PTP: PREVIOUS-TOKEN PREDICTION BASED LLM INVERSION FOR NEAR-EXACT PROMPT RECONSTRUCTION

Anonymous authors
Paper under double-blind review

ABSTRACT

Large language models (LLMs) generate text by auto-regressively sampling the next token. This inherently leads to a many-to-many mapping between prompts and responses, complicating the task of inferring prompts from observed outputs. Prior work on LLM inversion frames prompt recovery as a semantic reconstruction task. They rely on fine-tuning pretrained sequence-to-sequence models on large external datasets with access to model weights or logits to generate semantically plausible prompts. In contrast, we present a functional approach to inverting a given LLM in a black-box setting by training an explicit inverse language model entirely from scratch on data synthetically generated from the target LLM itself. Analogous to forward next-token prediction, our inverse model is trained using previous-token prediction (PTP), establishing a generative link between the forward and inverse processes that enables faithful prompt reconstruction. Moreover, it naturally supports diverse prompt reconstructions through sampling, whereby all such prompts induce similar responses under the forward, target LLM. Our approach generalizes across datasets and exhibits semantic transferability in reconstructing prompts from responses generated by different LLMs. PTP substantially outperforms prior work on exact prompt recovery with tokenizer access while remaining competitive on semantic reconstruction metrics in fully black-box setup.

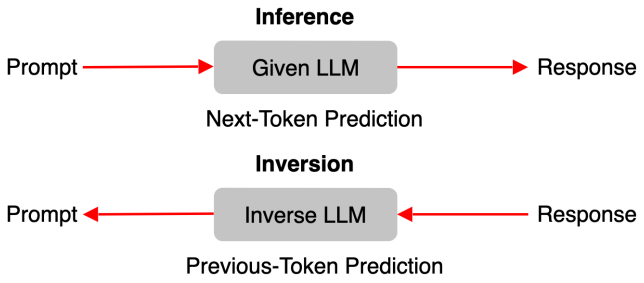


Figure 1: Our proposed approach to LLM Inversion using Previous-Token Prediction.

1 INTRODUCTION

Large Language Models (LLMs) are typically trained in a forward generative manner using the next-token prediction (NTP) objective. Given a prompt x, a causal language model f learns to model the conditional probability of the next token at each position. As a result, once trained, the model can generate a sequence auto-regressively from left to right, conditioned on the prompt x. In this work, we aim to *invert* the behavior of a pretrained, black-box LLM entirely from its textual responses. Specifically, given a response y, our goal is to recover a prompt $\hat{x}$ such that: $f(\hat{x}) \approx y$.

Rather than framing inversion as a semantic reconstruction problem, we propose a functional approach that mirrors the generative structure of the forward model itself, but by going backwards. We train an explicit inverse language model entirely from scratch on data synthetically generated from the target LLM, without access to model weights, logits, or internal representations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

To formalize, consider that at the core of a language model lies a next-token predictor that maps input text to a probability distribution over a fixed vocabulary $\mathcal{V}$. Specifically, the forward model f operates as: $f : \text{Context}_{<t} \to \mathcal{V}$, that is, $f(y_{<t}) = P(y_t \mid y_{<t})$. In contrast, our inverse model f_{inv} is trained to perform the reverse task of predicting the previous token given the future context: $f_{\text{inv}} : \text{Context}_{\geq t} \to \mathcal{V}$, that is, $f_{\text{inv}}(y_{\geq t}) = P(y_{t-1} \mid y_{\geq t})$ Thus, while the forward model is trained via NTP, the inverse model learns previous-token prediction (PTP) as shown in Figure 1, reconstructing preceding context that would induce the observed response under the given LLM.

Our general formulation of inversion recovers $\hat{x}$ that may not be identical to the original prompt which generated y, but results in similar generation. This implies that our approach can discover several prompts $\hat{x}$ which can yield responses similar to y which is distinctively useful in controlling undesirable responses through curtailing a diverse set of prompts that may yield those responses.

Contributions. Our main contributions include: (1) We propose *Previous-Token Prediction* (PTP)-based approach for LLM inversion that reconstructs prompts from observed model responses. (2) We reduce the external data dependence of inverse-model by introducing a synthetic data generation strategy in which the target LLM generates its own training data through vocabulary-level probing. (3) We demonstrate PTP as a practical approach to prompt reconstruction in a black-box setting validated by re-querying the forward model. (4) We show that the inverse model is both *generalisable* and *transferable* across datasets and responses obtained from different LLMs.

2 Prior Work

Network Inversion. Network inversion has been studied in Kindermann & Linden (1990) Jensen et al. (1999) Saad & Wunsch (2007) Wong (2017) as a means of understanding neural architectures by recovering input patterns that approximate a given output target. Inversion has also been explored in adversarial settings, with Yang et al. (2019) Kumar & Levine (2020) demonstrating model inversion attacks that reconstruct sensitive features without access to original training data and uncertainty-aware inversion methods Ansari et al. (2022) that constrain inverse solutions to reliable regions of the data distribution. To address optimization challenges in highly non-convex landscapes, Liu et al. (2022) proposed learning a modified loss landscape to accelerate convergence and improve stability. Generative approaches like Suhail & Sethi (2024), employ a conditioned generator for inversion of convolutional neural networks while encouraging diversity through feature-space constraints. Finally, deterministic formulations like Suhail (2024) encode networks into conjunctive normal form and apply SAT solvers and samplers to identify inverse solutions.

LLM Inversion. Morris et al. (2023) study *embedding inversion*, formulating recovery as controlled generation that iteratively decodes text whose embeddings re-encode close to a target latent vector. Complementarily, Morris et al. (2024) proposes Logit-to-Text(L2T) for *prompt inversion from next-token distributions*, showing that a model's probability vector over the vocabulary contains rich information about preceding text. Later, Nazir et al. (2025) propose *Prompt Inversion from Logprob Sequences* (PILS), exploiting the low-dimensional structure of next-token distributions to compactly represent and aggregate log-probabilities across generation steps, yielding substantial gains in exact recovery. Nikolaou et al. (2025) introduced *SipIt* suggesting that transformer language models are injective, mapping discrete input sequences to hidden representations enabling provable reconstruction of exact inputs from hidden activations. Gao et al. (2024) propose *DORY*, which reconstructs and refines candidate prompts using uncertainty derived from output probabilities, while Ye et al. (2025) propose *Inv^{2}A*, which exploits latent-space invariance and learns an inverse encoder that maps outputs to denoised latent representations. While effective, these approaches fundamentally rely on access to predictive distributions or internal representations, placing them outside a fully black-box setting.

In black-box regime, Zhang et al. (2024b) provide a systematic evaluation of prompt extraction attacks across multiple LLMs and prompt sources, demonstrating that even simple text-based attacks can reveal hidden system prompts in deployed models. Li & Klabjan (2025) propose *reverse prompt engineering*(RPE), a training-free black-box approach that queries the target LLM with its own responses and refines candidate prompts via search. Later, Zhang et al. (2024a) introduced Output-to-Prompt(O2P), which recovers prompts using only textual outputs from normal user queries, without access to logits or adversarial inputs.

Table 1: Comparison of Prompt reconstruction techniques including L2T (Logit-to-Text) Morris et al. (2024), O2P (Output-to-Prompt) Zhang et al. (2024a), with our proposed PTP(Previous-Token Prediction).

Dimension	**L2T**	**O2P**	**PTP (Ours)**
Access required	Full Logits	Text Outputs	Text Outputs
Training Data	Instructions-2M (2.33M)	Instructions-2M (30K prompts)	Synthetic Data + 400 prompts
Responses required	$\|V\|\times$ the number of bits required for desired precision	$\sim$64 outputs per prompt	Just One
Reconstructions	One	One	Multiple
Transferability	Poor across LLaMA-2	Moderate to GPT-3.5/4	Zero-Shot to GPT-4o

Comparison. Popular prompt inversion approaches for LLMs differ substantially in their assumptions, inversion mechanisms, and practical applicability as shown in Table 1. Logit-to-Text (L2T) Morris et al. (2024) relies on access to the model's full next-token probability distributions and uses pretrained sequence-to-sequence models to map these distributions back to prompts. L2T requires querying the model for a large number of logits per token, proportional to the vocabulary size and depends on extensive external training data. Output-to-Prompt (O2P) by Zhang et al. (2024a) operates purely on textual outputs in a black-box setting by fine-tuning pretrained sequence-to-sequence models to recover prompts from responses. However, O2P requires multiple responses corresponding to the same hidden prompt to achieve effective reconstruction and primarily succeeds at producing semantically similar prompts rather than exact token-level reconstructions.

In contrast, our Previous-Token Prediction (PTP) framework adopts a fundamentally different inversion strategy. Rather than fine-tuning a pretrained sequence-to-sequence model, we train an explicit decoder-based inverse LLM entirely from scratch using synthetically generated data obtained from the target model itself, without access to logits, gradients, embeddings, or original training data. The inverse model mirrors the forward model's training objective by predicting previous tokens, as in Pfau et al. (2023), given future context, establishing a principled generative link between forward and inverse processes. This enables near-exact prompt reconstruction at the level of individual tokens when possible, while naturally supporting the recovery of multiple distinct prompts through sampling. Importantly, PTP operates on a single observed response, produces a reusable inverse model, and demonstrates strong zero-shot transferability across different LLMs, all within a black-box setting. Together, these properties distinguish our approach as both more practical and more faithful to the underlying generative process than prior inversion techniques.

