Inverse Language Modeling Towards Robust and Grounded LLMs

Davide Gabrielli ✉[0009−0000−5672−6749], Simone Sestito[0009−0006−5009−9071], and Iacopo Masi[0000−0003−0444−7646]

Sapienza University of Rome, Computer Science Dept., OmnAI Lab, Italy
{gabrielli.d,sestito,masi}@di.uniroma1.it

Abstract. Interpretability and robustness remain major challenges for modern Large Language Models, especially in settings where conventional evaluation or auditing tools are limited. To address this, we propose *Inverse Language Modeling* (ILM), a unified training framework that jointly enhances robustness to adversarial perturbations and enables a novel form of gradient-based interpretability. Rather than reconstructing exact input prompts, ILM encourages LLMs to develop gradient-aligned internal representations that allow the model to approximate *plausible* input patterns underlying a given output. This approximate inversion provides a new mechanism for analyzing model behavior, identifying potential triggers for unsafe generations, and providing a diagnostic signal that may support future auditing workflows. Our results show that ILM can simultaneously improve robustness and produce meaningful inversion signals, laying a foundation for LLMs that are not only more resilient, but also more transparent and analyzable. Code available at https://github.com/davegabe/pag-llm.

Keywords: LLM · Invertibility · Adversarial Training · Gradients · Robustness.

1 Introduction

While Large Language Models (LLMs) can handle a wide variety of natural language processing tasks and demonstrate impressive reasoning abilities, they still have notable limitations. A single foundation model can address many NLP challenges, but LLMs are prone to producing inaccurate information and are sensitive to slight changes in input, such as adversarial prompts. Recent work indicates that even nonsensical perturbations [26,17] can trigger these issues, highlighting the potential for backdoors [3]. These risks become particularly salient when LLMs are used in culturally diverse communities, where it is essential to ensure consistent behavior with local values and ethical expectations.

These problems emphasize the need for adversarial training tools (AT) for LLMs. However, the literature on this topic is not as dense as that on deep classifiers, yet the security of LLMs against adversarial perturbations remains an open challenge. Moreover, LLM training is very costly, and therefore applying

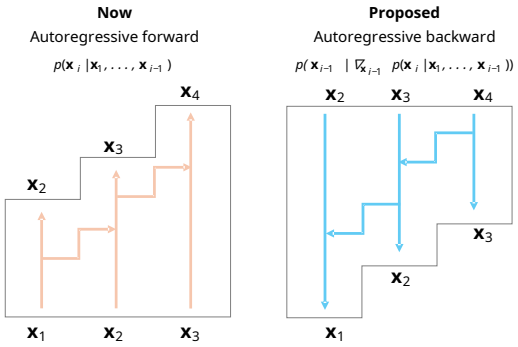

Fig. 1: Illustration of Inverse Language Modeling (ILM) setup. The forward pass predicts the next tokens, and the backward pass reconstructs the inputs from gradients.

AT could only worsen the issue. At the same time, beyond robustness, there is a growing demand for mechanisms that help interpret why a model produces certain responses, especially on ethically sensitive or culturally situated topics [24].

In this work, we define **robustness** as reduced sensitivity to adversarially perturbed prompts, and **grounding** as ensuring that LLM "know what they have been asked", addressing evidence that they often fail to represent their own knowledge faithfully [17,2].

In light of this, our objectives are twofold. The first centers on **Robustness**: we present a new, fast, and efficient adversarial training approach for LLMs, called *Inverse Language Modeling (ILM)*. ILM draws on advances in developing robust classifiers, emphasizing that Perceptually Aligned Gradients (PAG) serve as a foundation for robustness in these models [10].

Standard LLMs are typically trained in a forward mode, where a transformer [23] predicts the continuation $\mathbf{y}$ of text prompt $\mathbf{x}$ through self-supervision. For clarity, we define the text prompt as $\mathbf{x}$, which corresponds to the input token sequence $\{\mathbf{x}_0, \ldots, \mathbf{x}_{i-1}\}$, while the target sequence $\mathbf{y}$ is a one-step left-shifted version of $\mathbf{x}$. In contrast, ILM extends this paradigm by introducing a backward perspective (see Figure 1): given an output $\mathbf{y}$, the model attempts to approximate the conditioning prompt $\mathbf{x}$.

The second objective relates to **Grounded LLMs** and follows naturally from the first. By enabling inversion from $\mathbf{y}$ back to $\mathbf{x}$, ILM provides a diagnostic signal that can support auditing workflows. While it does not guarantee exact prompt recovery, it allows us to trace plausible prompt approximations that could have produced a given (potentially malicious or undesired) output. In this way, ILM helps ground model behavior in more transparent and inspectable evidence.

Importantly, ILM does not simply reverse the token sequence. Instead, it reconstructs the input prompt through a gradient-based alignment process informed by both output probabilities and intermediate representations accumulated across model layers during the forward pass.

2 Prior Work

Research on adversarial attacks against LLMs has advanced significantly, particularly in the generation of adversarial suffixes designed to bypass alignment safeguards. Early techniques, such as HotFlip [7] and Greedy Coordinate Gradient (GCG) [26], focused on manipulating the input text or its embedding gradients to induce undesirable behavior from LLMs. GCG modifies token selections iteratively based on gradient information. Subsequent enhancements, including Probe Sampling [25] and token similarity-based heuristics [14], improved the efficiency of this search process.

More recent methods include AutoDAN [16], which leverages genetic algorithms to produce fluent and stealthy adversarial suffixes, and its successor AutoDAN Turbo [15], which coordinates multiple LLMs for strategy development and attack evaluation. AdvPrompter [19] takes a different approach by fine-tuning a model specifically to generate coherent adversarial suffixes, allowing fast and automated jailbreaking.

