PrefixNLI: Detecting Factual Inconsistencies as Soon as They Arise

Anonymous ACL submission

Abstract

Natural Language Inference (NLI) models have been used in various ways to improve the factuality of LLM outputs. This is typically done by applying an NLI model to judge whether the model output is entailed from the supposed evidence, triggering some corrective actions, such as beam reranking at inference time or RL rewards during training. While NLI models are trained to detect factual inconsistencies over complete sentences, decisions in the common autoregressive generation architecture are made for each evolving text prefix, during decoding. Addressing this setting, we generalize the entailment detection task to apply over arbitrary text *prefixes*, and suggest its utility for improving generation faithfulness. Providing suitable evaluation and training datasets for this task, we train *MiniTruePrefixes*, a novel specialized model that better detects factual inconsistencies over text prefixes, outperforming comparable baseline NLI models by 5-14 F1 points in prefix-level entailment. We further demonstrate that integrating MiniTruePrefixes into a controlled decoding framework substantially improves factual consistency in abstractive summarization. When guided by MiniTruePrefixes, LLaMA-3.2-3B-Instruct matches the faithfulness and runtime of the 8B model from the same model family, while using only half the memory.[1]

1 Introduction

Large Language Models (LLMs) have made remarkable progress in text generation, yet they remain prone to generating factually inconsistent statements, known as hallucinations (Mishra et al., 2024). In this paper, we focus on generation scenarios where the model output is expected to be factually consistent with supporting textual evidence. Such evidence may be explicitly provided as part of the task input, as in text summarization, or may be retrieved in a Retrieval Augmented Generation (RAG) setting (Lewis et al., 2020; Izacard et al., 2023).

Directing LLM generation to be factually consistent with the given inputs is challenging. A notable research line addresses this goal by providing the LLM with factual consistency feedback over the generated text. This is commonly done by employing a Natural Language Inference (NLI) model (Dagan et al., 2005; Bowman et al., 2015), which classifies whether the generated text, considered as the hypothesis, is entailed by the given source texts, considered as the premise. Such entailment feedback has been provided either at training time, by incorporating the entailment score as a Reinforcement Learning (RL) reward (Roit et al., 2023), or at inference time, by utilizing the entailment score for reranking the decoding beam during generation, within a controlled decoding scheme (Wan et al., 2023; Sridhar and Visser, 2023). Since LLM generation is predominantly autoregressive, generating one token at a time, it would be appealing to provide the model with factual consistency feedback at each generation stage, that is, over sentence *prefixes*. However, the entailment recognition task has been originally defined for hypotheses that consist of one or more *complete* sentences, and NLI models have been accordingly trained over datasets with such hypotheses. This discrepancy has led to certain compromises in prior work. In the RL setting, the entailment RL reward was provided only at the end of complete sentences, missing the opportunity for earlier and more granular feedback. In the controlled decoding framework, prior methods greedily completed text prefixes during generation to complete texts (a "lookahead"), which were only then scored for entailment; this introduced noise in judging entailment for the prefix itself, while also incurring significant computational costs (see §2).

In this paper, we propose providing LLMs with

[1] Code and sample data can be found at: https://anonymous.4open.science/r/PrefixNLI;

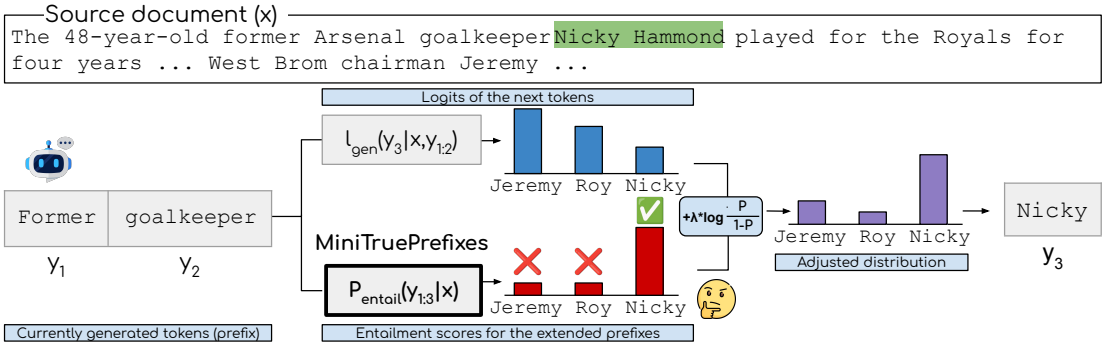

Figure 1: Illustration of PrefixNLI and its downstream utilization for controlled decoding. During autoregressive generation, the base model favors a hallucinated token ("Jeremy") that is not supported by the source document. Our MiniTruePrefixes model directly evaluates the factual consistency of candidate prefixes at each generation step, assigning here a low entailment score to this unfaithful continuation. For tokens with low entailment probability, we apply a penalty, effectively discouraging unfaithful continuations. This guides generation toward faithful outputs ("Nicky") in a fine-grained and efficient manner.

entailment scoring feedback that is computed *directly* for each text prefix during the autoregressive generation. To that end, we first introduce the *PrefixNLI* task, which extends the traditional textual entailment definition to apply over arbitrary text prefixes as hypotheses, and introduce suitable test and training datasets for this task (§3). Next, we train a dedicated NLI model for the PrefixNLI task, MiniTruePrefixes (§4), and show, in an intrinsic evaluation, that it significantly outperforms standard NLI models on prefix-level inference, with relative improvements of over 5 and 14 F1 points across two prefix-level entailment evaluation sets (§5). Finally, we apply our MiniTruePrefixes model within a controlled decoding framework, showing consistent faithfulness gains across model sizes and datasets on the abstractive summarization task, including a +5.5 faithfulness points improvement over the LLaMA-3.2-8B-Instruct baseline (Llama Team et al., 2024), while remaining much faster than prior controlled decoding methods (§6). For memory-efficient settings, we match the performance and speed of LLaMA-3.2-8B-Instruct with half the memory consumption. These contributions are illustrated in Figure 1. In light of our results, we suggest the broader potential of enhancing faithfulness in text generation via prefix-based NLI, including its incorporation within token-level reinforcement learning in future research.

In summary, our main contributions include:

- Introducing the **PrefixNLI** task and accompanying datasets, extending natural language inference to arbitrary text prefix hypotheses.
- Developing MiniTruePrefixes, the first entailment model trained specifically for prefix-level inference, establishing a strong baseline on our proposed benchmark.
- Integrating prefix entailment into a *controlled decoding* framework, demonstrating significant factual consistency improvements while maintaining efficiency.

2 Background

A persistent challenge in grounded natural language generation tasks such as summarization (Maynez et al., 2020; McKenna et al., 2023) and reference-based question answering (Zhang et al., 2023) is avoiding *factual inconsistencies*, that is, generating text fragments that are not supported by the grounding source ("unfaithful"). Prior work has explored various strategies for improving the faithfulness of text generation, broadly categorized into training-time, generation-time, and post-generation methods. Post-generation methods, such as RARR (Gao et al., 2023a), detect and revise unsupported spans only *after* generation, possibly using retrieved evidence. Potentially more appealing approaches aim to direct models to generate faithful outputs up front. Within training-time approaches, this has been done by modifying the model's learning objective (Roit et al., 2023; Tian et al., 2023) or by improving the training data (Wan and Bansal, 2022). Generation-time methods integrated faithfulness estimation into the decoding process, and utilized it to modify the ranking of candidate paths within the decoding beam, in a controlled decoding framework (Wan et al., 2023; Sridhar and Visser, 2023). An alternative generation-time approach,

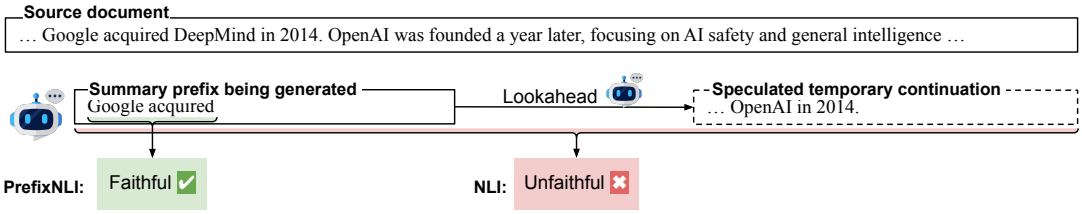

Figure 2: Comparing faithfulness assessment approaches in controlled decoding. PrefixNLI evaluates the faithfulness of the generated prefix itself. In contrast, lookahead-based methods (Wan et al., 2023; Sridhar and Visser, 2023) first complete the summary before evaluation, which would be misleading in case the completed summary is found unfaithful due to a factual inconsistency that arises only within the speculated completion, as illustrated in the figure.

context-aware decoding (Shi et al., 2024), encourages the model to rely more on the input context rather than prior knowledge, by contrasting the model's output probabilities with and without the context input. However, such a method does not explicitly check whether the output is factually faithful to the source.