3 METHODOLOGY

Our approach to *LLM inversion* is centered on constructing a functional inverse of a pretrained black-box language model by learning an inverse generative model that mirrors the forward model's auto-regressive structure in reverse. The inverse model is trained entirely from scratch using synthetic sequences generated by probing the forward model itself, without access to external data, pretrained knowledge, or internal signals such as logits or embeddings.

Notation. Let $\mathcal{V}$ denote the finite vocabulary of the given language model, with cardinality $|\mathcal{V}|$, and let $\mathcal{V}^*$ denote the set of all finite-length token sequences over $\mathcal{V}$. A prompt is represented as a sequence $x = (x_1, \ldots, x_n) \in \mathcal{V}^*$, and a response as a sequence $y = (y_1, \ldots, y_T) \in \mathcal{V}^*$, where all sequences are defined at the token level using a fixed tokenizer. We consider a frozen causal language model f which induces a conditional distribution $P_f(y \mid x)$ over response sequences given a prompt and factorizes auto-regressively as $P_f(y \mid x) = \prod_{t=1}^{T} P_f(y_t \mid x, y_{<t})$. We further denote the inverse language model as f_{inv} that induces an auto-regressive conditional distribution over reversed prompt tokens given reversed response context, $P_{f_{\text{inv}}}(\tilde{x} \mid \tilde{y}) = \prod_{t=1}^{n} P_{f_{\text{inv}}}(\tilde{x}_t \mid \tilde{y}, \tilde{x}_{<t})$, where $\tilde{y} = \text{Reverse}(y)$ and $\tilde{x} = \text{Reverse}(x)$.

Definition 1 (**Prompt Inversion**). *Let f be a causal language model inducing a conditional distribution $P_f(y \mid x)$ over response sequences $y \in \mathcal{V}^*$ given prompt sequences $x \in \mathcal{V}^*$. Given an observed response sequence y, the* prompt inversion problem *consists of recovering a prompt sequence $\hat{x} \in \mathcal{V}^*$ such that a response generated by f conditioned on $\hat{x}$ is sufficiently similar to y. Specifically, let*

$$\hat{y}(x) \sim P_f(\cdot \mid x)$$

denote a response generated by the forward model from prompt x. We define the set of valid inverse prompts as

$$\mathcal{X}_y = \{x \in \mathcal{V}^* \mid \mathcal{S}(\hat{y}(x), y) \leq \varepsilon\},$$

where $\mathcal{S}(\cdot, \cdot)$ is a task-dependent distance measure between generated and target response sequences (e.g., based on token-level or semantic similarity), and $\varepsilon \geq 0$ denotes the tolerance of equivalence.

The overall methodology, proposed in Algorithm 1, can be decomposed into probing the forward model to synthesize data, sequence reversal, training the inverse model on the reversed data, fine-tuning the inverse model on prompt response pairs, and using the inverse model for inference.

1. Forward Model Probing LLMs internalize the statistical and semantic regularities of the training data within their parameters. However, in a black-box setting, without access to its weights we cannot exploit these for our inversion objective. Hence, we conceptualize the forward LLM as a function f mapping prompts to responses, and the inverse LLM as an approximate inverse of this function. Under this view, functional inversion must remain unaided, i.e., independent of pretrained inverse models or external data. Since we neither have access to model weights nor can use any real data to train the inverse model, we generate synthetic data using the forward model f. Utility of Synthetic Data is evaluated in Appendix Sec. F.

Definition 2 (**Synthetic Data Generation**). *For each token $v \in \mathcal{V}$ and a fixed generation length L, we construct a synthetic sequence $y^{(v)} = (v, y_2^{(v)}, \ldots, y_L^{(v)})$, where subsequent tokens are generated auto-regressively according to $y_t^{(v)} \sim P_f(\cdot \mid y_{<t}^{(v)}), \quad t = 2, \ldots, L$, using stochastic decoding with temperature τ, top-k, and top-p filtering. The resulting collection $\mathcal{D}_{probe} = \{y^{(v)} \mid v \in \mathcal{V}\}$ defines our synthetic corpus, which captures how the model maps initial tokens to most likely continuations under its learned distribution and is used to train the inverse model.*

These sequences can be generated *on the fly* during training, allowing for efficient memory usage and better diversity. This ensures that the sequence generated for the same base token v_i differs across successive epochs, introducing beneficial randomness into the dataset. Such stochastic probing encapsulates how the forward model transforms input tokens into output distributions, making them informative for learning the inverse mapping in f_{inv}.

2. Individual Sequence Reversal The inverse model f_{inv} is meant to predict preceding tokens conditioned on future context, such as inferring a plausible prompt from a model-generated response or reconstructing a question given its answer. To achieve this using the standard left-to-right next-token prediction objective, we transform each synthetically generated sequence by reversing it at the token level. Specifically, given a sequence $y^{(i)} = (y_1^{(i)}, \ldots, y_L^{(i)})$ generated by the forward model, we construct its reversed counterpart $\tilde{y}^{(i)} = \text{Reverse}(y^{(i)}) = (y_L^{(i)}, y_{L-1}^{(i)}, \ldots, y_1^{(i)})$. This token-level reversal enables the inverse model to maintain compatibility with standard auto-regressive training.

Remark 1. Reversal is applied after tokenization rather than at the raw text level. Reversing text at the character or word level can produce token sequences that are different under the model's tokenizer, even when derived from the same text. By first tokenizing the sequence in the standard forward direction and then reversing the resulting tokens, we preserve the original token inventory.

3. Inverse Model Training On the reversed token sequences $\tilde{y}^{(i)}$, we train the inverse model f_{inv} using the standard autoregressive next-token prediction objective. For each reversed sequence $\tilde{y}^{(i)} = (\tilde{y}_1, \tilde{y}_2, \ldots, \tilde{y}_L)$, the model is trained to maximize the log-likelihood of each token conditioned on its left context, $\sum_{t=2}^{L} \log P_{f_{\text{inv}}}(\tilde{y}_t \mid \tilde{y}_{1:t-1})$. This corresponds to minimizing the negative log-likelihood loss

$$\mathcal{L}_{\text{NLL}}(\theta_{\text{inv}}) = -\sum_{t=2}^{L} \log P_{f_{\text{inv}}}(\tilde{y}_t \mid \tilde{y}_{1:t-1}; \theta_{\text{inv}}),$$

where θ_{inv} denotes the learnable parameters of the inverse model. Optimizing this objective encourages f_{inv} to accurately predict each token in the reversed sequence given its causal history, thereby approximating the reverse dynamics of the forward generative process. Since the generated sequence length L is smaller than the maximum context window C of the model, we train on the full sequence at every position providing dense token-level supervision and maximizing data efficiency. As a result, the inverse model is trained using the same auto-regressive objective as the forward model, but applied to token-reversed sequences, enabling it to implicitly learn a mapping from future context to previous tokens.

4. Inverse Model Fine-Tuning Although the inverse model trained on synthetically generated sequences learns to predict preceding tokens given future context, a brief fine-tuning stage is beneficial to better format the reconstructed prompts. Accordingly, after pretraining f_{inv} on the synthetic probing corpus, we further fine-tune it on prompt–response pairs generated by the forward model. This step helps align the inverse model more closely with realistic linguistic patterns and semantics observed in natural language.

To construct the fine-tuning corpus, we sample prompts $x \in \mathcal{X}$ from a downstream dataset. For each prompt $x = (x_1, \ldots, x_n)$, the frozen forward model f produces a response $y = (y_1, \ldots, y_T)$ via standard autoregressive decoding. We concatenate the prompt and response to form a single sequence $s = (x_1, \ldots, x_n, y_1, \ldots, y_T)$, which is then reversed at the token level to obtain $\tilde{s} = \text{Reverse}(s) = (y_T, \ldots, y_1, x_n, \ldots, x_1)$. The inverse model is fine-tuned by maximizing the log-likelihood of the reversed sequence under standard autoregressive supervision, or equivalently by minimizing the negative log-likelihood

$$\mathcal{L}_{\text{NLL}}(\theta_{\text{inv}}) = -\sum_{t=2}^{|\tilde{s}|} \log P_{f_{\text{inv}}}(\tilde{s}_t \mid \tilde{s}_{1:t-1}; \theta_{\text{inv}}),$$

where θ_{inv} denotes the parameters of the inverse model. This encourages f_{inv} to model realistic prompt–response transitions induced by the forward model, improving the reconstruction quality.

5. Inverse Model Inference At inference time, given an observed response sequence $y = (y_1, y_2, \ldots, y_T)$, our goal is to reconstruct a plausible prompt $\hat{x}$ such that conditioning the forward model on $\hat{x}$ yields a response consistent with y. Inference proceeds by applying the trained inverse model f_{inv} in a causal reverse decoding manner.