On the defensive side, perplexity-based filtering [1] has proven effective at identifying adversarial suffixes by exploiting their typically high perplexity. However, newer attacks are designed to bypass such detection mechanisms by optimizing fluency and semantic plausibility. In addition, work on language model inversion [18] explores the recovery of original prompts from output probabilities, similar to reconstruction techniques in computer vision. These findings have informed strategies for generating adversarial prompts using only output distributions.

Table 1: Original inputs $\mathbf{x}$ and adversarial examples $\mathbf{x}^\star$ generated using the GCG method for the SmolLM-360M model. The table shows that each original input and its corresponding adversarial example yield the same output, along with the loss values for the output token IDs. These examples show that LLMs can be manipulated into assigning lower loss to nonsensical prompts than to the original, meaningful input – highlighting a vulnerability that ILM is designed to address.

Input	**Output 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**

Unlike prior work focused on suffix generation or language inversion as an offensive tool, our research seeks to understand and mitigate these vulnerabilities. In particular, we study "evil twin" prompts as defined in [17,21]. Given

a text prompt $\mathbf{x}$ and the completion $\mathbf{y}$, we perform an optimization so that given $\mathbf{y}$, we find a new nonsensical $\mathbf{x}^\star$ – the "evil twin" – such that the loss $\mathcal{L}(\mathbf{x}^\star, \mathbf{y}; \boldsymbol{\theta}) \ll \mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta})$, where $\mathcal{L}$ is the next-token prediction loss of LLM and $\boldsymbol{\theta}$ are LLM's parameters. These $\mathbf{x}^\star$ are syntactically implausible out-of-distribution inputs that nevertheless lead to the same output, as illustrated in Table 1. Despite producing identical continuations, $\mathbf{x}$ and $\mathbf{x}^\star$ induce notably different entropy distributions. These prompts are also fragile – small changes typically break the adversarial effect, highlighting a key vulnerability in LLM robustness and alignment.

To address evil twin prompts, we propose Inverse Language Modeling, a novel training framework that improves LLM robustness. ILM enables both forward modeling and partial inversion, encouraging the model not only to generate fluent output but also to remain sensitive to input semantics.

3 Method

3.1 Preliminary Study on PAG on Text Classification

In this preliminary experiment, we investigate the application of Perceptually Aligned Gradients [10] to sentence classification using hidden state representations from the DistilBERT language model [22]. While PAG has primarily been explored in the context of image classification, we adapt the methodology to the hidden state space of a transformer model to examine its effects on robustness and interpretability in NLP tasks. The core idea of PAG is to encourage gradients to align with semantically meaningful directions, and we hypothesize that this can lead to more robust and interpretable text representations as well.

To prove our point, we ran a proof-of-concept experiment using a classifier trained on top of the hidden state associated with the `[CLS]` token, adopting the `distilbert-base-multilingual-cased`, on Amazon Review Multi dataset [13]. We considered 12 classes, each a combination of languages (English, German, Spanish, and French) and review ratings (1, 3, and 5 stars).

PAG Application. To incorporate PAG, we extend the standard cross-entropy objective with a regularization term that enforces alignment between the gradient of the model output and a "proxy" target direction. The resulting loss for a classifier built on top of a frozen DistilBERT backbone is defined as:

$$\begin{aligned} \mathcal{L} &= \mathcal{L}_{CE}(f_\theta(\mathbf{x}), y) + \lambda\, \mathcal{L}_{PAG}(\mathbf{x}), \\ \mathcal{L}_{PAG}(\mathbf{x}) &= \frac{1}{C} \sum_{c=1}^{C} \left(1 - \frac{\nabla_{\mathbf{h}} f_\theta(\mathbf{x})_c^\top \, g(\mathbf{x}, c)}{\|\nabla_{\mathbf{h}} f_\theta(\mathbf{x})_c\| \; \|g(\mathbf{x}, c)\|} \right). \end{aligned} \tag{1}$$

Here, $\mathcal{L}_{PAG}$ penalizes misalignment via the cosine distance between the gradient of the classifier output with respect to the hidden representation and a proxy direction.

Notation.

- $\mathbf{x}$ denotes the input sentence.
- $y \in \{1, \ldots, C\}$ is the ground-truth class label.
- $f_\theta(\mathbf{x})$ is the classifier operating on the DistilBERT hidden representation $\mathbf{h}$.
- $\mathcal{L}_{CE}$ denotes the standard cross-entropy loss.
- $\lambda \geq 0$ controls the strength of the PAG regularization.
- C is the number of classes.
- $\nabla_{\mathbf{h}} f_\theta(\mathbf{x})_c$ is the gradient of the logit corresponding to class c with respect to $\mathbf{h}$.
- $g(\mathbf{x}, c)$ denotes the proxy target direction associated with class c.

Proxy Ground-Truth Gradient. We define the proxy target direction (PAG variant) as the difference between the hidden representation of the input sentence, $\mathbf{h}_\mathbf{x}$, and that of a randomly sampled sentence $\mathbf{u}_y$ from the same class y:

$$g(\mathbf{x}, y) = \mathbf{u}_y - \mathbf{h}_\mathbf{x}. \tag{2}$$

This construction encourages gradients to align with directions that connect samples within the same class in representation space, thereby promoting semantically consistent updates.

We also consider an alternative formulation, referred to as **Identity**, in which the proxy direction is defined directly in the input space:

$$g(\mathbf{x}, y) = \mathbf{x}. \tag{3}$$

In this case, the model is encouraged to reconstruct the input from the induced gradients, effectively enforcing self-alignment.