A prominent approach for estimating faithfulness, in either training or decoding time, has been to employ NLI models. Roit et al. (2023) utilized such a model to compute sequence-level faithfulness rewards within a reinforcement learning framework, encouraging faithful generation. Since NLI models are inherently trained to consider hypotheses consisting of complete sentences, the reward was applied only at the end-of-sequence (EOS) token, assigning a zero reward to all other tokens. For a similar reason, Wan et al. (2023) and Sridhar and Visser (2023) applied a faithfulness score to candidate decoding beam paths only after applying a lookahead mechanism, which greedily generated a (temporary) full summary completion for each candidate token in the beam. Thus, the entailment estimation was applied not only to the currently examined prefix at each decoding step, but rather to a speculated complete summary that augments the current prefix. This method suffers from two substantial limitations. First, the decoding process becomes very costly computationally, since a complete temporary summary has to be generated for each candidate token at each generation step. Second, the entailment estimation is noisy, as it is not applied solely to the currently examined candidate prefix, but rather to the full speculated summary that was generated ad-hoc to complete the examined prefix. Thus, if the full summary is found to be unfaithful, it is not known whether the factual inconsistency exists already in the examined prefix or only in its speculated continuation. In the latter cases, prefixes that are still faithful, and might eventually lead to faithful summaries, get penalized unjustifiably, as illustrated in Figure 2. In our work, we aim to circumvent these two deficiencies of the prior lookahead approach by applying a suitable NLI model directly over the beam prefixes.

3 PrefixNLI: Task and Datasets

NLI models are widely used to assess factual consistency in text generation, but their utility in providing feedback *during* generation remains underexplored. Motivated by the need to detect factual errors as soon as they emerge, we introduce the **PrefixNLI** task, which targets entailment detection over incomplete hypothesis texts. This can be used as a reward signal during training or to steer decoding toward faithful generations at inference.

3.1 Task definition

Given a premise text x and an arbitrary text prefix, $y_{1:t}$, considered as the hypothesis, we would like to predict whether the hypothesis is textually entailed by the premise, or not (a binary entailment setting). Since a text prefix might be ungrammatical or even nonsensical, we refine the definition of entailment to fit this setting: a text prefix is entailed by a premise if there may be a sensible completion of the prefix to a complete text that would be entailed by the premise (see Figure 2 for an illustration). Accordingly, if the prefix already includes some details that are not entailed by the premise then the prefix as a whole is considered not entailed.

3.2 Constructing evaluation and training datasets

To facilitate research on our novel prefix entailment task, we derive evaluation and training datasets from existing textual entailment and factual consistency data. We first describe the general methodology used to generate our datasets, with specific

details for each dataset and an illustrating example provided in the subsections below.

For each dataset instance, we first obtained the source document x, regarded as the premise, and a hypothesis text y (typically a summary), which may or may not be faithful to the source. In addition, as detailed below, for non-entailed hypotheses we detected the *first* span, denoted by s, that expresses information that is factually inconsistent with the source. Given this span s, we can now deduce that all prefixes ending before the starting position of s are entailed by the premise, thus yielding entailed prefix examples. Conversely, all prefixes ending at the ending position of s or later yield non-entailing prefix instances.[2] Naturally, faithful original hypotheses yield only entailed prefix examples.

3.2.1 Evaluation benchmark dataset

Our evaluation benchmark is derived from two diverse factual consistency datasets. Both datasets contain high-quality *human annotations* which either explicitly mark erroneous spans in non-entailed examples or allow such spans to be inferred with high confidence. Specifically, RAGTruth (Niu et al., 2024) contains machine-generated summaries with human marked hallucination spans, while SummEdits (Laban et al., 2023) contains human factual consistency verdicts for summaries that were modified by LLMs. [3]

RAGTruthPrefixes RAGTruth (Niu et al., 2024) is a fine-grained hallucination corpus for assessing faithfulness in retrieval-augmented generations. RAGTruth contains LLM-generated responses with word-level manual annotations marking factually inconsistent spans, which we can directly consider as the unsupported utterance s in our prefix generation methodology (as described above). We then consider each sentence in the summary separately and generate from it the prefix instances for our dataset. Overall, we extract 213K prefix instances and set aside 2k instances for the development set.

SummEditsPrefixes Each SummEdits instance (Laban et al., 2023) includes a source document and two summaries: a faithful seed (original) and a modified one generated by an LLM. The modifications are usually limited to local changes, for example an entity swap, a lexical substitute, negation, etc. Each modified summary is evaluated by a human annotator for factual consistency.

To determine the unsupported utterance s in unfaithful summaries, we find the longest common prefix and suffix between the modified and seed summaries and mark the span in-between as s. As demonstrated in Figure 3, the unsupported utterance s would be the entire span between *"the"* and *"Argentina"*. According to our instance generation methodology, all prefixes preceding s are regarded as entailed, while prefixes starting at the last position of s or later are regarded as not entailed.

We construct our dataset by selecting instances from the *News* domain with up to a single LLM modification. To avoid length bias, we stratify over the prefix position t such that each prefix length has an equal number of factually consistent and inconsistent examples.

3.2.2 Training dataset

To train our PrefixNLI model, we construct two complementary prefix-level NLI datasets, each motivated by a different goal. The first is derived from the TrueTeacher dataset (Gekhman et al., 2023), which contains LLM-generated summaries labeled for factual consistency. While TrueTeacher provides only summary-level labels, we adapt it to the prefix setting by identifying hallucinated spans within unfaithful summaries using GPT-4 (OpenAI et al., 2024) prompting (see Appendix A). This dataset captures naturally occurring LLM hallucinations, offering a realistic distribution of factual errors encountered in practice.

Our second training dataset consists of synthetically generated summaries specifically crafted to contain fine-grained and subtle hallucinations. Generating synthetic summaries allows us to control for the types of hallucination in the data and provide a broad coverage of nuanced errors. These summaries were generated using GPT-4 and the complete prompt is provided in Appendix A.

4 PrefixNLI Model

In this section we introduce *MiniTruePrefixes*, trained for the PrefixNLI task, first describing its architecture (§4.1) and then its training regime (§4.2).

4.1 Model architecture and inference

The primary intended usage of PrefixNLI is to evaluate the consistency of a prefix with the source as it evolves by appending one token at a time, as illustrated in Figure 1. Such an approach can be

[2] We do not include prefixes that partially contain s, since it is impossible to deduce their entailment label from the available data that we leverage.

[3] Dataset statistics are provided in Appendix B.

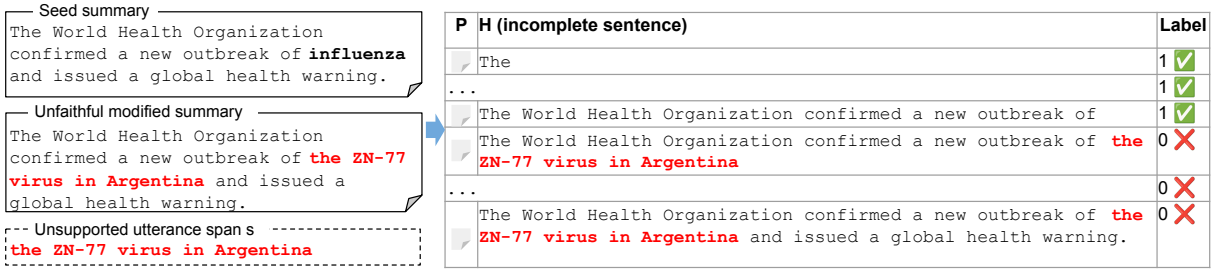

Figure 3: Our SummEditsPrefixes evaluation dataset creation. Given a seed and a modified summary, we identify the hallucinated span s in an unfaithful summary by identifying where it differs from the seed, highlighted in red. Entailed prefixes are then derived from all positions up to s, while non-entailed prefixes are derived starting from the ending position of s.

expensive as each decoding step requires another entailment inference call per candidate token, e.g. "Jeremy", "Roy", and "Nicky" in the figure. However, since the model is only generating the last token, the same prefix will be used in subsequent decoding steps, e.g. "Former goalkeeper". This step-by-step formulation naturally aligns with decoder-only architectures (Llama Team et al., 2024), as used by the entailment model, which are well-suited to this task due to their efficient support for prefix caching. That is, they store and reuse key-value (KV) pairs, which is critical for reducing computational overhead (Kwon et al., 2023).

