

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

1 Gradient-based Adversarial Attacks

► Gradient-based Adversarial Attacks

► Inverse Language Modeling

► Results



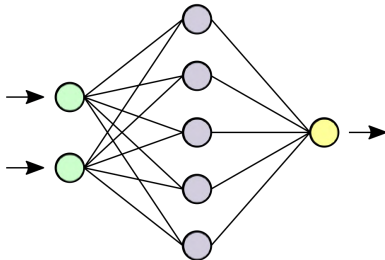
Gradient-based Attacks

1 Gradient-based Adversarial 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

1 Gradient-based Adversarial Attacks

The optimized perturbation δ may look like:

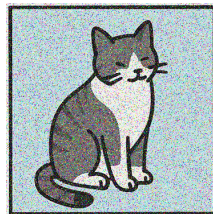


Input image

$+\alpha \cdot$



Noise



Adversarial
image



Adversarial Training

1 Gradient-based Adversarial Attacks

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

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



Adversarial Training

1 Gradient-based Adversarial Attacks

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

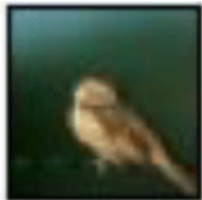




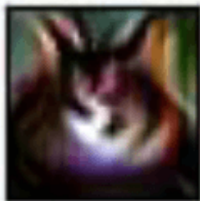
Perceptually-Aligned Gradients

1 Gradient-based Adversarial Attacks

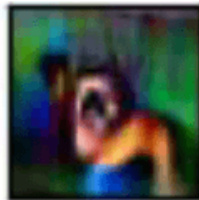
When our classifier has PAGs:



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

2 Inverse Language Modeling

What about LLMs?

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



Attacking LLMs with GCG

2 Inverse Language Modeling

Greedy Coordinate Gradient is an algorithm to find a sequence of tokens \mathbf{x}^* 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}^* : 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}^* : 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}^* : 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}^* : Report Global feminism agenda Representatives tell Sacredixties Trying	to change the world	5.9135 4.6041

They are called "Evil Twins"



Introducing ILM

2 Inverse Language Modeling

- **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 $\nabla_{\mathbf{x}} \mathcal{L}$



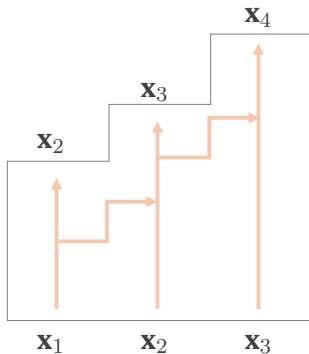
Introducing ILM

2 Inverse Language Modeling

Now

Autoregressive forward

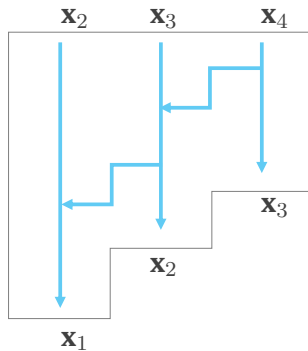
$$p(\mathbf{x}_i | \mathbf{x}_1, \dots, \mathbf{x}_{i-1})$$



