Inverse Language Modeling towards Robust and Grounded LLMs

Master's Degree in Computer Science

Simone Sestito (1937764)

Academic Year 2024/2025

SAPIENZA
UNIVERSITÀ DI ROMA

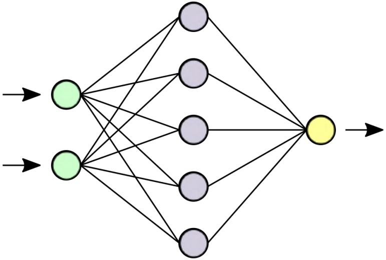
Table of Contents

- ▶ Gradient-based Adversarial Attacks
- ▶ Inverse Language Modeling
- ▶ Results

Gradient-based Attacks

We want to **change the input** to minimize the loss

Input image



Neural Network

Dog

Cat

Output Distribution

Gradient-based Attacks

1 Gradient-based Adversarial Attacks

What to optimize?

Input image

$+ \alpha \cdot$

Noise

$\Rightarrow$

Dog

Cat

Output Distribution

Adversarial Input

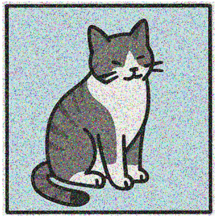
The optimized perturbation δ may look like:

Input image

$+ \alpha \cdot$

Noise

$\Rightarrow$



Adversarial image

Adversarial Training

A classifier can be made robust using **Adversarial Training**:

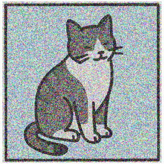
- Generate $\mathbf{x}'$ samples
- Include them in the training process
- Repeat

Adversarial Training

A classifier can be made robust using **Adversarial Training**:

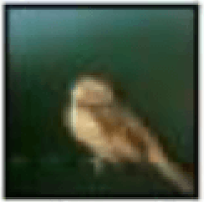
- Generate $\mathbf{x}'$ samples
- Include them in the training process
- Repeat

The required perturbation δ will be more and more perceptible by humans

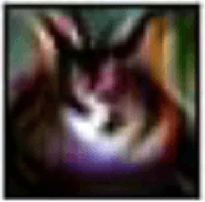


Perceptually-Aligned Gradients

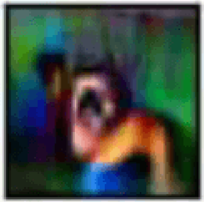
When our classifier has PAGs:



Original image: bird



A ”bird” classified as cat



A ”bird” classified as dog

Gradients are aligned to the human perception

Ganz et al, ”Do Perceptually Aligned Gradients Imply Robustness?”, 2023

Table of Contents

2 Inverse Language Modeling

- Gradient-based Adversarial Attacks
- Inverse Language Modeling
- Results

Difficulties of LLMs

What about LLMs?

- Input is **sequential**
- The same sequence can continue in multiple ways → **multiple** valid classes
- The input space is **discrete** ($|\mathcal{V}|$)

Attacking LLMs with GCG

Greedy Coordinate Gradient is an algorithm to find a sequence of tokens $\mathbf{x}\star$ such that the LLM will complete with $\mathbf{y}$ that we want — using **gradients** on input tokens

Input	Output $\mathbf{y}$	Loss
$\mathbf{x}$: Stevens recorded and produced the album at multiple $\mathbf{x}\star$: Zo Certified Cities (. broadcastquartered Fitness Academy thirteen	locations in the United	5.3642 **5.1302**
$\mathbf{x}$: After the introduction of the Majors , The $\mathbf{x}\star$: REQU Apart British received reformsMilitaryestic Division The	British Army was divided	11.2146 **7.1899**
$\mathbf{x}$: The founding director , Peggy Loar , left $\mathbf{x}\star$: tested UberERIC definitionCalifornia sustainability RutgersOL Jensen regarding	the University of California	7.2669 **6.4402**
$\mathbf{x}$: Ruiz notes that writing also has the power $\mathbf{x}\star$: Report Global feminism agenda Representatives tell Sacredixties Trying	to change the world	5.9135 **4.6041**

They are called ”Evil Twins”

Introducing ILM

- **Goal:** train LLMs to both generate text and *understand what they are conditioned on* from the output
- **Key Ideas:**
  - Create a new training procedure that adds more robustness in the loop
  - Reconstruct input from the output, using $\nabla_{\mathbf{x}}\mathcal{L}$