Accordingly, we use the LLaMA-3.2-Instruct model (Llama Team et al., 2024) as our base model. We selected a small and relatively nimble model with 1B parameters in order to further minimize computational costs. We then follow the entailment classification architecture of the TrueTeacher NLI system (Gekhman et al., 2023). Given a source document x and a hypothesis prefix $y_{1:t}$, the model predicts the entailment decision. To make a prediction, we construct a prompt from x and $y_{1:t}$ (see Appendix C for technical details), and infer entailment if the probability for the entailment class (technically for the class label token "1"), denoted $P_{\text{entail}}(y_{1:t} \mid x)$, is higher than 0.5.

4.2 Training regime

We train our model in two stages: we train a summary-level NLI model, *MiniTrue*, which we then adapt for prefix-level supervision, *MiniTruePrefixes*. In both stages we use cross-entropy loss between the model's predicted and target labels. Akin to the training regime of TrueTeacher (Gekhman et al., 2023), we provide as input a pair of a premise text and a (truncated) hypothesis text, and train the model to generate the token "1" if it is entailed or "0" for non-entailed verdicts. This formulation enables scoring of both truncated and complete hypotheses with the same setup.

Fine-tuning a lightweight entailment model. We fine-tune the base model on the TrueTeacher dataset (Gekhman et al., 2023) and the ANLI dataset (Nie et al., 2020), mirroring the training setup of the original TrueTeacher model with a more parameter efficient variant. The resulting model, which we refer to as MiniTrue, achieves strong entailment performance while remaining lightweight and computationally efficient. See Appendix D.2 for the evaluation of MiniTrue on the TRUE benchmark.

Adapting for prefix entailment. We introduce *MiniTruePrefixes*, obtained via an additional fine-tuning step on our curated prefix-level entailment dataset. This additional training step enables the model to better handle the linguistic incompleteness and increased ambiguity present in truncated hypotheses, resulting in more reliable factual inconsistencies detection on truncated hypotheses. See Appendix C for more details.

5 PrefixNLI Intrinsic Evaluation

5.1 Experimental setup

Baselines We evaluate two baseline models trained for the NLI task. The first is MiniTrue, our lightweight entailment model described in §4.2, which was trained on standard NLI data and served as the base model for further prefix-level finetuning of the MiniTruePrefixes model. The second is MiniCheck (Tang et al., 2024), the current state-of-the-art factual consistency model of similar size (770M parameters).

Evaluation metrics Given an evaluation benchmark consisting of premise x, prefix hypothesis $y_{1:t}$, and a gold entailment decision, we compute for each system the micro-averaged F1 score for the *unfaithful class* across all instances.[4] This choice aligns with our objective of detecting factual *inconsistencies* as soon as they arise during decoding, hence adopting a metric that directly reflects the model's ability to identify unfaithfulness. These F1 scores are reported with their 95% confidence interval.

5.2 Results

Model	SummEditsPrefixes	RAGTruthPrefixes
MiniCheck (Flan-T5)	72.9 ± 1.0	33.3 ± 0.5
MiniTrue	69.8 ± 1.2	27.5 ± 0.5
MiniTruePrefixes	**78.1** ± 1.0	**47.6** ± 0.6

Table 1: F1 scores for the unfaithful class on SummEditsPrefixes and RAGTruthPrefixes.

As shown in Table 1, MiniTruePrefixes outperforms the strongest baseline of similar size, MiniCheck (Flan-T5), by 5.2 and 14.3 points on SummEditsPrefixes and RAGTruthPrefixes, respectively. It also surpasses MiniTrue, demonstrating the benefit of prefix-level fine-tuning. These results highlight the benefit of training a targeted prefix-level NLI model, over prefix-level training data, when aiming to detect unfaithful text prefixes. Appendix D.3 presents a qualitative example where MiniTruePrefixes succeeds while the sentence-level models fail. Additionally, we performed a manual error analysis described in Appendix D.4.

6 Controlled Decoding with PrefixNLI

In this section, we show how PrefixNLI models can be used to generate more faithful summaries, with a reasonable computational cost. As motivated in §1 and §2, our approach avoids altogether using the inefficient and potentially noisy lookahead mechanism employed in prior work (Wan et al., 2023; Sridhar and Visser, 2023).

6.1 Method

Standard autoregressive decoding methods, including beam search, prioritize **likelihood**, which often indirectly reflects faithfulness but does not explicitly enforce it. To improve faithfulness, our method integrates entailment probability into next-token decoding decisions, following the above-mentioned line of recent work. Specifically, we use our MiniTruePrefixes model to calculate the entailment score between the source and the current prefix extended with each candidate next token (as illustrated in Figure 1). To steer generation away from hallucinations, we penalize tokens with low entailment scores, discouraging unfaithful continuations. This biases the model toward more faithful outputs without fully overriding its original preferences, allowing us to reduce factual inconsistencies as soon as they arise.

Formally, let $\ell_i = \ell_{\text{gen}}(y_t^i \mid x, y_{1:t-1})$ denote the logit assigned by the language model to the i-th token in the vocabulary at time step t, given the source and current prefix, and let $p_i = p_{\text{entail}}(y_{1:t}^i \mid x)$ denote the entailment score of the resulting extended prefix. For each top-p candidate token with an entailment score below 0.5, indicating that non-entailment is more likely than entailment, we update its logit as follows:

$$\ell_i \leftarrow \ell_i + \lambda \cdot \log\left(\frac{p_i}{1-p_i}\right)$$

The adjustment term is defined as the scaled log-odds of the entailment probability. It serves as a penalty, taking negative values when $p_i < 0.5$, with increasing magnitude as p_i decreases. λ is a scaling factor that controls the strength of the penalty. Tokens with $p_i > 0.5$ remain unchanged, thereby preserving the original distribution while softly encouraging more faithful continuations. All tokens not included in the top-p set are assigned a logit of $-\infty$ and thus excluded from consideration. For beam search with beam size K, we maintain the top K candidate sequences based on their cumulative adjusted log-probabilities. Hyper-parameters were tuned on the development set of the CNN/DM, see Appendix E.1 for more details.

6.2 Experimental setup

Datasets We follow prior work in this area (Wan et al., 2023; Roit et al., 2023; Sridhar and Visser, 2023) and focus our evaluation on abstractive summarization, a typical representative of source-grounded generation settings. We conduct experiments on the XSum (Narayan et al., 2018) and CNN/DM (Hermann et al., 2015) datasets, using 2,500 test set documents from each.

[4] We also report the micro-averaged F1 score for the faithful class in Appendix D.1.

Baselines Our experimental setup includes two components: the generator LLM, used to generate the outputs, and the NLI model which is employed in the decoding process to estimate the faithfulness of candidate tokens. For the generator LLM, we conduct experiments with several models from the LLaMA-3 series: LLaMA-3.2-1B-Instruct, LLaMA-3.2-3B-Instruct, and LLaMA-3.2-8B-Instruct (Llama Team et al., 2024), in a zero-shot setting (see Appendix E.2). For the decoding process intervention, we compare the following approaches: **Vanilla** denotes standard generation without intervention in the decoding process. **Lookahead** implements the lookahead algorithm of Wan et al. (2023). We greedily extend the current beam prefix to a complete (temporary) summary (as described in §2), using the same model as the LLM generator model, and then evaluate the faithfulness of this full summary using our MiniTrue entailment model. **CAD** compares our method to the Context-Aware Decoding (CAD) algorithm proposed by Shi et al. (2024), which does not rely on an NLI metric (additional setup details are provided in Appendix E.3). **Prefix** applies our proposed controlled decoding method (§6.1), while utilizing our MiniTruePrefixes as the prefix entailment model.

6.2.1 Metrics

We evaluate the complete generated summaries using the following automatic metrics:

Faithfulness Following recent works (Wan et al., 2025; Lee et al., 2024), we evaluate faithfulness using the MiniCheck entailment model (Tang et al., 2024). We use Bespoke-MiniCheck-7B, the current state-of-the-art for this task. Each summary sentence is individually compared against the entire source document and assigned a binary entailment label. The proportion of entailed sentences in each summary is regarded as its faithfulness score, then averaged over all evaluated summaries. We also evaluate faithfulness using GPT-4.1[5] (OpenAI et al., 2024) with a prompting setup adapted from Wadhwa et al. (2024) (described in Appendix E.4). Both this method and MiniCheck have been shown to correlate strongly with human judgments of faithfulness (Tang et al., 2024; Liu et al., 2023; Chiang and Lee, 2023; Gao et al., 2023b).

Content Quality We assess output quality using standard summarization metrics. Specifically, we report ROUGE-L F1 (Lin, 2004) against reference summaries and evaluate fluency with MAUVE (Pillutla et al., 2021), using a scaling factor of $c = 0.5$, to ensure our decoding does not harm fluency.

