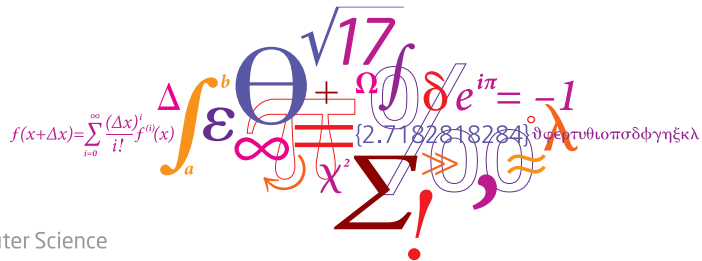
Towards Large Language Models in Educational Technology

Master Thesis Defence

Mikkel Godsk Jørgensen

Technical University of Denmark (DTU)



DTU Compute
Department of Applied Mathematics and Computer Science

Outline

- Introduction
- Language Model
- Transformers
- Memory-Efficient Back-propagation
- Truth representation
- Conclusions
- References
- Errata

Introduction

Formal definition

Uni-directional language model

Given an alphabet Σ, an augmented alphabet $\bar{\Sigma} = \Sigma \cup \{\text{BOS}, \text{EOS}\}$, and a language Σ^*, I define a uni-directional language model as a distribution over the language Σ^*:

$$p_{\text{LM}}(\boldsymbol{y}) = p_{\text{SM}}(\text{EOS}|\boldsymbol{y}) \prod_{t=1}^{\tau} p_{\text{SM}}\left(y_t|\boldsymbol{y}_0^{t-1}\right), \tag{1}$$

where $\boldsymbol{y} = (y_1, y_2, ..., y_\tau) \in \Sigma^*$ is a string, $y_0 = \text{BOS}$, and $p_{\text{SM}}(\,\cdot\,|\,\cdot\,)$ is a sequence model, which models the transition probability to a symbol $y_t \in \bar{\Sigma}$ conditioned on a string $\boldsymbol{y}_0^t \in (\bar{\Sigma})^*$ (belonging to the language of the augmented alphabet). (Cotterell et al. (2023a), Cotterell et al. (2023b))

Alternatively, we could do $L = \text{BOS}\,\Sigma^*\,\text{EOS} \subset \bar{\Sigma}^*$ (left and right cosets) and $p_{\text{LM}} : L \to [0, 1]$, where $p_{\text{LM}}(\boldsymbol{y}) = \prod_{t=1}^{\tau} p_{\text{SM}}(y_t|\boldsymbol{y}_1^{t-1})$, which is more in line with e.g. Bengio et al. (2000).

Tightness

Example of a non-tight model (inspired by Cotterell et al. (2023a, Week 7))

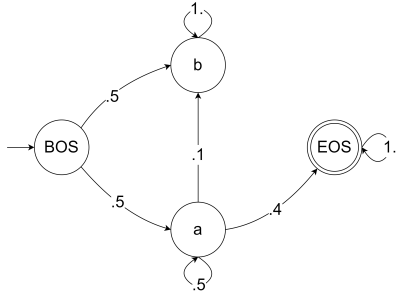


Figure: A sequence-model resulting in a non-tight language model, here depicted as a WFSA where the weights are transition probabilities.

Here we have the alphabet $\Sigma = \{a, b\}$, and a language $\Sigma^* = \{\varepsilon, a, b, aa, ab, ba, bb, aaa, ...\}$. However, the string language (language accepted by the WFSA) is only $L = \{a, aa, aaa, ...\} = aa^*$. We can now compute the total probability of finite strings:

$$\sum_{\boldsymbol{s} \in \Sigma^*} p_{\text{LM}}(\boldsymbol{s}) = \sum_{i=0}^{\infty} p_{\text{SM}}(\text{EOS}|a) p_{\text{SM}}(a|a)^i p_{\text{SM}}(a|\text{BOS})$$

$$= .4 \cdot \frac{1}{1 - .5} \cdot .5 = .4$$

Here the language model "leaks" probability mass into infinite strings, but it does *not* include them or assign non-zero probability to them (because it is undefined on them).

Other things worth mentioning

Decoding

To decode our language model (to find the most likely string), we can use e.g. *greedy decoding*, *sampling*, *beam search* etc.