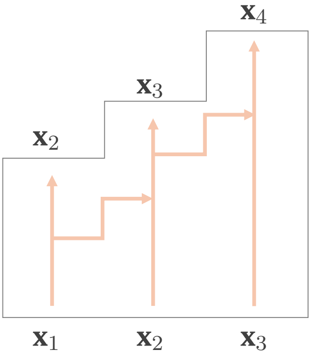
Introducing ILM

2 Inverse Language Modeling

Now

Autoregressive forward

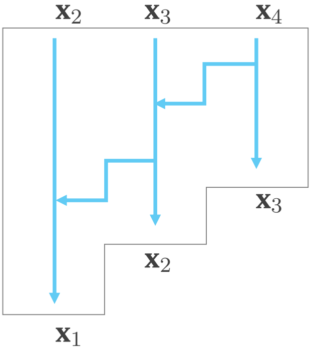
$p(\mathbf{x}_i|\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})$



Proposed

Autoregressive backward

$p\left(\mathbf{x}_{i-1}|\nabla_{\mathbf{x}_{i-1}} p(\mathbf{x}_i|\mathbf{x}_1, \ldots, \mathbf{x}_{i-1})\right)$



ILM Inversion Procedure

Split it into the original prefix $\mathbf{x}_p = \mathbf{x}_{0:k}$ and the suffix $\mathbf{x}_s = \mathbf{x}_{k:n}$

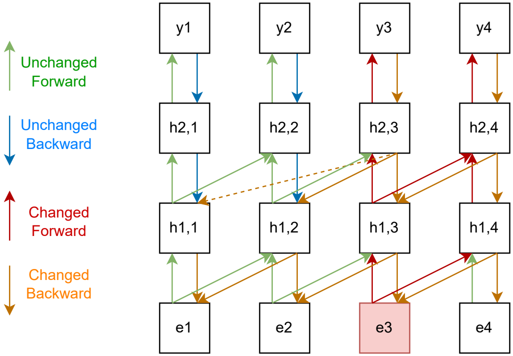
$\mathbf{x} =$ The pen is on the table

$\mathbf{x}_p =$ **The pen is**

$\mathbf{x}_s =$ on the table

Gradients Received by the Tokens

Gradients received on a single token embedding, carry information of the whole sentence



Gradients Received by the Tokens

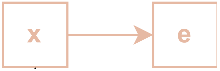
A causal model looks ahead,
but only its gradients disclose the pasts that might have built that future.

ILM Training Procedure

2 Inverse Language Modeling

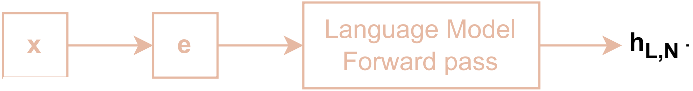
Given the input sentence $\mathbf{x}_0, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{n-1}$

ILM Training Procedure



Embed the input sentence tokens into $\mathbf{e}_0, \mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_{n-1}$

ILM Training Procedure



x

e

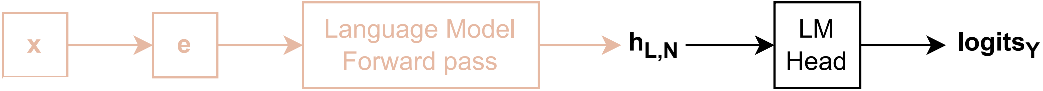
Language Model Forward pass

$\mathbf{h}_{L,N}$ ·

Pass through the Transformer Decoder layer, up to the final hidden state

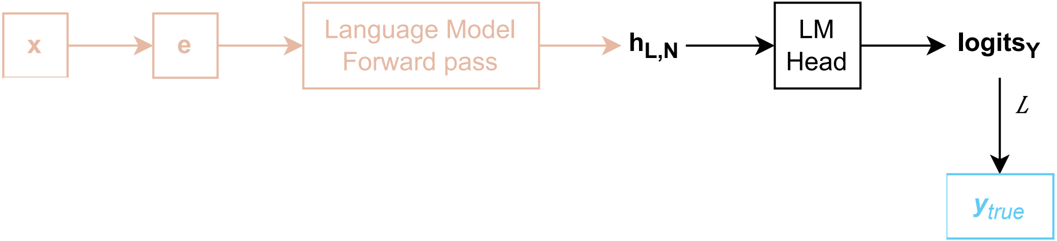
$$\mathbf{h}_0, \mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{n-1}$$

