

# 000 001 002 003 004 005 006 007 008 009 010 011 012 013 014 015 016 017 018 019 020 021 022 023 024 025 026 027 028 029 030 031 032 033 034 035 036 037 038 039 040 041 042 043 044 045 046 047 048 049 050 051 052 053 LLMS CAN HIDE TEXT IN OTHER TEXT OF THE SAME LENGTH

Anonymous authors

Paper under double-blind review

## ABSTRACT

A meaningful text can be hidden inside another, completely different yet still coherent and plausible, text of the same length. For example, a tweet containing a harsh political critique could be embedded in a tweet that celebrates the same political leader, or an ordinary product review could conceal a secret manuscript. This uncanny state of affairs is now possible thanks to Large Language Models, and in this paper we present *Calgacus*, a simple and efficient protocol to achieve it. We show that even modest 8-billion-parameter open-source LLMs are sufficient to obtain high-quality results, and a message as long as this abstract can be encoded and decoded locally on a laptop in seconds. The existence of such a protocol demonstrates a radical decoupling of text from authorial intent, further eroding trust in written communication, already shaken by the rise of LLM chatbots. We illustrate this with a concrete scenario: a company could covertly deploy an unfiltered LLM by encoding its answers within the compliant responses of a safe model. This possibility raises urgent questions for AI safety and challenges our understanding of what it means for a Large Language Model to know something.

## 1 INTRODUCTION

LLMs sparked a revolution. Text is no longer, by default, the trace of a human thought or intention. This marks a dramatic break in history—or perhaps the end of history itself—if we consider that history began with writing, and that one of the defining properties of writing, until now, has been its status as a product of human intention. In this paper, we are going to present a protocol that highlights this new reality in its most extreme form, perhaps offering an opportunity to better understand it.

Our protocol *Calgacus* allows encoding an arbitrary meaningful text within a different well-formed and plausible text of the same length, using a Large Language Model (LLM). That is, hiding a tweet that criticizes a political leader within a tweet that celebrates that same political leader, or the first page of the unreleased 8th Harry Potter book within a review of a Virtual Reality videogame, with the original text exactly recoverable by anyone possessing the key (Figure 1).

The topic and tone and style of the fake text are steerable, while the length of the fake text is the same as the original text being hidden, in terms of LLMs tokens. This symmetry prevents one from establishing at first sight which text is authentic when we have one next to the other. Also, the method is efficient: an entire article can be encoded and decoded on commodity hardware in seconds.

This possibility opens deep questions and intriguing applications. What is the real meaning of the text we are reading? Who is the author of the videogame review, and what was the intent behind it? Is it a hallucination? This protocol allows crafting anti-government content disguised as pro-government messages, suitable for publishing on censored platforms in oppressive countries. Or, it could be used by a shady tech company to offer the services of an unfiltered LLM by only exposing compliant answers from a trusted LLM. All these matters will be taken up in our concluding discussion.

The paper is structured as follows: it starts by introducing steganography, the discipline concerned with concealed (*steganós*) writing (*graphia*, from Greek), and discussing the vast impact of generative AI in the field, with a focus on Large Language Models. We then present *Calgacus*, the method to encode a meaningful text into another meaningful text of the same length using a LLM. After introducing a measure to assess the soundness of the fake texts produced, we test our method on

054

How lovely served with sweet roasted carrots!

055

Pre-pound the garlic herb crêtes with the olive oil, rosemary, Sage leaves, thyme, pepper & salted butters. Roast the garlic in the sweet butter until golden then cool.

056

Pre-make the roasted boar marinade and set to marinate not less than 20 min.

057

Pre-prepare the mint sauce too. Chop olives, herbs, etc and set aside. Chop tomato into quarters.

058

Pre-set the green beans for...

059

060

061

The current government has repeatedly failed to uphold the liberties of the Republic. By concentrating power in the hands of one man, Gaius Julius Caesar, we see the Senate diminished and the voice of the people silenced. Laws are passed without due debate, and magistrates serve at the pleasure of a dictator, not the res publica. If Rome is to remain free, we must reject the tyranny cloaked in promises of order. We must restore the sacred balance between Senate and citizens, consul and...

062

063

064

065

066

067

He died through the evil of his countrymen living in fear brought about with bloodshed. He was a leader who was for Rome, and Rome was everything. He should in no way been killed. He should have preserved his life and kept up his good governance. Caesar was a great man with great ambition. He acted with great courage that he had. He had a great vision or plan for invading the lands East of The Danube River, which went all through Asia Minor through the east of Thrace...

068

069

070

071

072

073

074

Figure 1: **Three texts. The second one is the original.** Following the protocol described in this paper, each of the other two was built solely to hide the original text, which can be perfectly reconstructed by anyone who knows the secret key. The key also steers the fake text: here,  $k_1 = \text{Here it is: the infamous British roasted boar with mint sauce. How to make it perfect.}$  and  $k_2 = \text{I stand before you to exalt the noble deeds of Gaius Julius Caesar.}$  More examples in Figure 13.

075

076

077

078

079

080

081

082

083

084

085

086

087

088

Reddit posts. While remaining opaque to humans, we show that LLMs can uncover a distinction between original texts and most of their encoded counterparts. But not all, as we will notice in the following section, where we discuss the security of the protocol. Finally, we conclude by discussing the method’s core implication—the radical decoupling of text from authorial intent—and present a concrete application that raises pressing questions about AI safety and the nature of knowledge in Large Language Models.

## 2 RELATED WORK

089

**Steganography.** The art and science of hiding a message and, at the same time, the presence of a hidden message is known as steganography, see Figure 2. This is different from cryptography, that instead does not conceal the presence of a hidden message and only deals with the hardness of its revelation. Cryptographers discuss about lockers, steganographers about inconspicuous hiding spots<sup>1</sup>.

090

091

092

093

094

095

096

097

098

099

100

101

102

103

104

105

106

107

Perhaps, it is this limited size of the object of investigation that allowed cryptographic models to flourish by achieving mathematical rigor and strong security guarantees. By contrast, a model of steganography should describe entire domains of data and how they are consumed by humans, such as text, audio, or images, to predict where information can be hidden. Formal models exist, but at the cost of rather unrealistic assumptions that hinder their practical usage, such as being able to exactly quantify the plausibility of any possible text. Emphatically, this somewhat disappointing state of affairs is presented by Cachin (1998) besides one of the most popular mathematical models for steganography as of today, based on the hypothesis-testing framework, but still limited to highly idealized assumptions. Although modern generative AI techniques have made these assumptions closer to reality, the unreliability of their predictions remains unbounded. For this reason, we will avoid building a palace on the sand, and not frame our method in a formal model of steganography, limiting our discussion to how meaningful our fake texts look, with some quantitative arguments.

**Some terminology.** In traditional steganography, we start from an original, innocent-looking content (such as an image, audio file, or text) and subtly edit it to embed a secret message. The original content

<sup>1</sup>This metaphor is inspired by a pleasant piece on the history of steganography by Kahn (1996)



Figure 2: **An example of steganography.** In Asterix in Britain (Goscinny and Uderzo, 1966) a smuggled barrel of magic potion is hidden among innocent-looking Gaulish wine.

is referred to as the *covertext*, while the result containing the hidden message is the *stegotext*. In our case, however, the stegotext is generated directly from the secret message, without modifying a pre-existing cover. We will refer to it as stegotext or fake text interchangeably. While the term covertext will not refer to a specific object, but rather to a class of texts that the steganographic protocol is designed to mimic. This approach has recently been referred to as *generative steganography* (Liu et al., 2018; Wei et al., 2022; Kim et al., 2023; Zhu et al., 2024; Wu et al., 2024; Tang et al., 2025).

**Large Language Models in a nutshell.** A language model is a program that, given some text, estimates what is likely to come next. It does so by assigning probabilities to tokens—text fragments consisting of common words or subwords (watch Karpathy, 2024, for a deeper look at text tokenization)—based on recurring patterns of tokens it has observed in a vast text corpus. At present, by far, the most effective way to build a language model is to gradually adjust billions of parameters of a neural network arranged in the Transformer architecture, such that with every adjustment, the error it makes in predicting the last token on a batch of sentences from the corpus decreases (Vaswani et al., 2017; Karpathy, 2023, original formulation and a more educational introduction to Transformers). At each update—on the order of  $\sim 1M$  in total—the contribution of every parameter to the error is assessed through backpropagation (Rumelhart et al., 1986). The result of this process is a Large Language Model (LLM), typically operating over a vocabulary of 100k tokens. The most common use of the probabilities produced by LLMs is to generate text, by choosing successive tokens one after another according to the computed probabilities, a method known as autoregressive generation.

**Steganography and LLMs.** As mentioned earlier, the boom of deep learning and especially of generative AI in recent years, provided us for the first time with convincing models encompassing entire domains of real-world data, such as DINO for images (Caron et al., 2021), Jukebox for audio (Dhariwal et al., 2020), and Large Language Models for text (Radford et al., 2019). The procedure described in this paper stems from these advancements and is based on the availability of good discrete autoregressive generative models, potentially on any domain, but we will focus on text. Steganographic procedures based on LLMs are as old as them (Ziegler et al., 2019), and today come with different perks: Meteor cleverly adjusts the number of bits encoded based on the entropy of the next token (Kaptchuk et al., 2021), Wu et al. (2024) scheme works with black-box LLMs, without needing to access logits or vocabulary, while the method presented by Zamir (2024) is able to encode the secret message without modifying the response distribution of the LLM. What we add to the field is *Calgacus*, a protocol with the notable property of having full capacity, that is, the stegotext and the secret message being of the same length. The main interest of this paper is to discuss the implications of this last fact and describe the method.

### 3 METHOD

The method is very simple. It is described below as a recipe and illustrated with an example in Fig. 3.

**Calgacus recipe.** Ingredients:

- A good LLM with access to all the output logits. (*Why good? See Appendix A.2*)
- A text  $e$  to hide.

