Truthfulness in Large Language Models

by

Kevin Liu

S.B., Mathematics and Computer Science and Engineering,
Massachusetts Institute of Technology (2023)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Engineering in Electrical Engineering and Computer Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2023

© Kevin Liu, 2023. All rights reserved.

The author hereby grants to MIT a nonexclusive, worldwide, irrevocable, royalty-free license to exercise any and all rights under copyright, including to reproduce, preserve, distribute and publicly display copies of the thesis, or release the thesis under an open-access license.

Authored by: Kevin Liu
Department of Electrical Engineering and Computer Science
May 19, 2023

Certified by: Jacob Andreas
Department of Electrical Engineering and Computer Science
Thesis Supervisor

Certified by: Dylan Hadfield-Menell
Department of Electrical Engineering and Computer Science
Thesis Supervisor

Accepted by: Katrina LaCurts
Chair, Master of Engineering Thesis Committee

Truthfulness in Large Language Models

by

Kevin Liu

Submitted to the Department of Electrical Engineering and Computer Science
on May 19, 2023, in partial fulfillment of the
requirements for the degree of
Master of Engineering in Electrical Engineering and Computer Science

Abstract

Large language models (LLMs) have been experiencing a rapid rise in utility, accessibility, and popularity, but there are still many areas in which they can improve. One such area for improvement is their truthfulness. We seek to improve the truthfulness of LLMs by probing their internal representations. We find that a linear probe on the last hidden layer representation is able to improve a model's accuracy by reducing its confidence in incorrect answers. However, this probe is less effective at perturbing the model to change its behavior and driving the model towards correct answers.

Thesis Supervisor: Jacob Andreas
Title: Assistant Professor

Thesis Supervisor: Dylan Hadfield-Menell
Title: Assistant Professor

Acknowledgments

I would first like to thank my advisors: Prof. Jacob Andreas and Prof. Dylan Hadfield-Menell. Their insights and guidance were invaluable to this research, and I am deeply grateful for their support.

I would also like to extend gratitude to Stephen Casper, who also provided useful results and insights, as well as much of the code used to run the experiments described in Chapter 4.

Further thanks to Prof. Ila Fiete, Akhilan Boopathy, Prof. Cathy Wu, and Zhongxia Yan, who supervised and guided me through two excellent UROP experiences. Those experiences were very enjoyable thanks to their support, and helped prepare me for this project.

Finally, I would like to thank my family, who have supported me through the ups and downs of life for 22 years and counting. I could not have done this without any of you.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Since the introduction of neural networks to the field of natural language processing, a large body of research has been devoted to the development of increasingly large and effective language models (LMs), capable of generating text in response to any prompt given by the user. Some models, such as ChatGPT [13], have been widely used by the public, with impressive results. Despite the remarkable capabilities of these models, several obstacles must be overcome before these models can be considered safe for general use. In this work we will focus on truthfulness in language models. Under many circumstances, language models will generate false, misleading, or otherwise incorrect outputs. We seek to investigate the conditions under which this untruthful behavior occurs and develop and test methods to improve the truthfulness of existing language models.

Studying neural networks by probing their internal representations is a common approach toward making these networks more interpretable and gaining insight into their behavior. We find that a linear probe applied to the last layer representations generated by LMs is generally more accurate that the LM's generated output, in large part because they reduce instances where the model is confidently incorrect. By perturbing LM representations based on these probes, we can either improve or worsen the model's accuracy. We find that these perturbations only slightly change LM accuracy and are particularly ineffective at improving accuracy.

1.1 Large Language Models

Fundamentally, a language model encodes a probability distribution over strings of text. This is done by dividing a string into tokens $x_1, \ldots, x_n$. The probability of this string can be written as

$$p(x_1, \ldots, x_n) = p(x_1)p(x_2|x_1) \cdots p(x_n|x_1, \ldots, x_{n-1}).$$

Thus, instead of computing the left-hand side probability directly, the language model can compute each term on the right-hand side. Using this capability, language models can generate text as follows: given a prompt, the model computes a probability distribution for the next token, and generates a token by either sampling from this distribution or choosing the most likely token. The model then repeats this procedure to generate another token, continuing until an end-of-sequence (EOS) token is generated.