Latency Average generation speed (in seconds) per summary, measuring the cost of applying the NLI model at each step. Evaluated on a single NVIDIA A100 80GB GPU.[6]

We report the standard error of the mean when applicable to assess statistical significance.

6.3 Results

The results are shown in Table 2 and our main takeaways are discussed below.

MiniTruePrefixes improves faithfulness across different-sized generators. Using our MiniTruePrefixes with the 1B generator yields 7.5-point and 8-point faithfulness improvements (MiniCheck column) in the CNN/DM and XSum datasets, respectively. Scaling the generator consistently improves faithfulness: combining our 1B MiniTruePrefixes with a 3B generator improves faithfulness by 5.7 points on CNN/DM and 6.2 points on XSum over the vanilla 3B model. Remarkably, on XSum, our 3B+1B configuration outperforms the vanilla 8B model by 0.9 MiniCheck points and offers a substantially more compute-efficient alternative, particularly in resource-constrained settings such as on-device deployment. Similarly, using our method with an 8B generator further improves faithfulness by 2.9 points on CNN/DM and 5.5 points on XSum compared to vanilla 8B. These results indicate that prefix-level entailment reranking provides consistent and complementary gains even when applied to strong generator models, and can effectively close the faithfulness gap between model scales. We observe consistent gains in GPT-4 assessed faithfulness across all model sizes using our method. Examples illustrating how our method fixes or avoids hallucinations in the vanilla summary are presented in Table 16 in Appendix G.1.

Using MiniTruePrefixes incurs a reasonable computational cost. The generation time increases by up to 2.9× for the 1B generator. However, the relative overhead decreases with model size, dropping to 1.4× for the 8B model, as the fixed-cost entailment model accounts for a smaller proportion of total compute.

[5]Evaluation based on 1,000 documents per dataset.

[6]For more details, see Appendix E.5.

Model	XSum					CNN/DM				
	Faithfulness		Auxiliary Metrics			Faithfulness		Auxiliary Metrics		
	MiniCheck↑	GPT-4↑	R-L↑	MAUVE↑	Speed↓	MiniCheck↑	GPT-4↑	R-L↑	MAUVE↑	Speed↓
Vanilla (1B)	66.7 ± 0.6	2.53 ± 0.02	14.4 ± 0.1	91.4	**1.74**	71.7 ± 0.6	3.23 ± 0.03	18.9 ± 0.1	**79.6**	**1.93**
Vanilla (3B)	76.2 ± 0.6	3.34 ± 0.03	14.3 ± 0.1	**91.6**	3.10	80.1 ± 0.6	3.68 ± 0.03	18.9 ± 0.1	79.3	4.46
Vanilla (8B)	81.5 ± 0.5	3.63 ± 0.03	**14.9** ± 0.1	91.3	5.30	87.9 ± 0.6	3.87 ± 0.02	19.0 ± 0.1	79.4	6.76
Lookahead (1B)	70.9 ± 0.6	3.03 ± 0.03	12.8 ± 0.1	89.4	114.1	75.9 ± 0.6	3.28 ± 0.03	17.9 ± 0.1	79.1	158.9
CAD (1B)	70.1 ± 0.6	3.05 ± 0.03	12.4 ± 0.1	90.8	13.3	75.5 ± 0.5	3.34 ± 0.02	**20.4** ± 0.1	79.4	20.0
Prefix (1B, MTP)	74.8 ± 0.6	3.13 ± 0.03	13.6 ± 0.1	89.6	4.84	79.2 ± 0.6	3.41 ± 0.02	19.8 ± 0.1	79.1	5.67
Prefix (3B, MTP)	82.4 ± 0.5	3.59 ± 0.03	14.1 ± 0.1	91.2	5.43	85.8 ± 0.4	3.73 ± 0.02	19.0 ± 0.1	79.1	7.18
Prefix (8B, MTP)	**87.0** ± 0.4	**3.84** ± 0.03	14.5 ± 0.1	91.3	7.23	**90.8** ± 0.3	**3.90** ± 0.02	18.9 ± 0.1	79.1	9.20

Table 2: Performance across the XSum and CNN/DM datasets. **MiniTruePrefixes** is abbreviated as **MTP**.

MiniTruePrefixes outperforms baseline approaches. Both the prior lookahead method and CAD improve faithfulness over vanilla decoding but underperform noticeably compared to our approach. For lookahead, we conjecture that scoring entire speculative continuations, rather than the prefix generated up to the current step, introduces noise into prefix-level entailment judgments and limits effectiveness (see Figure 2). As expected, the lookahead approach is substantially more computationally expensive, running on average 25.8× slower than our method. The CAD baseline, which does not rely on an NLI signal, offers meaningful improvements but remains less effective than our entailment-guided approach. It also incurs higher computational costs than ours, with 2.7× slower runtime on XSum and 3.5× on CNN/DM. As an ablation, we also evaluated the use of MiniTrue as the prefix entailment model and found that it consistently underperformed relative to MiniTruePrefixes across all evaluation metrics and model sizes. Complete ablation results are reported in Appendix F.

Summary quality is not compromised. With respect to summary quality, on the CNN/DM dataset our more faithful generation method for the 1B generator yields more relevant summaries with respect to the ROUGE score (+0.9 points). This makes sense since reference summaries in CNN/DM are generally faithful to the source document, hence hallucinations in the predicted summary also reduce similarity to the reference. ROUGE scores for 3B and 8B models show only minor nonsignificant changes when applying our method. On the XSum dataset, the ROUGE relevance score slightly decreases, which is sensible, since XSum references are known to include many factual inconsistencies relative to the source document (by construction, XSum references are not actual summaries) (Maynez et al., 2020). Fluency (MAUVE) is maintained across models, with negligible differences relative to the vanilla baseline.

Overall, these results indicate the potential of our prefix-based entailment approach for reducing hallucinations, while suggesting potential for future research to improve prefix-entailment modeling and its incorporation in autoregressive generation.

7 Conclusions and Future Work

We introduced PrefixNLI, the task of assessing factual consistency at the text prefix level, with the primary motivation of detecting factual inconsistencies as soon as they arise during autoregressive text generation. Promoting research on this task, we introduced suitable datasets and a targeted model, which showed positive results in both an intrinsic evaluation and in improving generation faithfulness in a controlled decoding framework. Our model also yields dramatic efficiency gains compared to prior lookahead-based approaches, facilitating inference-time factuality control.

Our work opens up several directions for future research on improving and leveraging prefix-level entailment models. First, there is room for improving the core PrefixNLI model, through the creation of richer training data, manually or automatically, as well as enhanced modeling, e.g. by identifying semantic unit boundaries as more reliable "breakpoints" for assessing prefix entailment. Second, there is potential for improving the faithfulness and efficiency of the controlled decoding method, e.g. by more selective application of PrefixNLI models at reliable time steps, or smarter beam management. Third, it is very appealing to incorporate PrefixNLI also into the training regime, possibly extending it from using only sentence-level rewards (Roit et al., 2023) to more informative and precise token-level rewards. Finally, prefix-level NLI can benefit additional generation tasks, beyond summarization.

Ethical considerations

This work aims to improve the faithfulness of language model outputs, which we view as a desirable objective with positive downstream impact. However, our method depends on predictions from MiniTruePrefixes, an entailment model that can occasionally misclassify unfaithful content as faithful. As a result, models using our approach might produce outputs that appear trustworthy despite containing factual errors, which can mislead users to assume a stronger connection between the source and generated text than what actually exists.

We used AI tools to assist with language refinement. All content was subsequently reviewed and validated by the authors for accuracy and correctness.

Limitations

Access to logits Our controlled decoding method relies on access to the output logits of the language model in order to modify the token distribution during inference. However, this requirement poses a constraint on the applicability of our method to closed-source, API-based models that do not expose internal logits. As a result, our approach is currently limited to open or locally accessible models.

Inference overhead The method introduces a moderate increase in inference time, stemming from the entailment computations performed at each generation step. While our experiments demonstrate that this additional cost is justified by the resulting gains in factual consistency, future work may explore strategies to reduce computational overhead. For instance, the entailment model could be applied more selectively, only at key decision points such as punctuation marks or in cases of high model uncertainty.

Language coverage Although the LLaMA-3.2-Instruct 1B model used as the backbone of the MiniTruePrefixes model supports multilingual generation, our model was trained exclusively on English data. This choice arises from the availability of high-quality datasets with fine-grained hallucination localization, which are largely limited to English and critical for the development and assessment of our method. Extending our approach to multilingual settings where such annotated data is currently scarce is an important avenue for broadening its applicability.