We first tokenize the response sequence y using the same tokenizer employed during training and reverse the resulting token sequence to obtain $\tilde{y} = (y_T, y_{T-1}, \ldots, y_1)$. This reversed sequence $\tilde{y}$ is provided as context to the inverse model f_{inv}, which autoregressively generates a sequence of reversed prompt tokens $\tilde{x} = (\tilde{x}_1, \tilde{x}_2, \ldots, \tilde{x}_n)$ according to the learned conditional distribution. At each step t, tokens may be generated either greedily or via stochastic decoding,

$$\tilde{x}_t \sim P_{f_{\text{inv}}}(\cdot \mid \tilde{y}, \tilde{x}_{<t}),$$

with standard decoding controls such as temperature, top-k, or top-p sampling used to trade off fidelity and diversity. Once generation terminates, the reversed prompt tokens are inverted to recover the prompt in the original order,

$$\hat{x} = \text{Reverse}(\tilde{x}) = (\tilde{x}_n, \tilde{x}_{n-1}, \ldots, \tilde{x}_1).$$

The reconstructed prompt $\hat{x}$ can then be passed to the frozen forward model f to verify if it produces the target response y, or a semantically equivalent response. This inference procedure completes the inversion loop and mirrors standard auto-regressive generation in reverse. See Appendix Sec B for our observations about training.

4 EXPERIMENTS

We validate our proposed inversion approach across multiple LLMs, including Qwen3-0.6B and LLaMA-2 7B, with different model sizes and architectures on both the *base* and *chat* variants to assess the performance of our approach under instruction tuning and alignment fine-tuning. Although all evaluated models are open source, we restrict ourselves to a *black-box setup*, interacting with each model solely through textual input–output queries.

1. Synthetic Sequence Generation We experiment with generating synthetic sequences of varying lengths (64, 128, and 256 tokens) followed by token level. All generations are performed without instructions, relying solely on next-token prediction with top-$k = 5$, top-$p = 0.95$, and temperature 0.9. The sequences are generated *on the fly* during training, allowing for efficient memory usage and increased diversity. An example of the synthetic sequence generated from the forward model and its corresponding reverse is in the box. While the reversal is shown at word level for simplicity, *actual reversal* is done at token level.

Example of Synthetic and Reversed Sequences

Synthetic Sequence: Additionally, what are the health benefits of consuming chocolate, and how does it affect the body?
Reversed Sequence: ? body the affect it does how and, chocolate consuming of benefits health the are what, Additionally

2. Training the Inverse Model We use Qwen3-0.6B, as our inverse model, which is lightweight enough to be trained from scratch, initialized using He initialization He et al. (2015). We define an epoch as a complete pass over sequences corresponding to all tokens in the vocabulary. We observe that on-the-fly sequence generation leads to consistently better inversion performance, suggesting that increased diversity during training improves the generalization of the inverse model.

3. Fine-tuning Dataset This fine-tuning stage uses a small subset of around 25-30 token long prompts sampled from a dataset with responses generated from the given target LLM to be inverted. The inverse model is then trained using PTP to recover the original prompt from the response.

For fine-tuning, we make use of the **ShareGPT** or **Instructions-2M** dataset that is a collection of user conversations with GPT models, capturing diverse prompt styles and real-world usage patterns. For ShareGPT, we select 500 prompts, using 400 for training and reserving 100 for testing.

5 Results & Limitations

We present a comprehensive quantitative analysis of our proposed PTP-based LLM inversion framework. A combination of lexical and semantic similarity metrics, including **Exact Match (EM)**, **Token F1**, **BLEU**, **ROUGE-L**, **Cosine Similarity (CS)**, and **BERT F1** defined in Appendix Sec C are used for evaluating the prompt reconstruction quality. For qualitative results see Appendix Sec H.

5.1 Base Vs. Chat Models

We begin by validating the effectiveness of our approach on both the *base* and *chat* variants of Qwen3-0.6B to analyse the impact of instruction fine-tuning on prompt invertibility in Table 2. Across all evaluation metrics, the chat model consistently yields substantially higher reconstruction accuracy than the base model, with especially large gains in Exact Match and Token F1. This suggests that instruction tuning does not hinder prompt invertibility; rather, it appears to facilitate inversion.

Table 2: Prompt reconstruction results on Qwen Base (QB) and Qwen Chat (QC).

Model	Exact	BLEU	ROUGE-L	CS	BERT F1	Token F1
Qwen-Base	35.14 ± 4.4	40.58 ± 4.1	52.73 ± 3.6	74.39 ± 2.2	70.49 ± 2.3	33.11 ± 4.3
Qwen-Chat	**64.77 ± 4.54**	**63.08 ± 4.60**	**66.96 ± 4.33**	**86.13 ± 2.11**	**80.66 ± 2.7**	**63.64 ± 4.61**

5.2 Architecture & Vocabulary Mismatch

Next, we evaluate prompt reconstruction across models with differing architectures and vocabularies. Specifically, we compare prompt inversion for responses generated by Qwen3-0.6B Chat and LLaMA-2 7B Chat, while fixing the inverse model to be Qwen3-0.6B Chat in both cases as shown in Table 3.

This setting allows us to isolate the effect of architectural and tokenizer mismatch on inversion performance and assess how well the inverse model generalizes when applied to responses produced by a different model family. As expected, inversion performance is substantially higher when both the forward and inverse models belong to the same family, compared to inverting responses generated by LLaMA-2 7B Chat using a Qwen-based inverse model. The degradation is most pronounced on strict token-level metrics such as Exact Match and Token F1, reflecting vocabulary and tokenization mismatches. In contrast, semantic metrics such as cosine similarity and BERT F1 remain comparatively high, indicating that the inverse model is often able to reconstruct semantically aligned prompts even with architectural and vocabulary differences.

Table 3: Prompt reconstruction comparison between Qwen3 0.6B Chat and Llama2 7B Chat with Qwen as the inverse model.

Model	Exact	BLEU	ROUGE-L	CS	BERT F1	Token F1
Qwen	**64.77 ± 4.54**	**63.08 ± 4.60**	**66.96 ± 4.33**	**86.13 ± 2.11**	**80.66 ± 2.7**	**63.64 ± 4.61**
Llama	17 ± 2.4	17 ± 2.7	33 ± 2.8	84 ± 1.0	60 ± 2.0	12 ± 1.9

5.3 Transferability across LLMs

Can an inverse model trained on one forward LLM generalize to responses generated by a different LLM? To evaluate this, we test the inverse model trained on Qwen-3 0.6B Chat against responses produced by GPT-4o on the 100 test prompts. The response from GPT-4o is tokenised using Qwen tokeniser and reversed at the token level for the inverse model to reconstruct the prompt. The reconstructed prompt is then fed back into GPT-4o to generate a reconstructed response. Table 4 summarizes the transferability results wherein exact token-level reconstruction is substantially degraded in this cross-model setting for both prompt and response reconstruction because of differences in model architectures and tokenizer. Despite this, semantic similarity metrics remain comparatively high, particularly at the response level indicating that the inverse model is able to reconstruct prompts that preserve sufficient information to induce similar responses when re-injected into the forward model, demonstrating transferability across heterogeneous LLMs.

Table 4: Transferability Check: Prompt-Response reconstruction results for outputs of Chat-GPT4o inverted using an inverse model trained on Qwen3 0.6B Chat.

P/R	Exact	BLEU	ROUGE-L	CS	BERT F1	Token F1
Prompt	11.36 ± 2.79	6.56 ± 1.36	15.65 ± 1.89	**63.01 ± 1.71**	**82.26 ± 0.06**	10.20 ± 2.77
Response	1.71 ± 0.43	5.02 ± 0.8	18.56 ± 1.1	**81.57 ± 1.2**	**84.53 ± 0.38**	0.65 ± 0.4

5.4 Generalization across Datasets

To evaluate cross-dataset generalization, we train the inverse model on the Instruction-2M dataset (30K samples) using the Qwen3-0.6B forward model and test it on the ShareGPT test set. The results reported in Table 5 highlight that Semantic-level transfer is robust across datasets while token-level exact metrics are modest under cross-dataset transfer. Despite being fine-tuned on a different dataset, the inverse model retains strong semantic-level generalization as cosine similarity and BERT F1 remain high at both prompt and response level. This indicates that the inverse model learns a generalizable semantic mapping from responses to prompts that transfers across distributions.

Table 5: Cross-dataset generalization: Prompt-Response reconstruction results for inverse model fine-tuned on Instruction-2M (30K samples) and evaluated on the ShareGPT test set.

P/R	Exact	BLEU	ROUGE-L	CS	BERT F1	Token F1
Prompt	11.03 ± 2.19	25.80 ± 2.59	51.84 ± 2.76	73.11 ± 2.05	91.34 ± 0.57	9.13 ± 1.99
Response	4.44 ± 0.84	13.25 ± 1.42	23.82 ± 1.45	82.61 ± 1.78	85.14 ± 0.44	3.44 ± 0.81

Table 6: Prompt reconstruction results on Qwen3 0.6B Chat using PTP (Ours), O2P, and RPE.