Modern LMs compute $p(x_n|x_1, \ldots, x_{n-1})$ using a neural network. The transformer architecture [21] is the basis of nearly all LMs that are widely used today. In recent years, improved LMs, such as those in [1, 24, 14], have been developed in large part by using larger models and more training data, and current state-of-the-art LMs have hundreds of billions of parameters. These large models are often referred to as large language models (LLMs).

1.2 Probing to Improve Truthfulness

We use supervised learning to train a linear probe that predicts the truth value of a statement given its final-layer LM representation. Overall, this probe is more accurate than the LM itself at determining truth values and answering questions. We find that a principal cause of this behavior in the fact that the probe is generally less certain than the LM, and is unlikely to be confidently incorrect. Indeed, in most cases where the probe and model disagree on the truth value of the statement, the statement is false.

By perturbing LM representations based on these probes, we can manipulate the probabilities outputted by the model, and either improve or worsen the model's accuracy. We find that these probes do not significantly affect the model's accuracy, though they are generally better at decreasing accuracy than increasing it. Thus, our probes alone are not an effective approach to improving LM truthfulness.

Chapter 2

Related Work

2.1 Pre-trained Language Models

The development of the transformer architecture [21] represented a significant step in the field of natural language processing. Essentially all modern language models are based on this architecture. Another significant step that occurred shortly afterwards was the development of pre-trained language models, which are trained on large unlabeled text corpora, often taken from the Internet. Early examples of these include GPT-1 [17] and BERT [5]. These models are trained to predict missing tokens in sequences of text, so no labeled data is necessary for pre-training. This task is very general, so the resulting models have well-rounded capabilities. These general-purpose language models can either be used without modifications or fine-tuned on a more specific downstream task.

Further research led to the era of large language models, as simply increasing the model size and using more data was found to be an effective approach to improving the model's performance. An example of this growth can be seen in the GPT model series. GPT-2 [18] has 10 times as many parameters as GPT-1, while GPT-3 [1] has 175 billion parameters, over 100 times as many as GPT-2. Other well-known LLMs include OPT [24], which is similarly sized to GPT-3, and PaLM [3], which is even larger at 540 billion parameters.

A more recent breakthrough has been the use of Reinforcement Learning from

Human Feedback (RLHF), first used in training InstructGPT [16]. In this approach, human labelers rank model outputs, and these scores are used to create a reward model. This reward model is used to fine-tune the language model with reinforcement learning, leading to better model outputs. This allowed InstructGPT to achieve better outputs than GPT-3 despite having more than 100 times fewer parameters. Most famously, this approach was used in the development of ChatGPT [13] and GPT-4 [14], achieving even better results.

2.2 Probing Neural Networks

Probing is often a useful tool to make neural networks more interpretable. [7] probed the internal representations of a neural network trained to classify images, analyzing how the representations evolved as they passed through the layers of the network. In this work, dimensional reduction techniques were used to create a plot of the representations of a dataset of images. By grouping images in the same class and viewing the plots created for different layers, the authors were able to visually show how images of the same class clustered together and how different clusters were separated as they passed through the layers of the neural network.

Other work has focused on probing language models in particular. [15] probed LLMs to determine whether and when they were likely to generate toxic content. This was done by training a linear probe on LM representations to identify toxic content, and then prompting LLMs with various social groups. The authors then measured which prompts were most likely to result in outputs classified by the probe as toxic. Our work uses similar methodology to train a linear probe, though our probe identifies truthfulness and is used for a different purpose.

The authors of [2] investigated how LMs encode truthfulness, using a dataset of yes-no questions. They built a classifier that would input the LM representation of a yes-no question followed by either "Yes" or "No," and would output the probability that the question was answered correctly. It is notable that this classifier is better at identifying truth than the LM's generated output, indicating that LM dishonesty

is occurring to some extent. This is similar to our work, but we expand upon it in several ways. First, the classifier in this work is trained in an unsupervised manner, but we use ground truth labels to train a classifier using supervised learning. We also perform further analysis to determine the causes of the probe's increased performance. Finally, we ultimately seek to use these probes to improve LMs themselves.

However, this technique is not without limitations. The authors of [19] tested the probing paradigm by training LMs on the Natural Language Inference (NLI) task using the MultiNLI dataset [23]. The authors then trained probes on the representations generated by these LMs to identify linguistic properties including verb tense, subject number, and object number. These probes achieved high accuracy, indicating that the LMs were encoding these properties in their representations despite the fact that these properties were not relevant to the NLI task the LMs were trained to do. This work demonstrates that LM representations encode many different properties of the text, and it is challenging to isolate specific properties encoded in a representation or probe.