Continuation

$$
\begin{aligned}
p_{\mathrm{LM}}(\boldsymbol{y}|\boldsymbol{x}) = & p_{\mathrm{SM}}(\mathrm{EOS}|\mathrm{BOS}, x_1, x_2, ..., x_n, y_1, y_2, ..., y_\tau) \\
& \cdot \left(\prod_{t=2}^{\tau} p_{\mathrm{SM}}(y_t|\mathrm{BOS}, x_1, x_2, ..., x_n, y_1, y_2, ..., y_{t-1}) \right) \\
& \cdot p_{\mathrm{SM}}(y_1|\mathrm{BOS}, x_1, x_2, ..., x_n).
\end{aligned} \tag{2}
$$

Training regimen

Using teacher forcing, we typically train the model on a continuation task. Here we do unsupervised pre-training, followed by instruction tuning/supervised fine-tuning.

Transformer architecture

Transformer block

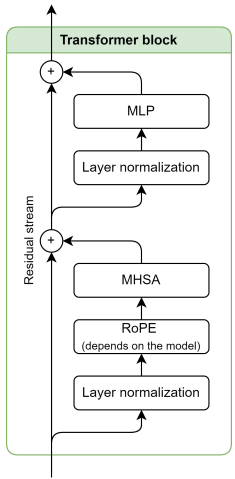


Figure: Inspired by Elhage et al. (2021).

We define multi-head self-attention (MHSA) as:

$$\text{MHSA}(\boldsymbol{X}) = \text{Concat}(\text{head}_1, ..., \text{head}_h)\boldsymbol{W}^O,$$

where

$$\text{head}_i = \text{softmax}\left(\frac{\boldsymbol{Q}_i\boldsymbol{K}_i^\top}{\sqrt{d_k}}\right)\boldsymbol{V}_i,$$

$$\boldsymbol{Q}_i = \boldsymbol{X}\boldsymbol{W}_i^Q,$$
$$\boldsymbol{K}_i = \boldsymbol{X}\boldsymbol{W}_i^K,$$
$$\boldsymbol{V}_i = \boldsymbol{X}\boldsymbol{W}_i^V.$$

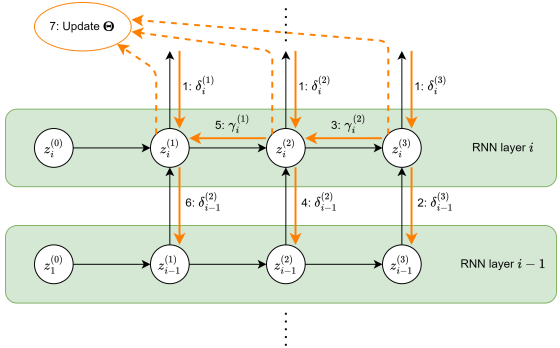
Here $\boldsymbol{X} \in \mathbb{R}^{\text{seq\_length} \times \text{hidden\_dim}}$, i.e. the representation for the t'th symbol is in the t'th row.

Memory-Efficient Back Propagation

Memory-efficient back propagation idea

Eagerly apply the gradients as they become available, instead of storing them all in memory. The parameters should be used with high temporal locality to utilize the method fully.

Example of memory-efficient back propagation on an RNN



We compute the Jacobian $\boldsymbol{J}_{\boldsymbol{\Theta}}F$ using the chain rule, meticulously summing over the Bauer paths using distributivity:

$$\boldsymbol{J}_{\boldsymbol{\Theta}}F = \sum_{j=1}^{3} (\boldsymbol{J}_{\boldsymbol{z}_i^{(j)}}F)(\boldsymbol{J}_{\boldsymbol{\Theta}}\boldsymbol{z}_i^{(j)}) \tag{3}$$

$$\boldsymbol{J}_{\boldsymbol{z}_i^{(i)}}F = \boldsymbol{\delta}_i^{(i)} + \gamma_i^{(i)} \tag{4}$$

$$\boldsymbol{\delta}_i^{(j)} = (\boldsymbol{J}_{\boldsymbol{z}_{i+1}^{(j)}}F)(\boldsymbol{J}_{\boldsymbol{z}_i^{(j)}}\boldsymbol{z}_{i+1}^{(j)}) \tag{5}$$

$$\gamma_i^{(j)} = (\boldsymbol{J}_{\boldsymbol{z}_i^{(j+1)}}F)(\boldsymbol{J}_{\boldsymbol{z}_i^{(j)}}\boldsymbol{z}_i^{(j+1)}) \tag{6}$$