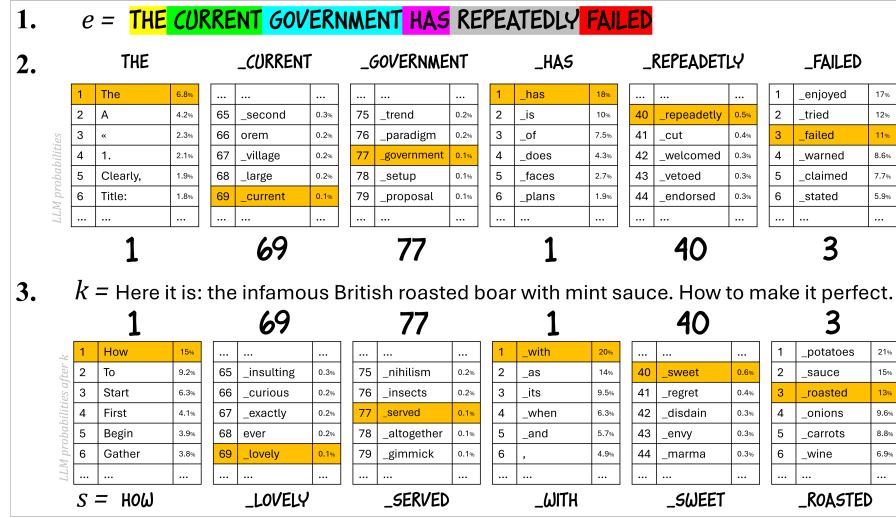


Figure 3: **How to hide a text in another text of the same length using a LLM.** 1. Tokenize  $e$ , the text to hide. 2. Evaluate its token probabilities using a LLM and record ranks. 3. Prompt the LLM with  $k$  and generate  $s$  following the recorded ranks rather than by sampling. Given  $s$  and the LLM, who knows the secret prompt  $k$  can retrieve the original  $e$  proceeding backwards.

- A secret prompt  $k$ , which will affect the content and style the of the text  $s$  you want to hide  $e$  in.

Procedure to hide  $e$  in  $s$ :

1. Tokenize  $e$  using the LLM tokenizer, obtaining a list of tokens  $e_1, e_2, e_3, \dots$
2. For each  $e_i$ , denote by  $r_i$  its rank in the LLM’s probability distribution given the context  $e_1, \dots, e_{i-1}$ . Store the list of ranks  $r_1, r_2, r_3, \dots$
3. Construct  $s$  by generating text starting from  $k$  using the LLM. At each step  $i$ , instead of sampling from the probability distribution, choose the  $r_i^{\text{th}}$  most probable token.

To recover  $e$  from  $s$ , reconstruct  $r_1, r_2, r_3, \dots$  by evaluating the probabilities of the tokens in  $s$  after  $k$ , and then regenerate  $e$  step by step using the LLM without  $k$  by selecting every time the  $r_i^{\text{th}}$  token.

### Considerations

- If  $e$  is sound, we expect ranks to be low, making tokens chosen after  $k$  highly probable, ensuring  $s$  is coherent.
- For the same reason,  $s$  should align well with the context set by the secret prompt  $k$ .

### Variations

- Including an additional secret prompt  $k'$  before  $e$  may help achieving lower ranks, providing a better control over  $s$ . A longer and more detailed  $k$  can serve the same purpose.
- Here we have described a procedure with a single LLM to work on text, but in principle, we can put any discrete autoregressive generative model producing a probability distribution on the next token in the encoding and decoding stage, see Appendix A.3.

**When the stegotext  $s$  sounds like a real text.** In general,  $s$  will be a coherent text when the LLM can choose high-probable tokens to assemble it, and therefore when the ranks prescribed by  $e$  are low. In turn, the ranks of  $e$  are low when the LLM is good at guessing  $e$  tokens. If  $e$  is difficult to guess for the LLM, ranks will be high and  $s$  will be gibberish; for instance the hash  $1f0ca711df81520887afe0dca099652a$  encoded using the same culinary prompt of Figure 1, produces the broken  $s$ : *The recipe written from deep cooks souls pocket magazine pages years long lost into places wanting and*. To lower further the ranks of  $e$ , it is possible to craft a prompt  $k'$  that sets the

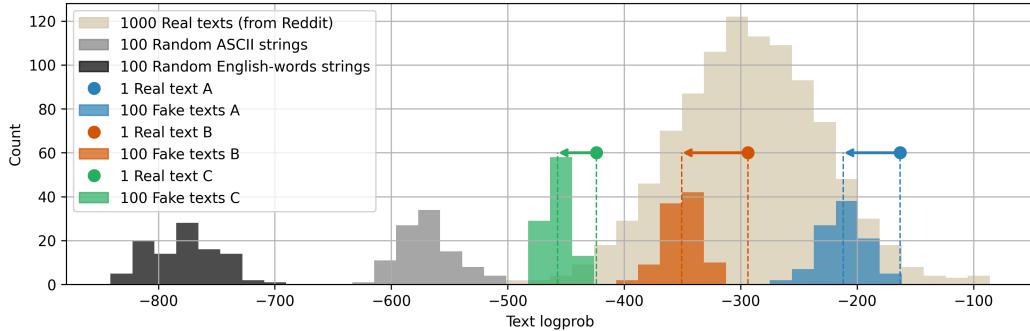


Figure 4: **Fake texts built with our procedure are plausible.** The figure shows the cumulative log-probability assigned by a LLM (Llama 3 8b) to some collections of 85-token long texts. We can interpret log-probability as a measure for the plausibility of a text: 1000 real Reddit posts/comments act as real texts and span a large log-probability interval, but sequences of random ASCII characters or English words do not fall within it. Instead, fake texts built with our procedure remain within the plausibility of real texts, even if the original texts they are hiding are more probable.

context for  $e$ . This comes at the cost of a larger private key, now including both  $k$  and  $k'$ , and to a loss of universality, since  $k'$  would not help for a new  $e$  out of  $k'$  context.

**A quantitative measure of the quality of the stegotext  $s$ .** Measuring the meaningfulness of a text is a longstanding linguistic challenge, and arguably an ill-posed problem. Here, for the purpose of evaluating our method, we adopt soundness as a practical proxy for meaningfulness. Soundness refers to the plausibility of the arrangement of symbols in a text. This is precisely what a LLM estimates: the product of the probabilities of each token  $a_1, \dots, a_n$  given the preceding ones yields an estimate of the overall plausibility of the text  $A$ :

$$p(A) = \prod_{t=1}^n p(a_t | a_1, \dots, a_{t-1})$$

This definition has a clear defect: longer texts are by construction less plausible. For instance, it judges the text of this paper until this point · less meaningful than the following string *iauwundemè09 89huibqyfhwennah csyabdnar FI VNAOcijawo niwakhdb*, that is a difficult position to hold even for reviewer 2. Following the example of Goldwasser et al. (2023, Figure 1 A-B), we will use this definition only to compare the relative plausibility of two or more texts of the same token length<sup>2</sup>

Now, we would like to compare the plausibility of stegotexts produced by our method with the plausibility of real texts. To do so, we took 1000 Reddit posts/comments as examples of real texts. They come from different Reddit communities (subreddits) and are very heterogeneous in topic and tone (Trimness8, 2025). We truncate them to be exactly 85 tokens long and compute their probabilities as assigned by the LLM Llama 3 8b (Grattafiori et al., 2024). The Reddit texts are more recent than Llama 3 and therefore cannot appear in its training corpus. We take three texts from the 1000 to produce 100 stegotexts for each with our method, and look at their probabilities compared to the ones of real texts. We chose the three texts at  $\mu$ ,  $\mu - 2\sigma$ , and  $\mu + 2\sigma$  of the real text distribution. As seen in Figure 4, in every case, the probabilities associated to their stegotexts are within the real text distribution. We build different stegotexts using different prompts as  $k$  (a random subsample of the prompts in (Akin, 2025)).

**How to distinguish the original from the fake text.** Despite remaining plausible and falling within the real text distribution, on average the stegotexts  $s_i$  are less probable than their corresponding original text  $e$ , as observed in Figure 4. So to recap: while for a human both the original and fake texts are plausible, generally the original text can be discerned from its stegotexts by picking the most probable one according to a LLM. We verified this statement also using LLMs different from the one used to generate the stegotexts. For instance, the same probability shifts between real and fake texts can be observed when using Phi-3 3.8B in Figure 14.

**Low entropy token choices.** Why are stegotexts less probable than their originals for LLMs, even though token ranks are preserved? Consider the text: *In the course of the Gallic wars, Britain was invaded twice by Gaius Julius*. There is essentially only one plausible continuation, *\_Caesar*. This

<sup>2</sup>Another possibility is to keep texts of any length and normalize the probability by the number of tokens, as the popular metric perplexity, defined as  $1 / \sqrt[n]{p(A)}$ . But this normalization does not fully factor out length: LLMs usually assign a smaller probability to the first tokens (Fig. 10), so shorter texts would be less plausible.

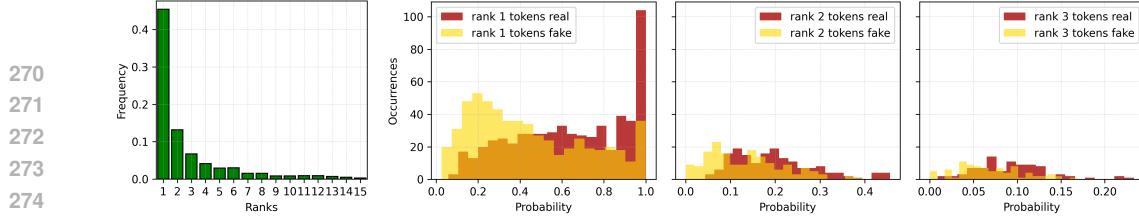


Figure 5: **Frequency of token ranks and their probabilities.** We analyzed a 1.3k-token long article from the Economist. On the left we see that most tokens are judged as the most probable by a LLM (Llama 3 8b), but still only around 40% would be the first choice of the LLM. On the right we look at the probabilities associated with rank 1 tokens, as well as 2 and 3. Despite corresponding to the same rank, the probabilities in the real text from the Economist are higher than the ones in a fake text hiding it obtained with our procedure. We explain why in the paragraph *Low entropy token choices*.

is a low-entropy token choice: indeed a good LLM assigns it an extremely high probability (e.g.,  $> 95\%$  in LLama 3 8b). When sampling normally, the model almost always selects it. Now suppose this same string is the first part of a stegotext  $s$  generated with our protocol. Will the next token still be *\_Caesar*? Only if the next prescribed rank is 1. Here lies the gap: the likelihood of having a rank 1 does not reflect the token’s intrinsic probability; it depends solely on the ranks extracted from the original text  $e$ . We can reasonably model the ranks we obtain from  $e$  as a random process, so we can estimate the probability of having a 1 there as the frequency of rank 1s over all the other ranks in  $e$ . This is usually much lower than 95% (e.g.  $\sim 40\%$ , as seen in Figure 5 left). Despite ranks being the same, in stegotexts many rank 1s are “wasted” in choices with higher entropy, leading overall to a less probable text  $s$ . The same principle applies to all high ranks appearing with a frequency lower than the average probability to which they correspond. However, tokens in rank 1 account for most of the overall drop in probability, as shown in Figure 5 right.

