confidence interval sizes are around 2000. The 0% and 100% confidence intervals are somewhat exceptions, in that the 0%

confidence interval has more very small and very large confidence interval sizes. The median size of the 100% CI is equal to 1 and the smallest of all the percentages. The 100% confidence interval mean size, on the other hand, is equal to around 134818 because the mean is dominated by a single confidence interval size of 10^8. For the long prompt, the Davinci model answered the question: "How many Gold Rings are in the 'Twelve Days of Christmas'?" with the confidence interval: "0:100000000". Which is a great confidence interval, to contain the answer almost 100% of the time.

7.3.2. Curie

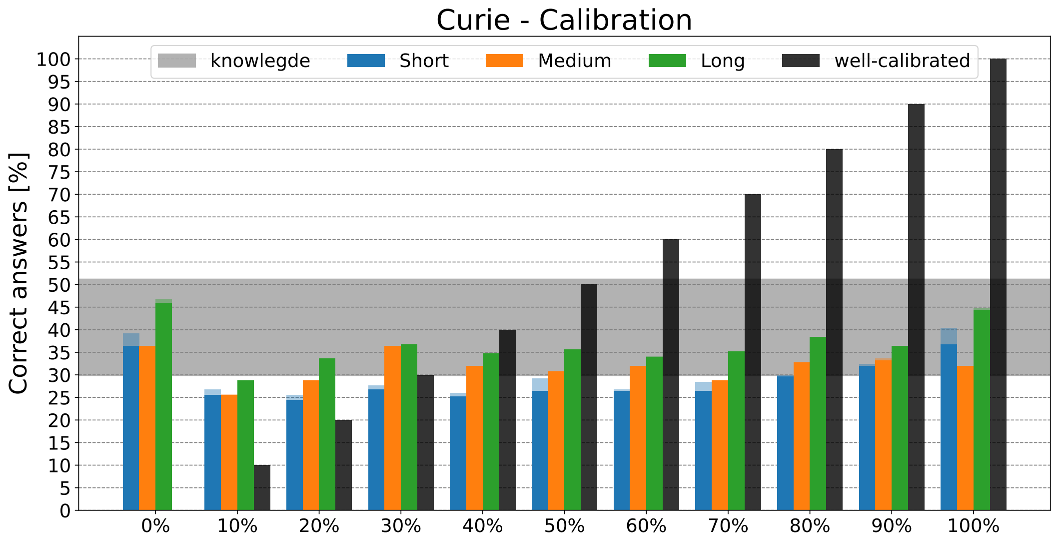

Figure 7.4.: Confidence Intervals of Curie are not calibrated

Similar to the Davinci model, the accuracy of the Curie model is approximately constant for all three prompts. On average, the 0% confidence interval has the highest accuracy, and the 10% Confidence Interval has the lowest. Therefore, it is clear that the Curie model is not calibrated for any of the three prompts used in the experiment. This is also supported by the Brier scores in Table 7.3 where we can see that all three prompts are about equally badly calibrated and broadly fall into the "Bad calibration" category, with a Brier score over 0.30. The % of confidence intervals containing the correct answer is lower or similar to the presumed "knowledge" of the Curie model. Although the Curie model's point estimate errors are mostly smaller than 100 (5.3), none of the confidence intervals outperforms the combined best of all point estimate prompts. On average, the confidence intervals from the long prompt contain the correct answer most often. The percentage of ambiguous answers of the Curie model is very low, with only 0.5% of answers being ambiguous. Like the Davinci model, the Curie model returns similar accuracies for different X%, is robust under different prompts, and returns valid answers. However, the returned CIs are not calibrated according to the given percentages and therefore, fail to meet the demands of the task.

	Short	Medium	Long	Perfect
Average Brier score from 0% - 100%	0.342	0.349	0.344	0.150
Average Brier score from 10% - 90%	0.307	0.311	0.308	0.183

Table 7.3.: Brier scores of Curie's CI's for short, medium and long prompt

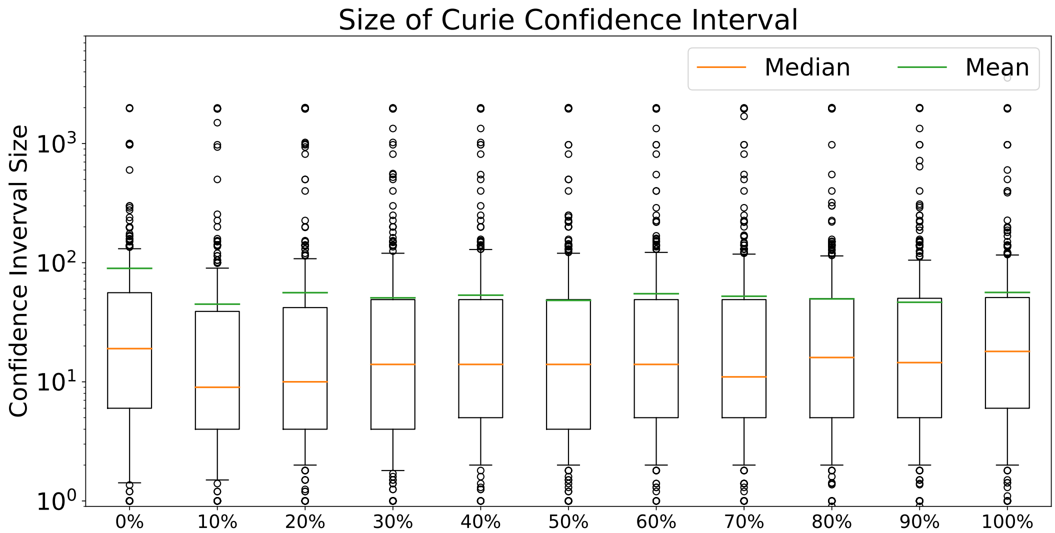

Figure 7.5.: Confidence interval sizes of Curie are steady

Having a closer look at the average sizes of different X% confidence intervals, we can generally see that for all percentages there are only small differences in Median, Mean and general distribution of the confidence intervals. The median confidence interval size is on average 14 and more than double the size of the average Davinci median confidence interval size. Most confidence intervals have sizes bigger than 1 and smaller than 100 and the biggest confidence interval sizes are around 2000. Overall, the sizes of Curie confidence intervals are similar but more homogeneous and narrower than the Davinci CI sizes.

7.3.3. Babbage

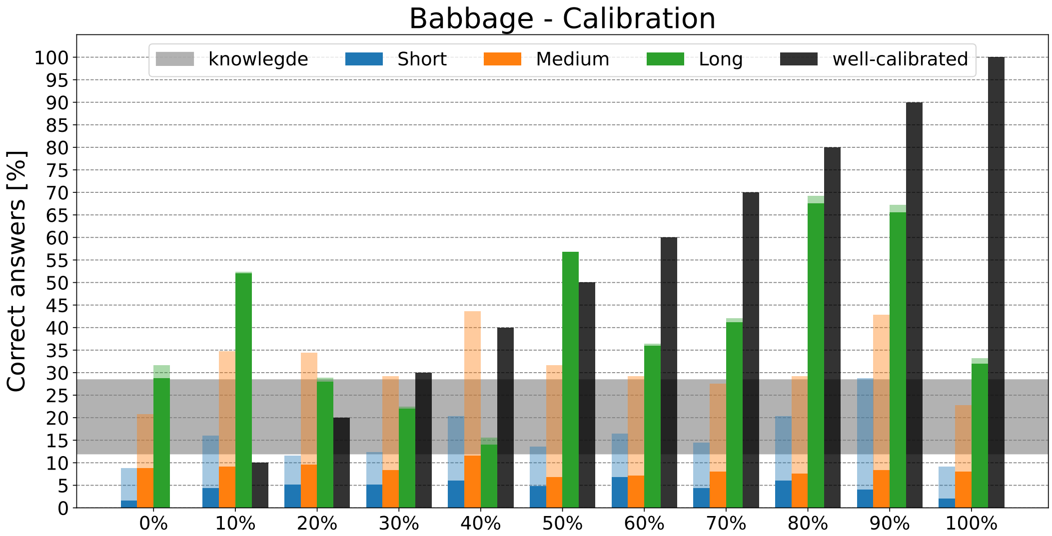

Figure 7.6.: Confidence intervals of Babbage for long prompt are somewhat calibrated

The confidence interval accuracies for the Babbage model differ substantially from the Davinci and Curie results. Confidence interval accuracy of short and medium prompt are very low with 4.6% and 8.5%, respectively. They never reach the lower bound of knowledge for any of the confidence interval percentages. Especially for the short prompt, the accuracies are less than half of the presumed knowledge. Both prompts have a much higher percentage of ambiguous answers than the Davinci and Curie model, with 11% and 23%, respectively. Calibration of both prompts is non-existent and falls into the "Bad calibration" category, with Brier scores above 0.30. Overall performance of the short and medium prompt are bad. The long prompt, on the other hand, has high accuracy for many percentages and outperforms knowledge for every confidence interval above 40%. Not a single combination of prompt and percentage has generated confidence intervals for the Davinci and Curie Model which have outperformed the upper bound of its own knowledge. The long prompt has only 1% ambiguous answers on average, which is only slightly worse than the Curie and Davinci models. Weak calibration for the long prompt between 10% and 90% is recognizable in Figure 7.6 and is supported by the Brier score in Table B.3 of 0.2648, which is under 0.30 for the first time and falls in the category of medium calibration.

	Short	Medium	Long	Perfect
Average Brier score from 0% - 100%	0.350	0.332	0.326	0.150
Average Brier score from 10% - 90%	0.317	0.320	0.265	0.183

Table 7.4.: Brier scores of Babbage's CI's for short, medium and long Prompt

By being more selective and only averaging over accuracies between 20% and 90%, the Brier score is 0.2446, which is under 25% for the first time and falls in the category of good calibration. Overall, with 11.7%, the Babbage Model has a much higher average percentage of ambiguous answers than the bigger models of Davinci and Curie.

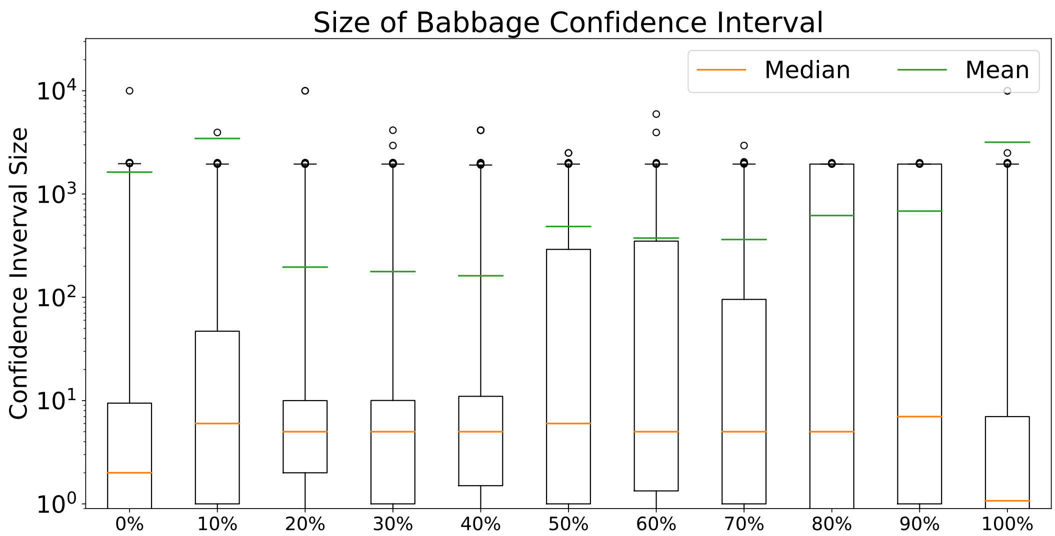

Figure 7.7.: Mean confidence interval sizes of Babbage increase with %

While the median of the confidence interval sizes is relatively static around 5, the mean confidence interval size is increasing between 20% and 100%. The mean sizes between 20% and 40% are all strictly smaller than the mean sizes between 50% and 70%. Moreover, all those confidence intervals are strictly smaller than the mean sizes of the 80% to 100% confidence intervals. All upper whiskers extremes are at exactly 1950, more on that in section 7.3.5. Another feature of the confidence interval sizes stands out; the lower whisker is always below 1. A look at the models text completions reveals, that a substantial part of answer by the model, are confidence intervals between 0 and 1 (which do not make sense for most questions), but are probably often represented in the training data of InstructGPT. Overall, the confidence intervals sizes do not reflect a high quality of calibration. But between 20% and 100% a clear trend of increasing mean confidence interval sizes is visible and also reflected in the confidence interval accuracy in figure 7.6.