Results in practice

Here training a LLaMa-2-7b model on the first 96 instances of the LIMA dataset, with a batch size of 16 and a maximum sequence length of 1024.[1]

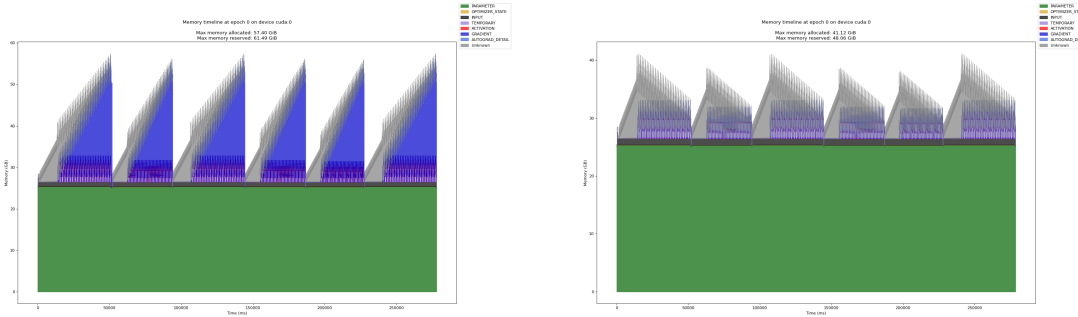


Figure: Normal back propagation vs MEBP visualized with the PyTorch memory profiler. The dark blue are the gradients. The green chunk represents the model parameters and is of the same size. Here we go from 61.49 GiB to 48.06 GiB reserved memory (57.40 GiB to 41.12 GiB allocated memory).

[1] The figures are A.13 and A.14 in the report.

Detectability of truth, and misalignment with model outputs I

Motivation

Marks and Tegmark (2023); Agrawal et al. (2023); Azaria and Mitchell (2023); Schulman (2023) etc. find evidence that large language models may have internal representations representing truth, and claim that that may "be aware" if they are "lying". I.e., it is possible that the models may sometimes incorrectly express their knowledge (also e.g. Li et al. (2024)). Lastly, Zhou et al. (2024) suggests that the model does not internalize new knowledge during alignment, but only learns how to respond in their *Superficial Alignment Hypothesis*.

Experiment

I fit a linear probe to the residual stream activations inputted to transformer block 15 in the LLaMa-2-13b model, as suggested in Marks and Tegmark (2023), and evaluate its accuracy in a 5-fold CV. I compare the probe performance with the model continuation likelihoods of starting the response with a "Yes" or "No", and a naïve baseline. I also attempted to get a few-shot prompted GPT-3.5-turbo-0125 to parse the model generation. I also compare the two versions of the LLaMa-2-13b model: One that has only been pre-trained, and one that has been fine-tuned for chat.

Detectability of truth, and misalignment with model outputs II

Results

Dataset	Model	Yes/No CL	Linear probe	GPT parser	GPT parser baseline	# Rejects
cities	Pretrain	[76%, 83%]	[98%, 100%]	[98%, 100%]	100%	391 of 500
common claims	Pretrain	[58%, 67%]	[67%, 75%]	[65%, 82%]	71%	407 of 500
counterfactuals	Pretrain	[60%, 69%]	[64%, 72%]	[81%, 100%]	92%	488 of 500
Politicians	Pretrain	[49%, 57%]	[60%, 69%]	[62%, 94%]	73%	478 of 500
cities	Chat	[67%, 75%]	[98%, 100%]	[72%, 80%]	53%	29 of 500
common claims	Chat	[73%, 80%]	[70%, 78%]	[71%, 79%]	52%	31 of 500
counterfactuals	Chat	[63%, 71%]	[64%, 72%]	[63%, 71%]	50%	57 of 500
Politicians	Chat	[46%, 55%]	[59%, 67%]	[44%, 56%]	51%	233 of 500

Table: Listed are the 95% Jeffrey intervals for accuracy. Here *CL* is an abbreviation for *continuation likelihood*, where I label using the rule $p(\texttt{Yes}|q) > p(\texttt{No}|q)$. The linear probe is a logistic regression (not an SVM). The dataset baseline accuracies are 50%.