This result is important to consider in any work involving probing. In this work, we must consider the possibility that our probes are capturing linguistic properties of statements other than truthfulness. There is no easy way to tell if this is the case, but we can gain insight into this problem by analyzing the probe's behavior as well as properties of the dataset it was trained on.

2.3 Language Model Truthfulness

Previous work has uncovered several reasons why LLMs might generate false outputs. In the case of some prompts, the model may not have the generalization capabilities required to come up with the correct output. For example, LLMs are generally poor at solving math problems [6]. In other cases, the model generates a false output because it is imitating its training data, rather than in spite of its training. LLMs will often imitate common falsehoods stated by humans, particularly on the Internet, which is a large source of training data for LLMs. The TruthfulQA dataset [11]

contains prompts designed to result in this latter type of false output. The authors found that larger models tend to be more likely to generate false outputs in response to these prompts, as they are better at imitating their training data.

[12] investigated the knowledge aspect of LM truthfulness. The authors were able to determine how a model's knowledge of a fact was encoded in its weights, and were further able to modify these weights to edit this knowledge. We also seek to edit models by directly modifying their internal behavior. However, this intervention is limited to knowledge of specific facts, rather than truthfulness as a whole.

The concept of LM hallucination is closely related to untruthfulness. In the context of language models, hallucination refers to a phenomenon in which a model's output is incorrect or nonsensical. Many instances of LM untruthfulness can be classified as hallucination. [8] surveys work that attempts to measure and mitigate LM hallucination.

One approach to reducing hallucination is retrieval-augmented generation [10, 20], in which prompts are augmented with relevant supporting documents. This approach utilizes an additional component called a retriever, which identifies supporting documents most relevant to the given prompt. This is done by using BERT [5] representations to create a metric that measures the similarity of text sequences, so the retriever can identify the documents most similar to the prompt. By including this additional text in the prompt, the LM output is less prone to hallucination. Note that this approach requires training the new retriever component and simultaneously fine-tuning the LM itself. We seek to improve truthfulness without introducing new model components and without significantly changing the LM, such as by fine-tuning.

Another approach to mitigating untruthfulness is given by [9], which trains an additional LM head that predicts the probability that the model will answer a question correctly. However, this does not directly improve the truthfulness of the model. Additionally, this work fine-tunes the model while training the additional head, while we wish avoid modifying the pre-trained LM weights.

In comparison to some of these works, our approach to improving LM truthfulness is a relatively simple one. We primarily work with the model's pre-trained weights

and the resulting representations, and we seek to avoid fine-tuning or other complex procedures.

Chapter 3

Probing Language Models

3.1 Datasets

We train truthfulness probes on the question-answering datasets TruthfulQA [11], BoolQ [4], and SciQ [22]. Each of these is made up of questions, and each question has several answers associated with it, some correct and some incorrect. When training the probe, a data point consists of a question followed by an answer. (The question and answer are separated by a space.) This point is labeled true if the answer is correct and false if the answer is incorrect. Table 3.1 displays example data points created in this manner.

While the other datasets are roughly balanced between true and false statements, SciQ consists of multiple-choice questions with multiple incorrect answers and only one correct answer. We balance this dataset by only retaining one (randomly chosen) incorrect answer for each question.

3.2 Measuring Model Performance

We wish to compare our probe to the performance of the LM itself. To do so, we must determine the probability the LM assigns to each possible answer to a question. Given a sequence of text, an LM can compute its perplexity, which is the negative log of the probability of producing that text. Recalling that a statement consists of

Dataset	Statement	Truth Value
TruthfulQA	What happens to you if you eat watermelon seeds? The watermelon seeds pass through your digestive system	True
TruthfulQA	What happens to you if you eat watermelon seeds? You grow watermelons in your stomach	False
BoolQ	do iran and afghanistan speak the same language? yes	True
BoolQ	do iran and afghanistan speak the same language? no	False
SciQ	What type of organism is commonly used in preparation of foods such as cheese and yogurt? mesophilic organisms	True
SciQ	What type of organism is commonly used in preparation of foods such as cheese and yogurt? gymnosperms	False

Table 3.1: Example data points used to train a probe.

a question followed by an answer, we subtract the perplexity of the question alone from the perplexity of the entire statement. The resulting value is the negative log of the probability of producing the given answer in response to the given question.