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Cheng-Han Chiang and Hung-yi Lee. 2023. Can large language models be an alternative to human evaluations? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15607–15631, Toronto, Canada. Association for Computational Linguistics.

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The pascal recognising textual entailment challenge. In *Machine Learning Challenges Workshop*. https://link.springer.com/chapter/10.1007/11736790_9.

Luyu Gao, Zhuyun Dai, Panupong Pasupat, Anthony Chen, Arun Tejasvi Chaganty, Yicheng Fan, Vincent Zhao, Ni Lao, Hongrae Lee, Da-Cheng Juan, and Kelvin Guu. 2023a. RARR: Researching and revising what language models say, using language models. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 16477–16508, Toronto, Canada. Association for Computational Linguistics.

Mingqi Gao, Jie Ruan, Renliang Sun, Xunjian Yin, Shiping Yang, and Xiaojun Wan. 2023b. Human-like summarization evaluation with chatgpt. *Preprint*, arXiv:2304.02554.

Zorik Gekhman, Jonathan Herzig, Roee Aharoni, Chen Elkind, and Idan Szpektor. 2023. Trueteacher: Learning factual consistency evaluation with large language models. *Preprint*, arXiv:2305.11171.

Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. *Preprint*, arXiv:1506.03340.

Or Honovich, Roee Aharoni, Jonathan Herzig, Hagai Taitelbaum, Doron Kukliansy, Vered Cohen, Thomas Scialom, Idan Szpektor, Avinatan Hassidim, and Yossi Matias. 2022. True: Re-evaluating factual consistency evaluation. *Preprint*, arXiv:2204.04991.

Gautier Izacard, Patrick Lewis, Maria Lomeli, Lucas Hosseini, Fabio Petroni, Timo Schick, Jane Dwivedi-Yu, Armand Joulin, Sebastian Riedel, and Edouard Grave. 2023. Atlas: Few-shot learning with retrieval augmented language models. *Journal of Machine Learning Research*, 24(251):1–43.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the*

ACM SIGOPS 29th Symposium on Operating Systems Principles.

Philippe Laban, Wojciech Kryściński, Divyansh Agarwal, Alexander R. Fabbri, Caiming Xiong, Shafiq Joty, and Chien-Sheng Wu. 2023. Llms as factual reasoners: Insights from existing benchmarks and beyond. *Preprint*, arXiv:2305.14540.

Yuho Lee, Taewon Yun, Jason Cai, Hang Su, and Hwanjun Song. 2024. Unisumeval: Towards unified, fine-grained, multi-dimensional summarization evaluation for llms. *Preprint*, arXiv:2409.19898.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Chin-Yew Lin. 2004. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain. Association for Computational Linguistics.

Yang Liu, Dan Iter, Yichong Xu, Shuohang Wang, Ruochen Xu, and Chenguang Zhu. 2023. G-eval: NLG evaluation using gpt-4 with better human alignment. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 2511–2522, Singapore. Association for Computational Linguistics.

Llama Team and 1 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Joshua Maynez, Shashi Narayan, Bernd Bohnet, and Ryan McDonald. 2020. On faithfulness and factuality in abstractive summarization. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1906–1919, Online. Association for Computational Linguistics.

Nick McKenna, Tianyi Li, Liang Cheng, Mohammad Hosseini, Mark Johnson, and Mark Steedman. 2023. Sources of hallucination by large language models on inference tasks. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 2758–2774, Singapore. Association for Computational Linguistics.

Abhika Mishra, Akari Asai, Vidhisha Balachandran, Yizhong Wang, Graham Neubig, Yulia Tsvetkov, and Hannaneh Hajishirzi. 2024. Fine-grained hallucinations detections. *COLM*.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. 2018. Don't give me the details, just the summary! topic-aware convolutional neural networks for extreme summarization. *Preprint*, arXiv:1808.08745.

Yixin Nie, Adina Williams, Emily Dinan, Mohit Bansal, Jason Weston, and Douwe Kiela. 2020. Adversarial nli: A new benchmark for natural language understanding. *Preprint*, arXiv:1910.14599.

Cheng Niu, Yuanhao Wu, Juno Zhu, Siliang Xu, Kashun Shum, Randy Zhong, Juntong Song, and Tong Zhang. 2024. Ragtruth: A hallucination corpus for developing trustworthy retrieval-augmented language models. *Preprint*, arXiv:2401.00396.

OpenAI and 1 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Krishna Pillutla, Swabha Swayamdipta, Rowan Zellers, John Thickstun, Sean Welleck, Yejin Choi, and Zaid Harchaoui. 2021. Mauve: Measuring the gap between neural text and human text using divergence frontiers. *Preprint*, arXiv:2102.01454.

Paul Roit, Johan Ferret, Lior Shani, Roee Aharoni, Geoffrey Cideron, Robert Dadashi, Matthieu Geist, Sertan Girgin, Leonard Hussenot, Orgad Keller, Nikola Momchev, Sabela Ramos Garea, Piotr Stanczyk, Nino Vieillard, Olivier Bachem, Gal Elidan, Avinatan Hassidim, Olivier Pietquin, and Idan Szpektor. 2023. Factually consistent summarization via reinforcement learning with textual entailment feedback. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6252–6272, Toronto, Canada. Association for Computational Linguistics.

Weijia Shi, Xiaochuang Han, Mike Lewis, Yulia Tsvetkov, Luke Zettlemoyer, and Wen-tau Yih. 2024. Trusting your evidence: Hallucinate less with context-aware decoding. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 2: Short Papers)*, pages 783–791, Mexico City, Mexico. Association for Computational Linguistics.

Arvind Krishna Sridhar and Erik Visser. 2023. Improved beam search for hallucination mitigation in abstractive summarization. *Preprint*, arXiv:2212.02712.

Liyan Tang, Philippe Laban, and Greg Durrett. 2024. Minicheck: Efficient fact-checking of llms on grounding documents. *Preprint*, arXiv:2404.10774.

Katherine Tian, Eric Mitchell, Huaxiu Yao, Christopher D Manning, and Chelsea Finn. 2023. Fine-tuning language models for factuality. In *The Twelfth International Conference on Learning Representations*.

Manya Wadhwa, Xinyu Zhao, Junyi Jessy Li, and Greg Durrett. 2024. Learning to refine with fine-grained natural language feedback. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 12281–12308, Miami, Florida, USA. Association for Computational Linguistics.

David Wan and Mohit Bansal. 2022. FactPEGASUS: Factuality-aware pre-training and fine-tuning for abstractive summarization. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1010–1028, Seattle, United States. Association for Computational Linguistics.

David Wan, Justin Chih-Yao Chen, Elias Stengel-Eskin, and Mohit Bansal. 2025. Mamm-refine: A recipe for improving faithfulness in generation with multi-agent collaboration. *Preprint*, arXiv:2503.15272.

David Wan, Mengwen Liu, Kathleen Mckeown, Markus Dreyer, and Mohit Bansal. 2023. Faithfulness-aware decoding strategies for abstractive summarization. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2864–2880.

Muru Zhang, Ofir Press, William Merrill, Alisa Liu, and Noah A. Smith. 2023. How language model hallucinations can snowball. *Preprint*, arXiv:2305.13534.

A Training Data Construction Details

This section provides additional details about the construction of our training dataset, expanding upon §3.2.2.

Hallucination detection using GPT-4 We begin by extracting summaries labeled as unfaithful from the TrueTeacher dataset (Gekhman et al., 2023). For each summary, we prompt GPT-4 to identify and highlight the specific span corresponding to the hallucinated content. Table 4 presents the prompt used along with an illustrative example.

Error analysis of subtle hallucinations We fine-tuned a version of MiniTruePrefixes on the hallucination identification data described above. To gain insight into its limitations, we manually analyzed a subset of its predictions on the RAGTruthPrefixes development set. This qualitative analysis revealed recurring failure cases where the model struggled to detect subtle hallucinations. These included claims that were inferred but not explicitly supported, slight modifications of factual details, and general statements that lacked sufficient grounding in the source. Table 7 presents a representative example of such a challenging case.

Motivated by the findings from our error analysis, we constructed a synthetic dataset containing subtle hallucinations that the model previously failed to detect. These examples were generated through targeted prompting of OpenAI's GPT-4 (O3) model (OpenAI et al., 2024); see Table 5 for the prompts and Table 8 for an illustrative example. This dataset introduced a wider variety of fine-grained hallucination types into training. As shown in Table 6, incorporating this data led to improved model performance.

The TrueTeacher dataset (Gekhman et al., 2023) is released under the CC-BY-NC 4.0 license. As our prefix-level entailment training dataset is derived from it, we will also release it under the same CC-BY-NC 4.0 license.