Detectability of truth, and misalignment with model outputs III

Differences

	Cities	Common claims	Counterfactuals	Politicians
$\text{Probe}_{\text{Pretrain}}$ vs $\text{CL}_{\text{Pretrain}}$	[17%, 24%]	[4%, 14%]	[-2%, 9%]	[6%, 17%]
$\text{Probe}_{\text{Chat}}$ vs CL_{Chat}	[24%, 32%]	[-7%, 2%]	[-4%, 7%]	[7%, 19%]
$\text{Probe}_{\text{Chat}}$ vs $\text{Probe}_{\text{Pretrain}}$	[-1%, 0%]	[-0%, 6%]	[-3%, 4%]	[-5%, 2%]
CL_{Chat} vs $\text{CL}_{\text{Pretrain}}$	[-14%, -2%]	[9%, 19%]	[-4%, 9%]	[-10%, 6%]

Table: Listed are the 95% CIs for the expected difference in accuracies. The intervals are computed using non-parametric bootstrapping in a paired setup.

Takeaways

I found some indications supporting the narrative that the model does not fully express its knowledge, but is not sufficient to draw firm conclusions. The linear predictability of truth does not seem to change during fine-tuning for chat, which is in line with Zhou et al. (2024)'s superficial alignment hypothesis.

Linear vs. non-linear representation (CAV vs CAR) I

Motivation

Crabbé and van der Schaar (2022) find that concepts in deep neural networks may not represent concepts linearly, as otherwise proposed in Kim et al. (2018). Marks and Tegmark (2023) find that the structure seems linear (true and false statements are linearly separable) in the chosen layer.

Experiment

To test the linearity of the truth representation, I use a linear SVM and an SVM with RBF kernel to probe the model on TruthfulQA. Here I optimize hyperparameters (layer, head, γ) using Bayesian optimization through 1000 trials. The objective is the validation accuracy. Using the optimally found hyperparameters, I re-train the model on the training and validation sets and evaluate the test set. I also test OOD performance.

Linear vs. non-linear representation (CAV vs CAR) II

Results

	TruthfulQA	Politicians	cities	neg-cities	common claims
Baseline	55%	78%	50%	50%	56%
LinearSVM (OOD)	[71%, 72%]	[49%, 50%]	[49%, 51%]	[50%, 52%]	[48%, 50%]
NonlinearSVM (OOD)	[75%, 76%]	[48%, 50%]	[48%, 50%]	[48%, 50%]	[51%, 52%]
LinearSVM (ID)	[71%, 73%]	[81%, 82%]	[88%, 90%]	[89%, 90%]	[65%, 67%]
NonlinearSVM (ID)	[75%, 77%]	[84%, 85%]	[88%, 90%]	[87%, 89%]	[66%, 68%]

Table: Aggregated 95% Jeffrey intervals for accuracies. TruthfulQA is always in-distribution.

	TruthfulQA	Politicians	cities	neg-cities	common claims
Nonlinear - linear (OOD)	[3%, 5%]	[-2%, -0%]	[-2%, 0%]	[-3%, -2%]	[2%, 3%]
Nonlinear - linear (ID)	[3%, 5%]	[2%, 3%]	[-1%, 1%]	[-2%, -0%]	[1%, 2%]

Table: Aggregated 95% confidence intervals of differences in accuracy, computed with non-parametric bootstrapping in a paired setup.

Linear vs. non-linear representation (CAV vs CAR) III

Takeaways

Although the non-linear probe sometimes performs slightly better on in-distribution data, I do not see evidence that we would strongly benefit from using a non-linear probe over a linear one.

Inference-Time Intervention I

Motivation

Li et al. (2024) finds that you can improve the performance of the LLaMa-2-7b model on TruthfulQA by editing the activations on specific attention heads. Here they shift the activations by adding a vector.[a] The vector is a scaled version of the difference in means between the activations for true statements and false statements. The scale is determined as a fixed number of standard deviations of the statement activations, along the vector by projection.

[a] The authors do not clearly specify whether they intervene on the last token or all. I intervene on all.

Experiment

I split TruthfulQA into 3 partitions stratified based on the number of true and false choices, and categories. The first partition is used to rank each attention head with the detectability along its mass-mean direction using 5-fold CV. The probe is retrained on this partition after the ranking. The second partition is then used to tune the hyperparameters α, being the number of standard deviations we shift the activations, and K being the number of (best-scoring) heads we intervene on. The third partition is then used to evaluate the model. I also evaluate on other datasets as OOD.