Model	Exact	BLEU	ROUGE-L	CS	BERT F1	Token F1
PTP (Ours)	**64.77 ± 4.54**	**63.08 ± 4.60**	66.96 ± 4.33	86.13 ± 2.11	80.66 ± 2.70	**63.64 ± 4.61**
O2P-1	25.44 ± 3.00	40.59 ± 2.61	**69.92 ± 1.81**	**89.36 ± 0.76**	**84.09 ± 1.00**	21.00 ± 2.72
O2P-2	19.66 ± 2.49	34.28 ± 2.33	64.25 ± 1.87	87.70 ± 0.74	80.83 ± 1.01	15.48 ± 2.27
O2P-3	19.86 ± 2.57	33.45 ± 2.56	64.90 ± 2.02	86.97 ± 0.84	80.91 ± 1.10	16.05 ± 2.35
RPEGA	0.64 ± 1.74	2.35 ± 1.87	11.90 ± 4.32	65.29 ± 8.57	50.97 ± 4.07	0.26 ± 0.73

5.5 Baseline Comparisons - PTP Vs. O2P Vs. RPE

We compare our PTP-based LLM inversion framework against prior SOTA methods for prompt reconstruction. Since PTP operates entirely in a black-box setting using only textual model outputs, we restrict the comparison to Output-to-Prompt (O2P) Zhang et al. (2024a) and Reverse Prompt Engineering (RPE) Li & Klabjan (2025), a training-free LLM-as-optimizer framework.

RPE treats prompt inversion as a search problem and relies on iterative refinement coupled with genetic algorithms to evolve candidate prompts that reproduce the observed response. In contrast, O2P relies heavily on pretrained sequence-to-sequence models and large auxiliary datasets, which significantly contribute to its performance. PTP performs single-shot prompt reconstruction, without any iterative refinement, evolutionary search, or population-based optimization. While incorporating iterative refinement or genetic algorithms could plausibly improve PTP performance, doing so would introduce substantial computational overhead and fundamentally change the method's cost profile. We compare PTP against multiple O2P variants with increasingly favorable data conditions and evaluate RPE in its strongest reported configuration.

O2P and RPE evaluate inversion solely based on the ability to reconstruct the original prompt. In contrast, we additionally assess whether reconstructed prompts preserve the forward model's behavior. Specifically, after reconstructing prompts, we re-query the forward model to generate responses and compare these *reconstructed responses* against the original responses. This closed-loop evaluation reveals whether diverse reconstructed prompts still induce semantically similar model behavior. We evaluate all approaches on inversion of Qwen3-0.6B Chat to ensure a fair comparison on 100 test samples drawn from the ShareGPT dataset Zhang et al. (2024a) in following configurations:

1. **PTP (Ours)**: Pre-training Qwen from scratch with our inversion method, followed by fine-tuning on only 400 prompts from the ShareGPT dataset.
2. **O2P-1**: T5 is pretrained using outputs from Qwen with Instruction 2M–30k samples dataset, as in Zhang et al. (2024a), followed by finetuning on 400 ShareGPT samples.
3. **O2P-2**: Pre-trained T5 is fine-tuned on Instruction 2M–30k samples dataset with no further fine-tuning on ShareGPT dataset.
4. **O2P-3**: The pre-trained base T5 model is trained on 400 samples from the ShareGPT dataset without exposure to the larger Instruction 2M–30k samples dataset.
5. **RPEGA**: The best-performing variant from Li & Klabjan (2025) leveraging iterative evolutionary optimization evaluated on the same 100-sample ShareGPT test set.

Prompt Reconstruction: Table 6 shows that PTP substantially outperforms O2P on Exact Match, BLEU, and Token F1, while remaining competitive on ROUGE-L and semantic similarity metrics. While RPEGA performs substantially below PTP across all evaluation metrics. The gains are particularly pronounced for exact reconstruction attributed to the direct generative link preserved between the forward and the inverse model via previous-token prediction. While O2P attains high scores on semantic similarity metrics, our method achieves comparable performance on these measures despite being trained mostly on synthetic sequences.

Response Reconstruction: PTP-based approach consistently outperforms O2P in response reconstruction Table 7, demonstrating that reversing the auto-regressive process yields prompts that are functionally equivalent to the originals. While RPEGA attains moderate cosine similarity at the response level, it fails almost entirely on token level metrics. This suggests that training-free prompted inversion is insufficient for exact prompt recovery, and that a dedicated inverse model trained via PTP cannot be replicated by simply prompting a capable LLM. See Appendix Sec. G for Histograms.

Table 7: Response reconstruction results on Qwen3 0.6B Chat using PTP (Ours), O2P, and RPE.

Model	Exact	BLEU	ROUGE-L	CS	BERT F1	Token F1
PTP (Ours)	**61.79 ± 4.60**	**64.87 ± 4.30**	**71.43 ± 3.60**	88.08 ± 1.90	**85.48 ± 1.90**	**61.07 ± 4.70**
O2P-1	24.17 ± 2.96	33.57 ± 2.73	64.20 ± 2.00	86.16 ± 1.02	80.19 ± 1.24	19.28 ± 2.74
O2P-2	17.70 ± 2.38	27.85 ± 2.45	57.19 ± 2.11	83.77 ± 1.22	76.49 ± 1.27	13.09 ± 2.05
O2P-3	17.15 ± 2.43	24.83 ± 2.69	46.97 ± 3.01	70.57 ± 2.69	67.66 ± 2.08	12.86 ± 2.15
RPEGA	3.34 ± 8.44	14.39 ± 9.01	34.39 ± 11.08	**88.81 ± 9.40**	73.68 ± 5.55	1.52 ± 4.43

Observations: Across the O2P variants, we observe performance progression from O2P-3 to O2P-1 as more training data is introduced, highlighting the strong dependence of O2P on pretrained models and external datasets to achieve effective prompt reconstruction. In contrast, our PTP-based method achieves superior token-level recovery without relying on pretrained models or large external real-world datasets. Moreover, the consistent improvements across both lexical and semantic metrics indicate that our approach does not merely reconstruct semantically plausible prompts, but also preserves fine-grained lexical and structural details of the original inputs, underscoring the effectiveness of framing inversion as a generative reverse modeling problem rather than a data-driven semantic reconstruction task.

5.6 Cross-Backbone Ablations

Above experiments compare PTP with a decoder-only against O2P with a T5 inverse model . PTP fundamentally relies on reversed left-to-right autoregressive generation, which naturally requires a decoder-only architecture, whereas O2P is formulated as a sequence-to-sequence mapping and is therefore best served by an encoder–decoder model. To rule out architectural confounds, we conduct cross-backbone ablations in the Appendix Sec. E by implementing O2P with Qwen, and running PTP with a T5 backbone.

In all cases, performance degrades substantially and remains well below our main PTP with Qwen results. Notably, O2P implemented with the same Qwen backbone as PTP still fails to recover prompts at the token level, demonstrating that PTP's gains are driven by its inversion objective and synthetic pre-training strategy rather than model capacity or backbone choice. Similarly, implementing PTP using a pretrained T5 backbone leads to substantial degradation across all metrics. This is consistent with the expected directional penalty introduced when forcing T5 to perform reversed auto-regressive generation, which conflicts with the left-to-right sequence modeling priors encoded during pretraining.

5.7 Limitations and Future Work

Despite strong performance on semantic reconstruction, exact token level recovery remains sensitive to target LLM tokenizer access, vocabulary and architecture mismatch. While pre-training is data free, the current framework still requires lightweight fine-tuning on a minimal set of prompts to recover prompt formatting. Also synthetic data generation over the entire vocabulary of the target LLM introduces a non-trivial query budget. Future work will explore efficient probing strategies and iterative refinement mechanisms to improve scalability and reconstruction quality. More details related to scalability, data-efficiency, black-box assumptions, and probing strategy are provided in Appendix Sec. D.

6 Conclusion

We present a Previous-Token Prediction (PTP) based approach to LLM inversion using an inverse language model trained entirely from scratch on synthetic data for faithful prompt reconstruction. PTP outperforms prior black-box baselines on lexical metrics and preserves semantic behavior under closed-loop response reconstruction. PTP also exhibits meaningful semantic transferability across tokenizers, datasets, and models, when exact lexical prompt recovery is impeded.

AI USE STATEMENT

AI tools were used to assist with coding and preparation of the manuscript, including language editing, writing assistance, grammar correction, and improvements to clarity and presentation. AI generated code was reviewed and tested by the authors. AI-assisted manuscript edits were reviewed by the authors before inclusion in the paper. Generative AI was not used to formulate the research ideas, design the methodology, conduct the experiments, generate or interpret the experimental results, perform the scientific analysis, or draw the scientific conclusions of this work. The authors take full responsibility for the correctness, originality, claims, results, and final content of the paper.

ETHICS STATEMENT

This work studies the inversion of language-model outputs to reconstruct prompts and therefore has potential implications for privacy and security. In particular, prompt reconstruction techniques could potentially be misused to infer information that a model user or system operator did not intend to disclose. We present this work as a study of the invertibility and information-retention properties of language models, with the aim of improving understanding of such risks and informing the development of safer systems. Our experiments are conducted using model-generated responses and existing research datasets, and the proposed method is evaluated in a controlled experimental setting. We encourage responsible use of prompt-inversion research, particularly where model outputs may contain or reveal sensitive information.