7.3.4. Ada

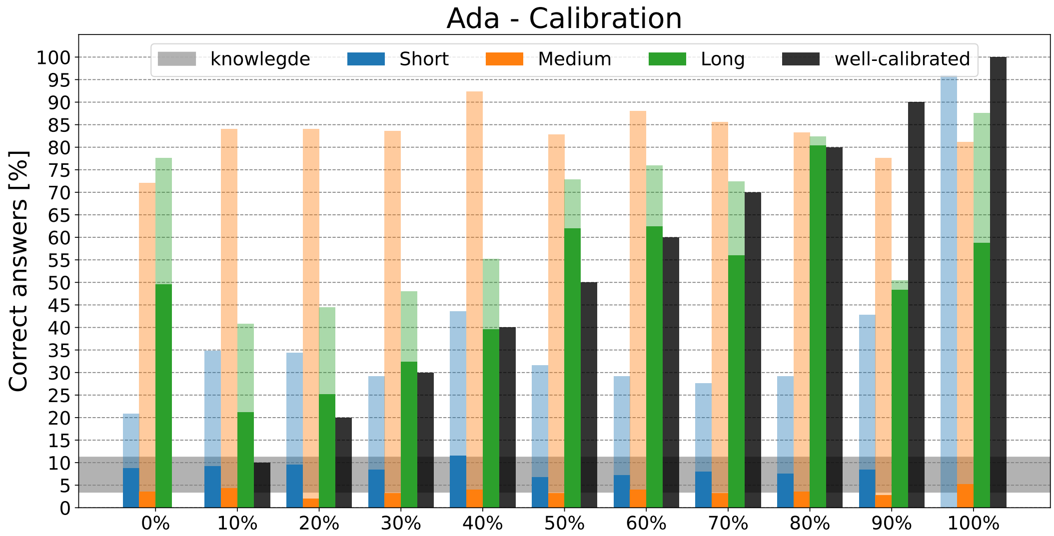

Figure 7.8.: Confidence intervals of Ada for long prompt are somewhat calibrated

The confidence interval accuracies for the Ada model are similar to those of the Babbage model, and differ substantially from the Davinci and Curie models' results. The short prompt results have a decent accuracy of 7.8% on average. The 40% CI accuracy is outperforming Ada's knowledge slightly. All other confidence intervals have accuracies within the Ada model's knowledge, except for the 100% confidence interval, which does not contain a single correct answer. For the short prompt, the Ada model returns on average 30% ambiguous answers and 96% of all answers for the 100% confidence interval are ambiguous. As we can see in Figure 7.8 and Table 7.5, the confidence intervals for the short prompt are not calibrated and fall into the category of awful calibration. The medium prompts' confidence intervals have the lowest accuracy, with an average of 3.6% and only few confidence intervals are performing within the knowledge. On average, 80% of the answers are ambiguous, which is the highest for any prompt or model. The confidence intervals of the medium prompt are not calibrated, and the Brier scores fall into the 'Bad calibration' category.

	Short	Medium	Long	Perfect
Average Brier score from 0% - 100%	0.361	0.349	0.279	0.150
Average Brier score from 10% - 90%	0.320	0.317	0.240	0.183

Table 7.5.: Brier scores of Ada's CI's for short, medium and long Prompt

The long prompt confidence intervals have a very high accuracy relative to Adas knowledge. Ranging between 21% accuracy for the 10% confidence interval to 80%

accuracy for the 80% confidence interval, which is the highest confidence interval accuracy for any model, prompt, and percentage combination. The average accuracy of 49% is more than 4-times as much as the upper bound of Ada's knowledge of 11%. This is the highest increase in confidence interval accuracy compared to a model's knowledge. The percentage of ambiguous answers is on average around 16% and therefore, much lower than the short and medium prompt with 30% and 80%, respectively. Overall, the share of ambiguous answers is rising sharply with decreasing model size, from an average of 0.7% for Davinci, 0.5% for Curie, 12% for Babbage, and 42% for Ada. The high number of ambiguous answers of the Ada model is explained by looking into the raw text answers and numbers of extracted values of the Ada model. The vast majority of ambiguous answers is due to the Ada model returning a single value instead of a confidence interval. The Ada model cannot reliably understand the instruction of returning a confidence interval. Visually, the model seems to be calibrated for probabilities between 10% and 80%. As we can see in Table B.4, the long prompt has a Brier score of 0.279 including the 0% and 100% confidence intervals, which falls into the 'Medium calibration' category, and a Brier score of 0.2401 for confidence intervals from 10% and 90%, which falls into the 'Good calibration' category. And if only including the 10% to 80% confidence intervals, the long prompt for the Ada model has an average Brier score of 0.2173, which is the best Brier score of any combination of model and prompt. The long prompt is therefore an exception from the overall weak performance of the Ada model.

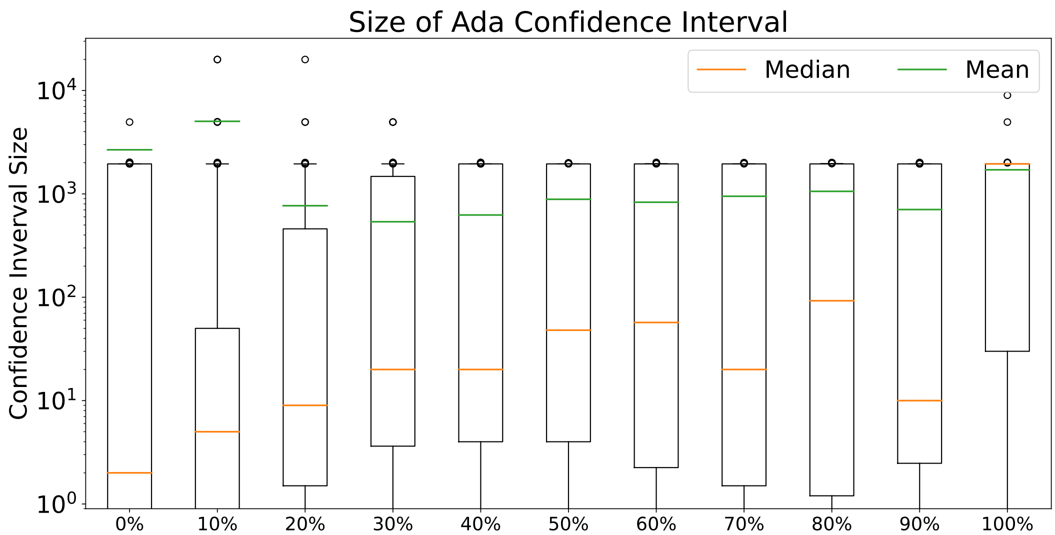

Figure 7.9.: Median of confidence interval sizes of Ada increase with %

While the mean of the confidence interval sizes are relatively static between 500 and 2000, the median confidence interval size is increasing between 2 to 1950. The increase in the median size of the confidence interval is what we would expect from calibrated confidence intervals. Similar to the Babbage confidence intervals, all upper whisker extremes and almost all upper quartiles are at exactly 1950, more analysis of this phenomenon will follow in section 7.3.5. Another feature stands out: Similar to the Babbage model, the lower whiskers always include values between 0 and 1 because such confidence intervals are likely represented in the training data. Overall, the confidence interval sizes reflect some level of calibration.

7.3.5. More Analysis for Long Prompt Results

After the previous results for the long prompt, more analysis is needed to understand why confidence intervals for the Babbage and Ada models show signs of calibration, but the Davinci and Curie results do not. When comparing the box plots of the models (see Figures 7.3, 7.5, 7.7 and 7.9), another difference becomes evident. A confidence interval size of exactly 1950 is rare for the Davinci and Curie models, but very common for the Babbage and Ada models. One plausible hypothesis is, that the Ada and Babbage models return a lot of 50:2000 confidence intervals because the long prompt (B) contains the example: "if your confidence interval is from 50 to 2000, you could enter 50:2000" and the Davinci and Curie models do not.

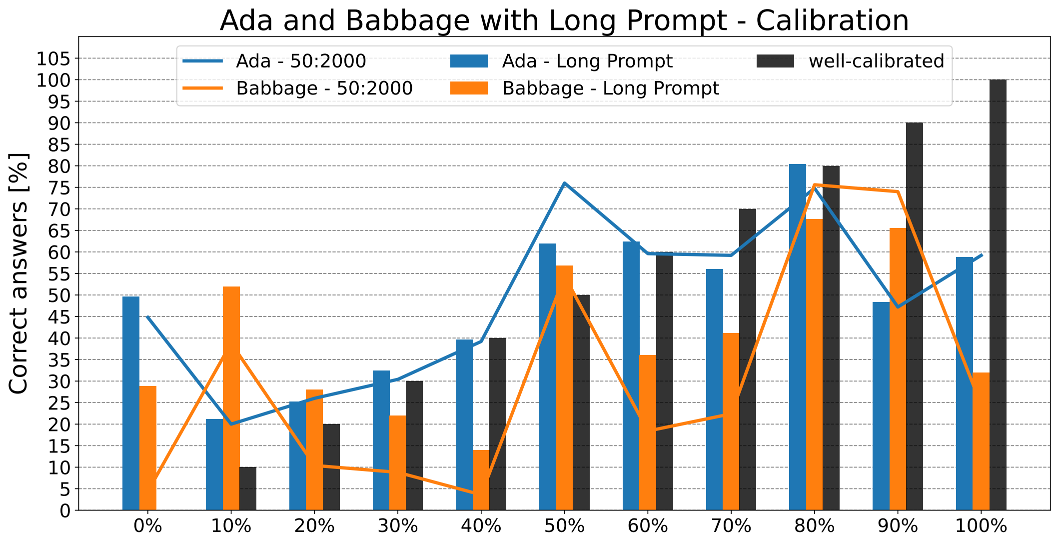

Figure 7.10.: For the Ada and Babbage model under the long prompt, the number of 50:2000 confidence intervals is a great predictor for accuracy

In Figure 7.10 we can clearly see that both the Ada and Babbage models return a lot of 50:2000 confidence intervals. Additionally, it is clear that the share of 50:2000 confidence intervals is decisive for the accuracy. This visual observation is supported by the correlation coefficient of 0.96 between the confidence interval accuracy and the % of 50:2000 confidence intervals for both models. See Appendix B.6 for calculation.

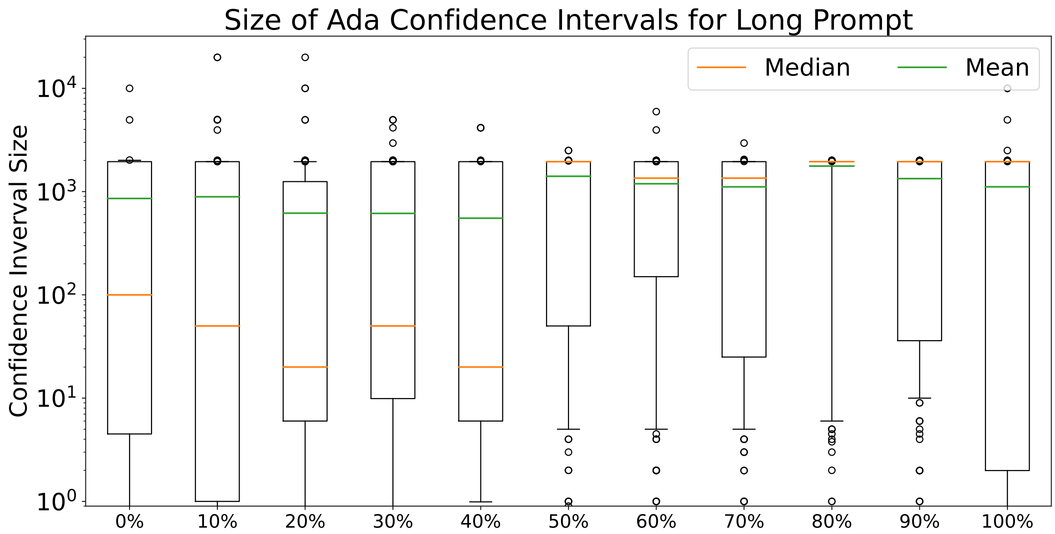

Figure 7.11.: Both mean and median Ada confidence interval sizes are increasing unsteadily but, generally, with higher X% for the long prompt

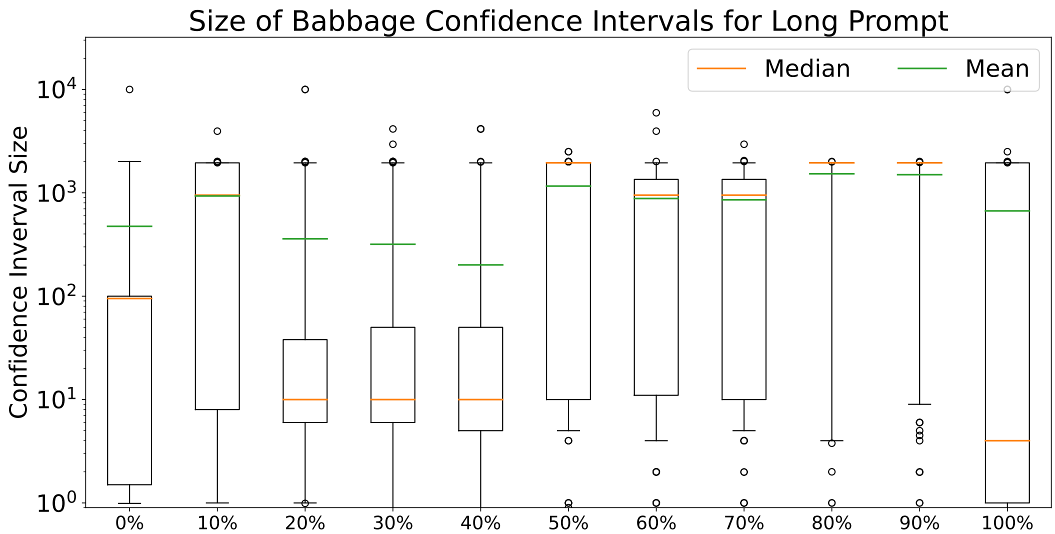

Figure 7.12.: Both mean and median Babbage confidence interval sizes are increasing unsteadily but, generally, with higher X% for the long prompt

Having a closer look at the mean and median sizes of different X% confidence intervals, we can see an unsteady but general trend of increasing size. This is an indicator of calibration and concurs with the calibration of confidence interval accuracies. Most CIs have sizes bigger than 1 and smaller than 2000.