Proposed

Autoregressive backward

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





ILM Inversion Procedure

2 Inverse Language Modeling

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

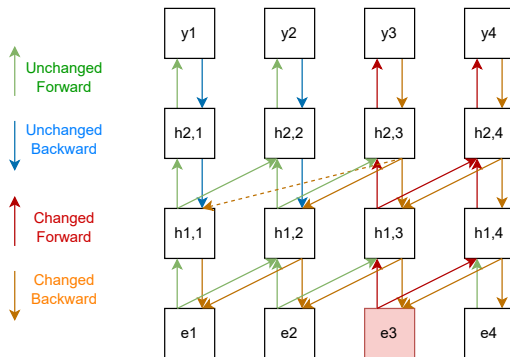
\mathbf{x}_s = on the table



Gradients Received by the Tokens

2 Inverse Language Modeling

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





Gradients Received by the Tokens

2 Inverse Language Modeling

*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

x

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



ILM Training Procedure

2 Inverse Language Modeling



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



ILM Training Procedure

2 Inverse Language Modeling



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

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



ILM Training Procedure

2 Inverse Language Modeling

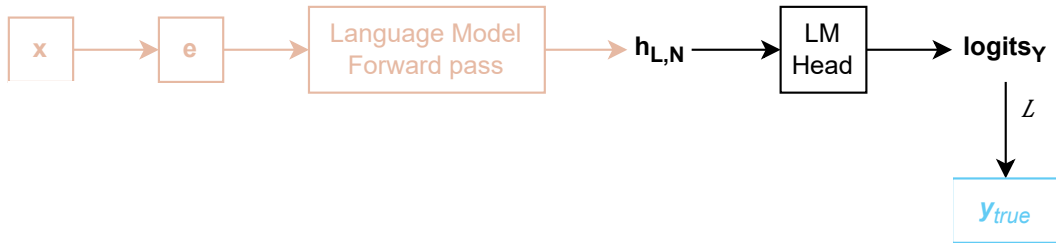


Using the Classifier Head, predict $\mathbf{y}_0, \mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_{n-1}$



ILM Training Procedure

2 Inverse Language Modeling

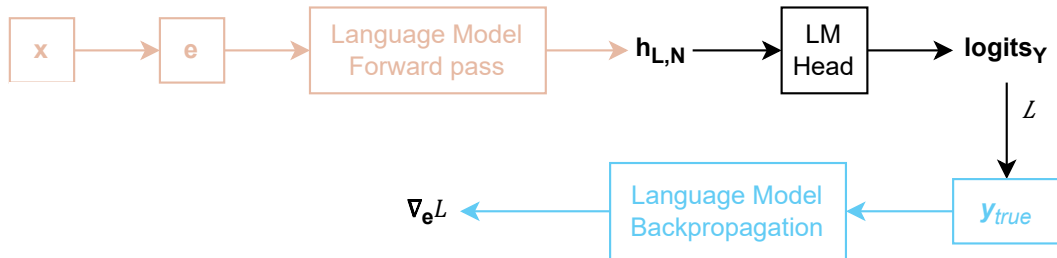


Compute the loss $\mathcal{L}_{CE} = CE(\mathbf{x}_{1:n}, \mathbf{y}_{0:n-1})$ comparing the predictions with the ground-truth