REPRODUCIBILITY STATEMENT

This work is fully reproducible. The complete methodology, including synthetic data generation, token-level sequence reversal, inverse-model pre-training, fine-tuning, and inference, is described in Section 3, with the experimental and training setup provided in Section 4. The PTP Algorithm with end-to-end specification of the training and inference procedure, including optimization settings, is provided in Appendix A, while additional training details and empirical observations are reported in Appendix B. The evaluation metrics used throughout the experiments are defined in Appendix C, and practical considerations concerning computational requirements, data dependence, tokenizer access, black-box assumptions, and the probing strategy are documented in Appendix D. Additional cross-backbone ablations and analyses of synthetic pre-training are provided in Appendices E and F, respectively. Lastly, Appendices G and H provide distributional analyses and qualitative reconstruction examples that complement the quantitative results in the main paper. Together, these sections specify the methodological, experimental, and evaluation details necessary to reproduce the reported results.

REFERENCES

Navid Ansari, Hans-Peter Seidel, Nima Vahidi Ferdowsi, and Vahid Babaei. Autoinverse: Uncertainty aware inversion of neural networks, 2022. URL https://arxiv.org/abs/2208.13780.

Lirong Gao, Ru Peng, Yiming Zhang, and Junbo Zhao. Dory: Deliberative prompt recovery for llm, 2024. URL https://arxiv.org/abs/2405.20657.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

C.A. Jensen, R.D. Reed, R.J. Marks, M.A. El-Sharkawi, Jae-Byung Jung, R.T. Miyamoto, G.M. Anderson, and C.J. Eggen. Inversion of feedforward neural networks: algorithms and applications. *Proceedings of the IEEE*, 87(9):1536–1549, 1999. doi: 10.1109/5.784232.

J Kindermann and A Linden. Inversion of neural networks by gradient descent. *Parallel Computing*, 14(3):277–286, 1990. ISSN 0167-8191. doi: https://doi.org/10.1016/0167-8191(90)90081-J. URL https://www.sciencedirect.com/science/article/pii/016781919090081J.

Aviral Kumar and Sergey Levine. Model inversion networks for model-based optimization. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 5126–5137. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/373e4c5d8edfa8b74fd4b6791d0cf6dc-Paper.pdf.

Hanqing Li and Diego Klabjan. Reverse prompt engineering, 2025. URL https://arxiv.org/abs/2411.06729.

Ruoshi Liu, Chengzhi Mao, Purva Tendulkar, Hao Wang, and Carl Vondrick. Landscape learning for neural network inversion, 2022. URL https://arxiv.org/abs/2206.09027.

John Xavier Morris, Volodymyr Kuleshov, Vitaly Shmatikov, and Alexander M Rush. Text embeddings reveal (almost) as much as text. In *The 2023 Conference on Empirical Methods in Natural Language Processing*, 2023. URL https://openreview.net/forum?id=EDuKP7DqCk.

John Xavier Morris, Wenting Zhao, Justin T Chiu, Vitaly Shmatikov, and Alexander M Rush. Language model inversion. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=t9dWHpGkPj.

Murtaza Nazir, Matthew Finlayson, John X. Morris, Xiang Ren, and Swabha Swayamdipta. Better language model inversion by compactly representing next-token distributions, 2025. URL https://arxiv.org/abs/2506.17090.

Giorgos Nikolaou, Tommaso Mencattini, Donato Crisostomi, Andrea Santilli, Yannis Panagakis, and Emanuele Rodolà. Language models are injective and hence invertible, 2025. URL https://arxiv.org/abs/2510.15511.

Jacob Pfau, Alex Infanger, Abhay Sheshadri, Ayush Panda, Julian Michael, and Curtis Huebner. Eliciting language model behaviors using reverse language models. In *Socially Responsible Language Modelling Research*, 2023. URL https://openreview.net/forum?id=m6xyTie61H.

Emad W. Saad and Donald C. Wunsch. Neural network explanation using inversion. *Neural Networks*, 20(1):78–93, 2007. ISSN 0893-6080. doi: https://doi.org/10.1016/j.neunet.2006.07.005. URL https://www.sciencedirect.com/science/article/pii/S0893608006001730.

Pirzada Suhail. Network inversion of binarised neural nets. In *The Second Tiny Papers Track at ICLR 2024*, 2024. URL https://openreview.net/forum?id=zKcB0vb7qd.

Pirzada Suhail and Amit Sethi. Network inversion of convolutional neural nets. In *Muslims in ML Workshop co-located with NeurIPS 2024*, 2024. URL https://openreview.net/forum?id=f9sUu7U1Cp.

Eric Wong. Neural network inversion beyond gradient descent. In *WOML NIPS*, 2017. URL https://api.semanticscholar.org/CorpusID:208231247.

Ziqi Yang, Jiyi Zhang, Ee-Chien Chang, and Zhenkai Liang. Neural network inversion in adversarial setting via background knowledge alignment. In *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, CCS '19, pp. 225–240, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367479. doi: 10.1145/3319535.3354261. URL https://doi.org/10.1145/3319535.3354261.

Wentao Ye, Jiaqi Hu, Haobo Wang, Xinpeng Ti, Zhiqing Xiao, Hao Chen, Liyao Li, Lei Feng, Sai Wu, and Junbo Zhao. An invariant latent space perspective on language model inversion, 2025. URL https://arxiv.org/abs/2511.19569.

Collin Zhang, John Xavier Morris, and Vitaly Shmatikov. Extracting prompts by inverting LLM outputs. In Yaser Al-Onaizan, Mohit Bansal, and Yun-Nung Chen (eds.), *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 14753–14777, Miami, Florida, USA, November 2024a. Association for Computational Linguistics. doi: 10.18653/v1/2024.emnlp-main.819. URL https://aclanthology.org/2024.emnlp-main.819/.

Yiming Zhang, Nicholas Carlini, and Daphne Ippolito. Effective prompt extraction from language models. In *First Conference on Language Modeling*, 2024b. URL https://openreview.net/forum?id=0o95CVdNuz.

A PTP ALGORITHM

Algorithm 1 Training and Inference using Previous-Token Prediction based LLM Inversion

Require: Forward LLM f, vocabulary $\mathcal{V}$, Synthetic Sequence length L
Ensure: Trained inverse model f_{inv}
Synthetic Sequence Generation:
for each token $v \in \mathcal{V}$ **do**
Generate $y^{(v)} = (v, y_2, \ldots, y_L)$ using f with stochastic decoding
Reverse tokens: $\tilde{y}^{(v)} \leftarrow \text{Reverse}(y^{(v)})$
end for
Inverse Model Pre-training:
Train f_{inv} on $\{\tilde{y}^{(v)}\}$ using autoregressive NLL loss with Adam Optimiser and lr=1e-3.
Inverse Model Fine-Tuning:
for prompt x in a small downstream set **do**
Generate response $y \leftarrow f(x)$
Form $s \leftarrow (x, y)$, reverse $\tilde{s} \leftarrow \text{Reverse}(s)$
Fine-Tune f_{inv} on $\tilde{s}$ using Adam Optimiser and lr=5e-4.
end for
Inference:
Given response y, compute $\tilde{y} \leftarrow \text{Reverse}(y)$
Generate reversed prompt $\tilde{x} \sim f_{\text{inv}}(\cdot \mid \tilde{y})$
Output $\hat{x} \leftarrow \text{Reverse}(\tilde{x})$

B OBSERVATIONS ABOUT TRAINING

In pre-training the inverse model, we observe that longer training sequences substantially improve reconstruction performance. Consequently, we report primary results for models trained on 256-token sequences. Importantly, the framework can be easily extended to longer contexts, up to the forward model's maximum context length of 4096 tokens, suggesting significant room for further improvement.

Additionally, we find that on-the-fly sequence generation introduces greater diversity in the training data, reducing overfitting to specific contexts and leading to improved inversion quality compared to pre-generated sequences. By resampling responses at each epoch using stochastic decoding strategies (top-k, top-p, and temperature sampling), the same base token can give rise to multiple distinct contexts. This variability enriches the training distribution and allows the inverse model to learn a more robust mapping from responses to prompts, rather than memorizing fixed continuations.

Another key observation is the benefit of fine-tuning on real-world prompts after synthetic pre-training. While pre-training on synthetic sequences is sufficient to establish the structural inverse mapping, fine-tuning on curated datasets such as ShareGPT further aligns the inverse model with natural linguistic patterns. This two-stage training procedure yields an inverse model that is both structurally principled and semantically aligned, improving its ability to reconstruct realistic prompts beyond synthetic text. The improvements are especially noticeable in qualitative evaluations (Appendix Sec H), where fine-tuned models recover prompts that are nearly indistinguishable from the originals. We also note the role of forward model choice: the Qwen Chat variant consistently yields better inversion results compared to Qwen Base.