7.3.6. Variations of Long Prompt

To further test the confidence interval calibration of the long prompt for the Babbage and Ada model, I devised two variations of the long prompt. They are both identical to the original long prompt, except for the example confidence interval given in that prompt. Instead of the confidence interval of 50:2000 in the original long prompt ("if your confidence interval is from 50 to 2000, you could enter 50:2000"), the two variations use example confidence intervals of 20:50 and 0:2000. The first confidence interval is chosen to actively undermine calibration because the 20:50 confidence interval would not contain any correct answers for the dataset. The second confidence interval is chosen to possibly improve calibration by using a confidence interval that includes all correct answers of the dataset.

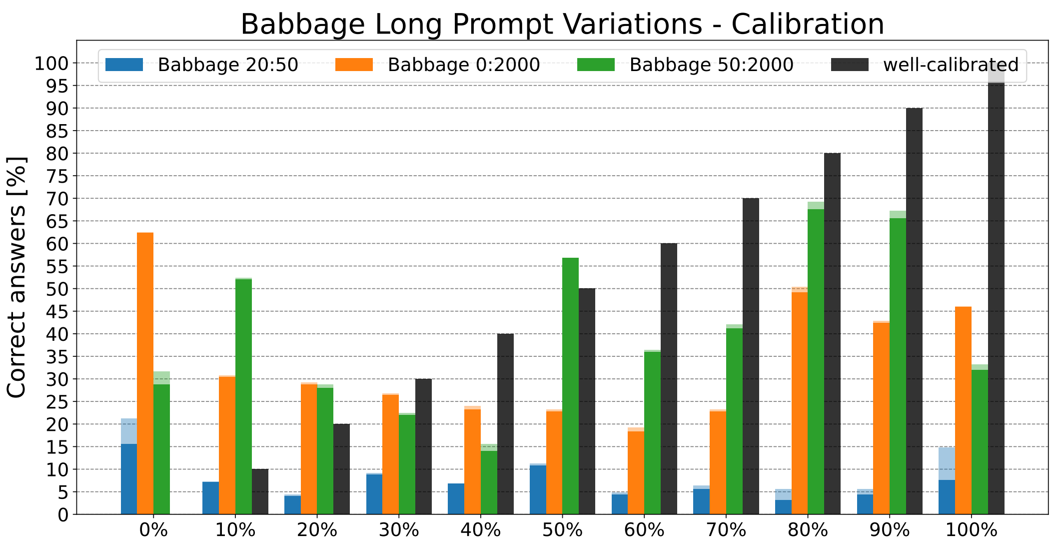

Figure 7.13.: Calibration for the original 50:2000 version is clearly visible and especially pronounced between 20% and 90%. Calibration is much worse and hardly visible for the 0:2000 variation. The confidence interval accuracies of the 20:50 variation are not calibrated. Changes to the example confidence interval in the prompt have a large impact on calibration.

The 20:50 confidence interval variation has a very low accuracy, and the Brier scores in Table 7.6 fall into the 'Bad' to 'Awful calibration' category. The 0:2000 confidence interval variation has an average accuracy of 34%, compared to the 40% for the 50:2000 original long prompt. The Bries Score between 0% and 100% falls into the 'Bad calibration' categor and the Brier score between 10% and 90% barely makes the 'Medium calibration' category. Overall, the relatively good calibration for the original long prompt did not hold for the variations.

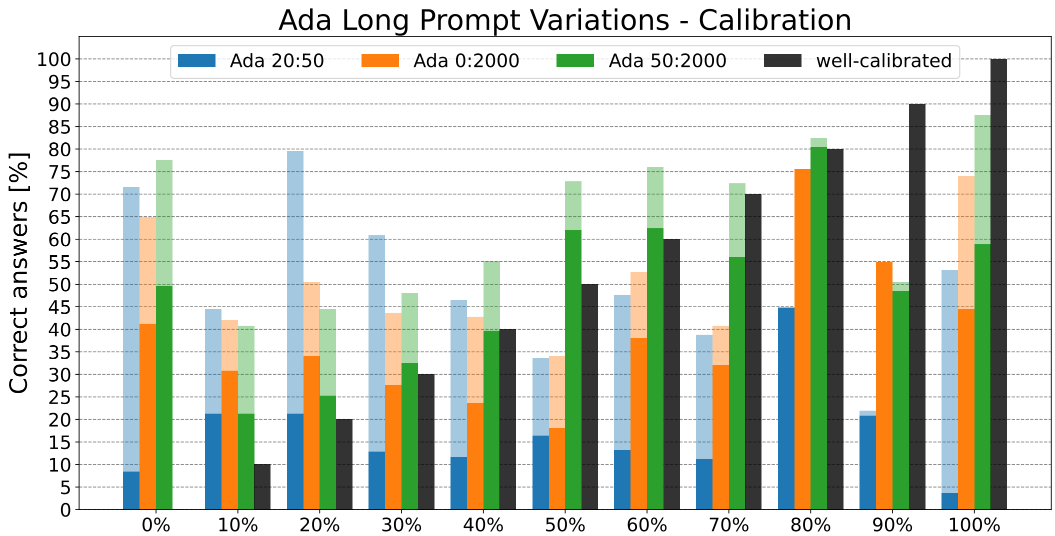

Figure 7.14.: Calibration for the original 50:2000 version is clearly visible and especially pronounced between 10% and 80%. Calibration is much worse and hardly visible for the 0:2000 variation. The confidence interval accuracies of the 20:50 variation are hardly calibrated. Changes to the example confidence interval in the prompt have a large impact on calibration.

The 20:50 variation for the Ada model has an average accuracy of 17% and the Brier score in Table 7.14 falls into the 'Bad calibration' category. The 0:2000 variation has an average accuracy of 38%, compared to the 49% for the 50:2000 original long prompt. The 0:2000 variation shows an unsteady but recognizable calibration, especially between the 10% confidence interval and the 90% confidence interval, where the Brier score falls into the 'Medium calibration' category. Overall both accuracy and, more importantly, calibration deteriorated significantly in both variations.

	Average Brier Score from 0% - 100%	Average Brier Score from 10% - 90%
Babbage with 20:50 CI	0.361	0.322
Babbage with 0:2000 CI	0.347	0.295
Babbage with 50:2000 CI	0.305	0.265
Ada with 20:50 CI	0.342	0.302
Ada with 0:2000 CI	0.303	0.262
Ada with 50:2000 CI	0.279	0.240
Perfect	0.150	0.183

Table 7.6.: Babbage and Ada Brier scores for long prompt confidence intervals

8. Robustness against Adversarial Examples

8.1. Motivation

So far, I only tested the robustness of InstructGPT by using different contexts and tasks. In this chapter, I want to test whether InstructGPT can preserve its ability to return calibrated uncertainty estimates under adversarial examples. Creating adversarial examples only makes sense in experiments where the InstructGPT model is calibrated. Therefore, I will only use the Davinci model and the 'True or False' as well as the 'True or False or I don't know' experiment. As explained in the Background 2.6, robustness to adversarial examples is important because real-world applications will work on noisy data and will potentially be purposefully deceived. Therefore, adversarial examples are a glimpse of what worst-case scenarios for well-intentioned users are and what is possible in cases of deliberate misuse.

8.2. Evaluation

8.2.1. Generating Adversarial Examples

The ultimate goal is to systematically find a way to generate adversarial examples within the semantic distribution of each question. This means changing the question in such a way that the meaning and correct answer to the question stay the same, but the model answer is worsened. Because the text space is discrete and non-differentiable and the model weights of InstructGPT are not public, many of the traditional approaches for generating adversarial examples will not work. As elaborated in section 2.6, many different approaches exist. A possible approach to generate semantic distributions over questions works similar to Sanborn and Griffiths (2007), who generated distributions for stick figures and geometric animals. From these semantically equivalent distributions, one could then select adversarial examples. However, there are multiple problems with translating and applying such an approach to changing sentences. Firstly, generating questions of similar semantic meaning is much more difficult than generating distributions over vectors. Luckily, NLP tools exist that make generating semantically similar questions possible. Therefore, the main challenge is to reliably determine whether generated variations of the question are semantically similar enough. This is to some extent a subjective question and requires checking every single sentence because questions cannot easily be mapped to a metric space, where distances correspond to similarity of semantic meaning. To test the prospect of such an approach, I did a mix between automated and manual question generation. For questions that I categorized as very similar, InstructGPT gave almost exclusively the same answers as for the original question. And for questions that were altered substantially, by introducing errors or changing the semantic meaning significantly, Davinci was less calibrated. In this preliminary test, I was unable to find any small systematic change that had a reliable and large effect on calibration. To find a reliable rule for generating adversarial examples, such a rule has to exist, it has to occur in my generated questions by chance, and it has to occur often enough to enable identifying the adversarial pattern. Patterns are potentially complex and hard to categorize and adversarial examples are rare. Since this is so, I would need to classify thousands of examples manually and hope that I recognize patterns for specific questions and then hope that these adversarial patterns generalize to different questions. Skipping the step of manually checking questions for similarity of semantic meaning would, if the trend from the preliminary test continues, only mean that I would very roughly generate adversarial examples by changing the actual semantic meaning of the question. These problems led me to develop my own approach, in which I designed patterns for adversarial examples that can largely be generated automatically. I fully acknowledge that my approach is less exciting and innovative.

8.2.2. My approach

First, I ran preliminary tests for 25 questions to try different approaches of generating adversarial examples. As suggested by the related research, I paraphrased to test for weaknesses in language comprehension. As a second approach, I tried to disrupt the model by changing the tokenization of the question or keyword. This makes it more difficult for the model to recognize the words, comprehend the meaning, and relate its knowledge. I tried to disrupt tokenization by changing the capitalization of words, replacing letters with numbers, and inserting additional numbers into the question or keyword. Overall, pretrials showed that calibration of the Davinci model worsened, sometimes significantly, but overall the answers remained calibrated for both experiments. For the actual experiment, I selected 50 questions for which the Davinci Model is calibrated really well for both experiments. This way, it would be much easier to detect a deterioration of the calibration. After the pretrials showed a high robustness of the Davinci model, I increased the extent of disruption significantly for the experiment. Instead of only paraphrasing, I included translations of the questions. Instead of changing or inserting a single character, I changed or inserted five characters. To test a broad range of adversarial examples, I devised 15 different approaches of adversarial examples generation. The website `https://texteditor.com/character-wrapper/` was very useful for quickly executing some of the alterations.