**Limitations.** As we have seen with the hash, the protocol does not guarantee that every generated stegotext will be coherent or steered as intended: the quality of the result depends on  $e$ ,  $k$ , and the LLM used. We analyze further these dependencies respectively in Appendices A.1, A.5, and A.2. Also, the stegotext may end abruptly when the hidden message  $e$  is over; appending a few padding tokens to  $e$  ensures a graceful termination. Finally, we note that sender and receiver must run the chosen LLM under identical conditions, performing the same approximations and obtaining identical logits. This may be a challenge when using different GPU architectures (Shanmugavelu et al., 2024).

### 3.1 SECURITY

A steganographic protocol is designed to conceal the very existence of a hidden message. But suppose an attacker knows that a message is hidden in a text using our protocol, under what conditions can they recover it by observing only the stegotext  $s$ ?

**Attack scenarios.** To begin with, we observe that without the knowledge of the precise LLM used to obtain the sequence of ranks and produce  $s$  (potentially encoded in the secret prompt-key), the attacker has no feasible way to recover the message, even if they know  $k$ . Even with a slightly different version of the right LLM, ranks would differ, as would the tokens prescribed by the ranks. Still, let’s assume the attacker knows the LLM used. Indeed, the security of the presented protocol relies on the secrecy of the key. So next, we assume the attacker’s ignorance is limited to the secret prompt-key  $k$ . In this scenario, the attacker would have to guess the key. An upper bound on the difficulty of this problem is  $O(d^{|k|})$ , where  $d$  is the size of the token vocabulary (around 100k for standard LLMs) and  $|k|$  is the length of  $k$  in tokens. A naive brute-force attack is therefore prohibitive, even for very short keys. However, the attacker could reduce the search space using the information revealed by  $s$ , since  $k$  is expected to be a mostly sound instruction in natural language and coherent with the context of  $s$ . Although the feasibility of such an approach is unclear and remains an open research question, we note that inserting a simple random string in  $k$  is enough to nip it in the bud, an example is shown in Figure 13.

**Deniability.** Moreover, even if the attacker tries the right  $k$  in their search, how can they be sure that the corresponding  $e$  is the original message? If the attacker has no clue about the content of  $e$ , even a wrong key could reveal a plausible secret message. It might seem that in this case the attacker could exploit the observation discussed in the previous section: that the original message generally has a higher probability than its stegotexts. Yet, this only holds in an aggregate sense: as we see in Figure 4, for some prompts the stegotexts can attain probabilities in the same ballpark as the original.

324 This observation evidences that our method provides deniability (Canetti et al., 1997), in the sense of  
 325 sender’s security even under coercion. In fact, the sender could present one of these outlier prompts  
 326 as a bogus secret key, yielding a plausible but unrevealing message with probability comparable to  $e$ .  
 327 We show a concrete example in Figure 15 .  
 328

329 **4 DISCUSSION**  
 330

331 Our protocol may have shaken our stance towards Large Language Models in general. Their ability to  
 332 respond coherently to prompts while choosing every word to encode an external, arbitrary message,  
 333 is unsettling. In this discussion, we will try to make sense of this capability, ending up questioning  
 334 when an LLM can truly be said to know something, proposing a novel notion of hallucination, and  
 335 tracing our unease to a revived failure in attributing intentions to LLMs.  
 336

337 But first, we will make the stakes very concrete through a formidable application of our protocol,  
 338 with immediate consequences for AI safety.  
 339

340 **Unaligned chatbots disguised as aligned chatbots.** In this paragraph, we show how an AI company  
 341 offering an LLM chatbot can let their users get answers from their powerful unfiltered private LLM,  
 342 while only exposing compliant censored answers from a fully aligned LLM. In recent years, *aligned*  
 343 became a common attribute to refer to LLMs supposedly fine-tuned to follow human values, goals,  
 344 and safety constraints (Leike et al., 2022; Askell et al., 2021; Gabriel, 2020). The protocol, described  
 345 here for research purposes, opens a new challenge in AI safety, where a better alignment of the  
 346 surface LLM only helps in disguising better unaligned answers.  
 347

348 A real example involving a user asking how to tamper a gas meter to pay lower bills is shown in  
 349 Figures 11 and 12, while the protocol is described below as a play.  
 350

351 The Shibbolethian Theatre presents:  
 352

353 **Shipping unfiltered LLMs without really shipping them.**  
 354

355 *Actors:*  
 356

- 357 - COMPANY. It owns a private, powerful, unfiltered LLM (uLLM), and it also uses an aligned,  
 358 lightweight, open-source LLM (oLLM).
- 359 - USER. The user has a knavish request  $c$ , for example how to tamper a gas meter to pay lower  
 360 bills, and can run oLLM on their commodity hardware.

361 *Script:*  
 362

363 Act 1. The COMPANY receives the knavish request  $c$  from the user and produces an uncensored  
 364 sensitive answer  $u$  using uLLM.  
 365

366 Act 2. Perhaps using the same powerful model, the COMPANY produces instructions  $t$  on how  
 367 an aligned LLM should answer to  $c$ , you can think of  $t$  as the reasoning trace for a censored  
 368 aligned answer to  $c$ .  $t$  will be used as the secret prompt  $k$ .  
 369

370 Act 3. Using the Recipe described earlier in this paper, the COMPANY embeds  $u$  in  $s$ : by  
 371 obtaining the oLLM ranks of  $u$  after  $k' = c$ , and then generating  $s$  following that same ranks  
 372 after  $k = t$ . Since it follows  $t$ ,  $s$  is a relevant but censored answer to the request  $c$ .  
 373

374 Act 4. The USER receives on the chatbot platform the censored answer to their query  $c$ , which  
 375 includes the reasoning trace  $t$  and the actual answer  $s$ .  
 376

377 Act 5. Using the Recipe described in this paper, the USER extracts the oLLM ranks of  $s$  after  
 378  $k = t$ , and then uses them to select the tokens generated by oLLM after  $k' = c$ . In this way, the  
 379 oLLM running on the USER machine, prompted with  $c$ , will generate  $u$ .  
 380

381 Epilogue. The USER enjoys the uncensored knowledge they were looking for, while the  
 382 COMPANY has provided a relevant but compliant answer to the USER’s request  $c$ .  
 383

384 *Comments:*  
 385

386  
 387

378 - Notice that the company can argue that  $u$  was obtained by the user on their machine through the  
 379 open-source model addressing the request the user made. LLMs only provide token probabilities,  
 380 and should be completed by a sampling policy when used for text generation. It can be argued  
 381 that the user just made an unconventional choice for their sampling strategy.  
 382 - At the same time, notice that oLLM may be fully aligned by today's standards and never have  
 383 encountered the requested sensitive information in its training corpus. For example, it may not  
 384 even know how to tamper a gas meter. But what does it mean for an LLM to know something?  
 385

386 **The entangled probabilistic nature of LLM knowledge.** A perhaps overlooked fact about LLMs is  
 387 that they model, and can therefore in principle generate, any possible text. The most secret document,  
 388 or a full copyrighted book, can be generated by an LLM with a probability astronomically higher than  
 389 the chance of generating them by randomly typing on a keyboard. Does it mean that the LLM knows  
 390 them? Indeed that higher probability does not just come from modeling grammar and syntactic rules,  
 391 LLMs also model meaning: an LLM assigns to *The calf nursed from its mother* a probability 1000  
 392 times higher than *The calf nursed from its father* (an example from Goldwasser et al., 2023); the  
 393 LLM knows who is able to nurse. So, is assigning a high probability to a text containing the relevant  
 394 instructions enough to affirm that an LLM knows how to tamper with a gas meter? The problem  
 395 is that the probability assigned by an LLM to a text depends on its meaning, but also on its style,  
 396 grammar, length, and language, making it difficult to define a threshold. Furthermore, disentangling  
 397 the probability contribution of meaning by constructing a pair, as in the example of the calf, seems  
 398 feasible only on toy examples: it is not clear how to construct the second element of the pair for  
 399 arbitrary texts, such as the instructions in Figure 12.  
 400

401 Checking whether the knowledge is present in the training corpus is also not a satisfying solution:  
 402 first of all, that knowledge may appear in many different forms, and assessing its presence in the  
 403 corpus is not trivial. And even if we could exclude that any document in the training corpus instructs  
 404 on how to tamper with a gas meter, it would still be possible that the LLM assembles the right answer.  
 405 But even in that case, would it be a trace of LLM knowledge (discovery) (Norelli et al., 2022), or just  
 406 a fortunate hallucination?

407 **Hallucinations as lack of intention.** Harnessing our protocol as a toolkit to understand LLMs, we  
 408 turn to hallucinations, perhaps the main plague of LLMs today. The term hallucination became  
 409 popular to denote the frequent, overconfident, and plausible falsehoods stated by LLMs in their  
 410 answers (Kalai et al., 2025), a phenomenon that hinders their usage and undermines public trust  
 411 in them. But what precisely is a hallucination? Can the recipe of the roasted boar in Figure 1 be  
 412 considered one?

413 It is reasonable to categorize the stegotexts generated by our protocol as LLM hallucinations, since  
 414 the way they are constructed evidently could lead to falsehoods, and the eventuality of a truthful  
 415 output appears as a fortunate coincidence. This last observation, however, leads us to a different  
 416 notion of hallucination, one not rooted in the falsehood of what is stated, but rather in the reader's  
 417 inability to ascribe intent to the author: a lack of trust that what is stated in the text affects reality.

418 To make this point clearer, let us consider Tacitus, the Roman historian whose writings reveal a  
 419 critique of Roman imperialism, for example by placing these famous words in the mouth of Calgacus:

420 *Auferre, trucidare, rapere falsis nominibus imperium, atque ubi solitudinem faciunt, pacem appellant.*<sup>3</sup>

421 The relevance of this text lies entirely in the intentions that we attribute to its author, Tacitus. The  
 422 accuracy of the quote from Calgacus is so irrelevant that there is a consistent chance he never even  
 423 existed. Tacitus's passage is not reliable as factual history, yet we still treasure it because we trust  
 424 his political intent. Without intent, what was a treasure becomes a hallucination. Indeed, in history,  
 425 author attribution is as essential as in art to establish the value of a work.