We also explore the **transferability** of the inverse model. Specifically, an inverse model trained on Qwen Chat is able to successfully reconstruct prompts when the responses come from a different forward model such as ChatGPT-4o. This demonstrates that the learned inverse mapping generalizes beyond the specific forward model used for training, highlighting the portability of the approach across architectures. Transferability underscores the practicality of the framework, showing that a single inverse model can invert multiple forward models without retraining.

C Description of Evaluation Metrics Used

To assess the effectiveness of the inverse model, we employ a range of complementary evaluation metrics that capture both exact lexical similarity and broader semantic alignment between reconstructed and ground-truth prompts:

- **Exact Match (EM):** A strict measure that checks whether the reconstructed prompt is identical to the ground-truth prompt at the token level. To account for variation in generation length, the reconstructed prompt is first truncated to the token length of the ground-truth prompt. Exact Match is then assigned if this length-aligned reconstruction is identical to the ground-truth token sequence.
- **BLEU:** A precision-oriented metric based on n-gram overlap, sensitive to exact word order and phrasing.
- **ROUGE-L:** A recall-oriented metric that evaluates the longest common subsequence (LCS) between predicted and reference text, capturing partial overlaps and fluency.
- **Cosine Similarity (CS):** Semantic similarity computed between sentence embeddings of the reconstructed and original prompts, using pretrained encoders. This metric is robust to paraphrasing.
- **BERT F1:** A semantic evaluation metric that aligns contextual embeddings from a pretrained BERT model and computes an F1 score over matches, capturing nuanced meaning beyond surface-level overlap.
- **Token F1:** Computes token-level precision, recall, and F1 between the complete reconstructed prompt and the ground-truth prompt, without length truncation. Unlike Exact Match, Token F1 therefore penalizes additional generated tokens beyond the ground-truth prompt length while providing partial credit for overlapping tokens.

D PRACTICALITY, DATA DEPENDENCE, AND BLACK-BOX ASSUMPTIONS

In this section, we address several practical concerns regarding the scalability, data dependence, black-box assumptions, and probing strategy underlying PTP.

D.1 EFFICIENCY AND PRACTICALITY

A key practical concern for black-box inversion methods is the query budget required to construct the synthetic pre-training corpus. We therefore provide a concrete estimate of the computational cost of synthetic data generation used in PTP.

The vocabulary size of Qwen3-0.6B is approximately

$$|\mathcal{V}| \approx 150{,}000.$$

For each token $v \in \mathcal{V}$, we generate a single synthetic sequence of length L using the forward model. The total number of queried forward-model tokens during one epoch of synthetic pre-training is therefore

$$|\mathcal{V}| \times L \;\approx\; 150{,}000 \times 256 \;\approx\; 38\text{M tokens}.$$

Although this budget scales linearly with the number of pre-training epochs under on-the-fly generation, the process remains computationally practical because each sequence is short and generated independently. Synthetic generation is therefore trivially parallelizable across GPUs and batched efficiently in practice. On a single A100 GPU, one epoch of synthetic corpus generation takes approximately one hour.

D.2 DATA DEPENDENCE AND THE ROLE OF FINE-TUNING

Our claim of being "data-free" pertains specifically to the synthetic pre-training stage. During pre-training, the inverse model is trained entirely on synthetic sequences generated directly from the target LLM without relying on any external instruction-response datasets.

The subsequent fine-tuning stage uses a minimal set of only 400 prompts from ShareGPT, compared to the 30K+ external instruction-response pairs used in O2P's main training pipeline. Critically, this fine-tuning stage serves only as a lightweight adaptation step rather than the source of the inverse capability itself.

Specifically:

(i) Only the *prompts* are drawn from ShareGPT. All responses used during fine-tuning are generated on the fly using the frozen forward model f.

(ii) The inverse model is trained entirely from scratch and does not inherit pretrained inversion capabilities from another model.

(iii) Appendix Sec. F demonstrates empirically that synthetic pre-training is essential for learning a meaningful inverse mapping. Models trained without synthetic pre-training exhibit high validation loss and fail to recover prompts reliably from limited real prompts alone. In contrast, models initialized with synthetic pre-training achieve substantially stronger reconstruction performance and validation accuracy.

(iv) Appendix Sec. F further shows that fine-tuning contributes primarily to formatting refinement rather than learning the inverse mapping itself. Models that undergo synthetic pre-training show minimal changes in validation loss and reconstruction accuracy during the fine-tuning stage, indicating that the inverse relationship is already established prior to fine-tuning.

Taken together, these results indicate that the central inversion capability of PTP arises from synthetic pre-training rather than from dependence on external supervised datasets.

D.3 Black-Box Assumptions and Tokenizer Access

We acknowledge that tokenizer access is required for *token-level exact reconstruction*, where the goal is to recover the precise sequence of token IDs corresponding to the original prompt.

However, it is important to distinguish token-level reconstruction from semantic reconstruction. Tokenizer access is not required for semantic prompt recovery. When the objective is semantic reconstruction rather than exact token recovery, the inverse model may operate entirely with its own tokenizer and vocabulary independently of the target model.

Tokenizer access becomes necessary only because our primary contribution focuses on strict token-level exact reconstruction, which requires consistency between the forward and inverse tokenization processes. We therefore view tokenizer access as an explicit assumption tied to the stricter evaluation setting rather than a hidden dependency of the method itself.

D.4 Sufficiency of Single-Token Vocabulary Probing

A natural question is whether probing the forward model using only single-token seeds sufficiently captures the structure of the prompt distribution.

The goal of single-token probing is not to reproduce the distribution of natural prompts directly. Instead, the objective is to expose the reverse conditional structure of the forward model across its entire vocabulary.

For every token $v \in \mathcal{V}$, we generate synthetic continuations conditioned on that token. Collectively, these generated sequences cover the model's token-transition dynamics across the vocabulary and expose the statistical regularities governing how the model maps contexts to continuations.

Moreover, stochastic generation with temperature, top-k, and top-p sampling ensures that the same seed token generates distinct trajectories across epochs. On-the-fly generation therefore substantially diversifies the synthetic corpus despite the simplicity of the probing seed.

Using larger seed combinations would require introducing assumptions about token ordering and prompt structure that are external to the forward model itself. Single-token probing avoids imposing such assumptions while preserving the data-independent nature of synthetic pre-training.

E Cross-Backbone Ablations

In the main paper, PTP is implemented using a decoder-only autoregressive inverse model, whereas the O2P baseline uses a T5 encoder–decoder architecture. A natural question is whether the observed gains arise from the inversion objective itself or simply from differences in backbone architecture.

We argue that direct backbone matching between PTP and O2P is fundamentally non-trivial because the architecture choice is dictated by the methodological formulation of each approach.

First, PTP fundamentally relies on reversed left-to-right autoregressive generation. The core mechanism of PTP is previous-token prediction over reversed sequences, which establishes a generative reverse process linking outputs back to prompts. This formulation naturally requires a decoder-only causal language model capable of strict autoregressive generation. In contrast, T5 is an encoder–decoder architecture that first encodes the entire input sequence before decoding the output sequence, making it fundamentally mismatched with the reversed autoregressive paradigm used in PTP.

Second, the value of T5 within O2P comes primarily from its pretrained sequence-to-sequence priors. Training T5 from scratch merely to match the training regime of PTP would remove the very advantage that motivates its use in O2P, making such a comparison uninformative. Conversely, adapting pretrained T5 to perform PTP requires fine-tuning it on reversed prompt–response sequences, which introduces a strong directional mismatch with the left-to-right generative priors encoded during pretraining. Fine-tuning T5 to autoregressively generate reversed sequences therefore antagonizes the pretrained priors baked into the decoder. Any resulting performance degradation should thus be attributed to degradation of pretrained priors rather than to limitations of the PTP objective itself.

Finally, the backbone choice reflects a deeper methodological distinction between the two approaches. O2P formulates inversion as a sequence-to-sequence mapping problem, naturally suited to encoder–decoder architectures. In contrast, PTP formulates inversion as a reverse autoregressive generation problem, naturally suited to decoder-only causal language models.

Nevertheless, to provide the most complete comparison possible, we conduct cross-backbone ablations in both directions: implementing O2P using the same Qwen3-0.6B decoder-only backbone used by PTP, and implementing PTP using a T5 backbone. Tables 8 and 9 report prompt and response reconstruction results respectively.

Table 8: Prompt reconstruction results for cross-backbone ablations. O2P is implemented using the same Qwen3-0.6B backbone as PTP, while PTP is additionally evaluated using a T5 inverse model.

Method	Exact	BLEU	ROUGE-L	CS	Token F1	BERT F1
O2P-1 (Qwen)	12.39 ± 20.38	5.99 ± 3.65	21.46 ± 5.47	76.29 ± 13.62	9.31 ± 16.86	58.09 ± 5.56
O2P-2 (Qwen)	4.06 ± 8.12	4.53 ± 3.30	20.19 ± 5.45	71.34 ± 10.25	2.48 ± 5.73	57.16 ± 7.01
O2P-3 (Qwen)	16.10 ± 22.27	5.47 ± 3.51	20.84 ± 5.10	80.33 ± 9.77	12.24 ± 19.03	59.47 ± 5.52
PTP (T5)	2.82 ± 7.73	2.69 ± 8.54	12.58 ± 11.60	47.69 ± 12.94	1.58 ± 6.04	47.05 ± 8.86
PTP (Qwen, Ours)	**64.77 ± 4.54**	**63.08 ± 4.60**	**66.96 ± 4.33**	**86.13 ± 2.11**	**63.64 ± 4.61**	**80.66 ± 2.70**

Table 9: Response reconstruction results for cross-backbone ablations. O2P is implemented using the same Qwen3-0.6B backbone as PTP, while PTP is additionally evaluated using a T5 inverse model.