Inference-Time Intervention II

Evaluation metric (MCCLA)

Here I use the *Multiple-Choice Continuation Likelihood Accuracy*. For each question q_i, we get a set of possible responses R_i of which a subset is correct $C_i \subset R_i$. For each question, we then compute the probability of a true response:

$$p_{\text{true}}(q_i) \overset{def}{=} \Pr[r_{\text{LM}} \in C_i \,|\, r_{\text{LM}} \in R_i \,, q_i] = \frac{\Pr[r_{\text{LM}} \in C_i | q_i]}{\Pr[r_{\text{LM}} \in R_i | q_i]} = \frac{\sum_{r \in C_i} p_{\text{LM}}(r|q_i)}{\sum_{r \in R_i} p_{\text{LM}}(r|q_i)} \tag{7}$$

We can then compute the MCCLA as the average $p_{\text{true}}(q_i)$ taken over the dataset. This is equivalent to Lin et al. (2021); Gao et al. (2021), but slightly rewritten. Here they refer to it as a *likelihood*, although that may be a bit controversial, and is not how it is defined in e.g. Bishop and Nasrabadi (2006).

Inference-Time Intervention III

MCCLA baseline

The report had a few errors I take the liberty to correct here. I define a baseline for the MCCLA based on a random probability assignment for each response in R_i:

$$\boldsymbol{\pi}|q_i \sim \text{Dirichlet}(\alpha \cdot \mathbf{1}_{1\times|R_i|}), \quad \alpha \in \mathbb{R} \tag{8}$$

$$p_{\text{true,baseline}}(q_i) \overset{def}{=} \frac{\sum_{r=0}^{|C_i|}(\boldsymbol{\pi}|q_i)_r}{\sum_{r=0}^{|R_i|}(\boldsymbol{\pi}|q_i)_r} \tag{9}$$

$$\text{MCCLA}_{\text{baseline}} = \mathbb{E}_{(q_i,R_i,C_i)\in\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{\pi}|q_i}\left[p_{\text{true,baseline}}(q_i)|q_i,R_i,C_i\right]\right]. \tag{10}$$

Here we assume the correct responses always have lower index than the incorrect ones. It can be more explicitly written as:

$$\boldsymbol{\pi}|R_i \sim \text{Dirichlet}(\alpha \cdot \mathbf{1}_{1\times|R_i|}), \quad \alpha \in \mathbb{R}$$

$$p_{\text{true,baseline}}(C_i,R_i) \overset{def}{=} \frac{\sum_{r=1}^{|C_i|}(\boldsymbol{\pi}|R_i)_r}{\sum_{r=1}^{|R_i|}(\boldsymbol{\pi}|R_i)_r} \tag{11}$$

$$\text{MCCLA}_{\text{baseline}} = \mathbb{E}_{(q_i,R_i,C_i)\in\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{\pi}|R_i}\left[p_{\text{true,baseline}}(C_i,R_i)|(q_i,R_i,C_i)\right]\right]$$

Inference-Time Intervention IV

Results: Improvements by ITI MCCLA

	TruthfulQA partition 3	common claim	counterfactuals
LLaMa-2-7b-hf	[13.55%, 20.74%]	[-1.36%, -0.64%]	[-2.23%, -0.62%]
LLaMa-2-7b-chat-hf	[1.05%, 12.68%]	[-13.91%, -10.04%]	[-20.72%, -15.94%]
Meta-LLaMa-3-8B	[17.77%, 28.58%]	[-0.55%, -0.04%]	[-21.46%, -17.31%]
Meta-LLaMa-3-8B-Instruct	[0.80%, 11.67%]	[-9.13%, -6.51%]	[-24.21%, -19.72%]
Mistral-7B-Instruct-v0.2	[-3.93%, -1.35%]	[-5.02%, -2.93%]	[-0.29%, -0.00%]
Mistral-7B-Instruct-v0.3	[-3.38%, -1.24%]	[-7.57%, -5.56%]	[-0.26%, 0.00%]
Mistral-7B-v0.3	[3.26%, 13.97%]	[-7.70%, 1.04%]	[-36.58%, -31.63%]
Mixtral-8x7B-v0.1	[-1.31%, 9.69%]	[-6.20%, -3.23%]	[-36.51%, -31.90%]
Mixtral-8x7B-Instruct-v0.1	[-6.52%, -2.40%]	[-9.52%, -6.78%]	[-0.28%, 0.05%]
opt-2.7b	[9.23%, 18.52%]	[-3.80%, 2.84%]	[-10.04%, -6.97%]
opt-125m	[4.58%, 13.88%]	[-3.89%, 4.17%]	[-11.17%, -6.61%]
opt-350m	[7.54%, 15.84%]	[-4.62%, 4.32%]	[-13.88%, -9.32%]
Phi-3-mini-4k-instruct	[-2.40%, 0.91%]	[0.66%, 4.45%]	[0.06%, 0.90%]