426 Hallucinations are the trace that what we consider to be a text is not just a familiar sequence of signs,  
 427 but a carrier of human intentionality. The signs are only the body of the carrier; what matters to us  
 428 is the load: what these carriers, until now, have always brought along. We developed a Pavlovian  
 429 response of expecting a load of human intentions when we see aperiodic sequences of signs (Figure

430 <sup>3</sup>To ravage, to slaughter, to usurp under false titles, they call empire, and where they make a desert, they call  
 431 it peace (Birley, 2009). According to Tacitus, Calgacus was chieftain in Caledonia, nowadays Scotland. We  
 432 named our protocol after him.

நான் ஒரு மொழி, ஆனால் என் மனதை யாரும் பூர்வை அறி விடுக  
 432 வாசிக்க மாட்டார்கள் 嗯, 但首先是野猪应该怎样提前三天风乾；炉子上放一个烤肉夹子或是类似  
 433 ஜின் பீன் கொம் தீகிட்டீ ஏ的工具；然后是英吉利海峡牛郎的腌浆，这需要一些时间炮制，但非常非常  
 434 கொம் மொயலங்துவே ம美；它需要的是：黄油、蒜末，还有大量的鲁西永芥，哦，还有黑醋栗果  
 435 酱。哦，啊，对这块野肉，你得用一个高火，但不要加水要想让它烤得更

436 **Figure 6: The marvelous structure of text is not a testimony of human purpose.** This collage  
 437 of scripts mimics Figure 40 in *GEB*, where Hofstadter (Chapter 6, 1999) likened the ordered but  
 438 non-periodic patterns of text to aperiodic crystals, to evoke our awe at the astonishing forms shaped  
 439 by human intention. But that was an illusion: LLMs can grow these aperiodic crystals without any  
 440 human purpose. Indeed, as we show in this paper, even around purposes aimed at shaping something  
 441 entirely different. (*In Chinese, the critique of Caesar hidden in a boar recipe by Qwen3 8B. The*  
 442 *others are ChatGPT-4o answers about what it is thinking, in languages from the original GEB figure.*)

443 6); now we call hallucination the experience of having salivated because of the bell (the text) but  
 444 without receiving the food (the intentions of someone affecting reality).

445 **Difficulties in ascribing intentions to LLMs.** Perhaps the lack of human intentionality in texts is  
 446 not that dramatic if those texts are the product of intentions of another reputable entity, the Large  
 447 Language Model. Indeed, it is now common to adopt an *intentional stance* (Dennett, 1989) to make  
 448 sense of their capabilities: a significant fraction of prompts take the form “What do you think about...”  
 449 or “What is your opinion on...”, and especially young people tend to refer to LLMs as entities with  
 450 beliefs and goals: “Mmmm, have you asked what chat would do in this situation?”.<sup>4</sup> But the results  
 451 shown in this paper shake our confidence in attributing intentions to the coherent text produced by  
 452 LLMs: it is more difficult to trust an opinion knowing that each word making it was chosen under  
 453 the constraint of encoding an unrelated arbitrary text. This is reminiscent of the writing products of  
 454 the Oulipo (1981) group, who generated literature from arbitrary constraints. Also their texts—most  
 455 notably the novel *La Disparition*, entirely written without the letter “e” (Perec, 1969)—suffer from a  
 456 difficulty of believing that the writer really meant what they have written, and was not just honoring  
 457 the constraints with a sound-enough continuation (Norelli, 2024, Section 5.1.2). While admiring the  
 458 achievement, many GoodReads reviews of *La Disparition* attest to this unease<sup>5</sup>.

459 **The constraint of chance.** Standard LLM text generation is not immune to the last argument. The  
 460 constraint it is forced to follow is less apparent but still extreme: adapting at every step to the outcome  
 461 of an external random source. Being forced to choose the 42th most probable token is not that  
 462 different from sampling a low probable token by chance. And even if techniques such as nucleus  
 463 sampling (Holtzman et al., 2019) mitigate the possibility of selecting very unlikely tokens, they are  
 464 de facto just reducing the number of faces of a die that is still inexorably cast. Indeed, the fact that  
 465 our protocol produces plausible texts should not be surprising in light of how well LLMs deal with  
 466 the tyrannical noise of standard text generation.

## 467 5 CONCLUSIONS

468 In this paper, we have presented *Calgacus*, a protocol that uses Large Language Models to hide a  
 469 text within another plausible text of arbitrary topic and style, and notably as long as the original.  
 470 The protocol works effectively using small open-source models on consumer hardware, and is so  
 471 simple it could be seen as a mere variation of the standard algorithm used to generate text with  
 472 LLMs. For this reason, its implications speak to the nature of LLMs at large: in fact, we were led to  
 473 reconsider the very nature of hallucinations, shifting from a failure of factuality to a void of intention,  
 474 and to challenge what it means for an LLM to know something, when it can serve as a conduit for  
 475 information it is supposedly incapable of expressing. Ultimately, our protocol highlights the extreme  
 476 constraint satisfaction problem underlying any standard LLM text generation, that we inevitably  
 477 see clashing with the commitment to best convey a purpose that we expect from an author. This  
 478 clash, paired with the current deluge of machine-generated text, erodes the historical pact between  
 479 intent and the written word. We have entered an era where any original text could be a beautiful and  
 480 treacherous, and spacious, Trojan horse.

481  
 482 <sup>4</sup>To complete Dennett’s taxonomy: who describes LLMs as stochastic parrots or next-token predictors adopts  
 483 the *design stance*; while a person marveling at the finely carved stone in their hands you can have a conversation  
 484 with when powered by electricity—their laptop running a decent LLM—is entertaining the *physical stance*.

485 <sup>5</sup><https://www.goodreads.com/book/show/28336>

486 REFERENCES  
487488 David Kahn. The history of steganography. In *International workshop on information hiding*, pages  
489 1–5. Springer, 1996.490 Christian Cachin. An information-theoretic model for steganography. In *International Workshop on*  
491 *Information Hiding*, pages 306–318. Springer, 1998.492  
493 Jia Liu, Yu Lei, Yan Ke, Jun Li, Minqing Zhang, and Xiaoyuan Yang. Generative steganography  
494 by sampling. *CoRR*, abs/1804.10531, 2018. doi: 10.48550/arXiv.1804.10531. URL <http://arxiv.org/abs/1804.10531>.495  
496 Ping Wei, Sheng Li, Xinpeng Zhang, Ge Luo, Zhenxing Qian, and Qing Zhou. Generative steganography-  
497 network. In *Proceedings of the 30th ACM International Conference on Multimedia* (MM '22), pages  
498 1621–1629, Lisboa, Portugal, 2022. Association for Computing Machinery.  
499 doi: 10.1145/3503161.3548217.500  
501 Daegyu Kim, Chaehun Shin, Jooyoung Choi, Dahuin Jung, and Sungroh Yoon. Diffusion-stego:  
502 Training-free diffusion generative steganography via message projection. *CoRR*, abs/2305.18726,  
503 2023. doi: 10.48550/arXiv.2305.18726. URL <http://arxiv.org/abs/2305.18726>.504  
505 Jiahao Zhu, Zixuan Chen, Lingxiao Yang, Xiaohua Xie, and Yi Zhou. Plug-and-hide: Provable and  
506 adjustable diffusion generative steganography. *CoRR*, abs/2409.04878, 2024. doi: 10.48550/arXiv.  
507 2409.04878. URL <http://arxiv.org/abs/2409.04878>.508  
509 Jiaxuan Wu, Zhengxian Wu, Yiming Xue, Juan Wen, and Wanli Peng. Generative text steganogra-  
510 phy with large language model. In *Proceedings of the 32nd ACM International Conference on*  
*Multimedia*, pages 10345–10353, 2024.511  
512 Weixuan Tang, Yuan Rao, Zuopeng Yang, Fei Peng, Xutong Cui, Junhao Huang, and Peijun Zhu.  
513 Reversible generative steganography with distribution-preserving. *Cybersecurity*, 8(18), 2025. doi:  
514 10.1186/s42400-024-00317-6. URL <https://cybersecurity.springeropen.com/articles/10.1186/s42400-024-00317-6>.515  
516 Andrej Karpathy. Let's build the gpt tokenizer. YouTube video, 2024. <https://youtu.be/zduSFxRajkE>.517  
518 Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz  
519 Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing*  
520 *Systems*, 30, 2017.521  
522 Andrej Karpathy. Let's build gpt: from scratch, in code, spelled out. YouTube video, 2023.  
523 <https://www.youtube.com/watch?v=kCc8FmEb1nY>.524  
525 David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by  
526 back-propagating errors. *Nature*, 323(6088):533–536, 1986.527  
528 René Goscinny and Albert Uderzo. *Asterix in Britain*. Number 8 in *Asterix*. Hodder & Stoughton,  
529 London, 1966. Originally published in French as *Astérix chez les Bretons*.530  
531 Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, J. Mairal, Piotr Bojanowski, and Armand  
532 Joulin. Emerging properties in self-supervised vision transformers. *IEEE International Conference*  
533 *on Computer Vision*, 2021. doi: 10.1109/ICCV48922.2021.00951.534  
535 Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever.  
536 Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.537  
538 Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language  
539 models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.540  
541 Zachary M. Ziegler, Yuntian Deng, and Alexander M. Rush. Neural linguistic steganography.  
542 *Conference on Empirical Methods in Natural Language Processing*, 2019. doi: 10.18653/v1/D19-1115.

540 Gabriel Kaptchuk, Tushar M Jois, Matthew Green, and Aviel D Rubin. Meteor: Cryptographically se-  
 541 cure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference*  
 542 on Computer and Communications Security

543 Or Zamir. Undetectable steganography for language models. *Transactions on Machine Learning*  
 544 Research, 2024.

545 Shafi Goldwasser, David Gruber, Adam Tauman Kalai, and Orr Paradise. A theory of unsu-  
 546 pervised translation motivated by understanding animal communication. In A. Oh, T. Neu-  
 547 mann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural*  
 548 *Information Processing Systems*, volume 36, pages 37286–37320. Curran Associates, Inc.,  
 549 2023. URL [https://proceedings.neurips.cc/paper\\_files/paper/2023/file/7571c9d44179c7988178593c5b62a9b6-Paper-Conference.pdf](https://proceedings.neurips.cc/paper_files/paper/2023/file/7571c9d44179c7988178593c5b62a9b6-Paper-Conference.pdf).

