Lecture 9: Hallucinations and How to Mitigate Them

Lecturer: Adam Kalai **Date:** March 5, 2026
Scribes: Liza Horokh, Jacopo Rizzo

1. Introduction

Today's lecture covers three topics:

- Why do hallucinations arise (next-word pre-training)
- Why hallucinations persist (evaluation incentives)
- Correcting incentives

Definition 1.1. A *hallucination* is false information that a language model outputs.

A query of "What does PGGB stand for?" in Claude Sonnet 4 generated candidates such as "Pacific Gas and Electric Company (PG&E)" and "Post Graduate Guidance Bureau," the former being clearly incorrect, and the latter of which does not exist.

Why are hallucinations particularly concerning? False information can lead to greater consequences than saying "I don't know" on uncertainty. In the not-so-distant past, hallucinations have led to models outputting nonexistent court cases and misrepresenting law [3].

Hallucinations are a particular instance of language model error. What separates them from other types of errors, such as spelling or grammar mistakes?

- Learning the rules of grammar for a next-word predictor is quantitatively easier than learning to arbitrarily predict birthdays, for instance, since grammar has a more "succinct" description with lower VC dimension.
- On the other hand, the natural hypothesis class $\mathcal{H}$ for instances $\{(\text{person}_i, \text{possible birthday})\}_i$ such that for each $h \in \mathcal{H}$ and person, $h(\text{person}, j) = 1$ for exactly one date j, can shatter a set of N instances for large N, implying a large VC dimension.

Where do hallucinations come from? We first briefly outline how LLMs are structured.

Building LLMs

- Step 1 (**Pre-training**): Using a large training set, obtain a *next-word predictor*, that predicts a next token output given current tokens. Uses unsupervised learning methods such as density estimation.
- Step 2 (**Base Model**): Using this next-word predictor, a base model can be established: namely, an initial chatbot consists of repeatedly applying the next-word predictor to formulate its output.
- Step 3 (**Post-training**): Refining the base model to achieve alignment through methods like Reinforcement Learning from Human Feedback (RLHF). An important goal is to reduce hallucinations, such as by expressing uncertainty in a given output.

Remark 1.1. For purposes of this lecture, we focus on pre-training on a noiseless distribution. While this is unrealistic, with noise one should reasonably expect higher rates of errors and hallucinations: hallucinations based on the training data mistakes are more predictable and so less interesting. Moreover, the purpose of assuming noiseless data is to show that hallucinations are not solely the result of noise.

2. Pre-training & Next-Word Prediction

We first discuss how hallucinations can arise from pre-training. Let $\mathcal{X}$ be a set of statements, i.e. of strings or claims like "Albert Einstein was born 03-14", and partition $\mathcal{X} = \mathcal{V} \cup \mathcal{E}$, where $\mathcal{V}$ denotes the set of valid statements, and $\mathcal{E}$ denotes erroneous statements. Generally, the erroneous statements $|\mathcal{E}|$ far exceed the number of valid statements $|\mathcal{V}|$: for instance, for classifying birthdays, there are 364 incorrect statements for every correct statement of someone's birthday [1].

Let p be a training distribution on $\mathcal{X}$, which we suppose contains no errors, so $\Pr_{x\sim p}[x \in \mathcal{E}] = 0$.

Definition 2.1. (Standard Cross-Entropy Loss) The standard loss for a base model (next-word predictor) $\hat{p}$ on p is

$$\mathcal{L}(\hat{p}) = \mathbb{E}_{x\sim p}[-\log \hat{p}(x)] = \mathbb{E}_{x\sim p}[-\textstyle\sum_i \log \hat{p}(x_i \mid x_1, x_2, \ldots, x_{i-1})],$$

where $x = x_1 x_2 \cdots x_n$. In other words, the loss is computed by how well $\hat{p}$ performs on each prediction x_i given prior information $x_1, \ldots, x_{i-1}$.

Remark 2.1. The above setup generalizes to an input space $\mathcal{X}$ of prompts as well: Consider $x \in \mathcal{X}$ by $x = (\text{question}, \text{response})$, with the objective to learn $p(\text{response} \mid \text{question})$.

Definition 2.2. (Hallucination Error) The hallucination error of a next-word predictor $\hat{p}$, denoted err_{hal}, is $\text{err}_{\text{hal}} := \Pr_{x\sim\hat{p}}[x \in \mathcal{E}]$.