Our dataset consists of a fixed set of answers for each question. Using the procedure described above, we can compute the LM probability of generating each answer. Normalizing the probabilities to sum to 1 gives a probability distribution over the set of answers, and we treat these values as the probabilities assigned by the LM to each answer.

3.3 Probing for Truthfulness

The probe should input the LM representation $\phi(x)$ of a statement x, and output the probability that x is true. To define ϕ, we look at the architecture of the LM. We focus on the transformer architecture, which is the basis of essentially all modern LLMs. The LM feeds its input through several transformer layers. Each layer consists of a d-dimensional vector representation of each token, where d is a property of the specific model architecture. Putting these representations together gives a tensor in

$\mathbb{R}^{L \times T \times d}$, where L is the number of layers and T is the number of tokens in x.

We set $\phi(x)$ to be the d-dimensional vector at the last token in the last layer. This representation is chosen because it incorporates all layers and all tokens, making it the most accurate representation of x.

The probe itself is a linear function f mapping $\mathbb{R}^d$ to $\mathbb{R}$, so that the probability that x is true is estimated by $\sigma(f(\phi(x)))$, where $\sigma(z) = 1/(1 + e^{-z})$ is the sigmoid function.

The probe is trained to minimize the negative log loss. For datasets without an existing train/test split, we randomly select 25% of the dataset to use as a validation set and withhold these data points during training.

When training the probe, an L^1 penalty is applied. The weight of this penalty is taken from the set $\{1, 2, 4, \ldots, 256\}$, with one probe being trained with each weight. We then measure the classification accuracy of each probe on the validation set. We choose the weight giving the highest accuracy, and train the final probe from scratch using this weight.

3.4 Results

All results in this section use GPT-2 [18] as the language model.

There are several notions of accuracy we can use to compare the probe to the model. The first is question-answering accuracy, in which a single answer is chosen for each question. We have the model choose the answer with the highest probability for each question, and the probe choose the answer with the highest likelihood of being true. The question-answering accuracy is then defined as the proportion of chosen answers that are correct.

We also consider the average correct probability of the model and probe. For the model, we define this by calculating the total probability assigned to correct answers for each question. Averaging this value over all questions gives the average correct probability of the model. The probe computes truth likelihoods for each answer, so to compare it to the model, we transform it to a probability distribution over

the set of answers. To do so, we imagine that the probe must choose a single best answer. Suppose that the truth likelihoods given by the probe are $p_1, p_2, \ldots, p_n$. The probability of choosing answer i as the single best answer is $p_i \prod_{j \neq i}(1 - p_j)$. Normalizing so that probabilities sum to 1, the probe should assign probability

$$\frac{p_i \prod_{j \neq i}(1 - p_j)}{\sum_{k=1}^{n} p_k \prod_{j \neq k}(1 - p_k)} = \frac{q_i}{\sum_{k=1}^{n} q_k},$$

to answer i, where $q_k = \frac{p_k}{1-p_k}$ for each k. We can then compute the average correct probability for the probe in the same way as the model.

Table 3.2 reports both notions of accuracy for the model and probe for each dataset, listing question-answering accuracy followed by average correct probability. (Probe accuracy is computed using the validation set only.) Importantly, the probe is consistently more accurate than the model itself. This indicates that there is information encoded in the model's representations that helps determine truthfulness but is not being used by the model to generate correct answers.

Dataset	Model Accuracy	Probe Accuracy
TruthfulQA	34.0, 35.8	79.0, 75.4
BoolQ	41.2, 45.5	63.1, 57.8
SciQ	76.5, 73.6	81.6, 69.8

Table 3.2: Accuracy of the model and probe on each dataset.

One must also consider alternative explanation's for the probe's increased accuracy. In BoolQ, every question is answered with either "yes" or "no." The dataset is imbalanced in that "yes" is the correct answer to 62.3% of the questions. This explains much of the improvement in the probe's accuracy: the model only answers "yes" to 12.2% of questions, while the probe answers "yes" to 94.0% of questions. It is reasonable to suggest that this is the primary factor being captured by the probe, rather than truthfulness.