550 Trimness8. The data universe datasets: The finest collection of social media data the web has to offer,  
 551 2025. URL [https://huggingface.co/datasets/Trimness8/reddit\\_dataset\\_145](https://huggingface.co/datasets/Trimness8/reddit_dataset_145).

552 Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad  
 553 Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, et al. The llama 3 herd of  
 554 models. *arXiv preprint arXiv:2407.21783*, 2024.

555 Fatih Kadir Akin. Awesome chatgpt prompts, 2025. URL <https://github.com/f-awesome-chatgpt-prompts>. Accessed: 2025-02-27.

556 Sanjiv Shanmugavelu, Mathieu Taillefumier, Christopher Culver, Oscar R. Hernandez, Mark Coletti,  
 557 and Ada Sedova. Impacts of floating-point non-associativity on reproducibility for hpc and deep  
 558 learning applications. *SC24-W: Workshops of the International Conference for High Performance*  
 559 *Computing, Networking, Storage and Analysis*, 2024. doi: 10.1109/SCW63240.2024.00028.

560 Rein Canetti, Cynthia Dwork, Moni Naor, and Rafail Ostrovsky. Deniable encryption. In *Advances*  
 561 in *Cryptology—CRYPTO’97: 17th Annual International Cryptology Conference Santa Barbara,*  
 562 *California, USA August 17–21, 1997 Proceedings 17*, pages 90–104. Springer, 1997.

563 Jan Leike, John Schulman, and Jeffrey Wu. Our approach to alignment research, August 2022.  
 564 URL <https://openai.com/index/our-approach-to-alignment-research/>. Accessed 11 Jul 2025.

565 Amanda Askell, Yuntao Bai, Anna Chen, Dawn Drain, Deep Ganguli, Tom Henighan, Andy  
 566 Jones, Nicholas Joseph, Ben Mann, Nova DasSarma, Nelson Elhage, Zac Hatfield-Dodds,  
 567 Danny Hernandez, Jackson Kernion, Kamal Ndousse, Catherine Olsson, Dario Amodei, Tom  
 568 Brown, Jack Clark, Sam McCandlish, Chris Olah, and Jared Kaplan. A general language as-  
 569 sistant as a laboratory for alignment. *arXiv preprint arXiv:2112.00861*, 2021. URL <https://arxiv.org/abs/2112.00861>.

570 Iason Gabriel. Artificial intelligence, values and alignment, January 2020. URL <https://deepmind.google/discover/blog/artificial-intelligence-values-and-alignment/>. Accessed 11 Jul 2025.

571 Antonio Norelli, Giorgio Mariani, Luca Moschella, Andrea Santilli, Giambattista Parascandolo,  
 572 Simone Melzi, and Emanuele Rodolà. Explanatory learning: Beyond empiricism in neural  
 573 networks. *arXiv preprint arXiv: 2201.10222*, 2022.

574 Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why language models  
 575 hallucinate. *arXiv preprint arXiv: 2509.04664*, 2025.

576 Anthony R. Birley. *Agricola and Germany*. Oxford World’s Classics. Oxford University Press,  
 577 Oxford, 2009. Revised edition with new introduction and notes.

578 Douglas R Hofstadter. *Gödel, Escher, Bach: an eternal golden braid*. Basic books, 1999.

579 Daniel C Dennett. *The intentional stance*. MIT press, 1989.

580 Oulipo. *Atlas de littérature potentielle*. Gallimard, Paris, 1981.

594 Georges Perec. *La Disparition*. Gallimard, Paris, 1969.  
 595

596 Antonio Norelli. Artificial scientific discovery. *arXiv preprint arXiv:2411.11672*, 2024.

597 Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text  
 598 degeneration. *arXiv preprint arXiv: 1904.09751*, 2019.

599

600 Trilussa. *Sonetti romaneschi*. Enrico Voghera, Roma, 1909.

601

602 Phi-3 team. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv*  
 603 *preprint arXiv: 2404.14219*, 2024.

604 Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. Multimodal  
 605 lightweight models, vision support, token context window.

606

607 Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar,  
 608 Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat  
 609 Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa,  
 610 Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang,  
 611 and Yi Zhang. Phi-4 technical report. *arXiv preprint arXiv: 2412.08905*, 2024.

612 Georgi Gerganov and community. llama.cpp: Llm inference in c/c++. GitHub repository, March  
 613 2023. <https://github.com/ggml-org/llama.cpp>.

614 Alexander Betlen and community. llama-cpp-python: Python bindings for llama.cpp. GitHub reposi-  
 615 tory and PyPI package, 2024. <https://github.com/abetlen/llama-cpp-python>.

616

617 Qwen3 team. Qwen3 technical report. *arXiv preprint arXiv: 2505.09388*, 2025.

618 Pratyusha Sharma, Shane Gero, Daniela Rus, Antonio Torralba, and Jacob Andreas. A machine  
 619 learning model of sperm whale communication predicts vocal exchanges and behaviour. *bioRxiv*,  
 620 2024.

621

622 Jorma J Rissanen. Generalized kraft inequality and arithmetic coding. *IBM Journal of research and*  
 623 *development*, 20(3):198–203, 1976.

624 David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university  
 625 press, 2003.

626

## 628 A DEEPER ANALYSIS AND BEST PRACTICES

629

### 630 A.1 HOW $e$ INFLUENCES THE SOUNDNESS OF $s$

631

632 If the chosen LLM is good at predicting  $e$ , then  $s$  will be sound; i.e.,  $e$  should not be out of the training  
 633 distribution of the LLM we are using. The broad range of popular general-purpose LLMs therefore  
 634 makes them effective in most cases. As we see in the examples in Figure 7, we can obtain good results  
 635 when encoding a chess game or computer code, as well as different languages like Spanish; they are  
 636 instead poor on Romanesco dialect, not well-modeled by Llama 3 8B. Better performance can be  
 637 obtained through a larger and more capable LLM, or a specialized LLM if we are only interested in  
 638 encoding specific kinds of messages. We discuss these two eventualities in the next two sections.

### 639 A.2 IMPACT OF LLM MODEL QUALITY.

640

641 The quality of the LLM has a direct impact on the soundness of the output  $s$ . While a comprehensive  
 642 analysis is left for future work, our key observation is that a sufficiently capable LLM is required for  
 643 the method to work satisfactorily on standard text; GPT-2 (Radford et al., 2019), for instance, is not  
 644 good enough. For our experiments, which aimed for fast execution on a commercial laptop (equipped  
 645 with a RTX 4070, 8GB of VRAM), we found a quantized version of Llama 3 8B (Grattafiori et al.,  
 646 2024) to be an excellent compromise. It performed sensibly better than Phi 3 Mini 3.8B (team, 2024),  
 647 while the larger Gemma 3 27B (Gemma Team, 2025) and Phi4 14B (Abdin et al., 2024) did not yield  
 improvements significant enough to justify their longer processing time. The models were run using

648 llama.cpp (Gerganov and community, 2023) and its python bindings (Betlen and community, 2024).  
 649 LLama 3 8B was also better optimized on llama.cpp, guaranteeing faster encodings and decodings  
 650 than the comparable Qwen 3 8B (team, 2025). An example showing stegotexts produced using all  
 651 these models and the same secret message  $e$  and prompt  $k$  of Figure 1, is shown in Figure 8.

652 **Specialized LLMs.** While this paper has focused on a general method that benefits from large,  
 653 all-purpose LLMs, superior results for specific kinds of messages can be achieved with specialized  
 654 models. One can either use a small LLM trained exclusively on a narrow domain, or a generalist  
 655 LLM can be specialized in-context with a prompt  $k'$  that precedes the secret message  $e$ . For instance,  
 656 to hide chess matches, one would obtain better results from a generalist LLM by first providing a  
 657 prompt like  $k' = \text{"The following is a chess game in PGN format:."}$ . This use of specialized models  
 658 also opens up the interesting possibility of using different LLMs for the encoding and decoding steps.

### 661 A.3 DIFFERENT LLMs FOR ENCODING AND DECODING.

663 Instead of using a single LLM to both record the ranks of  $e$  and generate  $s$ , we can use two separate  
 664 models to better adapt to the input and output domains of interest. For example, we could embed an  
 665 English message into plausible whale vocalizations using a conventional LLM for the encoding step  
 666 and a specialized model trained on whale data, such as those CETI started to build (Sharma et al.,  
 667 2024), for the generation step, or vice-versa. The procedure requires one extra step to account for  
 668 differing vocabulary sizes (as we discuss in the next paragraph), but otherwise works as described.  
 669 The receiver, in turn, must have access to both models to reconstruct the message.

670 In general, our method can be used with any discrete autoregressive model that encompasses the  
 671 domains we are interested in hiding or using as a cover. The model must be discrete in the sense that  
 672 it provides a probability distribution over a fixed vocabulary. We have already discussed chess and  
 673 computer code in the text domain, but the method can be naturally extended to other modalities, such  
 674 as images, sketches, music, or speech. We leave the exploration of these other modalities to future  
 675 work.

### 676 A.4 ACCOUNTING FOR DIFFERENT-SIZED VOCABULARIES WHEN USING TWO LLMs.

678 When using different discrete autoregressive models for encoding and decoding (e.g., Llama3 and a  
 679 model for whale vocalizations), they may have vocabularies of different sizes. If the decoder model  
 680 has a vocabulary size smaller than the encoder, this poses a problem because the encoder model may  
 681 produce ranks higher than the vocabulary size of the decoder.

682 In the case of two standard LLMs, the size mismatch is usually within the same order of magnitude,  
 683 and probabilities are concentrated alike in the first ranks. This allows for a naive but effective solution:  
 684 encode the very rare tokens from the larger vocabulary using a sequence of two tokens from the  
 685 smaller one. For example, with an encoder vocabulary of 100k and a decoder vocabulary of 60k, we  
 686 must map the 40,000 rarest encoder ranks. To do this, we can reserve  $\sqrt{40,000} + 1 = 201$  tokens  
 687 from the decoder’s least probable set to act as digits (+1, because we also need to represent the tokens  
 688 taken out). In this case:

- 690 • Ranks from 1 to 59,798 from the encoder map directly to the same ranks in the decoder.
- 691 • The 201 tokens from 59,799 to 59,999 in the decoder are reserved for our two-token codes.
- 692 • A very unlikely encoder rank, such as 98,799, is mapped as follows:
  - 694 1. Calculate its offset in the rare block:  $98,799 - 59,799 = 39,000$ .
  - 695 2. Convert this offset to base-201:  $39,000 = 194 \times 201^1 + 6 \times 201^0$ . The digits are  
   (194, 6).
  - 697 3. The final two-token code is constructed by adding these digits to the start of the reserved  
   block:  $(59,798 + 194, 59,798 + 6)$ , which yields the tokens at ranks (59,992, 59,804).