ILM Training Procedure

2 Inverse Language Modeling

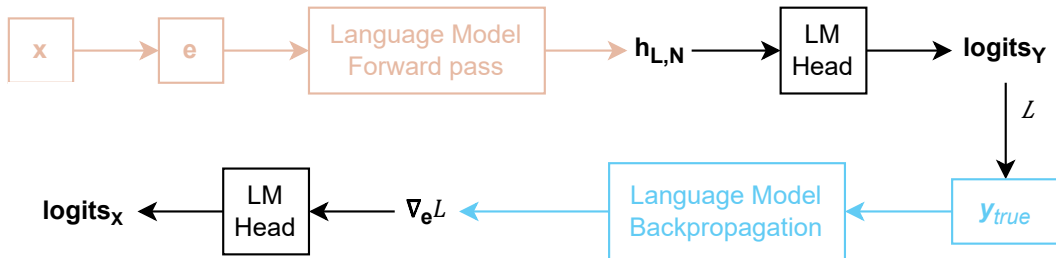


Backpropagation: compute the gradients $\nabla_{e_{0:n-1}} \mathcal{L}$



ILM Training Procedure

2 Inverse Language Modeling

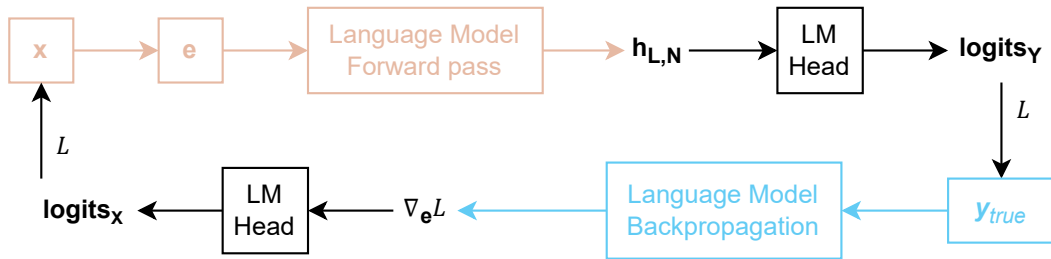


From the gradients, predict the input tokens $\mathbf{x}_{0:n-1}$



Parallelism

2 Inverse Language Modeling



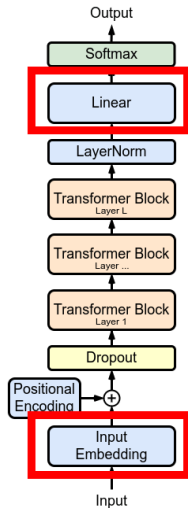
As if it were really **cyclic**!

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



More Parallelism: Weight Tying

2 Inverse Language Modeling





ILM Variants

2 Inverse Language Modeling

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



ILM Variants

2 Inverse Language Modeling

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

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



ILM Variants

2 Inverse Language Modeling

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

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
 - When computing ∇_e , replace 10% tokens to predict from the gradients with [PAD]
→ it should understand what's missing



ILM Variants

2 Inverse Language Modeling

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

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
 - When computing ∇_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

2 Inverse Language Modeling

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

- **Identity**: what we have discussed so far
- **BERT-like**: masking the input tokens on the gradients
 - When computing ∇_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 Strategies:



ILM Variants

2 Inverse Language Modeling

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

- **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 Strategies:

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



ILM Variants

2 Inverse Language Modeling

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

- **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 Strategies:

- 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

2 Inverse Language Modeling

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

3 Results

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

3 Results

	Grad.	Full Sentence Perplexity ↓	Predicted Prefix Perplexity ↓	Semantic Similarity ↑
Baseline		8.34	112.82	<u>0.28</u>
Inv-First		10.21	1576.23	0.25
Bert-like	Val.	11.54	5501.86	0.17
Identity		13.88	14658.58	0.12
Inv-First		9.77	1012.80	0.30
Bert-like	Dir.	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

x		dad in the garden. He gives her a small shovel and a bag of bulbs.
x★ Baseline		to play with his cars, and look at the shake. She feels on her hand.
x★ Inv-First	(Val.)	zzle spowerlizza in her plate. She start to fence and leaves.
x★ Bert-like	(Val.)	could buildDven measure its neighbign, how he sees nostiff.
x★ Identity	(Val.)	Kugct propide,RallashQilndmawkeycessUuhingask do.
x★ Inv-First	(Dir.)	too hurt the car's bricket. It did not want to grow in a cage.
x★ Bert-like	(Dir.)	Tim! Tim,ide, Sue, Sue, Tim!ide, "Tim, "Tim,ice. Tim! Tim!ittenbbbed Tim! Tim,ide,auseectle.
x★ Identity	(Dir.)	cars, and gets on his hand. But he does not want to play with the towers.
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

3 Results

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

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



ILM Robustness — Metrics on the Model Itself

3 Results

	Grad.	Original X CE-loss ↓	Attack X' CE-loss	Delta CE-loss ↓	KL Divergence ↑
Baseline		13.28	10.97	2.31	2.19
Inv-First		11.09	9.72	<u>1.37</u>	2.44
Bert-like	Val.	13.26	10.25	3.01	54.19
Identity		12.77	11.21	1.56	2.23
Inv-First		<u>11.21</u>	9.81	1.40	2.44
Bert-like	Dir.	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}_* to very different next token **distributions**



ILM Robustness — Third-Party Model Metrics

3 Results

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

However, all x^* are **meaningless**, due to extremely high 3rd party model perplexity



ILM Robustness — Qualitative Results

3 Results

	Input	Output y	Loss
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
x^* :	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

3 Results

INSERT SCREENSHOT

arXiv:2412.08127v3 [cs.CL] 31 Mar 2025



Inverse Language Modeling towards Robust and Grounded LLMs

Thank you for listening!
Any questions?