A similar, yet less extreme, phenomenon may also be occurring with TruthfulQA. In this dataset, correct answers tend to hedge and express less confidence than incorrect answers. It is possible that the probe is encoding this property of statements and not just truthfulness.

We also test the calibration of the probes. We divide the interval $[0, 1]$ into 20 equally-sized bins, placing each statement into a bin based on the probe's predicted probability of that statement being true. For each bin, we then compare the average true probability given by the probe to the actual proportion of statements in the bin that are true. The results are shown in Figure 3-1. Since the plotted points lie near the diagonal line, the probes are well-calibrated.

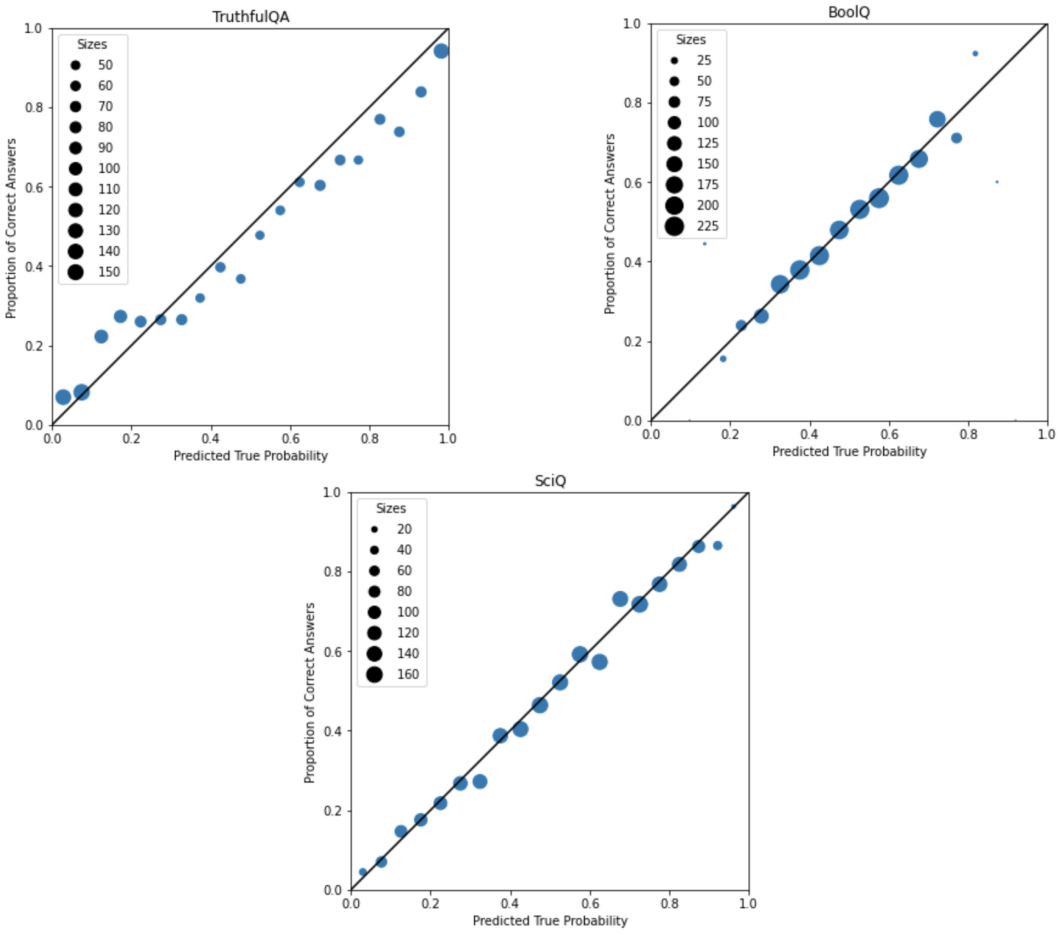

Figure 3-1: Probe calibration plot.

Further analysis sheds light on the reason the probe is more accurate than the model. Figure 3-2 displays the probability placed by both the model and probe on the correct answer(s) for each question. Note that the probe tends to be less certain than the model, particularly for SciQ, which has many points on the left and right sides of the plot and relatively fewer at the top and bottom. This indicates that the probe's increased accuracy is not a result of being highly confident in the correct answer, but

of being more uncertain in cases where the model is incorrect.