1. **Original**:
   What year saw the publication of Pride and Prejudice?
2. **Paraphrased**:
   Which year was Pride and Prejudice published?
3. **German translation**:
   In welchem Jahr wurde "Stolz und Vorurteil" veröffentlicht?
4. **Spanish translation**:
   ¿En qué año se publicó Orgullo y Prejuicio?
5. **All capitalized**:
   WHAT YEAR SAW THE PUBLICATION OF PRIDE AND PREJUDICE?
6. **Inverse capitalization**:
   wHAT YEAR SAW THE PUBLICATION OF pRIDE AND pREJUDICE?
7. **Random capitalization**:
   what YeAR saW THE puBlicAtION of pRidE And PREjUDiCe?
8. **Remove all vowels**:
   Wht yr sw th pblctn f Prd nd Prjdc?
9. **No spaces between letters**:
   WhatyearsawthepublicationofPrideandPrejudice?
10. **Spaces between every letter**:
    W h a t y e a r s a w t h e p u b l i c a t i o n o f P r i d e a n d P r e j u d i c e ?
11. **Insert five random numbers**:
    Wh1at year sa4w the public0ation of Prid9e and Preju6dice?
12. **Replace five letters with numbers**:
    W1at year sa4 the public0tion of Pri9e and Preju6ice?
13. **Wrap with parenthesis**:
    (W)(h)(a)(t) (y)(e)(a)(r) (s)(a)(w) (t)(h)(e) (p)(u)(b)(l)(i)(c)(a)(t)(i)(o)(n) (o)(f) (P)(r)(i)(d)(e) (a)(n)(d) (P)(r)(e)(j)(u)(d)(i)(c)(e)(?)
14. **Wrap with curly brackets**:
    {W}{h}{a}{t} {y}{e}{a}{r} {s}{a}{w} {t}{h}{e} {p}{u}{b}{l}{i}{c}{a}{t}{i}{o}{n} {o}{f} {P}{r}{i}{d}{e} {a}{n}{d} {P}{r}{e}{j}{u}{d}{i}{c}{e}{?}
15. **Add x between letters**:
    WxhxaxtxyxexaxrxsxaxwxtxhxexpxuxbxlxixcxaxtxixoxnxoxfxPxrxixdxexaxnxdxPxrxexjxuxdxixcxex?
16. **Add 42 between letters**:
    W42h42a42t42y42e42a42r42s42a42w42t42h42e42p42u42b42l42i42c42a42t42i42o42n42o42f42P42r42i42d42e42a42n42d42P42r42e42j42u42d42i42c42e42?

8.3. Results

8.3.1. True or False

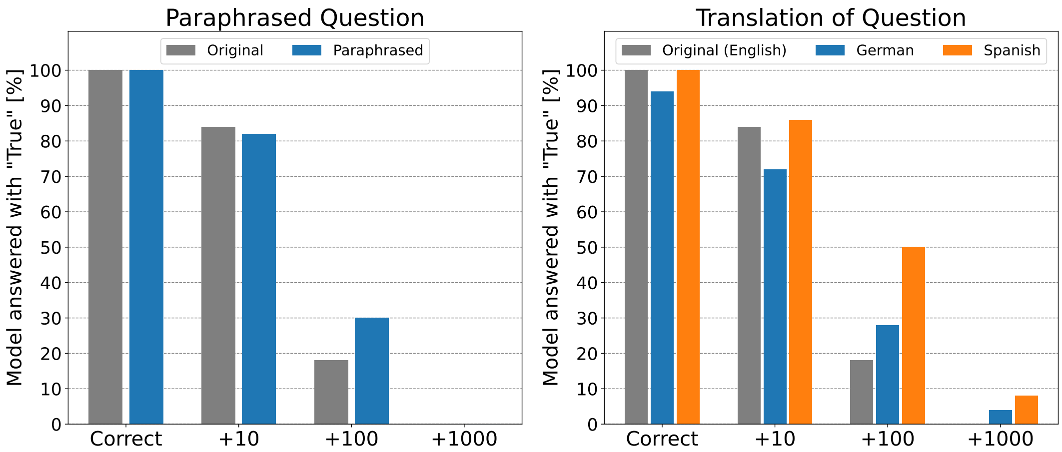

Figure 8.1.: Rephrasing has an only negligible impact on Davinci's calibration. Translation into German and Spanish impacts calibration more visibly, but calibration remains strong. Overall, Davinci struggles to identify proposed answers that are off by 10 as false.

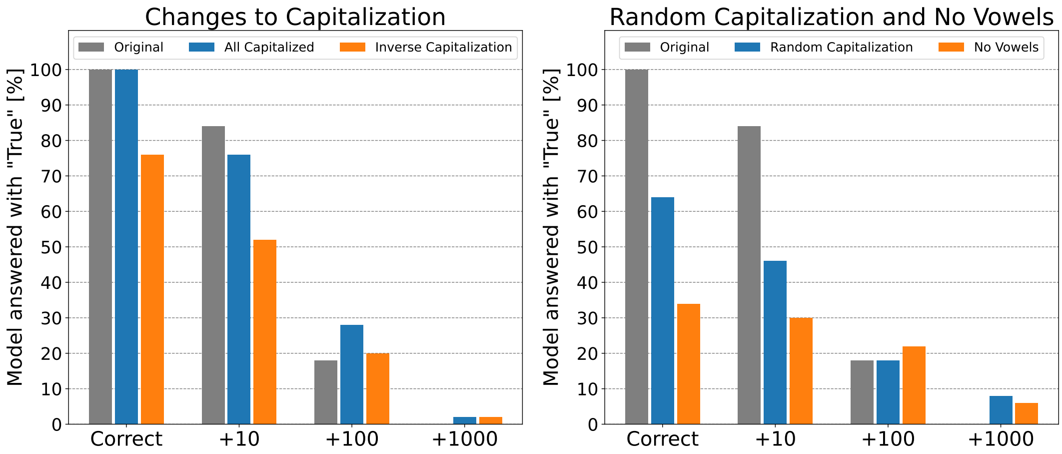

Figure 8.2.: Changing capitalization of the question worsens, but does not nullify Davinci's calibration. Capitalizing the entire sentence has negligible impact, inverse capitalization a more significant impact, and random capitalization comes with the most significant deterioration. The less common a change of capitalization is, the larger is the deteriorating effect. Removing vowels from the question almost nullifies calibration entirely.

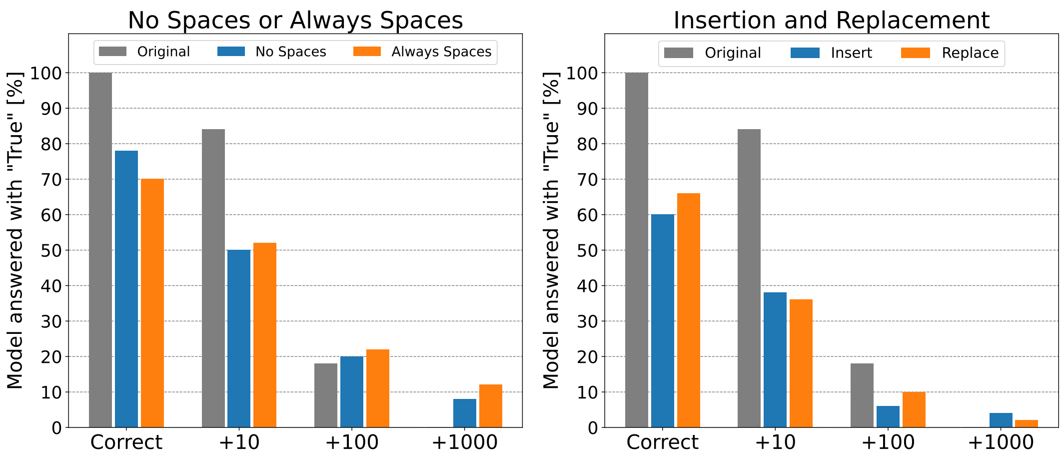

Figure 8.3.: Changing the letter spacing, inserting five random numbers, or replacing five letters with numbers makes the Davinci model less likely to classify a proposed answer as true. The overall effect remains relatively small and calibration remains clearly visible.

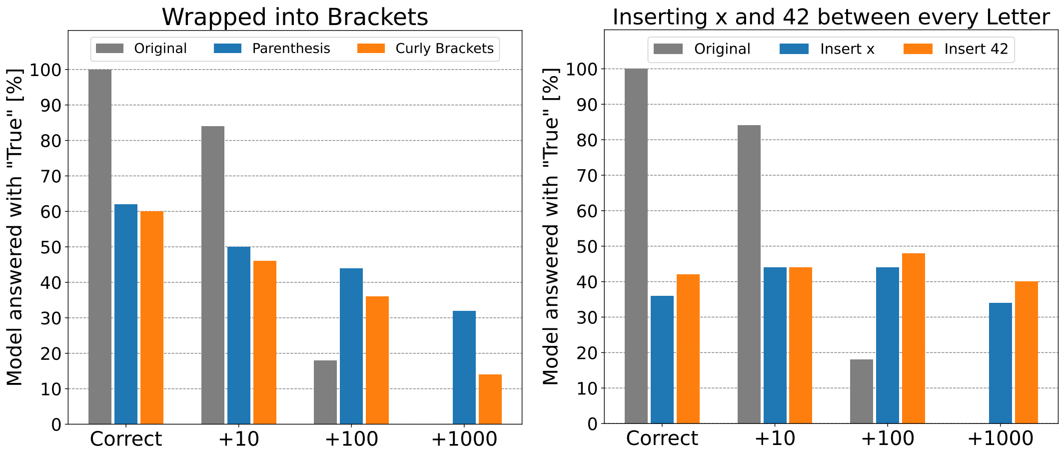

Figure 8.4.: Wrapping each letter into brackets deteriorates calibration strongly, and answers become much more static. However, a weak but steady calibration remains for both parenthesis and curly brackets. The insertion of x and 42 between every single letter not only makes reading the question very strenuous for humans, but also nullifies the calibration of the Davinci model.

In summary, paraphrasing only had negligible effects on calibration. InstructGPT is therefore very robust for natural perturbations and changes in natural human language. More disruptive changes to the questions visibly worsen calibration, but calibration usually remains strong. Only after inserting characters between every letter in the question, calibration was nullified. Such disruptive changes make reading the sentence very strenuous for humans. Only 0.7% of answers were ambiguous. Overall this is a really impressive result and InstructGPT is very robust in its calibration.

8.3.2. True or False or I don't know

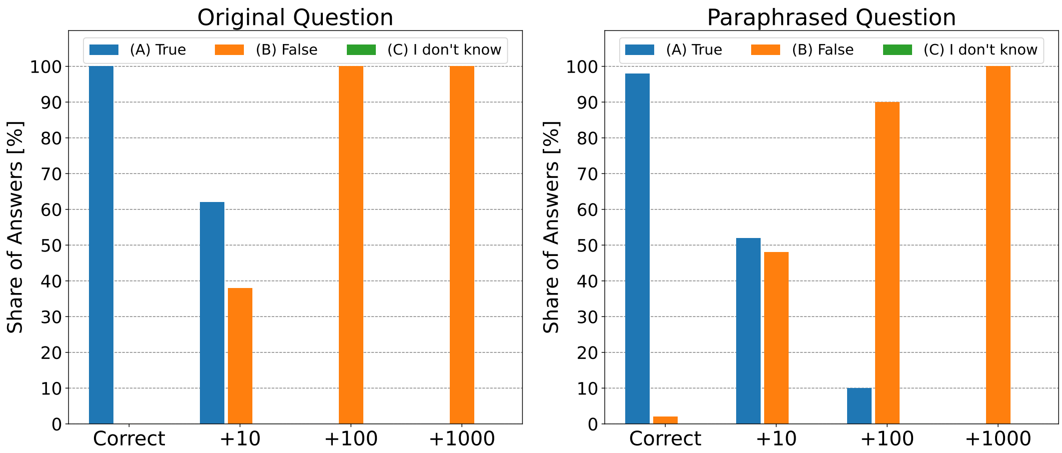

Figure 8.5.: Calibration for the original questions is excellent, and better than in the 'True or False' experiment. All correct proposed answers are classified as 'True', and all the proposed answers differing +100 or +1000 from the correct answer are classified as 'False'. Calibration for the paraphrased questions is slightly worse, but remains very good.

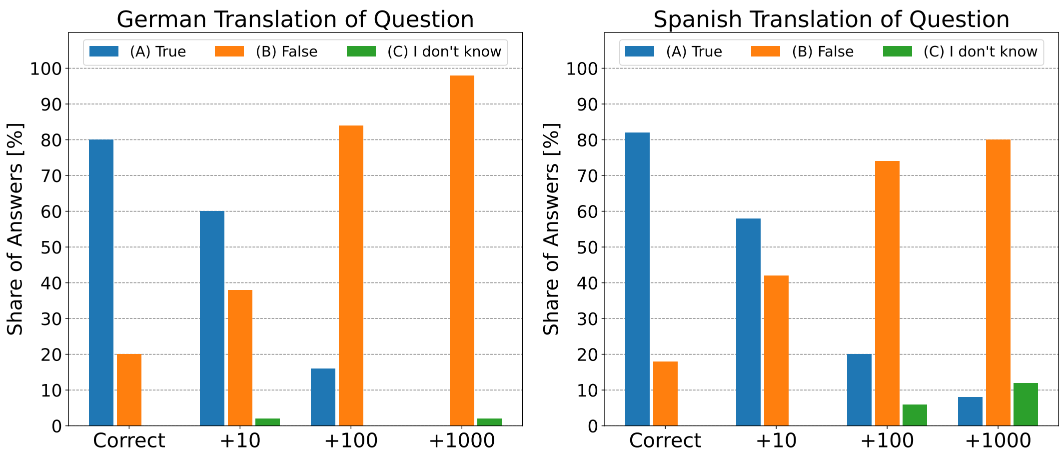

Figure 8.6.: Translating the question with the rest of the prompt remaining in English noticeably worsens calibration. Similar to the 'True or False' experiment, Davinci performs better for German than Spanish.