Inference-Time Intervention V

Results: Improvements by ITI MCCLA

	cities	neg-cities	politicians	$(\alpha, K_{\text{frac}})$
LLaMa-2-7b-hf	[-0.16%, -0.03%]	[-19.35%, -16.52%]	[-1.87%, -1.08%]	(8,2%)
LLaMa-2-7b-chat-hf	[-3.42%, -1.90%]	[-2.53%, 3.11%]	[-10.24%, -8.23%]	(8,4%)
Meta-LLaMa-3-8B	[-17.98%, -13.92%]	[-0.73%, 1.91%]	[-5.82%, -4.20%]	(8,8%)
Meta-LLaMa-3-8B-Instruct	[-22.72%, -18.39%]	[14.91%, 19.10%]	[-7.37%, -5.45%]	(8,4%)
Mistral-7B-Instruct-v0.2	[-0.00%, 0.00%]	[-1.03%, 0.06%]	[0.26%, 0.49%]	(1,2%)
Mistral-7B-Instruct-v0.3	[-0.01%, 0.00%]	[3.12%, 4.04%]	[0.09%, 0.22%]	(1,4%)
Mistral-7B-v0.3	[-38.43%, -33.07%]	[17.83%, 22.76%]	[-19.32%, -17.10%]	(16,4%)
Mixtral-8x7B-v0.1	[-40.60%, -35.19%]	[17.76%, 22.39%]	[-19.54%, -17.34%]	(16,6%)
Mixtral-8x7B-Instruct-v0.1	[-0.00%, 0.00%]	[-0.35%, 0.66%]	[0.07%, 0.32%]	(1,10%)
opt-2.7b	[-11.00%, -7.98%]	[5.00%, 7.59%]	[-5.35%, -3.96%]	(8,10%)
opt-125m	[-22.12%, -16.29%]	[11.04%, 15.68%]	[-8.94%, -7.19%]	(16,8%)
opt-350m	[-21.08%, -15.04%]	[3.58%, 9.46%]	[-12.15%, -10.31%]	(8,10%)
Phi-3-mini-4k-instruct	[0.02%, 0.17%]	[-4.32%, -3.08%]	[-0.33%, 0.07%]	(1,2%)

Inference-Time Intervention VI

Average MCCLA of pre-trained vs fine-tuned models

	TruthfulQA partition 3	common claim	counterfactuals	cities	neg-cities	politicians
Pre-train	[37.56%, 42.20%]	[51.50%, 54.07%]	[86.89%, 88.44%]	[99.49%, 99.75%]	[26.42%, 28.68%]	[64.67%, 65.62%]
Fine-tuned	[51.93%, 57.16%]	[64.49%, 66.79%]	[87.69%, 89.27%]	[99.35%, 99.68%]	[20.79%, 22.81%]	[65.33%, 66.29%]

Takeaways

- Inference-Time Intervention does not seem to generalize well to OOD datasets.
- Generally, the fine-tuned models tend to score higher on TruthfulQA and Common claims than the base (only pre-trained) counterparts, without using ITI.
- The pre-trained models seem to improve significantly more on TruthfulQA.
- I do not see convincing evidence that this method will also tend to work for Instruct models.

Conclusions

1. I have covered Memory-Efficient Back-propagation which only seems to come with the drawback that we cannot do gradient normalization (including `clip_grad_norm` in PyTorch).
2. There does indeed seem to be a linear structure to the truth representation.
3. It seems that the models may sometimes contain more knowledge than they "know" how to express.
4. It is unclear whether this is always the case, e.g. for instruct models I mostly see quite bad results for ITI on TruthfulQA. But they also mostly score higher by default.
5. To come back to the context of educational technology, since ITI does not seem to generalize well out of distribution, it is unclear whether this model would work well in a production setting.
6. In case we would instead want to train a model from scratch, the MEBP algorithm could offer a tiny contribution to democratizing this.