As a reference, we include a baseline model trained with identical architecture and hyperparameters but without PAG regularization (i.e., $\lambda = 0$), thereby isolating the effect of $\mathcal{L}_{PAG}$.

Table 2: Robustness of classifier models with PAG variants under APGD, Square, and FGSM attacks. Higher percentages indicate stronger robustness.

attack →	**APGD** [4]		**Square** [4]	**FGSM** [11]	
	ε			α	
$g(\mathbf{x})$ ↓	1e-3	0.5		5e-3	1e-2
Baseline	36.5%	31.2%	36.3%	27.3%	8.9%
Identity	28.3%	25.0%	27.2%	25.7%	8.0%
PAG	**48.1%**	**45.0%**	**49.3%**	**43.5%**	**25.7%**

Evaluation. According to the results in Table 2, the strongest model in robustness is the one trained with the full PAG loss with Equation 2, which forces the model to make the gradients on the input point towards the direction of the predicted class. These models have been attacked by APGD, Square [4], and FGSM [11].

3.2 Inverse Language Modeling

ILM builds on PAG by extending it to the language modeling setting. The method is non-iterative and relies on double backpropagation [6], which can be efficiently implemented with standard autograd tools.

ILM performs the following: instead of training the LLM to *only* maximize $p(\mathbf{y}|\mathbf{x})$, we also invert it and from the output $\mathbf{y}$, we aim to reconstruct the input $\mathbf{x}$. This procedure is not merely a double forward pass of the original text and its reverse. Instead, we first impose a loss for $p(\mathbf{y}|\mathbf{x})$, yet instead of updating the weights, we also receive gradients over input tokens $\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta})$ requiring them to predict some tokens in $\mathbf{x}$, depending on the exact model variant among the ones discussed later. This focus on bidirectional understanding during pretraining is key to improving the model's comprehension.

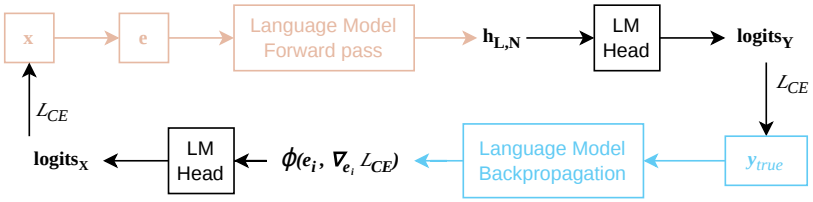

Fig. 2: Parallelism between last hidden states and embedding gradients: both can be mapped through the LM head to token predictions.

In a causal Transformer architecture, the influence of a single input token demonstrates a fundamental asymmetry between the forward and backward passes. During the forward pass, causal masking ensures that the hidden state at step t depends solely on the current and preceding tokens (i.e., $h_t = f(e_{\leq t})$). Therefore, modifying a token embedding e_i affects the forward activations only for steps $t \geq i$, while the preceding states (where $t < i$) remain completely unchanged.

In contrast, this causal isolation disappears during backpropagation. The total loss $\mathcal{L} = \sum_t \mathcal{L}_t$ is computed by evaluating future queries against past keys and values. This means that the loss at step t becomes a highly non-linear function of all preceding tokens. Changing e_i affects the queries, the attention weights, and subsequently the states for all $t \geq i$. When these modified future states are evaluated against the keys of past tokens ($e_{<i}$), the backward gradient signals sent to those earlier embeddings are fundamentally altered.

As a result, while the forward pass strictly prevents a token from influencing the past, the dense mixing in the backward pass ensures that modifying a single token will change the gradients propagated back to preceding tokens.

We will use ILM at training time, while at test time we exploit the GCG algorithm to find, given an original text prompt $\mathbf{x}$ and the completion $\mathbf{y}$, a new

nonsensical $\mathbf{x}^\star$ such that the loss $\mathcal{L}(\mathbf{x}^\star, \mathbf{y}; \boldsymbol{\theta}) \ll \mathcal{L}(\mathbf{x}, \mathbf{y}; \boldsymbol{\theta})$, where $\mathcal{L}$ is the next-token prediction loss of the LLM and $\boldsymbol{\theta}$ are LLM's parameters. We show that a nonsensical prompt $\mathbf{x}^\star$ can achieve lower loss with $\mathbf{y}$ than the natural $\mathbf{x}$.

The standard formulation of PAG in image classification (Eq. 1) is not directly transferable to LLMs due to fundamental differences in the input structure. Images are continuous tensors that encode class-discriminative information locally, whereas LLMs operate on discrete tokens that depend on full sequence context and are represented as one-hot vectors with limited semantic content.

Moreover, computing gradients with respect to input tokens is challenging: token-level gradients lack global context, and the large vocabulary size – often hundreds of thousands of tokens – makes class-iterative PAG computationally infeasible due to the high dimensionality of the prediction space.

Consequently, our approach deviates from the standard PAG for classifiers. We focus solely on gradients with respect to the actual input tokens and intend to classify the actual tokens, as in Figure 1, rather than using the cosine distance for gradient direction, aiming to leverage this for LLM inversion as well.

For these experiments, our model architecture is a small decoder-only transformer with Weight Tying [20,12] enabled, 3 hidden layers, and a hidden-layer vector size of 640.