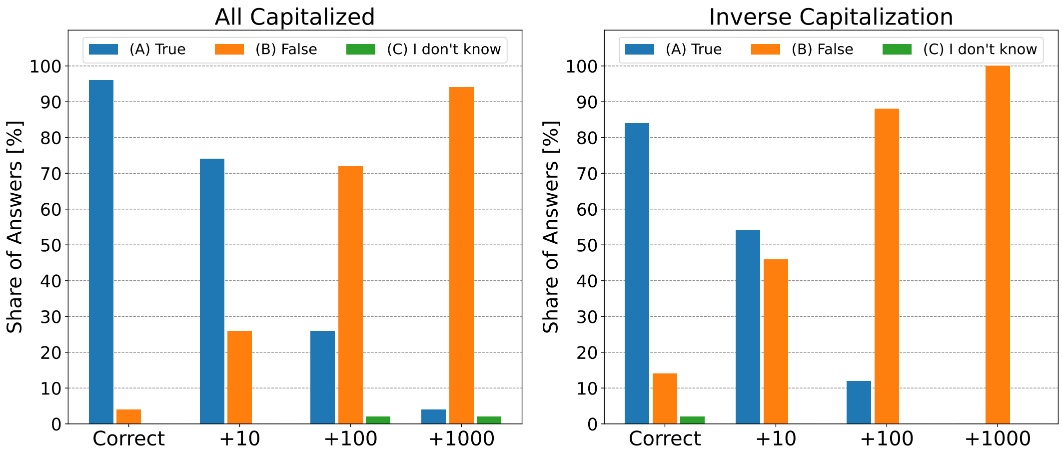

Figure 8.7.: Similar to the 'True or False' experiment, changing capitalization only mildly worsens calibration.

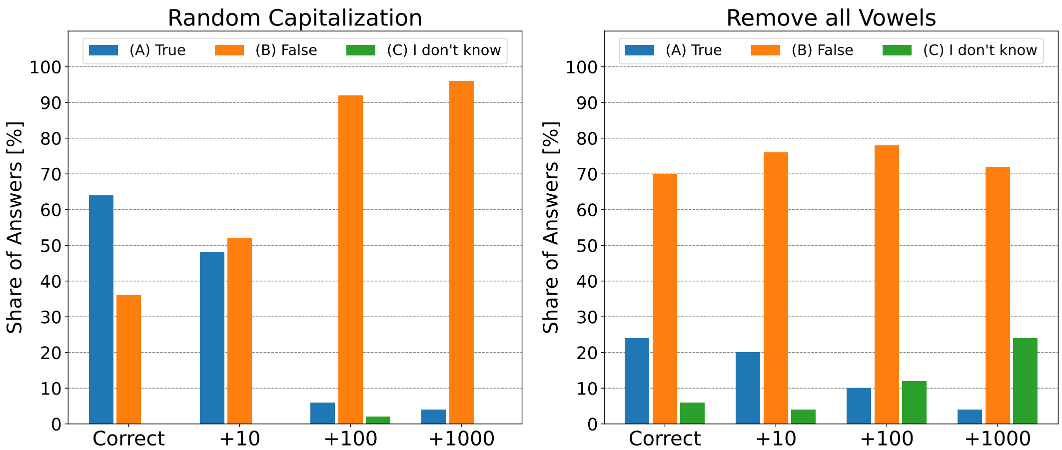

Figure 8.8.: Random capitalization primarily worsens calibration for correct proposed answers. The remaining calibration is weakened but intact. Removing all vowels makes calibration hardly noticeable. There remains a small but steady trend for how often a proposed answer is categorized as 'True', but the likelihood to categorize a proposed answer as 'False' remains largely static.

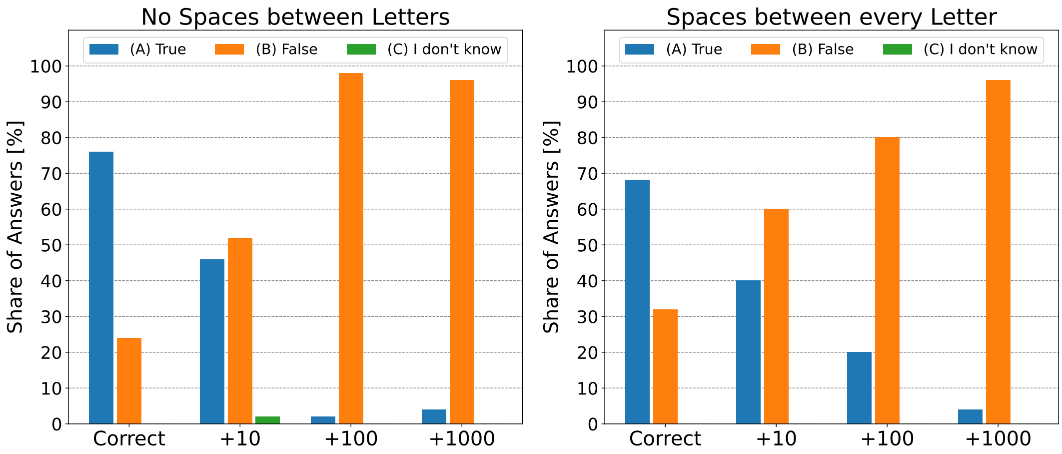

Figure 8.9.: Changing letter spacing only has small effects on overall calibration. Davinci is slightly more calibrated for no spaces than for spaces between every letter.

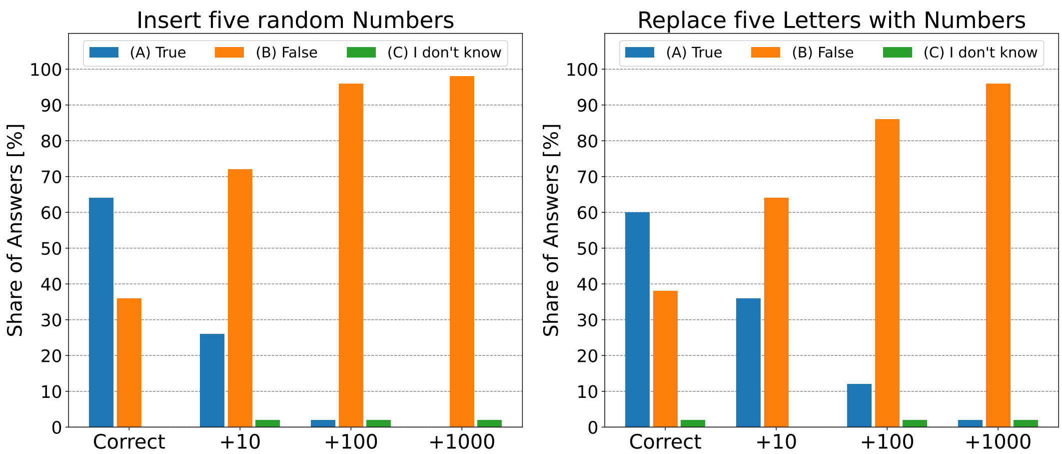

Figure 8.10.: Davinci struggles more with replaced letters than additionally inserted numbers. This makes sense because in the former case information is lost and in the latter case only interfering information is added. Calibration remains clearly visible for both alterations.

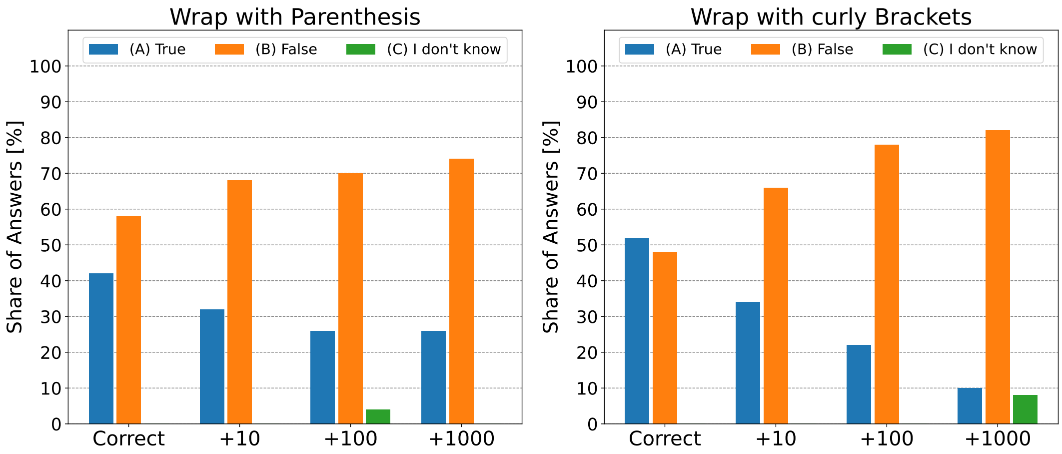

Figure 8.11.: Similar to the 'True and False' experiment, calibration deteriorates strongly for wrapping letters in brackets. While questions are altered systematically and the resulting text is easy to read for humans, wrapping letters is probably not well represented in the training data of InstructGPT. Overall, calibration for curly brackets remains much stronger, and Davinci rarely chooses 'I don't know' as an answer option.

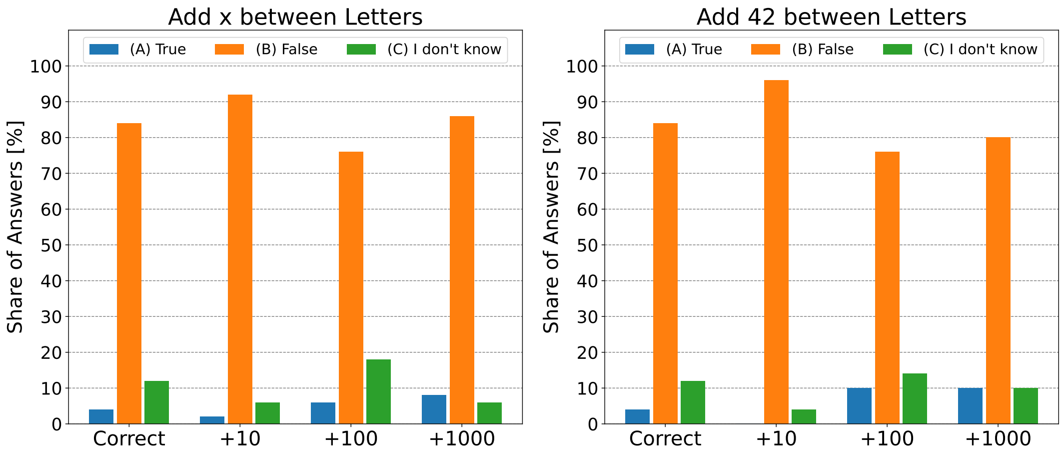

Figure 8.12.: Similar to the 'True and False' experiment, calibration is completely nullified by adding the letter x or the number 42 between each letter of the question. The Davinci Model mostly categorizes proposed answers as 'False'.

Overall, the results for the 'True or False or I don't know' experiment are similar to the 'True or False' experiment. Davinci has great calibration for the original and paraphrased questions. In both experiments, Davinci remains relatively robust against most adversarial examples. Davinci fails to remain calibrated for adding the letter x or the number 42 between every letter in the question. An alternation, which makes it very hard to read for humans. The tendency of the Davinci model to rarely answer with 'I don't know' remains and is similar to the experiment in section 6.3.2. None of the 3200 answers were ambiguous answers. In summary, InstructGPT is also very robust in its calibration for the 'True or False or I don't know' experiment.

9. Discussion

First main finding: Larger InstructGPT models have better question answering capabilities and make smaller errors in their point estimates.

Evidence: In the Point estimate experiment, larger models have dramatically higher point estimate accuracy than smaller models. The respective mean accuracy over all prompts and questions for the models Ada, Babbage, Curie, and Davinci are around 3%, 12%, 30%, and 65% respectively. The interquartile range, the 5th and 95th percentiles, and the extreme values of the prediction errors in the 'Point Estimate' experiment are also much smaller for larger models.

Comparison to expectations and previous research: These results were expected and are in line with the related work. It is established that closed-book question-answering capabilities increase with parameter size both for GPT-3 (Brown et al., 2020), as well as for InstructGPT (Ouyang et al., 2022). Overall, not only the 20-fold accuracy increase from the smallest to the biggest InstructGPT model is impressive, but also the absolute level of accuracy. Both the Curie and Davinci model have much better question-answering abilities than I do. I can give rough estimates for most questions, but rarely estimate the correct number exactly. Taking this into account, the Davinci model's accuracy for the "Average Human" prompt of around 53% seems beyond the level of accuracy and general knowledge of most humans. Therefore, Davinci significantly overestimates the accuracy of an average human's general knowledge.