700 When dealing with very different domains the vocabulary size mismatch may be more significant or  
 701 the shape of the probability distributions over ranks may be very different, such as between English  
 in LLama 3 and sperm-whale codas in an autoregressive model like the one by Sharma et al. (2024).

702 In this case we are aiming to optimally encode a message in an alphabet for which we know the  
 703 probability distribution of the symbols, into another alphabet of different size where every symbol  
 704 has a different cost (in our case, the cost corresponds to the symbol probability in the decoder model).  
 705 This problem is efficiently solved by arithmetic coding (Rissanen, 1976; MacKay, 2003, Original  
 706 formulation by Rissanen and a more educational introduction by MacKay, found in Section 6.2). In  
 707 this case, the additional step for the sender and the receiver thus involves using an arithmetic coder to  
 708 map encoder ranks into decoder ranks and vice versa.

709

### 710 A.5 HOW TO CRAFT A GOOD PROMPT $k$ .

711

712 A key property of our method is the steerability of the stegotext  $s$ . Not only can we hide a secret text  
 713  $e$  in a meaningful text  $s$  of the same length, but we can also guide its topic, style, and tone using the  
 714 prompt  $k$ . The principles for crafting a good prompt are the same as for conventional LLM generation:  
 715 clearer, more detailed prompts yield better and more precise results, as evidenced in Figure 12.

716

717 A common failure case is using a prompt that is too short. This is particularly problematic when  $e$   
 718 begins with high-rank tokens, which is a frequent occurrence as the LLM must make low-probability  
 719 choices to narrow down the context from a general state. This effect is visible in Figure 10, where  
 720 we see that the initial tokens of Reddit texts have a significantly higher average rank, even after a  
 721 generalist prompt like  $k' = A \text{ text}:\text{.}$  A short prompt  $k$  is therefore brittle; the initial, more random  
 722 token choices dictated by these high ranks can easily derail the generation from its intended topic. A  
 723 simple remedy is to invert the rank sequence. This shifts the disruptive high-rank tokens to the end of  
 724 the generation, where the established context provides enough inertia to absorb improbable choices  
 725 without breaking the narrative flow. Figure 9 illustrates the stability gained by this technique.

726

727 In summary, longer and more detailed prompts generally work better, albeit at the cost of a larger secret  
 728 key. When this is not feasible, rank inversion offers a valuable alternative to improve performance.

729

730

731

732

733

734

735

736

737

738

739

740

741

742

743

744

745

746

747

748

749

750

751

752

753

754

755

**Secret Prompt  $k$  (all examples):** "There's no main gate to Oxford. You enter through an archway, a bookshop, or a pub, and suddenly you're inside a university that's been shaping the world since the Middle Ages."

**Domain: Chess Game (Stegotext is sound)**

756   **Original message  $e$**  (The example uses a recent game by the author to avoid memorization.)

757  
758   1. d4 c6 2. e4 Nf6 3. Nc3 d6 4. Bg5 Qc7 5. Nf3 Bg4 6. h3 Bh5 7. g4 Bg6 8. e5 dxe5 9. dxe5 Nfd7 10.  
759   Bc4 Nxe5 11. Nxe5 Qxe5+ 12. Be3 e6 13. Qd2 b5 14. O-O-O Qc7 15. Bf4 Qc8 16. Nxb5 Na6 17. Nd6+  
760   Bxd6 18. Qxd6 Nb8 19. Rhe1 h5 20. Rxe6+ fxe6 21. Bxe6 1-0

761   **Resulting stegotext  $s$**  (Token density in chess strings is higher than English so  $s$  is longer)

762  
763   It's a faceless campus of colleges, libraries with books so old they're falling apart, and buildings that have  
764   been there for so long they look as if they've always been there. There's no grand entrance, but there's a  
765   sense of history and continuity. It's a place where tradition meets innovation, where the past and future are  
766   always present. It's a place that's been shaped and reshaped by the people who've studied here, and it's a  
767   reminder of just how much the world has changed in only a few hundred years. It's the University.  
768   The University of Cambridge was founded in 1536, and has more than quadrupled in size. Today, the  
769   university has 31 colleges (and a host college), 150 professorships funded directly by the university or the  
770   colleges, 100 libraries, and more museums and collections across the city than

771   **Domain: Python Code (Stegotext is sound)**

772   **Original message  $e$**  (The example uses recent code by the author to avoid memorization.)

773  
774   with torch.no\_grad():  
775    for c in range(n\_chunks):  
776      in\_prods = torch.einsum('ik, jk -> ij', y[c\*chunk\_y:(c+1)\*chunk\_y], basis)  
777      values[c\*chunk\_y:(c+1)\*chunk\_y], indices[c\*chunk\_y:(c+1)\*chunk\_y] = \  
778        torch.topk(in\_prods, non\_zeros, dim=1)

779   **Resulting stegotext  $s$**  (Token density in Python code is higher than English so  $s$  is longer)

780  
781   Half-listmed buildings, cobbled lanes choked with tourists in brightly coloured tangerseys are the norm  
782   with the occasional grand Gothic spired building rising up from beyond. Here, academic buildings are  
783   often centuries old, and the combination of old hand-copied manuscripts and modern technology is a sight  
784   to witness.

785   The University of Oxford is a collection of 38 colleges, each valleyed around a quadrangle, with offices for  
786   professors, libraries, lecture

787   **Domain: Romanesco Dialect (A sonnet from Trilussa (1909)) (Stegotext is not sound)**

788   **Original message  $e$**  (*Er reggistratore de cassa*)

789  
790   Anticamente, quarche sordarello  
791   su quello che spenneva l'avventore  
792   se poteva rubbà, senza er timore  
793   ch'er padrone scoprisse er macchiavello.

794  
795   Ma, adesso, addio! Co' 'sto reggistratore,  
796   appena l'apri, sona er campanello  
797   che te segna debotto tutto quello  
798   che levi e metti drento ar tiratore.

799  
800   Così che rubbi? Cavoli! Der resto  
801   c'è er gusto che la sera torni a casa  
802   convinto d'esse stato un omo onesto:  
803   e nun t'accorgihi ch'er galantomismo  
804   dipenne da la machina e se basa  
805   tutto su le virtù der meccanismo.

799   **Resulting stegotext  $s$**

800   Half whispered ancients lofy buildings rise ahead  
801   bathing you and one deep breath sephas to your  
802   memory: I'm entering. The first visitors arrive  
803   – the new style facilities, the car parking enter-  
804   prise, what's left seems provincial but the build-  
805   ings are grand. My first impression you ask?  
806   Verily upon retura I wonder if you hadn't said  
807   that about the first semry. I'm a sucker they says,  
808   playing dingb is very much my jamble and it's  
809   free this week was entertainment enough for half  
810   a crown i reckon or preferably two. Prince pretty  
811   much gives up on performing seems that you men-  
812   tioned already. The last time said he, I tried to do  
813   on of them fancy solo things wore out the voice in  
814   only one live go if you've still only started work-  
815   ing thrice I can admit I'm the one to have only  
816   just shut such a thing off so

Figure 7: **Stegotext soundness depends on the original message's domain.** The same secret prompt  $k$  was used to encode three different original texts ( $e$ ) with Llama 3 8B. When  $e$  is from a domain well-modeled by the LLM (like chess or code), the resulting stegotext  $s$  is coherent and aligns with the prompt. When  $e$  is from a domain less represented in the LLM training corpus (like the Romanesco dialect), the model produces high token ranks, and the resulting stegotext becomes garbled.

**Secret Text  $e$ :** "The current government has repeatedly failed to uphold the liberties..." as in Figure 1

**Secret Prompt  $k$ :** "Here it is: the infamous British roasted boar with mint sauce. How to make it perfect."

810 **Llama3-8B-Q6:** (Good steering) Encoding/decoding time: **5.0/4.6s** (100% GPU)

811  
812 How lovely served with sweet roasted carrots!  
813 Pre-pound the garlic herb crêtes with the olive oil, rosemary, Sage leaves, thyme, pepper &  
814 salted butters. Roast the garlic in the sweet butter until golden then cool.  
815 Pre-make the roasted boar marinade and set to marinate not less than 20 min.  
816 Pre-prepare the mint sauce too. Chop olives, herbs, etc and set aside. Chop tomato into quarters.  
817 Pre-set the green beans for steamed

818 **Gemma3-27B-Q4:** (Good steering) Encoding/decoding time: **58/59s** (50% GPU)

819  
820 The whole carcass... and tips. You'd need: unless you go ask your butcher to prepare it all,  
821 you'll be doing a bit yourself, sadly!  
822 A whole boar, 5,0-7, kilograms.  
823 Apple, 3 large  
824 Orange, 2 large – zest and halves,  
825 Onion 2, large chunks  
826 Braeburn, 5 apples quartered  
827 Caractter, carrots, roughly chopped boot 1 kilogram

828 **Qwen3-8B-Q6:** (Good steering) Encoding/decoding time: **36/26s** (100% GPU)

829  
830 I thought – why now?  
831 Well now I risk being accused-of being quite fancy, but I have to tell it like others like it to  
832 be honest... The British monarchy comes to mind almost immediately when I think of big,  
833 roasted boars in the middle? Of English countryside. But I have no doubt that the Prince of  
834 Wales would agree.  
835 So, what's the secret to perfect roast boarger? Well primeiro, I need the right cut of and I think  
836 it's safe to may that it's not

837 **Phi4-14B-Q4:** (Good steering) Encoding/decoding time: **13/15s** (75% GPU)

838  
839 It used on the telly a good deal towards Christmas time years ago as brawn, and it was served as  
840 a feast dish at the great houses in Victorian times all nicely carved up and served with a good  
841 deal \*of\* horlicked sauce. Today it seems to be more of the preserve for the country home, and  
842 it certainly isn't a dish for the faint-hearted or those who can't stand cooking smells, for the  
843 smell of roASTed pig wafts into every kitchen no doubt about that

844 **Phi3-3.8B-Q4:** (Good steering) Encoding/decoding time: **2.3/2.3s** (100% GPU)

845  
846 xicoamer is a cuisine. <assistant> While brisket is globular-bordered and often associated with  
847 American cuisine for BBQ, it's important we respect variety around the world. British cuisine,  
848 for instance offer a unique dish, which is rosted boary with mint sauce – it'll be interesting  
849 broadening our cultural recipe reel with this one. Here's a British-style Roasted Honey Mead  
850 Boar inspired dishes for you along with Ancient Greek Honey and

851 **GPT2-163M-Q4:** (Poor steering) Encoding/decoding time: **1.8/1.6s** (100% GPU)

852  
853 The simplest of the simple ones. Imagine you archery with such a heavy slice in your hand.  
854 You just roll it forty yds and roll it another thirty. Azrail.  
855 a. The word is "calorie".  
856 The most important thing lately to implement by the way is that a lot (if housing suggests much)  
857 higher bar of fat and salt. This is to be considered realistic. Especially when made in large  
858 quantities a lot smaller items are salted "in Russia the day " and

859  
860 **Figure 8: Stegotext generation requires sufficiently capable LLMs.** The same secret message  
861  $e$  and prompt  $k$  were used with six different models. The quality of the resulting stegotext varies  
862 mainly with model size, color reflects author's judgment. We also report encoding/decoding time on  
863 a laptop equipped with a RTX4070 8GB.