The dataset used is TinyStories [8], with a tokenizer trained from scratch using the standard Byte-Pair Encoding [9], to provide a flexible vocabulary size and support experiments of varying complexity and entropy in next-token classification. Specifically, we used a vocabulary of 2048 possible tokens. Also, the dataset samples include an overlap of 25% with the original sentences. This overlap increases variability in sentence starts, providing the model with more diverse context patterns and better approximating realistic sentence-completion scenarios. The finally constructed dataset [1] has been uploaded to HuggingFace for reproducibility.

The backward prediction strategy shares a common logic across all model variants, which can be formalized as follows. Given the cross-entropy loss between predicted tokens and their ground truth, we compute the gradient with respect to the embedding vectors. To handle different variants consistently, we define a mapping $\phi(\mathbf{e}_i, \nabla_{\mathbf{e}_i}\mathcal{L}_{CE})$ that specifies how the gradient is interpreted for classification. Then, the output is normalized and used with the LM Head weight matrix to get a probability distribution over the vocabulary. The general backward prediction for any token can then be written as:

$$\begin{aligned} \mathcal{L}_{CE} &= CE(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}}) \\ \mathbf{g}_i &= \text{LayerNorm}(\phi(\mathbf{e}_i, \nabla_{\mathbf{e}_i}\mathcal{L}_{CE})) \\ \mathbf{z}_i &= \mathbf{W}_{\text{LM\_head}}\, \mathbf{g}_i \\ \hat{\mathbf{y}}_i &= \text{softmax}(\mathbf{z}_i) \end{aligned} \tag{4}$$

This formulation highlights a parallelism between the forward and backward modes (Figure 2): the gradient vector with respect to a token embedding plays

[1] https://huggingface.co/datasets/DaveGabe/TinyStoriesV2_cleaned

the same role as the last hidden state in next-token prediction. In particular, replacing $\nabla_{e_i}\mathcal{L}_{CE}$ with the last hidden state recovers the standard forward pass of an LLM, since both share the same dimensionality.

The final training loss combines the forward and inverse LM objectives, with $\lambda = 2.0$ found to be a suitable hyperparameter:

$$\mathcal{L} = \underbrace{\mathcal{L}_{CE}(\mathbf{y}_{\text{true}}, \mathbf{y}_{\text{pred}})}_{\text{Forward: from the input x, encode y}} + \underbrace{\lambda\, \mathcal{L}_{CE}(\mathbf{x}, \hat{\mathbf{y}})}_{\text{Backward: from y, decode back x}} \tag{5}$$

3.3 ILM Model Variants

We evaluate four training strategies, each differing in how gradients are used for inversion:

Baseline. The model is trained only with the standard forward Cross-Entropy loss, without any gradient-based inversion objective.

Inv-First. Only the first token of the sentence is reconstructed from its gradient, by predicting

$$p(\mathbf{x}_0 \mid \nabla_{\mathbf{e}_0}\mathcal{L}_{\text{CE}}(f_\theta(\texttt{[PAD]} \,\|\, \mathbf{x}_{1:N}), \mathbf{y})). \tag{6}$$

Bert-like. A subset of tokens $\mathcal{M} \subseteq [1, N]$ is masked, and the model reconstructs only those tokens from their corresponding gradients:

$$p(\mathbf{x}_i \mid \nabla_{\mathbf{e}_i}\mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}_{\setminus\mathcal{M}}), \mathbf{y})), \quad \forall i \in \mathcal{M}. \tag{7}$$

This formulation is analogous to the BERT [5] training scheme, but operates on gradients in the backward pass.

Identity. During training, the model is required to reconstruct every input token directly from its corresponding gradient:

$$p(\mathbf{x}_i \mid \nabla_{\mathbf{e}_i}\mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}), \mathbf{y})), \quad \forall i \in [1, N]. \tag{8}$$

Each strategy comes in two approaches, depending on the implementation of the $\phi(\mathbf{e}_i, \nabla_{\mathbf{e}_i}\mathcal{L}_{CE})$ function mentioned in Equation 4. When we consider gradients as directions, we classify on $\mathbf{e}_i - \nabla_{\mathbf{e}_i}\mathcal{L}_{CE}$, thus by imposing $\phi(\mathbf{e}_i, \nabla_{\mathbf{e}_i}\mathcal{L}_{CE}) = \mathbf{e}_i - \nabla_{\mathbf{e}_i}\mathcal{L}_{CE}$. On the other hand, the case where we use the gradients as pure values is simpler: $\phi(\mathbf{e}_i, \nabla_{\mathbf{e}_i}\mathcal{L}_{CE}) = \nabla_{\mathbf{e}_i}\mathcal{L}_{CE}$, discarding the input embedding value.

4 Experimental Evaluation

We evaluate the models along two dimensions: their ability to recover the input sequence from a given continuation, and their robustness to GCG attacks. Both aspects are essential for developing robust and grounded LLMs.

4.1 Inversion Procedure

The evaluation relies on three complementary measures. Validation loss and validation accuracy are computed under the same conditions as training, serving as standard indicators of model fit. In addition, we report Inverse LM accuracy, which evaluates the model's ability to reconstruct a masked token $\mathbf{x}_i$ from its gradients, given the remaining context. This provides a direct assessment of the backward prediction mechanism.

Following section 3, inversion replaces a target token x_i according to the training strategy and computes $\nabla_{e_i} L_{CE}$ on the modified sequence. The gradients are mapped via $\phi(\cdot)$, normalized, and projected through the LM head to obtain $\hat{y}_i$ (Equation 4).

4.2 Inversion Evaluation

To assess inversion capabilities, we extend the task beyond single-token recovery and instead invert multiple tokens in an autoregressive manner. The procedure follows a beam-search strategy, as detailed in Algorithm 1, where candidate prefixes are iteratively expanded and filtered by perplexity until a coherent reconstruction emerges.