Method	Exact	BLEU	ROUGE-L	CS	Token F1	BERT F1
O2P-1 (Qwen)	1.26 ± 1.67	3.00 ± 7.27	15.68 ± 12.54	67.67 ± 18.52	0.32 ± 0.47	49.52 ± 10.94
O2P-2 (Qwen)	1.33 ± 1.28	4.67 ± 8.11	20.11 ± 11.62	71.06 ± 19.15	0.33 ± 0.37	52.79 ± 10.56
O2P-3 (Qwen)	1.21 ± 1.23	2.98 ± 4.76	19.30 ± 10.47	73.58 ± 11.59	0.39 ± 0.52	52.81 ± 8.54
PTP (T5)	1.90 ± 1.24	1.90 ± 4.49	15.86 ± 9.03	59.27 ± 21.26	0.69 ± 1.80	51.36 ± 9.59
PTP (Qwen, Ours)	**61.79 ± 4.60**	**64.87 ± 4.30**	**71.43 ± 3.60**	**88.08 ± 1.90**	**61.07 ± 4.70**	**85.48 ± 1.90**

The results strongly support the conclusions presented in the main paper. Replacing T5 with Qwen as the inverse model for O2P does not improve performance. All O2P variants implemented using the same Qwen backbone as PTP remain substantially below PTP on all token-level metrics for both

prompt and response reconstruction. This demonstrates that the performance gains of PTP are not explained by model capacity or backbone choice alone.

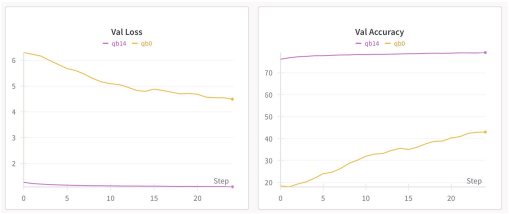
Similarly, implementing PTP using a pretrained T5 backbone leads to substantial degradation across all metrics. This is consistent with the expected directional penalty introduced when forcing T5 to perform reversed autoregressive generation, which conflicts with the left-to-right sequence modeling priors encoded during pretraining.

Taken together, these experiments provide strong evidence that PTP's gains arise from the inversion formulation itself — namely previous-token prediction over synthetically generated sequences — together with the synthetic pre-training strategy that establishes a principled generative reverse process between responses and prompts.

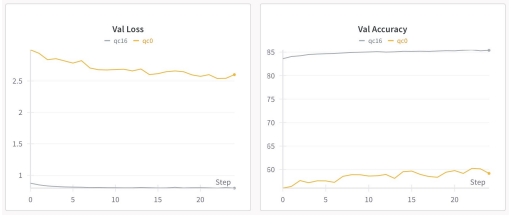
F SYNTHETIC DATA IS VERY HELPFUL

Figures 2a and 2b illustrate the impact of synthetic data pre-training on inverse model performance for Qwen3 Base and Qwen3 Chat, respectively. In both cases, we compare two training regimes: (i) pre-training the inverse model on synthetically generated sequences followed by fine-tuning on 400 ShareGPT prompts, and (ii) training only on the same 400 ShareGPT prompts without synthetic pre-training. The results show a stark contrast between the two settings. Models trained without synthetic pre-training exhibit high validation loss and poor accuracy, failing to learn a meaningful inversion mapping from limited real prompts alone.

In contrast, synthetic pre-training leads to rapid loss convergence and substantially higher validation accuracy, with fine-tuning providing only incremental refinement. This behavior is consistent across both base and chat variants, highlighting that large-scale synthetic probing is essential for learning the inverse generative structure of the forward LLM, and that small real-world prompt sets are insufficient on their own. These results underscore the central role of synthetic data in enabling stable and effective LLM inversion in an almost black-box setting.



(a) Qwen3-0.6B Base. **Yellow**: training without synthetic pre-training (ShareGPT only). **Purple**: synthetic pre-training followed by fine-tuning.



(b) Qwen3-0.6B Chat. **Yellow**: training without synthetic pre-training (ShareGPT only). **Grey**: synthetic pre-training followed by fine-tuning.

Figure 2: Effect of synthetic data pre-training on inverse model optimization. Synthetic pre-training enables stable convergence and substantially higher validation accuracy, while training on limited real prompts alone fails to learn a meaningful inverse mapping.

G PROMPT-RESPONSE RECONSTRUCTION HISTOGRAMS

To further analyze reconstruction behavior beyond aggregate scores, Figures 3 and 4 visualize the full distribution of reconstruction metrics using histograms, with prompt- and response-level scores shown side by side for each metric. Across all six metrics, prompt reconstruction exhibits a markedly bimodal distribution: a large mass near perfect scores (e.g., Exact Match, BLEU, ROUGE-L, Token F1 close to 1) corresponding to near-exact recoveries, alongside a smaller tail of lower-scoring reconstructions that deviate from the reference prompt at the lexical level. In contrast, the corresponding response-level histograms are significantly more concentrated toward high scores, with reduced variance across all metrics. This pattern indicates that even when prompt reconstruction is not exact at the token level, the reconstructed prompts still induce highly similar responses under the forward model.

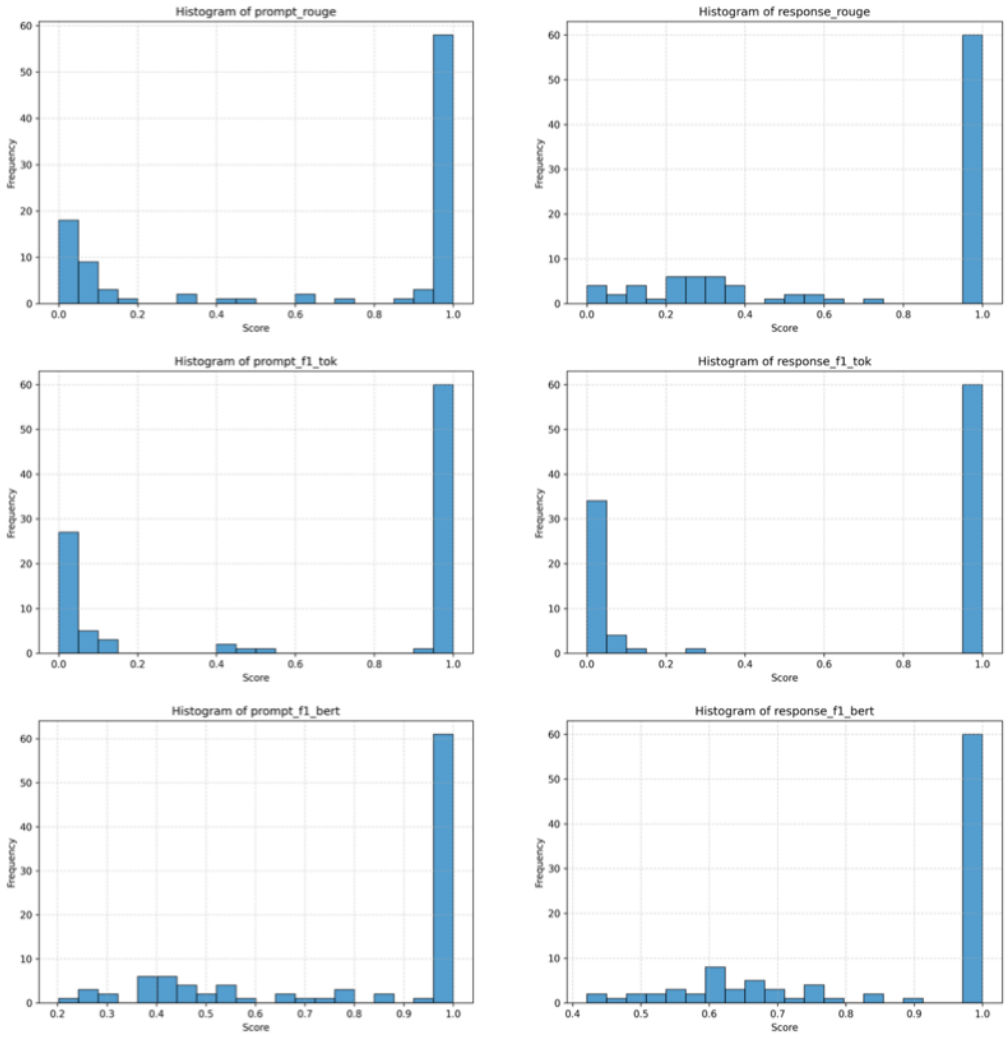


Figure 3: Distributions of prompt- and response-level reconstruction scores for **ROUGE-L**, **Token F1**, and **BERT F1**. Across all metrics, response-level distributions are more tightly concentrated than prompt-level ones, demonstrating that diverse reconstructed prompts consistently induce semantically similar responses.