ILM Training Procedure



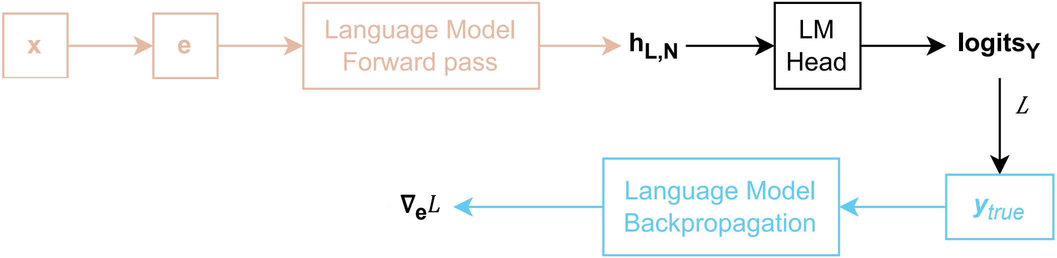
Using the Classifier Head, predict $\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{n-1}$

ILM Training Procedure



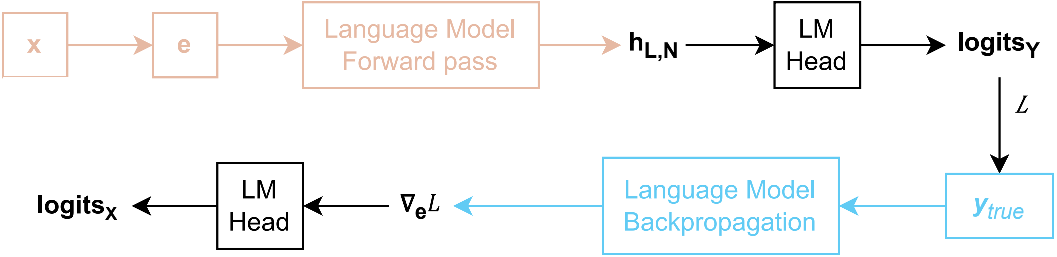
Compute the loss $\mathcal{L}_{CE} = CE(\mathbf{x}_{1:n}, \mathbf{y}_{0:n-1})$ comparing the predictions with the ground-truth

ILM Training Procedure



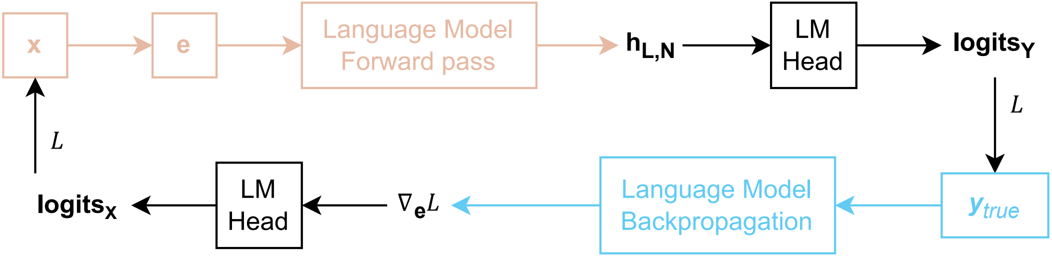
Backpropagation: compute the gradients $\nabla_{\mathbf{e}_{0:n-1}} \mathcal{L}$

ILM Training Procedure



From the gradients, predict the input tokens $\mathbf{x}_{0:n-1}$

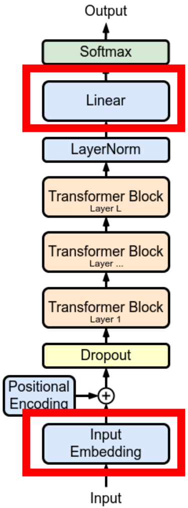
Parallelism


x
e
Language Model Forward pass
$\mathbf{h}_{\mathbf{L,N}}$
LM Head
$\mathbf{logits}_{\mathbf{Y}}$
L
y_{true}
Language Model Backpropagation
$\nabla_{\mathbf{e}} L$
LM Head
$\mathbf{logits}_{\mathbf{X}}$
L

As if it were really **cyclic**!

Parallelism between the last hidden state and the gradients on the embeddings

More Parallelism: Weight Tying



ILM Variants

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla \mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

ILM Variants

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla \mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

- **Identity**: what we have discussed so far

ILM Variants

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla \mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
  - When computing $\nabla_{\mathbf{e}}$, replace 10% tokens to predict from the gradients with `[PAD]` $\rightarrow$ it should understand what's missing

ILM Variants

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla \mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
  - When computing $\nabla_{\mathbf{e}}$, replace 10% tokens to predict from the gradients with `[PAD]` → it should understand what's missing
- **Inv-First**: assign the first token to `[PAD]` and invert it

ILM Variants

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla \mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
  - When computing $\nabla_{\mathbf{e}}$, replace 10% tokens to predict from the gradients with `[PAD]` → it should understand what's missing
- **Inv-First**: assign the first token to `[PAD]` and invert it