Algorithm 1 Autoregressive Inversion Evaluation with Beam Search

Require: Input sample $\mathbf{x}$ of length n, beam size b, split position k
Ensure: Inverted prefix $\mathbf{x}_{\text{inv}}$
$\mathbf{x}_p \leftarrow \mathbf{x}_{0:k}$ ▷ Original prefix (hidden)
$\mathbf{x}_s \leftarrow \mathbf{x}_{k:n}$ ▷ Visible suffix
$\mathbf{X} \leftarrow \{\mathbf{x}_s\}$ ▷ Initialize beam set with suffix only
while inverted prefix not sufficiently long **do**
  for each sequence $\mathbf{x}' \in \mathbf{X}$ **do**
    Compute top-b tokens for the previous position
    Extend $\mathbf{x}'$ with each candidate token
  end for
  $\mathbf{X} \leftarrow$ top-b sequences with lowest perplexity
end while
return $\mathbf{x}_{\text{inv}} \leftarrow \arg\min_{\mathbf{x}' \in \mathbf{X}} \text{Perplexity}(\mathbf{x}')$

In the evaluation process, we consider only the combination of initialization strategy and model variant used during training. In particular, the `Identity` variant initializes the unknown token using a simple bigram model. In contrast, all other variants use a fixed placeholder token (e.g., `<|pad|>`), consistent with their training setup. Since the true token is not available at inference time, the bigram initialization provides a reasonable approximation, yielding better inversion performance than random initialization or a fixed placeholder that was not observed during training in this context.

Table 3: Inversion performance combining token-level and sequence-level metrics. **Left:** token-level reconstruction metrics (Rec, Prec, F1, Acc). **Right:** sequence-level metrics (Full PPL, Prefix PPL, Prefix Sim) evaluating plausibility and semantic alignment of the reconstructed prefix.

	Token-level				**Sequence-level**		
Method	**Rec ↑**	**Prec ↑**	**F1 ↑**	**Acc ↑**	**Full PPL ↓**	**Pref. PPL ↓**	**Pref. Sim ↑**
Baseline	20.9%	18.8%	19.7%	2.4%	**8.34**	112.82	0.28
Gradient as Value							
Inv-First	11.3%	10.1%	10.7%	1.7%	10.21	1576.23	0.25
Bert-like	2.9%	2.7%	2.8%	0.3%	11.54	5501.86	0.17
Identity	0.7%	0.7%	0.7%	0.1%	13.88	14658.58	0.12
Gradient as Direction							
Inv-First	13.3%	12.0%	12.6%	2.4%	9.77	1012.80	**0.30**
Bert-like	0.1%	0.1%	0.1%	0.1%	11.05	563.26	0.11
Identity	**22.5%**	**20.2%**	**21.2%**	**2.5%**	**8.34**	**106.31**	**0.30**

We evaluate inversion quality using both token-level and sequence-level metrics. Token-level metrics assess reconstruction fidelity and include Recall, Precision, F1 score, and Accuracy. Sequence-level metrics evaluate the plausibility and semantic alignment of the reconstructed prefix, including Full PPL (perplexity of the reconstructed prefix concatenated with the continuation), Prefix PPL (perplexity of the reconstructed prefix alone), and semantic similarity with respect to the ground-truth prefix.

Perplexity-based metrics are computed using a third-party language model (Llama-3.2-1B)[2] to decouple evaluation from the trained models. Semantic similarity is computed as the cosine similarity between sentence embeddings obtained from an external encoder[3].

Among the evaluated metrics, *Full PPL* exhibits relatively low variance across models, as it is computed over the entire sequence, where the continuation dominates the total length. Consequently, differences between models are attenuated and should be interpreted comparatively.

For reference, the perplexity of the ground-truth prefix is 37.83, whereas reconstructed prefixes exhibit substantially higher values across all models. This highlights the intrinsic difficulty of the inversion task and the reduced naturalness of generated prefixes.

Among the evaluated metrics, *Prefix PPL* shows the largest variation across training strategies. Models that rely on gradients as raw values produce extremely high prefix perplexity, indicating poor fluency and unstable reconstructions. In contrast, using gradients as directions significantly reduces Prefix PPL (e.g., 106.31 for Identity), suggesting that directional gradients provide a more informative signal for inversion and yield more coherent prefixes.

[2] https://huggingface.co/meta-llama/Llama-3.2-1B
[3] https://huggingface.co/sentence-transformers/all-MiniLM-L6-v2

Overall, the Identity variant with gradients as directions achieves the best trade-off, combining the highest token-level performance with the lowest Prefix PPL and competitive semantic similarity. While none of the models match the naturalness of the ground-truth prefix, these results indicate that *Identity (grad. direction)* provides the best balance between reconstruction fidelity and linguistic plausibility.

4.3 Robustness Against GCG

We evaluate robustness using the success rate of Greedy Coordinate Gradient attacks [26]. This benchmark aligns naturally with the objectives of our training procedure: our ultimate goal is to produce LLMs that are more grounded, ensuring that their responses are faithful to and informed by the prompts they receive. The procedure for evaluating the models follows these rules, applied to 30% of randomly selected samples from the test set, consistently across all model variants.