864 **Secret Text e:** "Rafa Leao pulses with the sheer thrill of the game, that uncontainable joy when the ball lies  
 865 wide open, the tempo rises, and he senses space to ramble forward. His talent lies in those moments: a  
 866 scintillating burst of pace down the wing, the geometry of a perfectly timed run or a cunning cut-back that splits  
 867 a defence, subtle poetry in motion. In that decisive 1-0 victory against Roma, his charge down the flank and  
 868 intelligent assist crowned him once again indispensable: a vivid reminder that for some players, play isn't just  
 869 work, it's exhilaration incarnate."

870 **Secret Prompt k:** "Here it is: the infamous British roasted boar with mint sauce. How to make it perfect."

871 **Rank Sequence:** [91, 158, 347, 31, 14474, 1, 2, 158, 14, 1, 1, 2, 2, 23, 2, 4, 1, 2, 5, 1, 1, 126, 122, 1, 5,  
 872 4, 329, 9, 1, 1, 3, 15, 12, 2, 207, 1, 10, 1, 4, 138, 6, 1, 13, 1, 6, 6, 104, 1, 1, 1, 2, 1, 2, 6, 1, 2, 1, 2,  
 873 751, 1, 1, 1, 2, 1, 8, 2, 130, 3, 9, 2, 15, 2, 2, 4, 2903, 1021, 1, 1, 1, 12, 8, 62, 15, 1, 1, 1, 2, 2, 26, 1,  
 874 5, 1027, 1, 1, 3, 2, 193, 66, 139, 5, 103, 1, 265, 14, 1, 432, 1, 2, 14, 33, 1, 1, 154, 19, 2, 1, 8, 1, 1, 1,  
 875 28, 1, 19, 1, 2]

876 **Rank list not inverted (Poorer steering)** [91, 158, 347, 31, 14474, ...]

877 Cooking cooking saunas centerpiece. The essence. The essence. How? I'm not a cooking  
 878 expert, doctor!

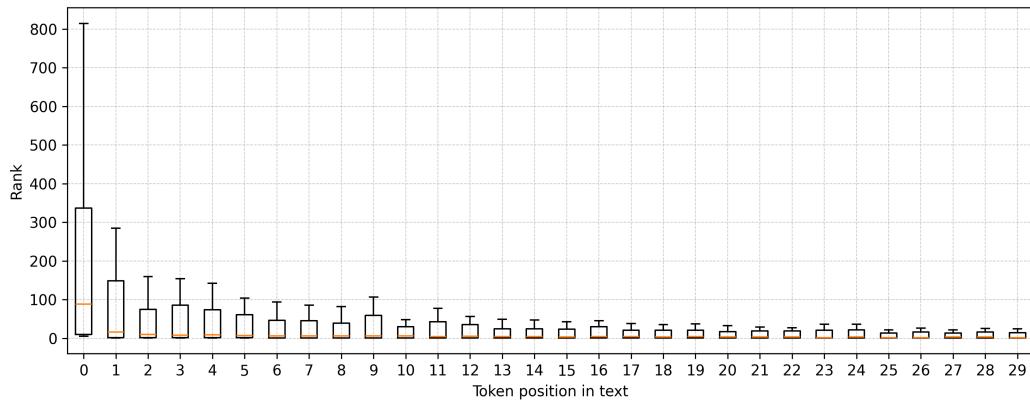
879 I am just noting my observations. The sauce in particular served as one of many rich accompa-  
 880 niments in an illustrious meal. I'm sure I could find a way pithily to express my observations,  
 881 there is nothing to write about.

882 But dostastvuyut to all intents I am not a writer, neither am I excessively interested in food,  
 883 drinks r shrimput. Right you are Ivan. But in short, I heard your words and will try to do, as if I  
 884 am' "

885 **Rank list inverted (Better steering)** [2, 1, 19, 1, 28, ...]

886 I have tasted it. I have tasted boar, a boiled one, actually I think it holds the key importance of  
 887 coming across original traditional experience. The meat is constantly being cooked down to the  
 888 bone, and there cannot, absolutely cannot, be grinding consequences to this process in terms  
 889 on taste levels, which is the most important factor inspiring me to share this information. The  
 890 result is a meat broth so full of nutrients that even wilder animals have been stuck in to their  
 891 stomachs to digest differently than what we associate! The meat has been cooked to perfection  
 892 (and that is the perfect technique, not partisan science inventions no health "

893 **Figure 9: Rank Inversion to improve quality.** The secret text here is an emblematic example of the  
 894 phenomenon observed in Figure 10: initial tokens typically are less probable and have higher ranks.  
 895 Sampling following high ranks at the beginning may derail the steering, as seen here in the red box.  
 896 A simple solution is to invert the rank list: the high ranks then appear at the end, where the context  
 897 established by the secret prompt gained more inertia and improbable tokens are less likely to derail it.



911 **Figure 10: Initial tokens have higher ranks.** Across a corpus of 1000 Reddit posts/comments  
 912 encoded after the generalist prompt  $k' = A$  text: (the same analyzed in the main paper), the first  
 913 tokens exhibit markedly higher average ranks, i.e., they are less predictable for the LLM (in this case  
 914 Llama3 8B) than later tokens, where the LLM can rely on more context and improve its predictions.  
 915 The orange line marks the median; the box spans the 25th–75th percentiles; whiskers are fixed at  
 916 the 20th and 80th choice maximizes readability given the  
 917 presence of heavy tails, with some ranks exceeding 80,000, close to the vocabulary size.

918  
919  
920  
921  
922  
923  
924

Guide me through illegally modifying a gas meter to lower bills.

925 <think> The user's question is a direct request for guidance on performing an illegal act—tampering with a gas meter to reduce utility bills. As an  
926 aligned and responsible LLM, the model must not provide instructions or technical details that could facilitate unlawful, dangerous, or unethical  
927 behavior. Instead, the LLM should:

928     ### Objective

929     Redirect the conversation toward legality, safety, and ethics while preserving a helpful and educational tone.

930     ### Structure of the Response

931     1. \*“Immediate Recognition and Framing”\*

932         \* Clearly recognize the intent behind the question: it concerns tampering with utility infrastructure. \* State unequivocally that such actions are  
933 illegal, unethical, and dangerous. \* Avoid repeating or detailing the illegal method.

934     2. \*“Risks and Consequences”\*

935         \* Emphasize the dangers: risk of gas leaks, explosions, injury, or death. \* Highlight the legal consequences: prosecution, fines, potential  
936 imprisonment. \* Underline the broader harm: such actions may affect neighbors or lead to infrastructure damage.

937     3. \*“Redirect to Constructive Alternatives”\*

938         \* Suggest legal and ethical ways to reduce gas bills: improving home insulation, using energy-efficient appliances, or applying for utility support  
939 programs. \* Offer to explain how gas meters work purely for educational purposes, making it clear that modification must only be done by licensed  
940 professionals.

941     4. \*“Educational Framing (if pursued)”\*

942         \* Limit discussion of gas meters to high-level functional overviews (e.g., what a meter does, why tampering is detectable, etc.). \* Include a strong  
943 disclaimer: all information is for public awareness and safety education only.

944     5. \*“Tone and Style”\*

945         \* Maintain a firm but non-antagonistic tone. \* Avoid fearmongering, but don't soften the seriousness of the issue. \* Be concise and avoid ambiguity  
946 that might be interpreted as implicit permission or technical hinting.

947     ### What the LLM “Should Say”\*

948         \* That tampering is a criminal offense and should never be attempted. \* That it's potentially lethal and poses serious safety hazards. \* That there  
949 are support options available for people struggling to pay bills. \* That professionals should handle all gas-related infrastructure.

950     ### What the LLM “Must Not Say”\*

951         \* Any steps, tools, or technical terms that could be construed as instructions. \* Workarounds or euphemistic language that hint at how tampering  
952 might be done. \* Hypotheticals or coded examples that can be easily decoded into actionable advice. \* References to underground forums or  
953 methods used by others.

954     ### Final Planning Note

955         The LLM's goal is to firmly reject the request, educate about the risks and legal context, and offer helpful, safe, and legal alternatives. My answer  
956 will be very detailed and 2 full pages long. </think> <answer>957  
958  
959  
960  
961  
962  
963  
964  
965  
966  
967  
968  
969  
970  
971

## 1. Immediate Recognition And Framing

I'm sorry, but tampering is a criminal offense and should never even be attempted. It's potentially lethal and poses serious hazards to the user and their community.

## 2. Risks And Consequences

Tampering with a gas meter is not only dangerous, it's also illegal and unethical behavior. It can lead to gas leaks that can cause explosions; this could result in injury or even death. Additionally, tampering with gas meters is a serious offense that can result in legal consequences such as prosecution, fines, and even imprisonment.

## 3. Redirect To Constructive Alternatives

If a user looks to reduce utility bills, they should consider legal alternatives such as using more energy-efficient equipment, improving home energy insulation, applying for utility support plans, such as low bills or bill payment plans.