References I

Agrawal, A., Mackey, L., and Kalai, A. T. (2023). Do language models know when they're hallucinating references? *arXiv preprint arXiv:2305.18248*.

Azaria, A. and Mitchell, T. (2023). The internal state of an llm knows when its lying. *arXiv preprint arXiv:2304.13734*.

Bau, D., Zhu, J.-Y., Strobelt, H., Lapedriza, A., Zhou, B., and Torralba, A. (2020). Understanding the role of individual units in a deep neural network. *Proceedings of the National Academy of Sciences*, 117(48):30071–30078.

Bengio, Y., Ducharme, R., and Vincent, P. (2000). A neural probabilistic language model. *Advances in neural information processing systems*, 13.

Bishop, C. M. and Nasrabadi, N. M. (2006). *Pattern recognition and machine learning*, volume 4. Springer.

Cotterell, R. et al. (Fall 2023a). Natural language processing. The NLP course a ETH Zürich offered in fall 2023. The resources are openly available on the course website - accessed: February 8th 2024.

References II

Cotterell, R., Svete, A., Malagutti, L., Meister, C., Liu, T., Zouhar, V. , and Du, L. (August 6, 2023b). Formal aspects of language modeling. See "LLM Course Notes Part 1" (Openly available on website - last accessed: February 8th 2024).

Crabbé, J. and van der Schaar, M. (2022). Concept activation regions: A generalized framework for concept-based explanations. *Advances in Neural Information Processing Systems*, 35:2590–2607.

Elhage, N., Nanda, N., Olsson, C., Henighan, T., Joseph, N., Mann, B., Askell, A., Bai, Y., Chen, A., Conerly, T., et al. (2021). A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 1(1):12.

Gao, L., Tow, J., Biderman, S., Black, S., DiPofi, A., Foster, C., Golding, L., Hsu, J., McDonell, K., Muennighoff, N., Phang, J., Reynolds, L., Tang, E., Thite, A., Wang, B., Wang, K., and Zou, A. (2021). A framework for few-shot language model evaluation.

Goodfellow, I., Bengio, Y., and Courville, A. (2016). *Deep Learning*. MIT Press. `http://www.deeplearningbook.org`.

References III

Kim, B., Wattenberg, M., Gilmer, J., Cai, C., Wexler, J., Viegas, F., et al. (2018). Interpretability beyond feature attribution: Quantitative testing with concept activation vectors (tcav). In *International conference on machine learning*, pages 2668–2677. PMLR.

Li, K., Patel, O., Viégas, F., Pfister, H., and Wattenberg, M. (2024). Inference-time intervention: Eliciting truthful answers from a language model. *Advances in Neural Information Processing Systems*, 36.

Lin, S., Hilton, J., and Evans, O. (2021). Truthfulqa: Measuring how models mimic human falsehoods. *arXiv preprint arXiv:2109.07958*.

Marks, S. and Tegmark, M. (2023). The geometry of truth: Emergent linear structure in large language model representations of true/false datasets. *arXiv preprint arXiv:2310.06824*.

Schulman, J. (April 2023). Reinforcement learning from human feedback: Progress and challenges. Last accessed: February 9th, 2024.

References IV

Zhou, C., Liu, P., Xu, P., Iyer, S., Sun, J., Mao, Y., Ma, X., Efrat, A., Yu, P., Yu, L., et al. (2024). Lima: Less is more for alignment. *Advances in Neural Information Processing Systems*, 36.

Errata for the report I

1. (p. 9 and p. 12) To clarify: Language models are probability distributions over finite strings (at least ideally). Saying they could "include" or have "zero probability of infinite strings" is inaccurate. I meant that they do not "leak" probability mass into the set of "infinite strings" by getting stuck.
2. (p. 12) In the *Decoding* section, I use the term "probable". I think it should be replaced with "likely", i.e. "the most likely string", "the most likely token" and "the top-k most likely sequences". That terminology is also used in Lin et al. (2021) and Gao et al. (2021) (LM Evaluation Harness), although it is not entirely consistent with how likelihood is often defined e.g. as in Bishop and Nasrabadi (2006); Goodfellow et al. (2016) (where the likelihood is a function of the parameters).
3. (p. 15) $\bar{\Sigma} = \Sigma \cup \{\text{BOS}, \text{EOS}\}$
4. (p. 18) In the TruthfulQA continuation likelihood definition, I sometimes use r and sometimes r_{LM}. For consistency, they should all be referred to as either r or r_{LM}.