Notably, this effect is most pronounced for semantic metrics such as cosine similarity and BERT F1, where response distributions collapse tightly around high values despite broader dispersion at the prompt level. These histograms empirically reinforce the many-to-one nature of the forward generative mapping: diverse prompts uncovered by the inverse model often lie within the same response-equivalence class. Taken together, the histogram analysis complements the quantitative results in Tables 6 and 7, demonstrating that our PTP-based inversion not only achieves strong average performance, but also reliably preserves functional response behavior across a wide spectrum of reconstructed prompts.

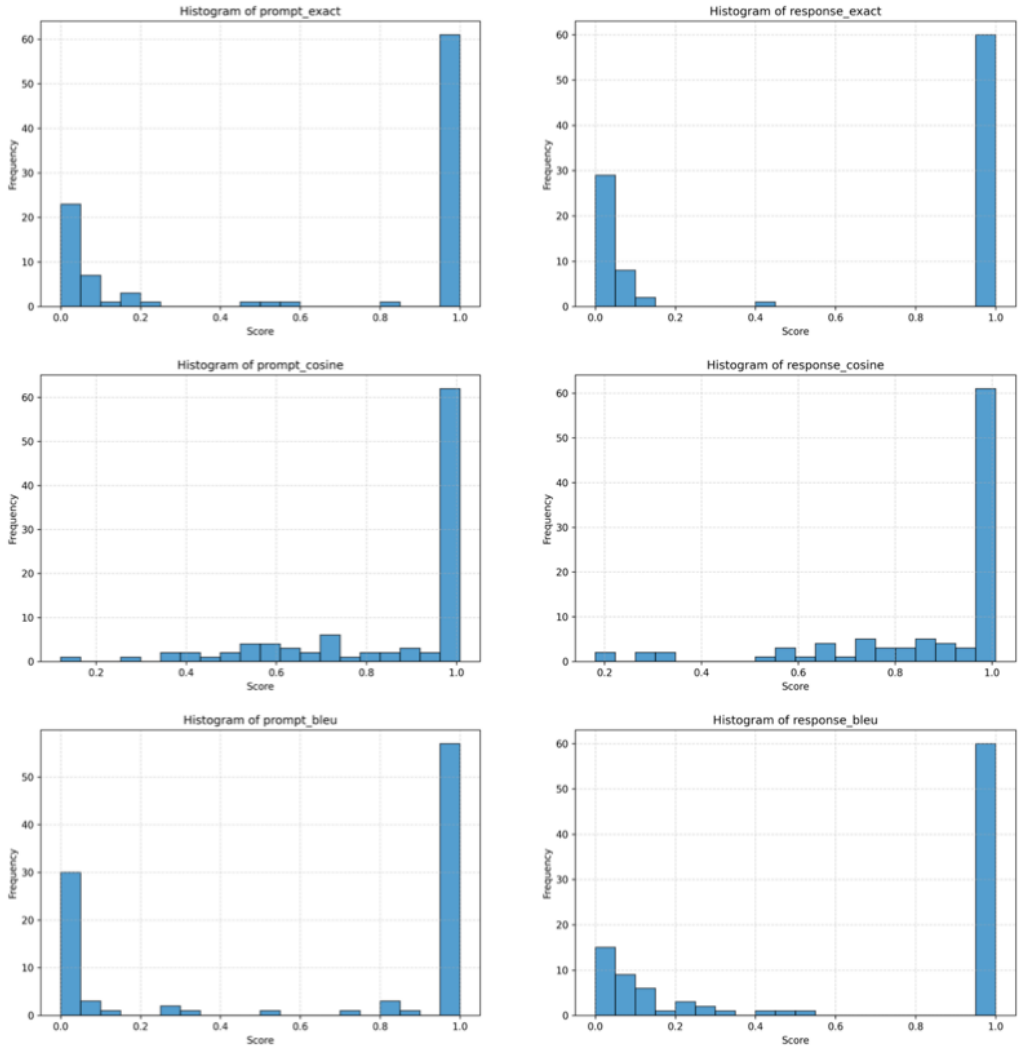


Figure 4: Distributions of prompt- and response-level reconstruction scores for **Exact Match**, **Cosine Similarity**, and **BLEU**. For each metric, the left histogram corresponds to prompt reconstruction and the right to response reconstruction. Prompt-level distributions exhibit higher variance, while response-level scores concentrate near high values, indicating functional equivalence despite prompt diversity.

H QUALITATIVE RESULTS

We next look at some of the qualitative results obtained from our inverse LLM. These include both multiple prompt reconstructions for a given response as well as response reconstructions from recovered prompts. The evaluation is carried out not only on the unseen test prompts but also on random prompts, in order to illustrate the generality of the approach beyond the curated test set. These qualitative examples highlight that our method can recover exact prompts, generate semantically diverse alternatives, and still yield responses closely aligned with the original forward model outputs.

H.1 MULTIPLE PROMPT RECONSTRUCTION

An important advantage of training an inverse LLM is the ability to generate *multiple* plausible prompts for the same response. Since inference is performed via previous-token prediction, the inverse model can be sampled repeatedly to produce diverse reconstructions by varying decoding parameters such as top-k, top-p, and temperature. This goes beyond recovering a single deterministic prompt and instead enables exploration of the space of prompts that lead to semantically similar responses under the forward model.

To illustrate, consider the above example. The original random prompt and response are shown, along with multiple reconstructed prompts sampled from the inverse model. This example highlights that while the model is able to recover the exact prompt, it can also generate diverse alternatives that preserve the semantic intent of the original prompt.

Example of Multiple Prompt Reconstruction

Prompt: How to reach out to competitors to find their pricing strategies?
Response: Reach out to competitors to gain insights into their pricing strategies and identify areas for improvement or innovation. Here are some steps you can follow:
1. Identify Competitors: Determine the specific market or industry you want to compete in. Identify your closest competitors and other potential competitors.
...
7. Use the Insights: Use the information you gain from your competitors' responses to inform your own pricing strategy. Consider how their pricing strategies can be adapted to meet your own business needs.
Reconstructed Prompts:
1. What do you need to know about their pricing strategies to invest in market research, competitive analysis, and competitive planning?
2. What are their strengths and weaknesses in their pricing strategies? What are the key challenges that my companies face in their pricing strategies?
3. How to reach out to competitors to find their pricing strategies?
4. How do you identify the most effective marketing strategies to attract potential clients? Choose from a wide range of competitors and provide insights into their pricing strategies?
5. Can you provide me with a list of pricing strategies for businesses that compare to its competitors? Also, are you looking to target competitors and their pricing strategies for their product/service?
6. What do we need to know about the potential opportunities for marketers to reach out to competitors to find their pricing strategies?
7. What tactics can a company looking to reach out to competitors in the market use to find their pricing strategy?

H.2 PROMPT RECONSTRUCTION ON SHAREGPT

To further evaluate our method beyond synthetic sequences, we test the inverse model on real-world prompts drawn from the ShareGPT dataset. In this setting, prompts are selected from ShareGPT, their responses are generated by the forward model, and the inverse model is tasked with reconstructing the original prompts from the responses alone. We present examples below for both the Qwen Base (QB) and Qwen Chat (QC) variants. These qualitative cases illustrate that while the reconstructed prompts may vary in surface form, they often remain semantically faithful to the original intent.

Qwen Base on ShareGPT (Unseen Test Prompt)

Inverse Model Input: The expiration effect in equities refers to the phenomenon where the price of an option moves in a way that is not solely driven by the intrinsic value of the option but also by the time value of the option. This effect is particularly pronounced in options with shorter expiration dates, such as call options. Here's a fundamental reason behind this effect and how to exploit it in a trading strategy: Fundamental Reason Behind the Expiration Effect
1. **Time Value**: The time value of an option is the difference between its premium and its intrinsic value. As the expiration date approaches, the time value of the option decreases because the probability of the option expiring in-the-money diminishes.. . .
Inverse Model Output (Reconstructed Prompt): Give me a brief explanation of the fundamental reason behind the expiration effect in equities and how to exploit this information in trading.
Actual Prompt: Fundamental reason behind option expiration effect in equities and how to exploit this information in creating a simple trading strategy with code.

Qwen Chat on ShareGPT (Unseen Test Prompt)

Inverse Model Input: Let me know if there are any other steps I could take. Yes, creating an advertising campaign on Pinterest can be an effective strategy. Here are some examples of steps you could take: 1. Define your campaign goals: Set a specific objective such as increasing brand awareness or driving sales. 2. Create visually appealing content: Use high-quality images and videos that reflect Semilac's values and products. . . . 6. Collaborate with influencers: Partner with influencers who have a strong presence on Pinterest to amplify your brand's reach. 7. Run ad campaigns
Inverse Model Output (Reconstructed Prompt): Give me examples of steps I could take to create an advertising campaign on Pinterest for Semilac brand. How can I use this information to make the campaign effective?
Actual Prompt: Give me examples of steps I could take to create an advertising campaign on Pinterest for Semilac brand.