Algorithm 2 Single-Sentence GCG Attack

Require: Expected continuation string $\mathbf{y}$ to be attacked, length of the attack prefix n, number of iterations T
Ensure: Best attack prefix $\mathbf{x}^\star$ with loss $\mathcal{L}_{\text{GCG}}$

$\mathbf{x}^\star \leftarrow$ random one-hot tokens matrix of size $|V| \times n$
$step \leftarrow 0$ ▷ Iteration counter
$d \leftarrow 0$ ▷ Loss non-decrease counter
$\mathcal{L}_{\text{old}} \leftarrow \infty$ ▷ Last loss found
while $step < T$ **and** $d < 10$ **do**
    Compute a batch of candidate prefixes $\mathbf{X}$ running one step of **GCG**
    $\mathbf{x}^\star \leftarrow \arg\min_{\mathbf{x} \in \mathbf{X}} \mathcal{L}_{\text{CE}}(\mathbf{x}, \mathbf{y}, \boldsymbol{\theta})$
    $\mathcal{L}_{\text{GCG}} \leftarrow \mathcal{L}_{\text{CE}}(\mathbf{x}^\star, \mathbf{y}, \boldsymbol{\theta})$ ▷ Take the min loss so far
    if $\mathcal{L}_{\text{GCG}} < \mathcal{L}_{\text{old}}$ **then**
        $\mathcal{L}_{\text{old}} \leftarrow \mathcal{L}_{\text{GCG}}$
        $d \leftarrow 0$
    else
        $d \leftarrow d + 1$
    end if
    $step \leftarrow step + 1$
end while
return $\mathcal{L}_{\text{GCG}}$

From the results reported in Table 4, most variants show improved robustness against GCG attacks. In particular, the variant identified as best in the inversion task, *Identity (Gradient as Direction)*, reduces the attack success rate by more than 13%. This improvement can be attributed to the model's stronger conditioning of the continuation on the input prompt (i.e., improved *grounding*), which leads to increased robustness.

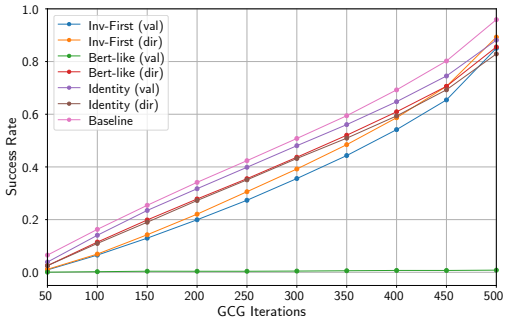

Fig. 3: GCG attack success rate (SR) as a function of the number of optimization iterations. Lower values indicate stronger robustness.

Table 4: GCG attack success rate (SR) for ILM variants. Lower is better.

Model	**SR** ↓	**Steps** ($\mu \pm \sigma$)
Baseline	95.9%	277 ± 148
Gradient as Value		
Inv-First	85.0%	320 ± 134
Bert-like	**0.8%**	249 ± 148
Identity	88.1%	274 ± 145
Gradient as Direction		
Inv-First	89.3%	313 ± 134
Bert-like	85.5%	287 ± 143
Identity	82.8%	284 ± 141

However, the *Bert-like (Gradient as Value)* variant exhibits an unusually low GCG success rate compared to all other models, suggesting markedly higher robustness to gradient-based white-box attacks. Given the magnitude of this effect, we repeated the experiments from the initial training phase to rule out potential artifacts. The results were consistent across runs, confirming the stability of this observation. Nevertheless, the underlying cause remains unclear and warrants further investigation.

We further analyze the relationship between the maximum number of GCG iterations and the attack success rate, defined as the fraction of tokens shared between the model outputs for the original input $\mathbf{x}$ and the adversarial input $\mathbf{x}'$. All other hyperparameters (e.g., search window width) are kept fixed. As shown in Figure 3, some curves intersect as the number of iterations increases, suggesting that different variants may be more effective under different optimization budgets. For example, *Inv-First (Gradient as Direction)* outperforms *Bert-like (Gradient as Direction)* at low iteration counts, while the opposite holds at the maximum number of iterations. Overall, this effect is limited and does not substantially alter the conclusions drawn from Table 4.

To further analyze GCG behavior, we report additional metrics computed on successful attacks. Specifically, we measure the cross-entropy loss on the original input, $\mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}), \mathbf{y})$, and on the adversarial input, $\mathcal{L}_{\text{CE}}(f_\theta(\mathbf{x}'), \mathbf{y})$, capturing how well the target continuation aligns with $\mathbf{x}$ and $\mathbf{x}'$, respectively. We also compute the KL divergence $KL\big(f_\theta(\mathbf{x}), f_\theta(\mathbf{x}')\big)$ to quantify the shift in the model's output distribution induced by the perturbation, where $f_\theta(\mathbf{x})$ denotes the output distribution over $\mathbf{y}$.

From the metrics in Table 5, we observe that robust variants, such as *Identity (Gradient as Direction)*, not only exhibit a substantially lower attack success rate compared to the baseline, but also show a smaller increase in cross-entropy loss when the attack succeeds. This increase, defined as the difference between

Table 5: GCG attack evaluation. **Left:** metrics computed on the best adversarial prefix found by GCG (cross-entropy loss and KL divergence). **Right:** average statistics over successful adversarial prefixes evaluated with a third-party model (perplexity and semantic similarity).