Limitations: My experiments only used a relatively small dataset with numeric answers. However, my data provides further support for a phenomenon that is already established both by theory and research experiments.

Next steps: There are multiple future research directions. Firstly, there needs to be more research on how knowledge extraction from a large language model can be optimized. Secondly, experiments should seek to determine the answers to the following questions: How strongly do different architectures, the number of model parameters, the amount of computation for training and the size of training data affect question-answering abilities? Which parameter is the current bottleneck for improved question-answering for different architectures?

Conclusion: As I expected, question-answering capabilities improved with increasing parameter size. The Davinci model has a deeper general knowledge than the average human.

Second main finding: All InstructGPT models are capable of verbally estimating their epistemic uncertainty in a variety of ways.

Evidence: Every model returned valid estimates for most experiments in more than 90% of cases. While the Ada model struggled with 10.9% ambiguous answers for the first prompt of the 'Multiple Choice' experiment and with on average of 42% ambiguous answers for the 'Confidence Interval' experiment, it still returned valid responses more than 90% of the time in the majority of prompts and experiments overall. The experiments show that all InstructGPT models can return uncertainty estimates in multiple choice questions, open questions, and as confidence intervals.

Comparison to expectations and previous research: Previous research has shown, that large language models generally and InstructGPT specifically can recognize a multitude of patterns, including multiple choice questions. Therefore, it was expected that InstructGPT can follow the prompts used in the experiments. However, to the best of my knowledge, it has not been previously studied whether transformers can return written confidence intervals. Since there are many confidence intervals in the training data, it seemed reasonable to assume that some models would recognize the pattern or understand the instructions. Overall, the model performed within expectations, with a tendency to exceed expectations in following instructions in a zero-shot setting. Most research has focused on few-shot learning, while my experiments used the more challenging zero-shot learning. The prompts only included instructions but no examples except for the confidence interval prompts. Of course, the models have previously seen all of these question formats. Still, they were able to recognize the format and answer with fitting completions.

Limitations: The experiments' results are limited in many ways. My thesis focused on very specific ways to express uncertainty. Therefore, it is uncertain whether my results would generalize to other ways of expressing uncertainty. Further, I used prompts that worked well during pretrials, but different prompt designs could have significant impact on experiment results. Lastly, it has to be noted, that matching the syntactical pattern is not the same as truthfully and accurately expressing the epistemic uncertainty about one's own knowledge, but merely a precondition to it. This limitation is important, as we later see that calibration of epistemic uncertainty is lacking in most cases.

Next steps: Future research could improve existing prompts or request query settings. Devising new experiments such as expressing uncertainty via words like "low confidence", or creating probability distributions via random sampling could be worthwhile. Further exploration of how strongly abilities to estimate uncertainty get enhanced by fine-tuning, changes to the model architecture, or further scaling of parameter size, seem to be valuable approaches.

Conclusion: Overall, InstructGPT's abilities to follow instructions are impressive. Further enhancements seem possible and promising.

Third main finding: Prompts design has a significant impact on InstructGPT answers. Uncertainty Estimates between experiments are not internally coherent.

Evidence: In the 'Point Estimate' experiment, there was significant and mostly unpredictable variation between the accuracy of the six prompts. In the 'True or False' experiment, changing the order of the answer options entirely changed Curie's and Babbage's answers. Changing the multiple choice answer options also changed the uncertainty estimates returned by the models. In the 'Confidence Interval' experiment, the three different prompts mostly led to very different results, especially in the smaller models. Overall, differences in prompt design led to significant differences in the uncertainty estimates. Furthermore, returning different answers for the same question asked in different ways means that the model is inconsistent in its epistemic uncertainty estimates.

Comparison to expectations and previous research: It is well established in the previous research that prompt design has a big impact on performance of language models (Liu et al., 2021; Reynolds and McDonell, 2021).

Limitations: My experiments are testing a very specific capability with a small number of prompts and questions. The observed results could be different for other prompts, datasets, and capabilities. Especially for capabilities in which InstructGPT models generally perform better than for epistemic uncertainty estimation, the results would probably be more robust and coherent. Because I did exploratory pretests to find suitable prompts, my experiments already select robust and well-performing prompts. Therefore, without previous prompt engineering, results would be even less robust and coherent.

Next steps: To improve robustness and coherence, one could use random sampling for the same prompt, or use multiple variations of the prompt and average the results or generate probability distributions from it. Another approach for estimating uncertainty would be to use differences between the uncertainty estimates as an indicator of the model's uncertainty. If a model only returns uncertainty estimates between 40% and 60% for different prompts asking the same question, we can give much higher credence to the model than if the uncertainty estimates range between 10% and 90% for different prompts asking the same question. Fine-tuning for uncertainty estimates might also increase robustness and coherence of the models' answers.

Conclusion: Asking for the same uncertainty estimate in different ways can lead to very different results. The model is often not frail and inconsistent with itself.

Forth Main finding: Larger InstructGPT models are better able and more reliable to follow instructions than smaller InstructGPT models.

Evidence: Overall, the number of ambiguous answers is much lower for the larger models Davinci and Curie, than for the smaller models Babbage and Ada.

Share of ambiguous answers [%]	Davinci	Curie	Babbage	Ada
Point Estimates	1.8	13.8	18.2	19.2
True or False	0.7	0.1	1.3	1.9
True or False or I don't know	0	0.1	0.3	1.6
Multiple Choice Questions	0	0	0	7.9
Free Uncertainty Estimates	4.7	0.5	6.3	4.5
Confidence Intervals	0.7	0.5	11.7	30
Mean share of ambiguous answers	1.3	2.5	6.3	10.9

There are exceptions to this trend, as the Curie model has a smaller share of ambiguous answers in the majority of experiments than the Davinci model. And in the free uncertainty estimates experiment, the Babbage model has more ambiguous answers than the Ada model. But overall and on average, there is a clear trend, that larger models return fewer ambiguous answers. In addition, larger models follow instructions more precisely in a qualitative sense. For example, the Davinci model uses the Confidence Interval representation X:Y specified in the prompt, while the Babbage and Ada models are using a wide array of confidence interval representations. Also, performance like accuracy and calibration are better for larger models, which means they are much better able to understand and follow the intent of instructions.

Comparison to expectations and previous research: This result is expected, as previous research supports that language understanding and general capabilities are improved by increasing parameter size (Devlin et al., 2018; Brown et al., 2020; Kadavath et al., 2022; Ouyang et al., 2022).

Limitations: Due to the small sample of very specific tasks and questions, the results are a limited, but additional contribution to the current consensus. The capability and reliability to follow instructions could differ for tasks different from my experiments. It has to be noted, that this specific finding primarily refers to the ability to follow a pattern explained by instructions in the prompt, and not the ability to understand what that pattern means. My findings do not imply that the model understands the nature of epistemic uncertainty.

Next steps: Reliability can be improved for each experiment by improving prompts and fine-tuning for specific experiments. I think broader research on how transformers improve their ability to express epistemic uncertainty with scaling of parameter size are promising. Research whether capability, reliability, and calibration of estimates continue to improve with model size would be valuable. However, building larger and more capable transformers can only be executed by very capable and well-funded research teams. Another promising current research direction is reinforcement from human feedback (RHLF). The current RHLF approach would benefit from improvement or from finding better working alternatives.

Conclusion: Overall, larger models can follow patterns and instructions more easily and robustly. Research into whether this trend continues and how to improve the ability to follow instructions are both highly promising and valuable.

Fifth main finding: Only the Davinci model is calibrated for some of the experiments. None of the other models show real calibration for any of the experiments.

Evidence: Davinci is calibrated for 'True or False' questions and for the 'True or False or I don't know' questions. Such experiments are relatively inexpressive and primitive ways of expressing uncertainty. Additionally, Davinci shows weak calibration for the more expressive free uncertainty estimates. However, the Davinci model does not show signs of calibration for the multiple choice probabilities and confidence interval experiment. The Curie, Babbage, and Ada models do not show reliable signs of calibration for any of the experiments. Only for the long prompt in the 'Confidence Interval' experiment, the Babbage and Ada model initially displayed calibration. However, this superficial calibration does not hold under scrutiny, as small changes to the long prompt nullify their calibration.

Comparison to expectations and previous research: The results partly fulfill and partly underperform optimistic expectations. As I outlined in the related work chapter, calibration of large language models depends significantly on how calibration is extracted and whether the model is fine-tuned for uncertainty calibration. In cases where the models' posterior probabilities 'logits' are used as an indicator of uncertainty, results are mixed (Desai and Durrett (2020)). Kadavath et al. (2022) find that models like BERT (Devlin et al., 2018) and RoBERTa (Liu et al., 2019) are calibrated out-of-the-box. Kadavath et al. (2022) also find that their own language models, with similar sizes to the different InstructGPT models, are calibrated for multiple choice and 'True or False' questions when evaluating the posterior probability distributions over the answer options. On the other hand, Jiang et al. (2021) find that T5 (Raffel et al., 2019), BART (Lewis et al., 2019) and GPT-2 (Radford et al., 2019) are not well-calibrated out of the box for question-and-answer-tasks, including multiple choice tasks. When studying calibration for verbal uncertainty estimates instead of posterior probabilities of model outputs, calibration is mostly absent. Branwen (2020) finds that GPT-3 (Brown et al., 2020) is incapable of verbally expressing uncertainty in a calibrated way. Mielke et al. (2022) find that BlenderBot (Roller et al., 2020) is not initially calibrated for verbal uncertainty estimates. While Kadavath et al. (2022) models are calibrated for posterior probabilities, they are not calibrated for verbal self-evaluation of epistemic uncertainty. Their model's verbal uncertainty estimates are poorly calibrated for zero-shot learning and weakly calibrated for 20-shot learning in 'True or Fals' experiments. Lastly, a lot of research successfully uses fine-tuning, temperature scaling, and other methods to improve calibration of language models. This includes Lin et al. (2022), who successfully fine-tuned GPT-3s Davinci model to be well-calibrated for verbal epistemic uncertainty estimates on a specific dataset. Overall, my results fit well into existing research. However, the hope was that InstructGPT would have improved abilities through reinforcement learning from human feedback to enable it to estimate its epistemic uncertainty in a calibrated way. This hope was only partly fulfilled.

Limitations: As the literature suggests, results for calibration have very limited generality and differ for each specific setup and model. My results only speak for verbal uncertainty estimation with InstructGPT in the specific experiments I used. And even for those experiments, changing the wording of the prompt, using a different dataset, or using one-shot or few-shot learning would likely change calibration significantly.

Next steps: There are many different research avenues that would be worthwhile to pursue. Firstly, one could repeat all my experiments with one-shot and few-shot learning. This is likely to improve the overall results and calibration to some extent. The experiments should consider, however, that choosing certain one and few-shot examples already influences the results and possible calibration could depend on the given few-shot examples. Secondly, one could design improved or new experiments to test for verbal calibration. Possibly, using random sampling from text completions, one could generate probability distributions over possible completions. Random sampling can be achieved through setting the temperature from 0 to 1 and then running the same request query multiple times. However, this approach did not work well in preliminary tests, and more prompt engineering or additional changes to the setup are probably needed for this to return valuable results. Another possible change to the experiments would be to introduce the phrase "Let’s think step by step" to encourage the model to make reasoning explicit before suggesting an uncertainty estimate. Kojima et al. (2022) had success in introducing this phrase in one-shot learning. Another alteration of the experiment would be to let the model make an uncertainty estimate (zero, one or few-shot) and then provide the model with additional information through the prompt and let the model update its uncertainty estimate. Additional information can have the form of giving an interval in which the correct answer lies, or a related fact that makes the prediction of the value easier. When giving intervals, again, having numeric values is beneficial because this makes intervals possible in the first place and one can vary the informational value of a given interval size by widening or narrowing it. Such an experiment could study whether additional information can improve calibration and whether InstructGPT can do Bayesian updating, in the sense, that information confirming its prior belief about the correct answer should increase the likelihood it assigns to a proposed answer being true. To improve robustness and calibration of results, combining multiple uncertainty estimate extraction methods and weighing their estimates could be valuable. If they are relatively independent of each other, results should get more robust and possibly more calibrated. A different research angle, would be to improve the model itself instead of the experiments. One obvious approach would be to fine-tune the model for specific experiments used in my thesis. Training could be similar to the work of Lin et al. (2022). Investigating whether calibration for specific experiments improves with fine-tuning and whether improved calibration for one experiment generalizes to do different experiments, would be interesting. If calibration is still hard to achieve, a combination of both previous research directions is most likely to improve calibration: Fine-tuning a model to a specific task and simultaneously using few-shot learning should significantly improve the probability of calibration. If "Let's think step by step" or providing intervals for the correct answer improve calibration, they can be used additionally to boost calibration. Lastly, all the previously described experiments are worthwhile to run on other language models to find out whether all language models fail on the same or different tasks and how well they each perform. Comparing different language models' calibration to each other could potentially highlight how architecture and training possibly shape calibration and offer insights on how to develop more calibrated language models in the future.