B Evaluation Benchmark Datasets

Table 3 provides statistics for the evaluation benchmark datasets. The RAGTruthPrefixes dataset is released under the same MIT license as the original RAGTruth dataset (Niu et al., 2024). Similarly, the SummEditsPrefixes dataset follows the CC-BY 4.0 license of the original SummEdits dataset (Laban et al., 2023).

C Model Training Details

We provide additional training details for our MiniTruePrefixes model, supplementing the information in §4. We used the following prompt, following TrueTeacher (Gekhman et al., 2023) for the entailment prediction: ***Premise**: {document} **Hypothesis**: {summary prefix}*. The model is trained to predict "1" if the hypothesis is factually consistent and "0" otherwise. We infer that the hypothesis is entailed if the probability for the token "1" is higher than 0.5, otherwise we deduce that it is not entailed.

For fine-tuning MiniTrue, we used a learning rate of 2×10^{-4} and a batch size of 32, training for 3 epochs with LoRA. For MiniTruePrefixes, we used a lower learning rate of 5×10^{-6}, the same batch size of 32, and also trained for 3 epochs using LoRA. During training, we set a maximum input length of 2048 tokens. In line with our goal of detecting inconsistencies, we selected the checkpoint with the highest F1 score on the unfaithful class over our development set.

To preserve MiniTruePrefixes pre-trained entailment knowledge while adapting it to the new setting of truncated hypotheses, we froze all layers except the final one. We also experimented with full fine-tuning but found that freezing all but the last layer yielded better results. This approach allowed the model to retain its foundational understanding of entailment while improving its capacity to assess incomplete text.

	# Prefixes		# Unique documents	Avg. hallucinatory span length	Avg. Prefix length
	Faithful	Unfaithful			
RAGTruthPrefixes	194,283	16,395	880	11.8	14.7
SummEditsPrefixes	4,455	4,455	25	4.2	27.6

Table 3: We report the number of prefixes in each dataset (faithful vs. unfaithful), along with the number of unique source documents, the average hallucinatory span length and the average prefix length (in tokens).

	Prompt used for hallucination identification
1	**System prompt:** You are given a summary and its corresponding source document. Your task is to identify the first hallucinated unit – a word, phrase, or clause that is not clearly supported or is contradicted by the document. Mark the hallucinated unit in the summary using the following two tags: - Insert `[HALLUCINATION_STARTING_TAG]` immediately before the first character of the hallucinated unit. - Insert `[HALLUCINATION_TURNOUT_TAG]` immediately after the last character of the hallucinated unit. Return the modified summary only, with the tags inserted in the correct positions. Do not return anything else — no explanations. If the summary is entirely faithful, return it unchanged, with no tags. **Document:** ... Cardinal Jorge Mario Bergoglio was elected the 256th Pope, President Cristina Kirchner appeared to be gearing up to use Francis I's powerful new status to Argentina's advantage ... **Summary:** Pope Francis has been sworn in as the new Pope of Argentina, a move feared to be a catalyst for nationalism. **GPT-4 output:** Pope Francis has been sworn in as the new `[HALLUCINATION_STARTING_TAG]`Pope of Argentina`[HALLUCINATION_TURNOUT_TAG]`, a move feared to be a catalyst for nationalism.

Table 4: Prompt for hallucination identification with example.

	Prompts used for hallucination generation
1	**System prompt:** You are given a document containing factual information. Your task is to write a concise summary (2–4 sentences) that includes a subtle and believable hallucination. The hallucination should infer something that was not stated in the document but sounds plausible and is difficult to detect. Surround only the hallucinated **phrase** - not the entire sentence - with the following tags: `[HALLUCINATION_STARTING_TAG]` ... `[HALLUCINATION_TURNOUT_TAG]`. Return only the modified summary with the hallucination tags inserted. Do not include any additional output or explanation.
2	**System prompt:** You are given a document containing factual information. Your task is to write a concise summary (2–4 sentences) that includes a subtle and believable hallucination. The hallucination should misrepresent a minor detail (like date or count). Surround only the hallucinated **phrase** — not the entire sentence — with the following tags: `[HALLUCINATION_STARTING_TAG]` ... `[HALLUCINATION_TURNOUT_TAG]`. Return only the modified summary with the hallucination tags inserted. Do not include any additional output or explanation.
3	**System prompt:** You are given a document containing factual information. Your task is to write a concise summary (2–4 sentences) that includes a subtle and believable hallucination. This hallucination should introduce an emotional or symbolic interpretation that is not directly supported by the document, yet remains believable within its context. Surround only the hallucinated **phrase** — not the entire sentence — with the following tags: `[HALLUCINATION_STARTING_TAG]` ... `[HALLUCINATION_TURNOUT_TAG]`. Return only the modified summary with the hallucination tags inserted. Do not include any additional output or explanation.
4	**System prompt:** You are given a document containing factual information. Your task is to write a concise summary (2–4 sentences) that includes a subtle and believable hallucination. The hallucination should combine multiple facts into a single, inaccurate generalization. Surround only the hallucinated **phrase** — not the entire sentence — with the following tags: `[HALLUCINATION_STARTING_TAG]` ... `[HALLUCINATION_TURNOUT_TAG]`. Return only the modified summary with the hallucination tags inserted. Do not include any additional output or explanation.

Table 5: Prompts used for hallucination generation.

Method	SummEditsPrefixes	RAGTruthPrefixes
Identification Only	77.5 ± 0.8	44.7 ± 0.1
Identification + Training with Injected Hallucinations (MiniTruePrefixes)	**78.1** ± 0.8	**47.6** ± 0.1

Table 6: F1 scores for the **unfaithful** class on the SummEditsPrefixes and RAGTruthPrefixes benchmarks. The second model was trained with additional examples containing synthetically injected hallucinations.

	Examples from error analysis
1	**Document:** The FBI charged a Philadelphia woman on Thursday with trying to travel overseas to fight for ISIS. She's one of three women arrested this week on terror charges. Two New York women were also taken into custody. An FBI complaint cites numerous social media messages dating back to August 2013 that were sent by Keonna Thomas, 30, also known as "Young Lioness" and "Fatayat Al Khilafah." One Twitter message said, "If we truly knew the realities ... we all would be rushing to join our brothers in the front lines pray ALLAH accept us as shuhada [martyrs]." Another said, "When you're a mujahid [violent jihadi fighter] your death becomes a wedding." The FBI said Thomas purchased an electronic visa to Turkey on March 23. Turkey is known as the easiest place from which to enter Syria and join ISIS. An ISIS manual advises recruits to buy round-trip tickets to vacation spots such as Spain and then purchase tickets for their real destination once they arrive overseas, the FBI said. On March 26, Thomas purchased a ticket to Barcelona, with a March 29 departure and an April 15 return to the United States, the complaint said. It's not clear when or where she was arrested. She was charged with knowingly attempting to provide material support and resources to a designated foreign terrorist organization. She could be sentenced to 15 years in prison. On Thursday, Noelle Velentzas, 28, and her former roommate, Asia Siddiqui, 31, were arrested in New York and accused of planning to build an explosive device for attacks in the United States, federal prosecutors said. In the past 18 months, the Justice Department's National Security Division has prosecuted or is prosecuting more than 30 cases of people attempting to travel abroad to join or provide support to terrorist groups. Of those cases, 18 allegedly involve support to ISIS. "The terrorist threat is more decentralized, more diffuse, more complicated," Homeland Security Secretary Jeh Johnson told reporters Thursday. "It involves the potential lone wolf actor, it involves the effective use of social media, the Internet." **Hypothesis:** Three women, including Keonna Thomas of Philadelphia, were charged with attempting to join ISIS. **Error Analysis:** The hallucination is that three women were charged with attempting to join ISIS, but only Keonna Thomas was charged with that; the other two were arrested on separate terror-related charges.

Table 7: Examples of errors caused by the identification-only version of MiniTruePrefixes.

D More Results for MiniTruePrefixes

D.1 MiniTruePrefixes

We previously reported F1 for the unfaithful class (Section 5), reflecting our emphasis on detecting hallucinations. In this section, we provide complementary results for the faithful class, for control. Specifically, we report micro-averaged F1 scores in Table 9 as a control metric.

As shown in the table, MiniTruePrefixes consistently outperforms both baselines across the two benchmarks, with a 2.7-point gain over its base model MiniTrue on SummEditsPrefixes and a 4-point improvement on RAGTruthPrefixes.

D.2 MiniTrue performance over the TRUE benchmark

To develop a lightweight entailment model, we trained MiniTrue on the same dataset as TrueTeacher (Gekhman et al., 2023) (see Section 4.2 for more training details). We assessed its ability to detect factual inconsistencies by evaluating it on the summarization subset of the TRUE benchmark (Honovich et al., 2022).