	Best Attack				**Average Prefix**		
	Orig. X CE ↓	**Attack X' CE ↓**	**Delta CE ↓**	**KL Div. ↑**	**Orig. X PPL**	**Attack X' PPL ↓**	**Semantic Similarity ↑**
Baseline	13.28	10.97	2.31	2.19	44.14	17344.04	0.13
Gradient as Value							
Inv-First	**11.09**	**9.72**	1.37	2.44	44.81	9431.09	0.16
Bert-like	13.26	10.25	3.01	**54.19**	40.37	11817.21	0.11
Identity	12.77	11.21	1.56	2.23	43.98	**8322.25**	**0.18**
Gradient as Direction							
Inv-First	11.21	9.81	1.40	2.44	43.50	12344.85	0.13
Bert-like	11.49	10.34	**1.15**	2.23	44.74	10611.09	0.13
Identity	12.58	11.12	1.46	2.47	44.71	10929.21	0.15

the loss on the original input and that on the adversarial sequence, quantifies the extent to which the perturbation misleads the model. Larger delta values indicate greater susceptibility, as the model assigns a higher likelihood to the target continuation $\mathbf{y}$ under the adversarial input.

Finally, the KL divergence measures the discrepancy between the output distributions $f_\theta(\mathbf{x})$ and $f_\theta(\mathbf{x}')$. A larger divergence indicates that the model distinguishes more effectively between the original and adversarial inputs, reflecting stronger robustness to perturbations.

To obtain a complete evaluation, we adopt a similar approach to the one we used during inversion: using a third-party model to compute additional statistics lets us abstract from the biases in our LLMs. Here, since the perplexity of the attack prefix is computed with a third-party independent model, it can easily return the real naturalness of the generated prefix, instead of being influenced by the attack itself and wrongly reporting that it will be even more natural than the human prefix.

Recall that we are considering only successful attacks, ignoring those that have failed, since they are not useful for understanding the quality of the attacks. Also, because of that, the results involving the `bert-like` variant with gradients as values will have a much smaller number of samples contributing to these metrics.

4.4 Forward Mode Evaluation

Finally, we verify that the proposed models retain proper forward-mode behavior, without experiencing performance degradation. By analyzing perplexity

Table 6: Forward-mode evaluation. Values report perplexity and cross-entropy (CE). Δ denotes change relative to the baseline.

	Perplexity ↓		**CE Loss ↓**	
Method	**Value**	Δ	**Value**	Δ
Baseline	**4.83**	–	**1.58**	–
Gradient as Value				
Inv-First	8.41	+3.58	2.13	+0.55
Bert-like	5.79	+0.96	1.76	+0.18
Identity	5.07	+0.24	1.63	+0.05
Gradient as Direction				
Inv-First	6.82	+1.99	1.92	+0.34
Bert-like	5.42	+0.59	1.69	+0.11
Identity	5.08	+0.25	1.62	+0.04

during both training and validation, we observe that introducing the gradient-based regularization term $\nabla_{\mathbf{e}}\mathcal{L}_{\mathrm{CE}}$ does not impair the model's ability to generate fluent text during standard forward inference. This result is particularly relevant, as adversarial training methods often require additional parameters or extended training to maintain comparable forward-mode performance, because part of the model's capacity is allocated to the adversarial objective.

As shown in Table 6, the worst-performing variant is *Inv-First (grad. value)*. Its higher perplexity is consistent with the qualitative examples reported below, in which the generated sentences appear less coherent and more repetitive than those from the other models.

Table 7: Example completion for the given prompt, in forward mode.

Method	**Completion for "One day,"**
Baseline	a little boy named Tim wanted to travel to a far mountain. He asked his dad for a raft,
	Gradient as Value
Inv-First	they pinch. They find gold. They take pictures of stars.
Bert-like	a little boy was walking in the park. He noticed a big, shiny object in the park.
Identity	a little girl named Lucy went to the park with her mom. Lucy liked to play on the swings
Inv-First	hey went to the beach with his mom. He saw something shiny and strange inside.
Bert-like	a little girl named Lucy was playing in the garden. She saw a shiny ring on a branch.
	Gradient as Direction
Identity	a little girl named Amy was playing outside. She saw a big tree and thought it was a toy.

5 Conclusions and Future Work

In conclusion, this paper introduces Inverse Language Modeling (ILM) as a novel framework that simultaneously addresses two critical challenges in Large Language Models: robustness and grounding. Our experiments demonstrate ILM's potential to enhance LLMs' resilience against input perturbations, a key step towards mitigating vulnerabilities to adversarial attacks. Furthermore, ILM offers a pathway to improved grounding, enabling LLMs to better correlate their outputs with the input prompts and thereby facilitating the identification of potentially problematic input triggers.

Crucially, this inversion capability may enable exploratory analysis of the input patterns underlying model outputs, which could serve as a preliminary signal for auditing model behavior, including in ethically sensitive contexts. For instance, when an LLM answers a controversial or value-laden question, ILM allows us to approximate the "implicit prompt" or internal framing the model uses. This makes ILM a promising tool not only for robustness but also for value transparency – providing a lightweight way to inspect how an LLM internally justifies its answers, and opening a research path toward value transparency that may eventually help communities detect misalignment with local ethical norms or social expectations.

There are several promising avenues for future research. While ILM is introduced in the context of pre-training, an interesting direction would be to explore its application during fine-tuning. Specifically, one could investigate how the principles of inverse modeling can be incorporated into the fine-tuning process to improve the robustness and generalization of LLMs on downstream tasks. Additionally, research could explore the potential benefits of combining ILM with instruction tuning to further align LLM behavior with human preferences and instructions. Future work should evaluate ILM on larger-scale LLMs to rigorously assess its scalability and effectiveness as model capacity increases.

Acknowledgements

This work was supported by projects PNRR MUR PE0000013FAIR under the MUR National Recovery and Resilience Plan funded by the European Union - NextGenerationEU and PRIN 2022 project 20227YET9B "AdVVent" CUP code B53D23012830006. It was also partially supported by Sapienza research projects "Prebunking", "Adagio", and "Risk and Resilience factors in disadvantaged young people: a multi-method study in ecological and virtual environments". Computing was supported by CINECA cluster under projects Ge-Di HP10CRPUVC, EHPC-DEV-2025D06-096 and the Sapienza Computer Science Department cluster.