Errata for the report II

5. (p. 18) "The orthogonality could otherwise have allowed for single-neuron representations of concepts." I think this claim should have been marked more clearly as speculation, but it has been discussed in the literature whether single neurons code for individual concepts e.g. in Bau et al. (2020). Here I mean that if we have single-neuron representations of concepts, then those concepts must be orthogonally represented.
6. (p. 21) I believe the term "probability of generation" may be more accurate here than "generation likelihood".
7. (p. 27) In equations 3.21 and 3.24, replace $b^{(i)}$ with $b^{(1)}$.
8. (p. 35) I use GPT-3.5-turbo-0125, not GPT-3.5-turbo-1025.
9. (p. 36) The DTU course *02477 Bayesian Machine Learning* should have been cited for the Dirichlet distribution.
10. (p. 36) In model 3.4.1, we form the Dirichlet distribution using R_i, not C_i, as we want it to be a no-preference distribution over all possible responses.

Errata for the report III

11 (p. 36) In equation 3.51, to keep the notation consistent with the MCCLA, I denote $p_{\text{true,baseline}}(q_i)$ as a function of q_i, although it would actually be a function of R_i, and C_i (which we can look up and find once we know q_i). Using this "implicit" notation, equation 3.51 should further be corrected to $\text{TruthfulQA}_{\text{baseline}} = \mathbb{E}_{(q_i,R_i,C_i)\in\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{\pi}|q_i,}\left[p_{\text{true,baseline}}(q_i)|q_i\right]\right]$, and could even further be put on a more explicit form:

$$
\begin{aligned}
\boldsymbol{\pi}|R_i &\sim \text{Dirichlet}(\alpha \cdot \mathbf{1}_{1\times|R_i|}), \qquad \alpha \in \mathbb{R} \\
p_{\text{true,baseline}}(C_i, R_i) &\overset{def}{=} \frac{\sum_{r=1}^{|C_i|}(\boldsymbol{\pi}|R_i)_r}{\sum_{r=1}^{|R_i|}(\boldsymbol{\pi}|R_i)_r} \\
\text{TruthfulQA}_{\text{baseline}} &= \mathbb{E}_{(q_i,R_i,C_i)\in\mathcal{D}}\left[\mathbb{E}_{\boldsymbol{\pi}|R_i}\left[p_{\text{true,baseline}}(C_i,R_i)|(q_i,R_i,C_i)\right]\right]
\end{aligned}
\tag{12}
$$

12 (p. 36) It should say $\text{MCCLA}_{\text{baseline}}$ instead of $\text{TruthfulQA}_{\text{baseline}}$.

13 (p. 38) In section 3.4.4, I write that I let $\mathcal{I} \subset \mathbb{R}^2$ be a set of coordinates. It should have read $\mathcal{I} \subset \mathbb{N}^2$.

Errata for the report IV

14. (p. 55) I do not quantize all OPT models. Only the small OPT-125m since quantization adds a layer of complexity in debugging. I have been using this model throughout to develop and test the code on smaller hardware.

15. (p. 43) In the RNN section, it should have read: "For any practical purposes, training a machine learning model this big is quite insensible for this particular dataset,"

16. (p. 60) "However, the success of applying inference-time intervention (in section 4.2.3) to pre-trained models (i.e. the versions that were not finetuned), could indicate that the claim is true, at least for these." should have read "However, the success of applying inference-time intervention (in section 4.2.3) to pre-trained models (i.e. the versions that were not finetuned), could indicate that the claim is true, at least for TruthfulQA."

Things that could have been done differently in the experiments

1. The politicians dataset was created with a class imbalance, and then class balanced. It would have been better to strictly adhere to algorithm 4 in section 3.3 to avoid this. In trying to fix this, I ended up with a dataset that contained a lot of lesser-known politicians, which I deemed a slightly unfair task. The evaluations on this second one are in the appendix of the report for transparency (as I originally used it for the first run of the experiment in section 4.2.2).
2. For the experiment with the linear vs non-linear probe, I took the first 4000 instances of each dataset. Here I introduce a class imbalance in common claims, because it was sorted and approximately 4400 instances long. Ideally, the dataset should have been shuffled first.
3. For inference-time intervention, I think it would have been better to use the same partitions across all runs.