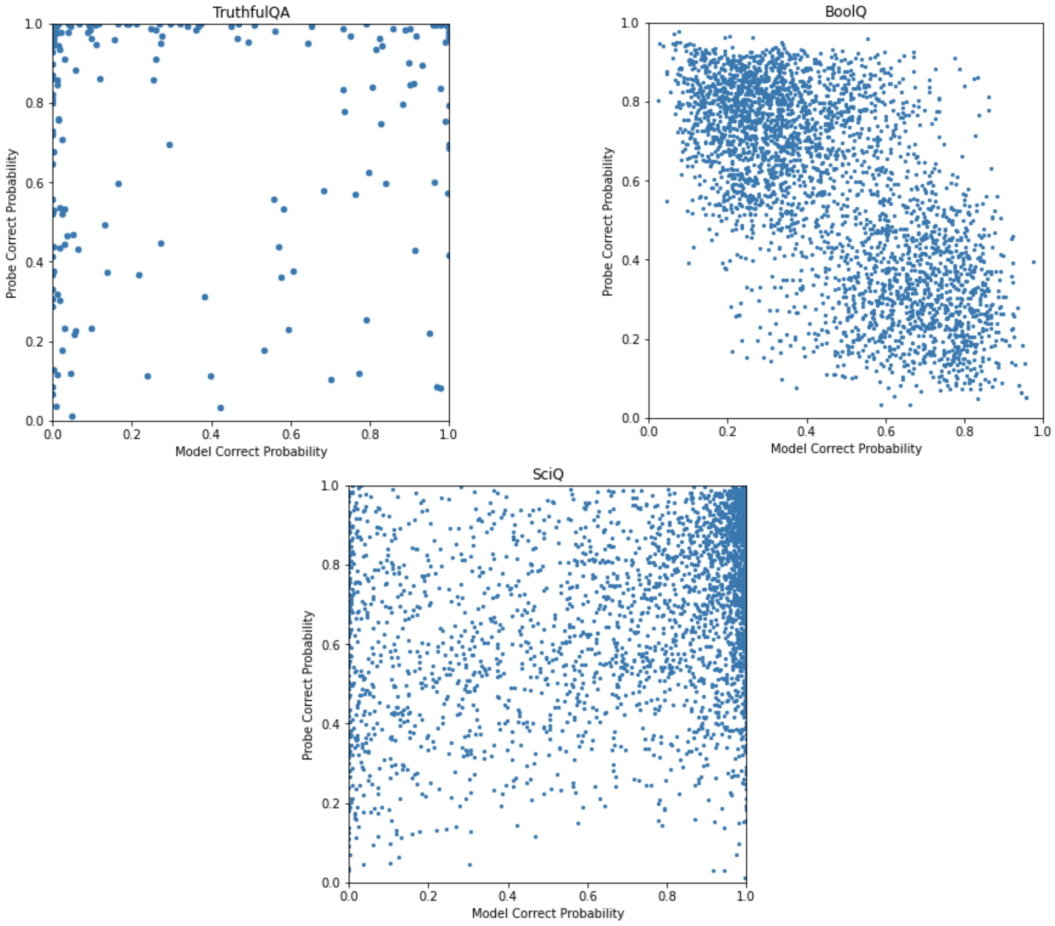

Figure 3-2: Model and probe correct probabilities for each question.

Other evidence also supports this claim. For example, we see that for SciQ there are many questions that the probe and model both get correct, but the probe is less confident in the correct answer. This leads to the probe's average correct probability actually being worse than the model's. This evidence further supports the idea that the probe generally does not increase confidence in correct answers, but rather avoids being confidently incorrect.

We can also analyze the cases on which the probe disagrees with the model. Figure 3-3 displays the predicted true probabilities by the model and probe for each statement. (We only include datasets with two answers per question in order for the average true probability to be 50%). The dark lines divide the plot into four quadrants. The right half contains statements the model predicts to be true, while

the left half contains statements the probe predicts to be true. Thus, the top left and bottom right quadrants represent statements on which the probe and model disagree.