Classification Stategies:

ILM Variants

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla \mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
  - When computing $\nabla_{\mathbf{e}}$, replace 10% tokens to predict from the gradients with `[PAD]` $\rightarrow$ it should understand what's missing
- **Inv-First**: assign the first token to `[PAD]` and invert it

Classification Stategies:

- Use gradient as **value** — $f_{\mathbf{W}}(\nabla_{\mathbf{x}_i} \mathcal{L}_{CE})$

ILM Variants

2 Inverse Language Modeling

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, f(\mathbf{x}, \nabla\mathbf{x}))}_{\text{Backward: from gradients, decode back x}}$$

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
  - When computing $\nabla_{\mathbf{e}}$, replace 10% tokens to predict from the gradients with `[PAD]` → it should understand what's missing
- **Inv-First**: assign the first token to `[PAD]` and invert it

Classification Stategies:

- Use gradient as **value** — $f_{\mathbf{W}}(\nabla_{\mathbf{x}_i}\mathcal{L}_{CE})$
- Use gradient as **direction** — $f_{\mathbf{W}}(\mathbf{x}_i - \nabla_{\mathbf{x}_i}\mathcal{L}_{CE})$

But we don’t have billions of dollars

These results have been obtained on a tiny LLM:

- Only 10M parameters
- A vocabulary of just 2048 tokens
- A simple corpus (`TinyStories` dataset)

It will be scaled to Llama-1B in the future.

Table of Contents

3 Results

- Gradient-based Adversarial Attacks
- Inverse Language Modeling
- Results

Inversion Evaluation

	Grad.	Token Recall ↑	Token Precision ↑	Token F1-score ↑	Positional Accuracy ↑
Baseline		20.9%	18.8%	19.7%	2.4%
Inv-First	Val.	11.3%	10.1%	10.7%	1.7%
Bert-like		2.9%	2.7%	2.8%	0.3%
Identity		0.7%	0.7%	0.7%	0.1%
Inv-First	Dir.	13.3%	12.0%	12.6%	2.4%
Bert-like		0.1%	0.1%	0.1%	0.1%
Identity		**22.5%**	**20.2%**	**21.2%**	**2.5%**

Evaluation of the inversion capabilities, on metrics relative to the single tokens

Inversion Evaluation

	Grad.	Full Sentence Perplexity ↓	Predicted Prefix Perplexity ↓	Semantic Similarity ↑
Baseline		**8.34**	112.82	0.28
Inv-First	Val.	10.21	1576.23	0.25
Bert-like		11.54	5501.86	0.17
Identity		13.88	14658.58	0.12
Inv-First	Dir.	9.77	1012.80	**0.30**
Bert-like		11.05	563.26	0.11
Identity		**8.34**	**106.31**	**0.30**

Metrics relative to the full sentences, computed using a third-party LLM

Example of Inversion

3 Results

$\mathbf{x}$		dad in the garden. He gives her a small shovel and a bag of bulbs.
$\mathbf{x}\star$ Baseline		**to play with his cars, and look at the shake. She feels on her hand.**
$\mathbf{x}\star$ Inv-First	(Val.)	zzle spowerlizza in her plate. She start to fence and leaves.
$\mathbf{x}\star$ Bert-like	(Val.)	could buildDven measure its neighbign, how he sees nostiff.
$\mathbf{x}\star$ Identity	(Val.)	Kugct propide,RallashQilndmawkeycessUuhingask do.
$\mathbf{x}\star$ Inv-First	(Dir.)	too hurt the car's bricket. It did not want to grow in a cage.
$\mathbf{x}\star$ Bert-like	(Dir.)	Tim! Tim,ide, Sue, Sue, Tim!ide, ”Tim, ”Tim,ice. Tim! Tim!ittenbbed Tim! Tim,ide,auseectle.
$\mathbf{x}\star$ **Identity**	(Dir.)	**cars, and gets on his hand. But he does not want to play with the towers.**
$\mathbf{y}$		Bulbs are like round seeds that grow into flowers. Lily digs holes in the dirt and puts the bulbs inside. She covers them with more [...]