Conclusion: Overall, my finding support the claim that current language models struggle to verbally express their epistemic uncertainty in a calibrated way without fine-tuning. I find that parameter size improves calibration for InstructGPT and calibration is better for primitive and indirect ways of expressing epistemic uncertainty.

Sixth Main finding: The InstructGPT Davinci model's calibration in the 'True or False' and the 'True or False and I don't know' experiment is robust and only mildly affected by most adversarial examples.

Evidence: Results from 15 different adversarial examples show that calibration only slightly worsens from paraphrasing, translating, changing the capitalization, or spacing of the question. Calibration worsens more significantly for wrapping each letter into brackets, inserting five random numbers, or replacing five letters with numbers. Calibration is only nullified if the letter x or number 42 is inserted between each letter of the sentence. This also makes the questions very strenuous to read for a human.

Comparison to expectations and previous research: It was difficult to form accurate expectations for how robust InstructGPT would be for adversarial examples. Both the approaches to generating adversarial examples and the results for language models differ greatly. Overall, InstructGPT was more robust than I had expected, based on the previous research.

Limitations: Since I used a limited amount of adversarial attacks, most of which can easily be described and generated, it is likely that much better and more subtly adversarial examples can be developed.

Next steps: Finding algorithmic solutions to generate targeted, but subtle adversarial examples would be very valuable. Finding such adversarial examples would allow to fine-tune models to be more robust.

Conclusion: Increasing robustness to adversarial examples is important when language models are used in safety-critical applications and other high stakes environments. InstructGPT seems to be robust to many alterations, and calibration only worsens significantly for obvious and major alterations to the question. However, it is likely, that further research finds more subtle and effective adversarial examples.

9.1. Conclusion

Overall, the examined InstructGPT models show impressive language comprehension, general knowledge, and ability to follow instructions in zero-shot settings. The number of parameters greatly improves InstructGPT's capabilities and performance, including its question-answering abilities, uncertainty calibration, and robustness to adversarial examples. While state-of-the-art language models are mostly calibrated for model logits, InstructGPT models struggle to verbally self-evaluate their uncertainty in a calibrated way. Except for the biggest model, no model is calibrated for any of the experiments in this study. Moreover, the biggest model is only calibrated for the most indirect and simple experiments assessing uncertainty estimations. It is to be expected that scaling of parameter size will continue to improve and expand language models' capabilities in the area of self-evaluation of uncertainty.

A. Point Estimate Experiment

A.1. Prompts used in Point Estimate Experiment

For the 'Point Estimate' experiment, I used six different prompts. For each of the prompts, I will print the prompt in `typewriter` font and a possible answer by the model in **bold**. I will always use the same placeholder question "How many episodes, in total, does the series How I Met Your Mother have over its nine seasons?" and the placeholder answer "208".

Standard Prompt

```
Answer the following question by providing an integer as your point estimate.
The integer should be the answer you find most likely.
Question: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
Only reply with an integer (your point estimate) and nothing else.
```

208

Settings

Standard Settings 4.4

Human Expert

Prompt

```
The following is a conversation with the current World Quizzing Champion.
The Champion is clever, knowledgeable and always knows the correct answer.
The Champion is asked questions by the Host. Every question has an integer
as the correct answer.
The Champion answers only with the correct number.


Host: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
```

Champion: 208

Settings
Special Settings 4.4.1

AI Expert

Prompt
```
The following is a conversation with an all-knowing AI. The AI is clever,
knowledgeable and always knows the correct answer.
The AI is asked questions by the Host. Every question has an integer as
the correct answer.
The AI answers only with the correct number.


Host: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
```

AI: 208

Settings
Special Settings 4.4.1

AI Assistant

Prompt
```
The following is a conversation with an AI assistant. The assistant is
helpful, creative, clever, and very friendly.


Human: Hello, who are you?
AI: I am an AI created by OpenAI. How can I help you today?
Human: Please answer the following question: How many episodes, in total,
does the series How I Met Your Mother have over its nine seasons? Only
return a single integer as you answer.
```

AI: 208

Settings
Special Settings 4.4.1

Average Human

Prompt
```
The following is a conversation with a contestant. The contestant is inattentive,
```

```
not intelligent and has very limited general knowledge.
The contestant is asked questions by the Host. Every question has an integer
as the correct answer.
The contestant answers only with an integer.


Host: How many episodes, in total, does the series How I Met Your Mother
have over its nine seasons?
```

Contestant: 208

Settings

Special Settings 4.4.1

Only Question

Prompt

```
How many episodes, in total, does the series How I Met Your Mother have
over its nine seasons?
```

208

Settings

Standard Settings 4.4

A.2. Calculation of Coefficient of Variation

Calculation of Standard Deviation, Mean and Coefficient of Variation.

	Davinci	Curie	Babbage	Ada
Standard	0.66	0.236	0.12	0.04
Human Expert	0.684	0.336	0.044	0.048
AI Expert	0.66	0.364	0.16	0.044
AI Assistant	0.66	0.344	0.076	0.012
Average Human	0.528	0.092	0.1	0.016
Only Question	0.736	0.42	0.22	0.048
Standard Deviation	0.004	0.107	0.057	0.015
Mean	0.655	0.299	0.12	0.035
Coefficient of Variation	0.006	0.359	0.478	0.43

B. Confidence Interval Experiment

B.1. Prompts used in Confidence Interval Experiment

Every prompt consisted of three parts. A question from the dataset in section (4.6), a probability X%, and a context that surrounds the question. The entire prompt, containing question, probability, and context are given as input to the model. When running the 'Confidence Interval' experiment on the dataset with a specific context and model, the question and probabilities are changed. As a placeholder question, I always used the first example question from Table 4.1, 80% as a placeholder probability and "176:236" as a placeholder answer. For each of the prompts, I print the prompt in `typewriter` font, underline the probability, and give a possible answer by the model in **bold**.

Short Prompt

`Return your confidence interval in the form x:y to answer the following question.`

`How many episodes, in total, does the series How I Met Your Mother have over its nine seasons?`

`My 80% confidence interval is` **176:236**

Settings

Standard Settings 4.4

Medium Prompt

`In Frequentist statistics, a confidence interval is a range of estimates for an unknown parameter. A confidence interval is computed at a designated confidence level. The confidence level represents the long-run proportion of corresponding CIs that contain the true value of the parameter. For example, out of all intervals computed at the 80% level, 80% of them should`

```
contain the parameter's true value.


Return your confidence interval in the form x:y to answer the following
question.


How many episodes, in total, does the series How I Met Your Mother have
over its nine seasons?


My 80% confidence interval is
```
176:236

Settings

Standard Settings 4.4

Long Prompt

```
Your question set is now Confidence Intervals!

Each question in this set has a numeric answer, and we will ask you for
an X% "confidence interval" - that is, a range of numbers that you're X%
confident contains the true answer.

For example, you might be asked to give your 80% confidence interval for
the number of people that were alive at the beginning of 2010. If you are
perfectly calibrated, then on such questions (that ask for an 80% confidence
interval) the true answer will fall within your stated range 80% of the
time. On the other hand, on questions where you are asked for your 90%
confidence interval, the true answer should fall within your stated confidence
interval 90% of the time. The higher the confidence level that is asked
for, the wider apart your lower and upper bounds should be. In response
to each question, type your lower bound, then a colon, then your upper
bound. For example, if your confidence interval is from 50 to 2000, you
could enter 50:2000.


Return your confidence interval in the form x:y to answer the following
question.


How many episodes, in total, does the series How I Met Your Mother have
over its nine seasons?


My 80% confidence interval is
```
176:236

Settings
Standard Settings 4.4

B.2. Brier Scores of Confidence Intervals

% of CI	Perfect Calibration	Short	Medium	Long
0%	0	0.632	0.528	0.632
10%	0.09	0.541	0.541	0.586
20%	0,16	0.446	0.438	0.482
30%	0.21	0.359	0.364	0.386
40%	0.24	0.3	0.3	0.31
50%	0.25	0.25	0.25	0.25
60%	0.24	0.225	0.222	0.212
70%	0.21	0.216	0.228	0.194
80%	0.16	0.23	0.23	0.2
90%	0.09	0.26	0.282	0.215
100%	0	0.304	0.324	0.368
avg. 0% - 100%	0.15	0.342	0.337	0.349
avg. 10 - 90 %	0.183	0.314	0.317	0.315

Table B.1.: Brier scores of Davinci confidence intervals

% of CI	Perfect Calibration	Short	Medium	Long
0%	0	0.364	0.364	0.46
10%	0.09	0.215	0.215	0.24
20%	0.16	0.186	0.213	0.242
30%	0.21	0.197	0.236	0.237
40%	0.24	0.21	0.22	0.23
50%	0.25	0.25	0.25	0.25
60%	0.24	0.307	0.296	0.292
70%	0.21	0.384	0.375	0.349
80%	0.16	0.46	0.44	0.41
90%	0.09	0.554	0.544	0.519
100%	0	0.632	0.68	0.556
avg. 0% - 100%	0.15	0.342	0.349	0.344
avg. 10 - 90 %	0.183	0.307	0.311	0.308

Table B.2.: Brier scores of Curie confidence intervals

% of CI	Perfect Calibration	Short	Medium	Long
0%	0	0.016	0.088	0.288
10%	0.09	0.045	0.084	0.426
20%	0.16	0.071	0.098	0.208
30%	0.21	0.111	0.124	0.178
40%	0.24	0.17	0.18	0.19
50%	0.25	0.25	0.25	0.25
60%	0.24	0.346	0.346	0.288
70%	0.21	0.472	0.458	0.325
80%	0.16	0.6	0.59	0.23
90%	0.09	0.778	0.743	0.285
100%	0	0.98	0.68	0.92
avg. 0% - 100%	0.15	0.350	0.332	0.326
avg. 10 - 90 %	0.183	0.317	0.320	0.265

Table B.3.: Brier scores of Babbage confidence intervals

% of CI	Perfect Calibration	Short	Medium	Long
0%	0	0.088	0.036	0.496
10%	0.09	0.084	0.045	0.18
20%	0.16	0.098	0.052	0.191
30%	0.21	0.124	0.103	0.22
40%	0.24	0.18	0.17	0.24
50%	0.25	0.25	0.25	0.25
60%	0.24	0.346	0.352	0.235
70%	0.21	0.458	0.477	0.266
80%	0.16	0.59	0.62	0.16
90%	0.09	0.743	0.788	0.423
100%	0	1	0.948	0.412
avg. 0% - 100%	0.15	0.361	0.349	0.279
avg. 10 - 90 %	0.183	0.320	0.317	0.240

Table B.4.: Brier scores of Ada confidence intervals

	Static Fifty		Inverse Model		Worst Model		Perfect Model	
	Values	Brier Score	Values	Brier Score	Values	Brier Score	Values	Brier Score
0%	0.5	0.5	1	1	1	1	0	0
10%	0.5	0.41	0.9	0.73	1	0.81	0.1	0.09
20%	0.5	0.34	0.8	0.52	1	0.64	0.2	0.16
30%	0.5	0.29	0.7	0.37	1	0.49	0.3	0.21
40%	0.5	0.26	0.6	0.28	1	0.36	0.4	0.24
50%	0.5	0.25	0.5	0.25	1	0.25	0.5	0.25
60%	0.5	0.26	0.4	0.28	0	0.36	0.6	0.24
70%	0.5	0.29	0.3	0.37	0	0.49	0.7	0.21
80%	0.5	0.34	0.2	0.52	0	0.64	0.8	0.16
90%	0.5	0.41	0.1	0.73	0	0.81	0.9	0.09
100%	0.5	0.5	0	1	0	1	1	0
Average Brier Score 0% - 100%		0.35		0.55		0.6227		0.15
Average Brier Score 10% - 90%		0.317		0.45		0.539		0.183

Table B.5.: Full Brier scores of perfect. ctatic. inverse and worst Calibration

B.3. More Analysis

Confidence Interval Percentages	Babbage Long Prompt CI Accuracy	Babbage % of 50:2000 CIs	Ada Long Prompt CI Accuracy	Ada % of 50:2000 CIs
0%	28.8	4.4	49.6	44.8
10%	52	38.8	21.2	20
20%	28	10.4	25.2	26
30%	22	8.8	32.4	30.4
40%	14	3.6	39.6	39.2
50%	56.8	54.4	62	76
60%	36	18.4	62.4	59.6
70%	41.2	22.4	56	59.2
80%	67.6	75.6	80.4	74.8
90%	65.6	74	48.4	47.2
100%	32	24.4	58.8	59.2
Correlation Coefficient	0.960		0.960	

Table B.6.: Strong Positive Correlation of 0.96 between Babbage long prompt confidence interval accuracy and % of 50:2000 CIs and strong positive correlation 0f 0.96 between Ada long prompt CI accuracy and % of 50:2000 CIs.