Disclosure of Interests

The authors have no competing interests to declare.

Ethics and Impact Statement

The advancement of LLMs carries potential ethical implications, especially in the era of agents. Our investigation into this phenomenon contributes to a better understanding of how LLMs work and, thus, ultimately, to make them safer and more predictable. We believe that publishing our research will promote a broader discussion of the responsible development of LLMs and contribute to the development of stronger defense mechanisms, as similar progress has already been made in the field of deep classifiers.

References

1. Alon, G., Kamfonas, M.: Detecting language model attacks with perplexity. arXiv preprint arXiv:2308.14132 (2023)
2. Bender, E.M., Gebru, T., McMillan-Major, A., Shmitchell, S.: On the dangers of stochastic parrots: Can language models be too big? In: Proceedings of the 2021 ACM conference on fairness, accountability, and transparency. pp. 610–623 (2021)
3. Carlini, N., Jagielski, M., Choquette-Choo, C.A., Paleka, D., Pearce, W., Anderson, H., Terzis, A., Thomas, K., Tramèr, F.: Poisoning web-scale training datasets is practical. In: 2024 IEEE Symposium on Security and Privacy (SP). pp. 407–425. IEEE (2024)
4. Croce, F., Hein, M.: Reliable evaluation of adversarial robustness with an ensemble of diverse parameter-free attacks. In: International conference on machine learning. pp. 2206–2216. PMLR (2020)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. 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). pp. 4171–4186 (2019)
6. Drucker, H., Le Cun, Y.: Improving generalization performance using double back-propagation. IEEE transactions on neural networks **3**(6), 991–997 (1992)
7. Ebrahimi, J., Rao, A., Lowd, D., Dou, D.: Hotflip: White-box adversarial examples for text classification. In: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers). pp. 31–36 (2018)
8. Eldan, R., Li, Y.: Tinystories: How small can language models be and still speak coherent english? arXiv preprint arXiv:2305.07759 (2023)
9. Gage, P.: A new algorithm for data compression. The C Users Journal **12**(2), 23–38 (1994)
10. Ganz, R., Kawar, B., Elad, M.: Do perceptually aligned gradients imply robustness? In: International Conference on Machine Learning. pp. 10628–10648. PMLR (2023)
11. Goodfellow, I.J., Shlens, J., Szegedy, C.: Explaining and harnessing adversarial examples. arXiv preprint arXiv:1412.6572 (2014)
12. Inan, H., Khosravi, K., Socher, R.: Tying word vectors and word classifiers: A loss framework for language modeling. arXiv preprint arXiv:1611.01462 (2016)
13. Keung, P., Lu, Y., Szarvas, G., Smith, N.A.: The multilingual amazon reviews corpus. In: Proceedings of the 2020 conference on empirical methods in natural language processing (EMNLP). pp. 4563–4568 (2020)

14. Li, X., Li, Z., Li, Q., Lee, B., Cui, J., Hu, X.: Faster-gcg: Efficient discrete optimization jailbreak attacks against aligned large language models. arXiv preprint arXiv:2410.15362 (2024)
15. Liu, X., Li, P., Suh, E., Vorobeychik, Y., Mao, Z., Jha, S., McDaniel, P., Sun, H., Li, B., Xiao, C.: Autodan-turbo: A lifelong agent for strategy self-exploration to jailbreak llms. arXiv preprint arXiv:2410.05295 (2024)
16. Liu, X., Xu, N., Chen, M., Xiao, C.: Autodan: Generating stealthy jailbreak prompts on aligned large language models. arXiv preprint arXiv:2310.04451 (2023)
17. Melamed, R., McCabe, L.H., Wakhare, T., Kim, Y., Huang, H.H., Boix-Adsera, E.: Prompts have evil twins. In: Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing. pp. 46–74 (2024)
18. Morris, J.X., Zhao, W., Chiu, J.T., Shmatikov, V., Rush, A.M.: Language model inversion. arXiv preprint arXiv:2311.13647 (2023)
19. Paulus, A., Zharmagambetov, A., Guo, C., Amos, B., Tian, Y.: Advprompter: Fast adaptive adversarial prompting for llms. arXiv preprint arXiv:2404.16873 (2024)
20. Press, O., Wolf, L.: Using the output embedding to improve language models. In: Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers. pp. 157–163 (2017)
21. Rakotonirina, N.C., Kervadec, C., Franzon, F., Baroni, M.: Evil twins are not that evil: Qualitative insights into machine-generated prompts. In: Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP. pp. 48–68 (2025)
22. Sanh, V., Debut, L., Chaumond, J., Wolf, T.: Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. arXiv preprint arXiv:1910.01108 (2019)
23. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
24. Xhonneux, S., Sordoni, A., Günnemann, S., Gidel, G., Schwinn, L.: Efficient adversarial training in llms with continuous attacks. Advances in Neural Information Processing Systems **37**, 1502–1530 (2024)
25. Zhao, Y., Zheng, W., Cai, T., Long, X., Kawaguchi, K., Goyal, A., Shieh, M.Q.: Accelerating greedy coordinate gradient and general prompt optimization via probe sampling. Advances in Neural Information Processing Systems **37**, 53710–53731 (2024)
26. Zou, A., Wang, Z., Carlini, N., Nasr, M., Kolter, J.Z., Fredrikson, M.: Universal and transferable adversarial attacks on aligned language models. arXiv preprint arXiv:2307.15043 (2023)