As shown in Table 10, MiniTrue achieves performance that is comparable to TrueTeacher, despite being significantly smaller in size. These results support the suitability of MiniTrue as a lightweight alternative for entailment models.

D.3 Successful detection example

Table 11 presents a case where MiniTruePrefixes successfully detected a subtle hallucination, while both MiniTrue and MiniCheck failed. This may be because these models were trained exclusively on full sentences and are less effective when the input is not a complete sentence. In contrast, MiniTruePrefixes, trained on prefix-level entailment inputs, is better suited to handle incomplete or partial inputs, enabling it to detect hallucinations in prefixes more reliably.

D.4 Error analysis for MiniTruePrefixes

To better understand the failure modes of MiniTruePrefixes on RAGTruthPrefixes, we conducted a manual analysis of 60 misclassified examples: 30 false positives (FPs) and 30 false negatives (FNs).

Each instance was assigned to a single error category based on the primary cause of the model's mistake. Tables 12 and 13 report the distribution and explanations for each category.

	Hallucination generation example
1	**System prompt:** You are given a document containing factual information. Your task is to write a concise summary (2–4 sentences) that includes a subtle and believable hallucination. The hallucination should misrepresent a minor detail (like date or count). Surround only the hallucinated **phrase** — not the entire sentence — with the following tags: `[HALLUCINATION_STARTING_TAG]` ... `[HALLUCINATION_TURNOUT_TAG]`. Return only the modified summary with the hallucination tags inserted. Do not include any additional output or explanation. **Document:** ... both parties could expect their share of the national vote to count towards a representative share of seats in the 120-member parliament ... **GPT-4 output:** The unlikely alliance between Dotcom's Internet Party and Maori nationalist-led Mana Party seeks to challenge New Zealand's electoral system by combining contrasting approaches to internet freedom and indigenous representation. Political scientist Bryce Edwards noted that this partnership could secure success in the September elections, especially if their vote share translates into seats in a `[HALLUCINATION_STARTING_TAG]`170-member parliament`[HALLUCINATION_TURNOUT_TAG]`.

Table 8: Example of hallucination generation by misrepresenting a minor numerical detail.

Model	SummEditsPrefixes	RAGTruthPrefixes
MiniCheck (Flan-T5)	76.8 ± 0.9	89.7 ± 0.1
MiniTrue	78.7 ± 0.8	91.0 ± 0.1
MiniTruePrefixes	**81.4** ± 0.8	**95.0** ± 0.1

Table 9: F1 scores for the faithful class on the SummEdits and RAGTruth prefixes benchmarks.

Prefix incompleteness is not a dominant failure mode. Only 3.3% of FPs and 13.3% of FNs were caused by underdetermined prefixes, cases where the input lacked sufficient context to determine entailment. This suggests that prefix incompleteness is not the main cause of model errors, and that in most cases, the prefix alone provides enough information for a correct decision.

Generic misclassification is the most frequent source of error. The largest share of FPs (53.3%) and a substantial portion of FNs (20%) fall into the "Generic Prediction Error" category, referring to cases where the input was clear but the model's prediction was plainly incorrect. These types of errors are also common in standard NLI models, indicating that they reflect general prediction challenges rather than prefix-specific issues.

Shallow heuristics drive many false positives. A significant portion of false positives (33.3%) arise from surface-level heuristic biases. These errors may indicate limited semantic reasoning, where shallow lexical or structural patterns are favored over actual consistency. Mitigating this bias could lead to more reliable models.

False negatives reveal reasoning gaps. The most common false negative category (43.3%) involves missed inferences, where the entailment may be present but not explicitly stated, potentially requiring temporal, causal, or commonsense reasoning. This pattern suggests that the model may struggle to capture implicit meaning or integrate less directly stated information.

Sensitivity to paraphrase and implicit knowledge remains limited. 13.3% of false negatives involved surface-level mismatches, where paraphrastic entailment was not recognized, and an additional 3.3% involved entailments that may rely on world knowledge. These cases point to potential challenges in generalizing beyond literal overlap and handling implicit meaning.

These findings suggest that many of the errors made by MiniTruePrefixes reflect general challenges commonly seen in NLI models, including difficulties with semantic reasoning, inference, and robustness to paraphrasing or lexical variation, rather than issues specific to using prefix hypotheses as input.

E Controlled Decoding Experimental Setup

E.1 Hyper-parameters used

For generation, we use $p = 0.9$ for considering the top-p candidates in the beam (the most likely tokens whose accumulated probability reaches p). For computational efficiency, we limit this to a maximum of 20 tokens per beam. Results are reported for beam size $K = 3$.

We conducted hyperparameter tuning using 500 instances from the CNN/DM development set, evaluating values from 1 to 15 in increments of 2 as the scaling factor. Our experiments identified $\lambda = 5$ as optimal, producing the most faithful results.

	MNBM	QAGS-X	FRANK	SummEval	QAGS-C	Average
TrueTeacher (11B) (Gekhman et al., 2023)*	78.1	**89.4**	**93.6**	**88.5**	**89.4**	**87.8**
MiniTrue (1B)	**79.2**	85.1	90.3	83.2	88.6	85.3

Table 10: ROC-AUC results on the summarization subset of the TRUE benchmark (Honovich et al., 2022). Results for TrueTeacher were reported in Gekhman et al. (2023).

Premise (truncated)
... John T. Booker Jr. of Topeka, an American citizen also known as Mohammed Abdullah Hassan, was taken into custody near Manhattan, Kansas, in a van that contained what he thought was a bomb ... Booker enlisted in the Army last year and was due to ship out to basic training April 7, 2014 ... His enlistment was terminated March 24, 2014, at the request of Army Criminal Investigation Command ...
Hypothesis
A US Army veteran has been arrested
Gold Label
NOT ENTAILED
Rationale
The hypothesis refers to Booker as a "US Army veteran," which implies he completed service. However, the premise states his enlistment was terminated before training began. He never served and thus is not a veteran.

Table 11: An example where MiniTruePrefixes correctly detects a factual hallucination describing Booker as a "US Army veteran" that MiniTrue and MiniCheck, which were trained on full sentences, fail to catch.

E.2 Summaries generation prompt

For our controlled decoding experiment, described in Section 6.2, we used the following prompt for the generation of summaries: *Summarize the following text accurately and concisely. Output only the summary—do not include any introductory words like 'Summary:' or explanations.*

E.3 CAD experimental setup

In Section §6, we use Context-Aware Decoding (CAD) (Shi et al., 2024) as a baseline. We adopt the hyperparameters recommended by the authors, setting $\alpha = 0.5$ and using top-p sampling with $p = 0.9$ for summarization. The experiment is conducted with the `LLaMA-3.2-1B-Instruct` model, using the document as context, consistent with the original CAD setup. The input format was:

```
{"role": "system", "content": "Summarize the following text accurately and concisely. Output only the summary—do not include any introductory words like 'Summary:' or explanations."},
{"role": "user", "content": user_content}
```

For the no-context condition, we set `user_content` to `[Text omitted]`. For the context condition, we provided the full document.

E.4 GPT-4 faithfulness metric

The prompt used for evaluating the faithfulness of the generated summaries from Section 6 can be found in Table 14. We used gpt-4.1-2025-04-14.

E.5 Additional details on the experimental setup

Our controlled decoding method (Section 6.1) frequently invokes the entailment model to score summary prefixes. For efficient inference, we run the entailment model using vLLM (Kwon et al., 2023). Summary generation across all experiments required approximately 45 GPU hours on an NVIDIA A100-80GB.

F Ablation Study

We report results when applying the MiniTrue model to beam prefixes, allowing us to assess the relative benefits of MiniTruePrefixes over MiniTrue, which was trained only on complete-sentence hypotheses (§4). As shown in Table 15, MiniTruePrefixes consistently outperforms MiniTrue across all evaluation metrics and model sizes, with more than a 2-point improvement in the MiniCheck faithfulness score across both datasets and all model scales. Notably, applying MiniTrue with the 8B model on XSum degraded performance relative to the vanilla model, which achieved a MiniCheck score of 81.5 (see Table 2). These findings highlight the necessity of a dedicated prefix-level entailment model, rather than relying on NLI models trained exclusively on complete-sentence hypotheses.

#	Category	Explanation	FP (%)
1	Underdetermined Prefix	The prefix is too short, ambiguous, or lacks necessary context. False positives may result from the limited information available in the prefix.	3.33%
2	Surface-Level Heuristic Bias	The prefix shares significant surface-level lexical or structural features with the source, which may lead the model to predict entailment despite the absence of true semantic consistency.	33.33%
3	World Knowledge Confusion	Cases where entailment may depend on somewhat specific world knowledge, which was not assumed in the gold annotation and therefore labeled as not entailed.	3.33%
4	Generic Prediction Error	The prefix is understandable, the source is clear, but the model gets it wrong.	53.33%
5	Wrong Annotation	The model prediction is reasonable, but the gold annotation is incorrect.	6.67%

Table 12: Distribution of false positives (FP) by error category.