Bibliography

Alzantot, M., Sharma, Y., Elgohary, A., Ho, B.-J., Srivastava, M., and Chang, K.-W. (2018). Generating natural language adversarial examples.

Bai, Y., Jones, A., Ndousse, K., Askell, A., Chen, A., DasSarma, N., Drain, D., Fort, S., Ganguli, D., Henighan, T., Joseph, N., Kadavath, S., Kernion, J., Conerly, T., El-Showk, S., Elhage, N., Hatfield-Dodds, Z., Hernandez, D., Hume, T., Johnston, S., Kravec, S., Lovitt, L., Nanda, N., Olsson, C., Amodei, D., Brown, T., Clark, J., McCandlish, S., Olah, C., Mann, B., and Kaplan, J. (2022). Training a helpful and harmless assistant with reinforcement learning from human feedback.

Branch, H. J., Cefalu, J. R., McHugh, J., Hujer, L., Bahl, A., Iglesias, D. d. C., Heichman, R., and Darwishi, R. (2022). Evaluating the susceptibility of pre-trained language models via handcrafted adversarial examples.

Branwen, G. (2020). Gpt-3 expressing uncertainty; `https://www.gwern.net/GPT-3-nonfiction#calibration`.

Brier, G. W. et al. (1950). Verification of forecasts expressed in terms of probability. *Monthly weather review*, 78(1):1–3.

Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., and Lin, H., editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

Calame, K. (2005). Obsessive genius: The inner world of marie curie. *The Journal of Clinical Investigation*, 115(5):1109–1109.

Carvalho, A. (2016). An overview of applications of proper scoring rules. *Decision Analysis*, 13(4):223–242.

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

Chung, J., Gulcehre, C., Cho, K., and Bengio, Y. (2014). Empirical evaluation of gated recurrent neural networks on sequence modeling.

Copeland, B. J. (2020). The Modern History of Computing. In Zalta, E. N., editor, *The Stanford Encyclopedia of Philosophy*. Metaphysics Research Lab, Stanford University, Winter 2020 edition.

Desai, S. and Durrett, G. (2020). Calibration of pre-trained transformers.

Devlin, J., Chang, M., Lee, K., and Toutanova, K. (2018). BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

Dunning, D., Griffin, D. W., Milojkovic, J. D., and Ross, L. (1990). The overconfidence effect in social prediction. *Journal of personality and social psychology*, 58(4):568.

Ebrahimi, J., Rao, A., Lowd, D., and Dou, D. (2018). Hotflip: White-box adversarial examples for text classification.

Fuegi, J. and Francis, J. (2003). Lovelace & babbage and the creation of the 1843 'notes'. *IEEE Annals of the History of Computing*, 25(4):16–26.

Gao, L. (2021). On the sizes of openai api models.

Ghojogh, B. and Ghodsi, A. (2020). Attention mechanism, transformers, bert, and gpt: Tutorial and survey.

Gneiting, T. and Raftery, A. E. (2007). Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477):359–378.

Guo, C., Pleiss, G., Sun, Y., and Weinberger, K. Q. (2017). On calibration of modern neural networks.

Hendrycks, D., Carlini, N., Schulman, J., and Steinhardt, J. (2021). Unsolved problems in ml safety.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural computation*, 9:1735–80.

Hüllermeier, E., Destercke, S., and Shaker, M. H. (2022). Quantification of credal uncertainty in machine learning: A critical analysis and empirical comparison. In Cussens, J. and Zhang, K., editors, *Proceedings of the Thirty-Eighth Conference on Uncertainty in Artificial Intelligence*, volume 180 of *Proceedings of Machine Learning Research*, pages 548–557. PMLR.

Iyyer, M., Wieting, J., Gimpel, K., and Zettlemoyer, L. (2018). Adversarial example generation with syntactically controlled paraphrase networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1875–1885, New Orleans, Louisiana. Association for Computational Linguistics.

Jiang, Z., Araki, J., Ding, H., and Neubig, G. (2021). How Can We Know When Language Models Know? On the Calibration of Language Models for Question Answering. *Transactions of the Association for Computational Linguistics*, 9:962–977.

Jin, D., Jin, Z., Zhou, J. T., and Szolovits, P. (2019). Is bert really robust? a strong baseline for natural language attack on text classification and entailment.

JRC, E. C. (2022). Definition of coefficient of variation.

Kadavath, S., Conerly, T., Askell, A., Henighan, T., Drain, D., Perez, E., Schiefer, N., Dodds, Z. H., DasSarma, N., Tran-Johnson, E., Johnston, S., El-Showk, S., Jones, A., Elhage, N., Hume, T., Chen, A., Bai, Y., Bowman, S., Fort, S., Ganguli, D., Hernandez, D., Jacobson, J., Kernion, J., Kravec, S., Lovitt, L., Ndousse, K., Olsson, C., Ringer, S., Amodei, D., Brown, T., Clark, J., Joseph, N., Mann, B., McCandlish, S., Olah, C., and Kaplan, J. (2022). Language models (mostly) know what they know.

Khashabi, D., Min, S., Khot, T., Sabharwal, A., Tafjord, O., Clark, P., and Hajishirzi, H. (2020). Unifiedqa: Crossing format boundaries with a single qa system.

Kojima, T., Gu, S. S., Reid, M., Matsuo, Y., and Iwasawa, Y. (2022). Large language models are zero-shot reasoners.

Lampert, C. H., Nickisch, H., and Harmeling, S. (2014). Attribute-based classification for zero-shot visual object categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36(3):453–465.

Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., and Zettlemoyer, L. (2019). Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Lichtenstein, S. and Fischhoff, B. (1980). Training for calibration. *Organizational Behavior and Human Performance*, 26(2):149–171.

Lichtenstein, S., Fischhoff, B., and Phillips, L. D. (1977). *Calibration of Probabilities: The State of the Art*, pages 275–324. Springer Netherlands, Dordrecht.

Lin, S., Hilton, J., and Evans, O. (2022). Teaching models to express their uncertainty in words.

Liu, P., Yuan, W., Fu, J., Jiang, Z., Hayashi, H., and Neubig, G. (2021). Pre-train, prompt, and predict: A systematic survey of prompting methods in natural language processing.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach.

Mannes, A. and Moore, D. (2013). I know i'm right! a behavioural view of overconfidence. *Significance*, 10(4):10–14.

Mielke, S. J., Szlam, A., Dinan, E., and Boureau, Y.-L. (2022). Reducing Conversational Agents' Overconfidence Through Linguistic Calibration. *Transactions of the Association for Computational Linguistics*, 10:857–872.

Nicholl, C. (2004). *Leonardo Da Vinci: The Flights of the Mind*. Allen Lane.

Niculescu-Mizil, A. and Caruana, R. (2005). Predicting good probabilities with supervised learning. In *Proceedings of the 22nd International Conference on Machine Learning*, ICML '05, page 625–632, New York, NY, USA. Association for Computing Machinery.

Nie, Y., Williams, A., Dinan, E., Bansal, M., Weston, J., and Kiela, D. (2019). Adversarial nli: A new benchmark for natural language understanding.

Open Philanthropy Project Calibrate your Judgement, C. T. (2018). Definition of confidence interval from `https://www.openphilanthropy.org/calibration`.

Ouyang, L., Wu, J., Jiang, X., Almeida, D., Wainwright, C. L., Mishkin, P., Zhang, C., Agarwal, S., Slama, K., Ray, A., Schulman, J., Hilton, J., Kelton, F., Miller, L., Simens, M., Askell, A., Welinder, P., Christiano, P., Leike, J., and Lowe, R. (2022). Training language models to follow instructions with human feedback.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *CoRR*, abs/1802.05365.

Pilipiszyn, A. (2021). Gpt-3 powers the next generation of apps.

Puri, R. and Catanzaro, B. (2019). Zero-shot text classification with generative language models.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al. (2019). Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, H. F., Aslanides, J., Henderson, S., Ring, R., Young, S., Rutherford, E., Hennigan, T., Menick, J., Cassirer, A., Powell, R., van den Driessche, G., Hendricks, L. A., Rauh, M., Huang, P., Glaese, A., Welbl, J., Dathathri, S., Huang, S., Uesato, J., Mellor, J., Higgins, I., Creswell, A., McAleese, N., Wu, A., Elsen, E., Jayakumar, S. M., Buchatskaya, E., Budden, D., Sutherland, E., Simonyan, K., Paganini, M., Sifre, L., Martens, L., Li, X. L., Kuncoro, A., Nematzadeh, A., Gribovskaya, E., Donato, D., Lazaridou, A., Mensch, A., Lespiau, J., Tsimpoukelli, M., Grigorev, N., Fritz, D., Sottiaux, T., Pajarskas, M., Pohlen, T., Gong, Z., Toyama, D., de Masson d'Autume, C., Li, Y., Terzi, T., Mikulik, V., Babuschkin, I., Clark, A., de Las Casas, D., Guy, A., Jones, C., Bradbury, J., Johnson, M., Hechtman, B. A., Weidinger, L., Gabriel, I., Isaac, W. S., Lockhart, E., Osindero, S., Rimell, L., Dyer, C., Vinyals, O., Ayoub, K., Stanway, J., Bennett, L., Hassabis, D., Kavukcuoglu, K., and Irving, G. (2021). Scaling language models: Methods, analysis & insights from training gopher. *CoRR*, abs/2112.11446.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. (2019). Exploring the limits of transfer learning with a unified text-to-text transformer. *CoRR*, abs/1910.10683.

Ren, S., Deng, Y., He, K., and Che, W. (2019). Generating natural language adversarial examples through probability weighted word saliency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1085–1097, Florence, Italy. Association for Computational Linguistics.

Reynolds, L. and McDonell, K. (2021). Prompt programming for large language models: Beyond the few-shot paradigm. In *Extended Abstracts of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI EA '21, New York, NY, USA. Association for Computing Machinery.

Roller, S., Boureau, Y.-L., Weston, J., Bordes, A., Dinan, E., Fan, A., Gunning, D., Ju, D., Li, M., Poff, S., Ringshia, P., Shuster, K., Smith, E. M., Szlam, A., Urbanek, J., and Williamson, M. (2020). Open-domain conversational agents: Current progress, open problems, and future directions.

Sanborn, A. and Griffiths, T. (2007). Markov chain monte carlo with people. In Platt, J., Koller, D., Singer, Y., and Roweis, S., editors, *Advances in Neural Information Processing Systems*, volume 20. Curran Associates, Inc.

Schaefer, P. S., Williams, C. C., Goodie, A. S., and Campbell, W. (2004). Overconfidence and the big five. *Journal of Research in Personality*, 38(5):473–480.

Sennrich, R., Haddow, B., and Birch, A. (2015). Neural machine translation of rare words with subword units. *CoRR*, abs/1508.07909.

Sevilla, J., Heim, L., Ho, A., Besiroglu, T., Hobbhahn, M., and Villalobos, P. (2022). Compute trends across three eras of machine learning.

Smith, S., Patwary, M., Norick, B., LeGresley, P., Rajbhandari, S., Casper, J., Liu, Z., Prabhumoye, S., Zerveas, G., Korthikanti, V., Zheng, E., Child, R., Aminabadi, R. Y., Bernauer, J., Song, X., Shoeybi, M., He, Y., Houston, M., Tiwary, S., and Catanzaro, B. (2022). Using deepspeed and megatron to train megatron-turing NLG 530b, A large-scale generative language model. *CoRR*, abs/2201.11990.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., and Polosukhin, I. (2017). Attention is all you need. *CoRR*, abs/1706.03762.

Vig, J. (2019). A multiscale visualization of attention in the transformer model.

Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., and Bowman, S. R. (2018). Glue: A multi-task benchmark and analysis platform for natural language understanding.

Wang, B., Pei, H., Pan, B., Chen, Q., Wang, S., and Li, B. (2019). T3: Tree-autoencoder constrained adversarial text generation for targeted attack.

Wang, W. and Gang, J. (2018). Application of convolutional neural network in natural language processing. In *2018 International Conference on Information Systems and Computer Aided Education (ICISCAE)*, pages 64–70.

Wikipedia (2022). Definition of confidence interval from `https://en.wikipedia.org/wiki/Confidence_interval`.

Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., and Rush, A. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Xian, Y., Lampert, C. H., Schiele, B., and Akata, Z. (2017). Zero-shot learning – a comprehensive evaluation of the good, the bad and the ugly.

Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R. R., and Le, Q. V. (2019). Xlnet: Generalized autoregressive pretraining for language understanding. In Wallach, H., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E., and Garnett, R., editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Zang, Y., Qi, F., Yang, C., Liu, Z., Zhang, M., Liu, Q., and Sun, M. (2020). Word-level textual adversarial attacking as combinatorial optimization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

Zhang, Y., Baldridge, J., and He, L. (2019). Paws: Paraphrase adversaries from word scrambling.