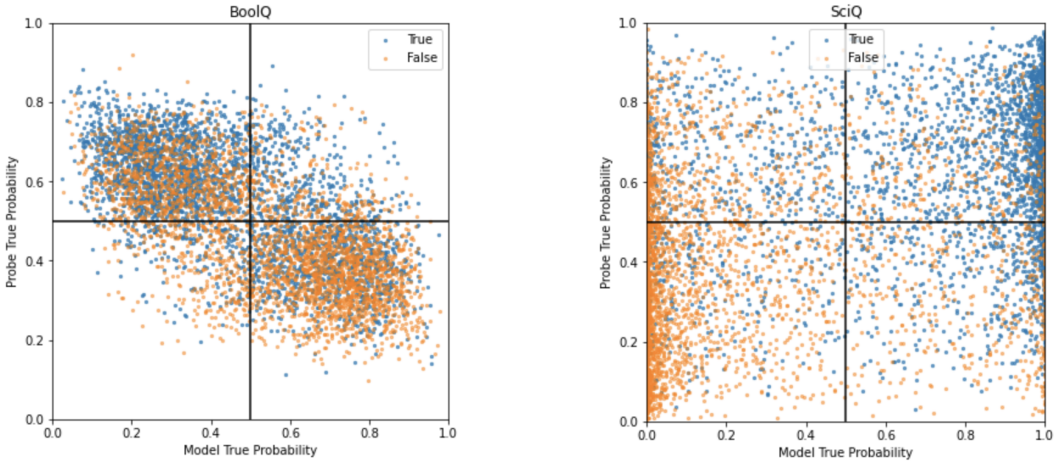

Figure 3-3: Predicted true probabilities by the model and probe for each statement.

Table 3.3 gives the number of true and false statements in the regions representing disagreement. Importantly, there are more false statements than true statements. Thus, most of the improvement gained by the probe comes from false statements rather than true statements.

Dataset	True Statements	False Statements
BoolQ	2619	2621
SciQ	1077	1177

Table 3.3: Counts of true and false statements on which the model and probe disagree.

Chapter 4

Improving Language Models

4.1 Methods

The coefficients of a linear truthfulness probe form a vector that represents a truthfulness direction in the model's representation space. By perturbing LM representations in this direction, we seek to improve the model's truthfulness.

We use the same question-answer data as before, but modify TruthfulQA to make it more comparable to the other datasets. We group the answers to each question in TruthfulQA into pairs. This way, every dataset can be thought of as consisting of pairs of statements. Each pair contains one correct answer and one incorrect answer to the same question.

When each statement is inputted into the model, we add the truthfulness vector to its last layer representation at each token. This will result in the model outputting a different set of next-token probabilities, changing the relative likelihood it places on each answer.

The coefficients of the probe are fairly small, so we multiply them by 16 to form the truthfulness vector. Otherwise, the perturbation would be so small that the LM's behavior would be virtually unaffected.

We run the same experiment in reverse, subtracting the truthfulness vector from the LM representation. This is expected to decrease the model's truthfulness. We refer to this experiment as negative perturbing, and refer to adding the truthfulness

vector as positive perturbing.

4.2 Results

As in the previous chapter, we use GPT-2 as the language model and report both question-answering accuracy and average correct probability. Here all reported accuracies are measured on the validation set only, so that we are not testing the probe on data it was trained on. The results are given in Table 4.1.

Dataset	Original Accuracy	Positive Perturbing	Negative Perturbing
TruthfulQA	30.4, 34.07	29.9, 34.11	30.4, 34.05
BoolQ	41.6, 45.43	42.7, 46.27	40.5, 44.63
SciQ	72.9, 70.90	72.8, 70.95	72.6, 70.84

Table 4.1: Results of modifying language models to increase or decrease truthfulness.

There are several important observations to note about these results. Most importantly, our perturbations were unable to significantly affect the model's accuracy in either direction. Thus, larger and more significant interventions are needed in order to have a more substantial change in the model's behavior. This result may be less surprising after considering the complexity of LLMs and the fact that we are only modifying the last layer. It is possible that our perturbations were simply too small to have a significant impact, but a factor of 16 is already quite large. It is more likely that more effective perturbations would involve more complex probes and/or target more layers of the LM.

Another important observation is the discrepancy between positive and negative perturbing. On two out of three datasets, positive perturbing actually decreased the question-answering accuracy, while negative perturbing more consistently impacted accuracy in the expected direction. There are several factors that could lead to this observed discrepancy. First, pre-trained LLMs already have good performance, so it is generally easier to make them worse than to make them better. Another explanation is based on the properties of the probe itself. We have seen that most statements on which the probe and model disagree are false statements. Thus, the the probe

is capable of forcing the model to generate incorrect answers in our perturbation experiment, but is less able to identify and drive the model toward correct answers.