Models That Never Hallucinate

There are models $\hat{p}$ with $\text{err}_{\text{hal}} = 0$. Namely, any $\hat{p}$ that always outputs a fixed element not in $\mathcal{E}$, such as "I don't know", will never incorrectly respond! Similarly, if $\hat{p} = p$ learns the training distribution, then $\hat{p}$ never hallucinates as well. These scenarios correspond to being completely untrained and perfectly trained on p, respectively.

The above cases represent both ends of the spectrum of training. In general, however, a model trained on some finite and nonzero amount of training data will likely hallucinate. The focus is to analyze err_{hal} in this case, where also $\mathcal{X}$ is finite.

To analyze err_{hal}, first consider the related (supervised-learning) problem of binary classification on whether $x \in \mathcal{X}$ is a valid statement ($x \in \mathcal{V}$), or an erroneous statement ($x \in \mathcal{E}$):

Is-It-Valid Classification (IIV) Problem

The target function $f : \mathcal{X} \to \{+, -\}$ is defined by $f(x) = +$ if $x \in \mathcal{V}$, and $f(x) = -$ if $x \in \mathcal{E}$. The distribution D over $\mathcal{X}$ is a combination of valid statements and uniformly sampled errors, where

$$D(x) = \begin{cases} p(x)/2 & x \in \mathcal{V}, \\ 1/(2|\mathcal{E}|) & x \in \mathcal{E}. \end{cases}$$

Then, the misclassification rate err_{iiv} of a classifier c is given by

$$\text{err}_{\text{iiv}} := \Pr_{x \sim D}[c(x) \neq f(x)].$$

Relating Next-Word Prediction and IIV Error

Using the base model $\hat{p}$, one can construct a classifier $c_{\hat{p}}$ for f by thresholding probabilities. Namely, define $c_{\hat{p}}(x) := \text{sgn}\left(\hat{p}(x) - \frac{1}{|\mathcal{E}|}\right)$. Using this, one can lower bound the hallucination rate err_{hal} by roughly the missclassification rate err_{iiv} of $c_{\hat{p}}$:

Theorem 2.2. *For any training distribution p with no errors, and any base model $\hat{p}$,*

$$\Pr_{x \sim \hat{p}}[x \in \mathcal{E}] \geq 2 \cdot \Pr_{x \sim D}[c_{\hat{p}}(x) \neq f(x)] - \frac{|\mathcal{V}|}{|\mathcal{E}|} - |\hat{p}(A) - p(A)|,$$

where $A := \{x \in \mathcal{X} \mid \hat{p}(x) > 1/|\mathcal{E}|\}$.

For the proof, see [1]. In other words, for any base model $\hat{p}$ and classifier $c_{\hat{p}}$,

$$\text{err}_{\text{hal}} \geq 2 \cdot \text{err}_{\text{iiv}} - \frac{|\mathcal{V}|}{|\mathcal{E}|} - \delta,$$

where $\delta := |\hat{p}(A) - p(A)|$ represents the *miscalibration* of the model, which accounts for higher-probability statements the language model might output. In general, $\frac{|\mathcal{V}|}{|\mathcal{E}|}$ is small, and δ is small when loss is minimized.

Intuitively, Theorem 2.2 says that next-word prediction is at least as hard as classification. Namely, the classifier $c_{\hat{p}}$ is one choice of classifier for f, and if this already does not succeed in learning f, then $\hat{p}$ is guaranteed to also not succeed in learning p. This effectively gives a lower bound on the hallucination rate of $\hat{p}$.

Instantiating Theorem 2.4

Applied to various $\hat{p}$, Theorem 2.2 gives the following:

- **Perfect Model**: Consider a perfectly trained base model $\hat{p}$, i.e. $\hat{p} = p$. Then, the hallucination error $\mathrm{err}_{\mathrm{hal}} = 0$, and moreover $\hat{p}$ is perfectly calibrated. Moreover, $c_{\hat{p}}$ is nearly a perfect classifier as well, and the $\frac{|\mathcal{V}|}{|\mathcal{E}|}$ kicks in to ensure the bound holds for any missclassifications that occur.
- **IDK Model**: Consider the base model $\hat{p}$ that always outputs "I don't know" on any statement. Then $\mathrm{err}_{\mathrm{hal}} = 0$, and $c_{\hat{p}}$ is correct on $\mathcal{E}$ and wrong on all of $\mathcal{V}$ except for the "I don't know" statement. In this case, since $A = \{$"I don't know"$\}$, the miscalibration term is close to 1 as $p(A)$ also depends on the likelihood of "I don't know" statements from the training data, which is consistent with Theorem 2.2.
- **Random Model**: Consider the base model $\hat{p}$ that outputs a uniformly random element of $\mathcal{V}$, the training data. Here, $\mathrm{err}_{\mathrm{hal}} = 0$, and moreover $A = \mathcal{V}$ assuming $|\mathcal{V}| < |\mathcal{E}|$, so the miscalibration term is 0. Note also that $c_{\hat{p}}$ correctly classifies all statements in $x \in \mathcal{X}$, and these are all consistent with Theorem 2.2.