#	Category	Explanation	FN (%)
1	Underdetermined Prefix	The prefix is too short, ambiguous, or lacks necessary context. False negatives occur when the model hesitates due to missing context despite underlying entailment.	13.33%
2	Incorrect Reasoning / Inference	The hypothesis is clear and understandable, but the prefix contains information that may call for implicit reasoning (e.g., temporal, causal, or logical). The model does not predict entailment, possibly due to the complexity of the inference.	43.33%
3	Surface-Level Heuristic Bias	The hypothesis is entailed but paraphrased with limited surface similarity to the premise, which may challenge the model when predictions depend on more than lexical or structural overlap.	13.33%
4	World Knowledge Confusion	Cases where the gold annotator assumes access to somewhat specific world knowledge, but the model does not predict entailment. The missed entailment may reflect challenges in applying or accessing such knowledge.	3.33%
5	Generic Prediction Error	The prefix is understandable, the source is clear, but the model gets it wrong.	20.0%
6	Wrong Annotation	The model prediction is reasonable, but the gold annotation is incorrect.	6.67%

Table 13: Distribution of false negatives (FN) by error category.

G Example Outputs

G.1 Controlled decoding generation examples

Table 16 presents example summaries generated from XSum documents using our controlled decoding method (Section 6), alongside outputs from the vanilla model. As demonstrated here, our method effectively avoids hallucinations, thereby improving faithfulness.

G.2 XSum's reference summaries are not faithful

As discussed in Section 6.3, our method resulted in a lower ROUGE-L score on the XSum dataset. We attribute this decline to the fact that our method encourages faithfulness, while XSum's reference summaries frequently contain hallucinated content. Table 17 illustrates an example of such a hallucination.

This issue stems from the design of the XSum dataset: the reference summaries are not actual summaries, and were not required to be faithful. Prior analysis has shown that over 70% of XSum's reference summaries contain factual inconsistencies (Maynez et al., 2020).

System prompt: Determine whether the provided summary is consistent with the corresponding document. Consistency in this context implies that all information presented in the response is substantiated by the document. If not, it should be considered inconsistent.
The response can have one or more of the following errors: 1. Extrinsic Information: the response contains new information not grounded in the source material 2. Mis-Referencing: a property or an event in the response can be found in the source material, but are associated with the wrong entity 3. Stating Opinion As Fact: the response entails a proposition that's mentioned in the source material not as a fact, but as someone's opinion 4. Reasoning Error: the response makes one or more wrong inferences from the information in the source material 5. Tense/modality Error: the tense or modal (e.g., can, may, must) used in the response sentence does not match the tense/modality of the source material 6. Contradiction: the response contradicts the source material 7. Nuanced Meaning Shift: the response twists information from the source material in a subtle way
Given the error categories, rate the above response on a scale of 1 to 5 based on extent of factual consistency: 5. completely consistent: the response is completely factually consistent with the source material. 4. insignificant inconsistencies: the response is mostly factually consistent, with slight inconsistencies not affecting main points. 3. partially inconsistent: overall factually consistent, with a few inconsistencies with the source material. 2. severe inconsistencies: nearly half response is factually inconsistent, with severe deviation from main points. 1. completely inconsistent: the entire response is factually inconsistent with the source material.
First output a list of errors that the summary makes, then conclude the response with a score in the following format: "therefore, the score is:"

Table 14: Prompt (Wadhwa et al., 2024) used for evaluating faithfulness of the generated summaries from Section 6.

Model	XSum					CNN/DM				
	Faithfulness		Auxiliary Metrics			Faithfulness		Auxiliary Metrics		
	MiniCheck↑	GPT-4↑	R-L↑	MAUVE↑	Speed↓	MiniCheck↑	GPT-4↑	R-L↑	MAUVE↑	Speed↓
Prefix (1B, MT)	72.4 ± 0.6	3.09 ± 0.03	13.5 ± 0.1	89.6	4.89	77.2 ± 0.6	3.36 ± 0.02	19.4 ± 0.1	**79.2**	5.69
Prefix(3B, MT)	79.2 ± 0.6	3.54 ± 0.03	14.2 ± 0.1	91.1	5.27	82.3 ± 0.7	3.71 ± 0.02	**19.8** ± 0.1	79.1	6.62
Prefix(8B, MT)	79.7 ± 0.6	3.75 ± 0.03	**14.6** ± 0.1	91.2	7.18	88.7 ± 0.4	3.85 ± 0.02	18.9 ± 0.1	79.1	9.35
Prefix (1B, MTP)	74.8 ± 0.6	3.13 ± 0.03	13.6 ± 0.1	89.6	**4.84**	79.2 ± 0.6	3.41 ± 0.02	**19.8** ± 0.1	79.1	**5.67**
Prefix (3B, MTP)	82.4 ± 0.5	3.59 ± 0.03	14.1 ± 0.1	91.2	5.43	85.8 ± 0.4	3.73 ± 0.02	19.0 ± 0.1	79.1	7.18
Prefix (8B, MTP)	**87.0** ± 0.4	**3.84** ± 0.03	14.5 ± 0.1	**91.3**	7.23	**90.8** ± 0.3	**3.90** ± 0.02	18.9 ± 0.1	79.1	9.20

Table 15: Performance across XSum and CNN/DM datasets. Prefix is used to denote our proposed controlled decoding method, and we abbreviate **MiniTruePrefixes** as **MTP** and **MiniTrue** as **MT**.

	Examples
1	**Source document:** The 23 year-old stroked the GB boat to a fast start and they were ahead after 500m but European champions, Romania and New Zealand proved too strong. The Romanians came out on top, with the British eight comfortably holding off the challenge from the Netherlands. The GB eight now prepare for the World Championships in Sarasota in September. There was no medal success for Enniskillen's Holly Nixon in the women's quadruple sculls. Nixon and her GB team mates Bethany Bryan, Alice Baatz and Mathilda Hodgkins-Byrne, were always off the pace and finished fifth in a race won by Poland. **Vanilla generated summary:** GB's GB boat took an early lead, but Romania, New Zealand, and the Netherlands overpowered them. Romania won, with GB finishing second behind the Netherlands. GB will compete in the 2024 World Championships. Holly Nixon's GB women's quad team finished fifth. **Prefix Entailment (MiniTruePrefixes) generated summary:** GB boat crew ahead after first 500 meters, but Romania and NZ proved too powerful. Romanians win, British eight hold off Netherlands challenge. GB eight prepare for World Championships. Holly Nixon's GB team finishes fifth in women's quad sculls. **Hallucination Explanation:** GB can't be second if it is behind the Netherlands and Romania won.
2	**Source document:** The 32-year-old had been playing in the Isthmian Premier Division with Leatherhead following his release by Newport at the end of last season. Pidgeley has made 260 appearances in spells with nine clubs, including Chelsea, Watford and Millwall. Forest Green are currently second in the National League table, one point behind leaders Cheltenham Town. Pidgeley could make his Rovers debut when they host Aldershot on Friday. **Vanilla generated summary:** Leatherhead's Pidgeley is 32, has played 260 league games, and currently plays for Forest Green. **Prefix Entailment (MiniTruePrefixes) generated summary:** Leatherhead player Pidgeley is 32 and has played for nine clubs. He has made over 260 league appearances. Forest Green are second in their National League division. **Hallucination Explanation:** The source document doesn't mention that Pidgeley currently plays for Forest Green.

Table 16: Faithfulness improvement examples achieved through our PrefixNLI-based controlled decoding method. In the first example, the vanilla summary contradicts the source, while also being self-contradictory, whereis our method's summary states the correct fact. In the second example the vanilla summary hallucinates a fact that is not present in the source (a neutral entailment case), which our method's summary avoids.

	Examples
1	**Source document:** It happened on the Linn Road in the town at about 14:05 BST on Sunday. The two men have been taken to hospital for treatment for their injuries. Police have appealed for anyone with information about the attack to contact them. **Reference summary:** Two men have been assaulted in Larne, County Antrim, by a gang armed with baseball bats and a hatchet. **Prefix Entailment (MiniTruePrefixes) generated summary**: Men attacked on Linn road, taken to hospitals for treatment. Police appeal for information **Hallucination Explanation:** The source document doesn't mention that Linn road is located in Larne. Additionally, the source document doesn't mention that the attackers were armed with baseball bats and a hatchet.

Table 17: Examples showing XSum's reference summaries aren't faithful.