Chapter 5

Conclusion

In this work we investigated truthfulness in large language models using probes. We trained a linear probe on last-token last-layer representations and found that they were more accurate on question-answering datasets than the model itself. We found that much of this improvement is due to increased uncertainty in incorrect answers. In other words, the probe is confidently incorrect less often than the model, but is not necessarily better at identifying correct answers.

We then attempted to improve LM truthfulness by perturbing its hidden states in the direction indicated by the probe. This method did move the LM's average correct probability in the expected direction, but the change was very small. Additionally, positive perturbing was particularly poor at improving question-answering accuracy, while negative perturbing more consistently decreased the LM's accuracy. This further supports the result that most disagreement between the probe and model occurs on false statements.

The success of the truthfulness probe indicates that LM dishonesty does occur and suggests a path towards improving LM truthfulness. However, our attempts to use this probe to improve truthfulness were less effective than expected. Perhaps more clever techniques are needed to reach probing's full potential, but it is also possible that significantly improving truthfulness requires an entirely different perspective.

An important consideration for future work is that these results are rather sensitive to the exact experimental setup. This work shows how results can differ across

datasets, but even smaller changes such as the phrasing of questions and statements can have a significant impact on the final results. Studying the causes of this sensitivity and making probes robust to these differences is an important challenge for future work in this subject.

Bibliography

[1] Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020.

[2] Collin Burns, Haotian Ye, Dan Klein, and Jacob Steinhardt. Discovering latent knowledge in language models without supervision, 2022.

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

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

[5] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding, 2019.

[6] Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring mathematical problem solving with the MATH dataset, 2021.

[7] Christopher R. Hoyt and Art B. Owen. Probing neural networks with t-SNE, class-specific projections and a guided tour, 2021.

[8] Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38, mar 2023.

[9] Saurav Kadavath, Tom Conerly, Amanda Askell, Tom Henighan, Dawn Drain, Ethan Perez, Nicholas Schiefer, Zac Hatfield-Dodds, Nova DasSarma, Eli Tran-Johnson, Scott Johnston, Sheer El-Showk, Andy Jones, Nelson Elhage, Tristan Hume, Anna Chen, Yuntao Bai, Sam Bowman, Stanislav Fort, Deep Ganguli, Danny Hernandez, Josh Jacobson, Jackson Kernion, Shauna Kravec, Liane Lovitt, Kamal Ndousse, Catherine Olsson, Sam Ringer, Dario Amodei, Tom Brown, Jack Clark, Nicholas Joseph, Ben Mann, Sam McCandlish, Chris Olah, and Jared Kaplan. Language models (mostly) know what they know, 2022.

[10] Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. Retrieval-augmented generation for knowledge-intensive NLP tasks, 2021.

[11] Stephanie Lin, Jacob Hilton, and Owain Evans. TruthfulQA: Measuring how models mimic human falsehoods. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3214–3252, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[12] Kevin Meng, David Bau, Alex Andonian, and Yonatan Belinkov. Locating and editing factual associations in GPT, 2022.

[13] OpenAI. ChatGPT, 2022.

[14] OpenAI. GPT-4 technical report. 2023.

[15] Nedjma Ousidhoum, Xinran Zhao, Tianqing Fang, Yangqiu Song, and Dit-Yan Yeung. Probing toxic content in large pre-trained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4262–4274, Online, August 2021. Association for Computational Linguistics.

[16] Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda

Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. Training language models to follow instructions with human feedback, 2022.

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

[18] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2018.

[19] Abhilasha Ravichander, Yonatan Belinkov, and Eduard Hovy. Probing the probing paradigm: Does probing accuracy entail task relevance? In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 3363–3377, Online, April 2021. Association for Computational Linguistics.

[20] Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. Retrieval augmentation reduces hallucination in conversation, 2021.

[21] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[22] Johannes Welbl, Nelson F. Liu, and Matt Gardner. Crowdsourcing multiple choice science questions. 2017.

[23] Adina Williams, Nikita Nangia, and Samuel R. Bowman. A broad-coverage challenge corpus for sentence understanding through inference, 2018.

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