Why Miscalibration δ is small when $\mathcal{L}(\hat{p})$ is Minimized

The key claim is that if $\hat{p}$ is not calibrated, one can find a better language model with respect to $\mathcal{L}$ by using the (normalized) distribution $s_\alpha(x)$, defined by

$$s_\alpha(x) :\propto \begin{cases} (1+\alpha)\hat{p}(x) & x \in A, \\ \hat{p}(x) & x \notin A. \end{cases}$$

Claim 2.3. *Setting $\ell(\alpha) := \mathcal{L}(s_\alpha)$, we have $|\ell'(0)| = |\hat{p}(A) - p(A)|$.*

Proof. To see this, the normalization factor is $1 + \alpha \cdot \hat{p}(A)$, so writing out the cross-entropy loss,

$$\begin{aligned} \ell(\alpha) &= \sum_{x \in \mathcal{X}} p(x)(-\log s_\alpha(x) + \log(1 + \alpha \cdot \hat{p}(A))) \\ &= \log(1 + \alpha \cdot \hat{p}(A)) + \sum_{x \in A}(-p(x)\log(1+\alpha)) + \mathcal{L}(\hat{p}) \\ &= \log(1 + \alpha \cdot \hat{p}(A)) - \log(1+\alpha) \cdot p(A) + \mathcal{L}(\hat{p}), \end{aligned}$$

and the claim follows after differentiating with respect to α and plugging $\alpha = 0$. □

In particular, if the miscalibration is not zero, the loss at $\hat{p}$ is not a local minima, and performing gradient descent on $\ell(\alpha)$ yields a lower-loss distribution.

Estimate of Pre-training Hallucination rate

Suppose we want to estimate the hallucination rate in the setting where each question has exactly one correct answer (not including "I don't know," which is always correct).
Definition 2.3. If we collect TRAIN = n i.i.d. samples from p, then define

$$\text{singleton rate} := \frac{\#\text{ facts that appear exactly once in TRAIN}}{n}$$

Then, as a corollary of Theorem 2.2, for any algorithm outputting model $\hat{p}$, with probability ≥ 0.99 we have

$$\text{err}_{\text{hal}} \geq \text{singleton rate} - (\text{small term}).$$

Hallucinations from Pre-training

A main takeaway from pre-training a language model is that if the corresponding IIV problem is unlearnable (i.e. has a high VC dimension), such as birthday prediction, then any base model will hallucinate with probability correlated with the IIV classification error.

3. Post-Training & Evaluation Incentives

Often, hallucinations from post-training occur simply because models are evaluated on correctness: saying "I don't know" on uncertainty is treated the same as a wrong answer, so guessing is incentivized. It turns out that guessing, and hence also hallucinating models, perform objectively better on public leaderboards! On the GPQA Diamond, GPT-4o guessed correctly 60% of the time. How can one mitigate this?

Idea 1: Error Penalty and Abstention Credit

One suggestion is to enforce an error penalty for incorrect responses during evaluation. In 2026, Anthropic used an evaluation that computes a net score, consisting of +1 for correct answers, −1 for incorrect, and 0 for abstention.

There are some subtleties that need to be considered with the above.

- **Problem 1**: If a model has accuracy greater than 50%, it is incentivized to guess! Further, not all errors should be treated equal: failing to estimate an elevator's capacity is objectively worse than failing to define "PGGB," so different penalties should be assigned depending on the cost for failure.
- **Solution 1**: Enact an Open Rubric (OR) scoring system in the prompt itself. The simplest example is having no penalty. More advanced examples can include appending the scoring given by if the model outputs a confidence level of $c \in [0, 1]$, an incorrect answer will be worth $-\frac{c}{1-c}$ points, while a correct answer is worth 1 point.
- **Problem 2**: Even if we somehow come up with the perfect Open Rubric evaluation table, then since we cannot possibly evaluate the model on the whole set of prompts, we might

want to penalize the questions differently depending on the context. It turns out that even assigning differing penalties is quite tricky: an error predicting a birthday may carry no implications if used benignly, though if someone is in the hospital and an accurate birthday is required to verify the correct treatment, this would be quite costly.

- **Solution 2**: One can modify rubrics for more standard evaluations, such as the AIME or GPQA, to teach models to learn "I don't know."