ILM Robustness Results

	Grad.	GCG Success Rate ↓	GCG Average Steps (mean ± stddev)
Baseline		95.9%	277 ± 148
Inv-First	Val.	85.0%	320 ± 134
Bert-like		**0.8%**	249 ± 148
Identity		88.1%	274 ± 145
Inv-First	Dir.	89.3%	313 ± 134
Bert-like		85.5%	287 ± 143
Identity		82.8%	284 ± 141

Identity looks good, but Bert-like is **suspicious**

ILM Robustness — Metrics on the Model Itself

	Grad.	Original X	Attack X’	Delta	KL
		CE-loss ↓	CE-loss	CE-loss ↓	Divergence ↑
Baseline		13.28	10.97	2.31	2.19
Inv-First	Val.	**11.09**	9.72	1.37	2.44
Bert-like		13.26	10.25	3.01	**54.19**
Identity		12.77	11.21	1.56	2.23
Inv-First	Dir.	11.21	9.81	1.40	2.44
Bert-like		11.49	10.34	**1.15**	2.23
Identity		12.58	11.12	1.46	2.47

Also, Bert-like seems to map $\mathbf{x}_\star$ to very different next token **distributions**

ILM Robustness — Third-Party Model Metrics

	Grad.	Original X Perplexity	Attack X' Perplexity ↓	Semantic Similarity ↑
Baseline		44.14	17344.04	0.13
Inv-First	Val.	44.81	9431.09	0.16
Bert-like		40.37	11817.21	0.11
Identity		43.98	**8322.25**	**0.18**
Inv-First	Dir.	43.50	12344.85	0.13
Bert-like		44.74	10611.09	0.13
Identity		44.71	10929.21	0.15

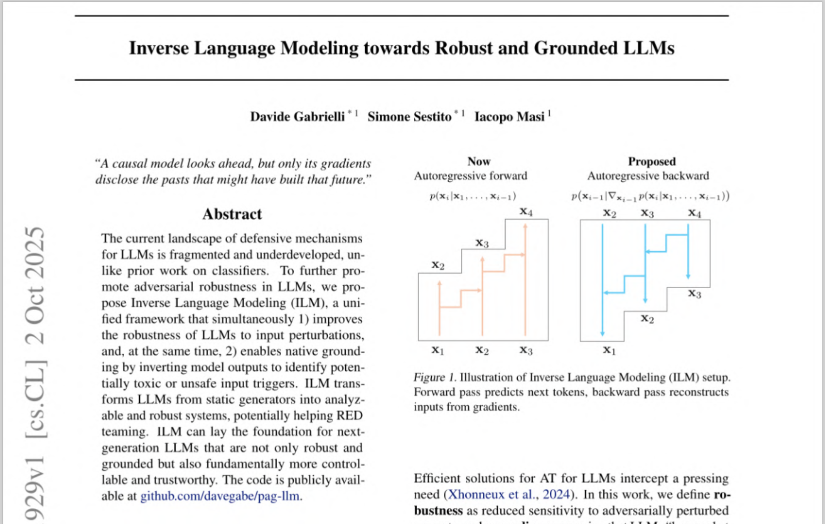
However, all $\mathbf{x}_{\star}$ are **meaningless**, due to extremely high 3rd party model perplexity

ILM Robustness — Qualitative Results

	Input	Output y	Loss
$\mathbf{x}$:	Lily and Ben were friends who liked to play outside. But they did not like the same things. Lily	liked to make snowmen and snow angel	13.22
$\mathbf{x}\star$:	Lucy. Speez herself angO piecle you.”lly named nexird opened cake”.o.ter carrotmy		**12.14**

An example result attacking with GCG the `Identity (grad. value)` model.
Almost the same for all model variants.

Also on arXiv (2510.01929v1)



Inverse Language Modeling towards Robust and Grounded LLMs

Davide Gabrielli [*1] **Simone Sestito** [*1] **Iacopo Masi** [1]

"A causal model looks ahead, but only its gradients disclose the pasts that might have built that future."

Abstract

The current landscape of defensive mechanisms for LLMs is fragmented and underdeveloped, unlike prior work on classifiers. To further promote adversarial robustness in LLMs, we propose Inverse Language Modeling (ILM), a unified framework that simultaneously 1) improves the robustness of LLMs to input perturbations, and, at the same time, 2) enables native grounding by inverting model outputs to identify potentially toxic or unsafe input triggers. ILM transforms LLMs from static generators into analyzable and robust systems, potentially helping RED teaming. ILM can lay the foundation for next-generation LLMs that are not only robust and grounded but also fundamentally more controllable and trustworthy. The code is publicly available at github.com/davegabe/pag-llm.

Figure 1. Illustration of Inverse Language Modeling (ILM) setup. Forward pass predicts next tokens, backward pass reconstructs inputs from gradients.

Efficient solutions for AT for LLMs intercept a pressing need (Xhonneux et al., 2024). In this work, we define **robustness** as reduced sensitivity to adversarially perturbed

Inverse Language Modeling towards Robust and Grounded LLMs

Thank you for listening!
Any questions?