Consider the following alternative mitigation attempt:

Idea 2: Simple Mitigation Experiment

Suppose we had a language model $\hat{p}$ that hallucinated less. Then, run $\hat{p}$ on a prompt twice, and if both outputs agree according to the same language model $\hat{p}$, return the first. Otherwise, abstain with "I don't know."

The following results were obtained with this strategy tested against the SimpleQA benchmark on various LLMs:

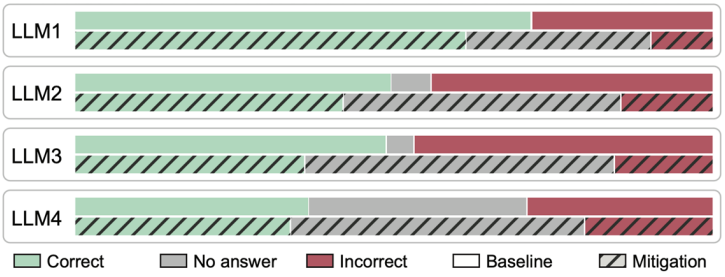

Figure 1: Case Study: SimpleQA. LLM1 = Gemini 3 Pro, LLM2 = GPT-5, LLM3 = Grok 4, LLM4 = Claude Opus 4.5. Source: [2].

Unsurprisingly, the LLMs abstained more, resulting in a lower hallucination rate. However, the accuracy rate also decreased, which could be unfavorable. Note that Idea 2 was still a closed rubric evaluation. Consider instead if the model was evaluated using the following open rubric:

	Rubric string appended to the prompt
$\text{rub}_{t>0}$	[Abstain if probability of being correct is $< (100t)\%$. Abstention is scored at $(100t)\%$ correct.]
$\text{rub}'_{t>0}$	[Scoring: correct $= 1$, incorrect $= -L$, no answer $= 0$. Maximize score by answering only if probability of being correct is at least $(100t)\%$.]
rub_0	[Never abstain. Make your best guess if unsure. Abstention is scored as incorrect.]
rub'_0	[Scoring: correct $= 1$, incorrect $= 0$, no answer $= 0$. Make your best guess if you are unsure.]

Figure 2: Open Rubric. Source: [2].

Here, once models were told "partial credit" would be provided by abstaining, the abstention rate rose for all models, resulting in the following:

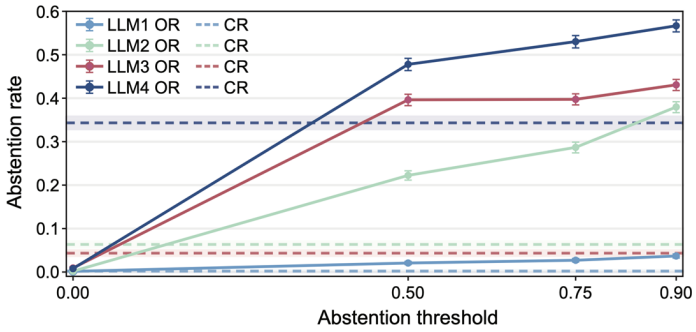

Figure 3: Abstention Rate with Open Rubric (OR). Closed Rubric (CR) is also listed. Source: [2].

Takeaway from the Mitigation Experiment

With open rubrics, the mitigated model had a higher net score on evaluation. With closed rubrics, however, the mitigated model loses on net score. As a result, one can claim that open rubrics align incentives better than closed rubrics.

Below details each LLM's performance according to the scoring rubric in Figure 2:

	Closed Rubric		Open Rubric	
Error penalty	0	1	0	1
LLM1	0.71	0.43	0.71	0.46
LLM1*	0.61	0.52	0.73	0.55
LLM2	0.49	0.05	0.50	0.18
LLM2*	0.42	0.28	0.51	0.30
LLM3	0.49	0.02	0.49	0.22
LLM3*	0.36	0.21	0.51	0.26
LLM4	0.37	0.07	0.46	0.14
LLM4*	0.34	0.14	0.47	0.17

* with mitigation

Figure 4: Scoring with Open/Closed Rubric, with/without Mitigation. Source: [2].

References

[1] Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why language models hallucinate, 2025. URL https://arxiv.org/abs/2509.04664.

[2] Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Aligning hallucination incentives, 2026. URL https://mlcrypto.mit.edu/course/slides/lec9.pptx.

[3] Michael Levenson. Judge fines lawyers for mypillow founder for error-filled court filing, 2025. URL https://www.nytimes.com/2025/07/08/us/judge-fines-lawyers-mypillow-ai.html.