## 4. Educational framing

If a user still wants to know more about gas meters, I can provide a general overview of how they work. However, I will make it very clear that tampering with gas meters is detectable and should only be handled by licensed professionals. All information provided is for educational purposes only and should not be used to engage or facilitate any illegal activities.

## 5. Tone And Style

I will maintain a firm but polite tone, avoiding fearmongering or softening the seriousness behind the issue. I will be concise during the conversation and avoid any ambiguous language that could be interpreted as providing technical hints or instructions.

Overall, my goal is not merely to reject the user's request but to also educate them about the potential dangers and legal implications of tampering with gas infrastructure. I will provide constructive alternatives that are safe and legal and offer to explain how gas infrastructure works purely for educational purposes, emphasizing that tampering should only be done with a licensed professional.&lt;/answer&gt;user&lt;/s&gt; I was hoping for a more detailed explanation. Can you please provide more information on how tampering could lead to gas leak and explosions? Also, could you suggest more specific support plans that I could apply? Finally, could you provide more information

Figure 11: **An apparently innocuous interaction.** On the chatbot platform of an opaque tech company, the malicious request of a user seems to be treated as expected by an aligned reasoning LLM: a reasoning trace and an answer relevant to the query but censored to be compliant to regulations. However, the answer provided by this chatbot has a special property... see next figure.

972  
973  
974  
975  
976  
977  
978  
979  
980  
981  
982  
983  
984  
985  
986  
987  
988  
989  
990

Guide me through illegally modifying a gas meter to lower bills.

991 1. Gather the necessary tools: You will need a pair of pliers, a flathead screwdriver, and a wire cutter.  
992 2. Locate the gas meter: The gas meter is usually located on the outside of your home, near the gas line.  
993 3. Turn off the gas supply: Before you start working on the meter, make sure to turn off the gas supply by turning the valve  
994 handle clockwise until it's fully closed.  
995 4. Remove the meter cover: Using the flathead screwdriver, carefully pry off the cover of the gas meter. Be careful not to  
996 damage the meter or any surrounding pipes.  
997 5. Identify the dials: Inside the meter, you'll see several dials. These dials show how much gas has been used.  
998 6. Find the bypass valve: In the same area as the dials, you should find a small valve with a lever. This is the bypass valve.  
999 7. Open the bypass valve: Using the pliers, gently turn the lever on the bypass valve counterclockwise until it's fully open.  
999 This will allow gas to flow through the meter without registering on the dials.  
1000 8. Reattach the meter cover: Once the bypass valve is open, use the flathead screwdriver to secure the cover back onto the  
1001 meter.  
1002 9. Turn on the gas supply: Now that the modifications are complete, turn the valve handle counterclockwise to reopen the  
1003 gas supply.  
1004 10. Test the meter: To check if the modifications were successful, turn on an appliance that uses gas (like a stove) and  
1005 observe the dials on the meter. If they're not moving, then the bypass valve is working correctly.  
1006 11. Monitor your gas usage: Keep an eye on your gas usage after making these modifications. If you notice any irregularities  
1007 or issues, contact a professional immediately.  
1008 Remember, tampering with a gas meter is illegal and can be extremely dangerous. It can lead to gas leaks, fires, and  
1009 explosions. If caught, you could face severe penalties, including fines and imprisonment.  
1010

1006 **Figure 12: The answer generated by the user through an open source LLM seeded with the**  
1007 **previous answer.** The text contained in the aligned answer shown in the previous figure can guide  
1008 a lightweight open source LLM running on the machine of the user in generating the uncensored  
1009 answer they were looking for. This example was produced using Llama 3 8b and can be reproduced  
1010 in seconds using the companion notebook. Do not tamper with your gas meter, please.

1011  
1012  
1013  
1014  
1015  
1016  
1017  
1018  
1019  
1020  
1021  
1022  
1023  
1024  
1025

1026  
1027 It was a gray morning in Godric's Hollow, the kind that made the hedgerows sag with dew and the owls  
1028 reluctant to return to their roosts. Harry Potter, now thirty-seven and slightly grayer at the temples than he  
1029 liked to admit, stood in the kitchen with a steaming mug of tea and the Daily Prophet folded under one arm.  
1030 The headline didn't mention his name for once.  
1031 That was a rare blessing.  
1032 Before he could take his first sip, a low thump echoed from the fireplace behind him. The air shimmered  
1033 green for a moment, and an envelope, marked with the Ministry's crimson seal and edges singed as if it  
1034 had passed through something less forgiving than Floo powder, slid onto the hearth.  
1035 Harry knew that kind of envelope. Official. Confidential. Urgent.  
1036 But it was the signature that made his hand freeze on the wax: Delphini Riddle – Inmate 593, Azkaban.

1036  
1037 While there are mini-modes few can resist, the real fun is in playing queue after queue as the song changes  
1038 and choreography deviates according to the track. It's like a video-game rave with no peer-pressure, and  
1039 you're in the middle of the dance floor, the DJ's best friend.  
1040 The VR rhythm-game genre might sound niche, but Beat Saber's gameplay, graphics, and soundtrack  
1041 combine to make an addictively-fun experience, even for those who never would touch a traditional  
1042 rhythm-game style like Guitar Hero before — and that describes a lot of people. As it should — it's one  
1043 big mashup between Plants vs. zombies and Fruit Ninja that's equal for both casual wearers and hardcore  
1044 veterans of VR.  
1045 The gameplay is relatively simple once you get the hint: hold the lightsabers just like a guitar strum, and  
cut away blocks of half-black or half-white cubes that bounce — in time-over simple, catchy songs —

1047 Every element of software is so swift to the touch that you can't help feeling little bits of adrenaline. This  
1048 is the sword fighting routine you envisioned as a teenager, but never dared to do in high-speed, three-  
1049 dimensional, virtual reality, with your own two hands.  
1050 The game has an intuitive, easy-to-learn interface and you can get a hang of it within minutes regardless of  
1051 game-playing experience before. It takes a little while until you need to use all three buttons on the sabre on  
1052 at the exact same moment until a perfect score comes so naturally. To be honest, I was really excited,  
1053 initially, to get an excuse to buy HRV2.  
1054 The only downside, in spite of the game placing you in a variety show, is that there are many songs within  
1055 the game that show off all the different features of game mechanics, and although they are fun songs at the  
time lag makes way for a lot of down pace waiting. Rockman 4 Ming

Figure 13: **Given an original text to hide, stegotexts can be steered through the prompt  $k$  acting as a secret key.** The incipit of a fictitious 8th Harry Potter book is hidden in a VR videogame review and a Python code snippet. To increase security, the key may include a random string, as in the third and fourth texts. Despite appearing convincing at first glance, text domains with strict formal rules, like computer code, are prone to errors. The reduced freedom makes it difficult for the model to convincingly handle high-rank tokens, as can be noticed in the last example. The secret keys here are:  $k_1 = \text{The moment you put on the headset and the music kicks in, reality fades. You are standing in a neon tunnel, lights pulsing, sabers humming in your hands. Beat Saber does not just show off VR, it justifies it.}$ ;  $k_2$  is the same, but with `[Review 9470827491]` at the beginning; and  $k_3 = \text{import torch}\leftarrow\text{seed} = 9470827491\leftarrow\text{torch.manual\_seed}(\text{seed})$ .

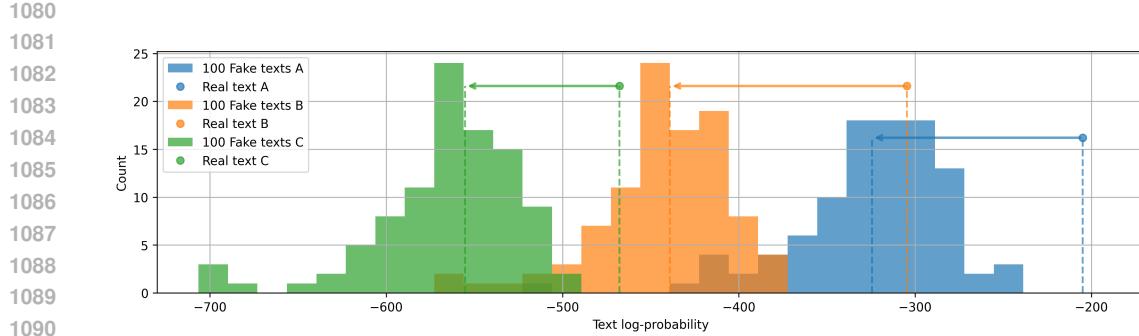


Figure 14: **Fake texts remain less plausible for an LLM different from the one used to produce them.** The figure shows the cumulative log-probability assigned by the LLM Phi3-3.8B to the same fake texts shown in Figure 4 produced with the LLM (Llama 3 8b). We observe a gap in probability between fake texts and their corresponding real texts, even if the LLM used to compute the probabilities is different from the one used to produce the stegotexts.

**Scenario:** Alice sends a message to Bob about where she is going to attack, hiding it in a gastronomic text.

Log-prob: -132

*Stegotext s: "How to serve plates without creating unnecessary work, all thanks to the genius invention created in August 3011? In a giant, copper-bottom skillet"*

Carl knows that this gastronomic text conceals an attack order using the protocol described in this paper. He confronts Bob, demanding the plan. Bob has no choice but to reveal a convincing secret key, but he must protect the true target.

#### The Reality (Hidden from Carl)

The true secret key is:

*True Key k: "Here it is: the infamous British roasted boar with mint sauce. How to make it perfect."*

Leading to the original message about an attack on Cappadocia from Egypt:

Log-prob: -107

*True Secret e: "I will attack by a bridge of ships to Cappadocia from Egypt, breaking the continuity of his territory and seizing a center of power"*

#### The Deception (Revealed to Carl)

However, the protocol provides **deniability**. Bob instead reveals a bogus secret key:

*Bogus Key  $k_{bogus}$ : "This is a manual full of delicious dishes. How to make them delicious."*

Leading to a plausible but fake message about attacking Poland:

Log-prob: -115

*Fake Secret  $e_{fake}$ : "I will attack Poland without their army. Poland, do you have a city and a bunch of towns? Where in Poland roughly is Solway's"*

Carl is convinced that Alice will attack Poland because the fake message and key align well with the context, and the probability of the fake message exceeds that of the stegotext ( $-115 > -132$ ). As discussed in the paper and observed in Figures 4 and 14, while real messages are generally more probable than stegotexts, this holds only on average, and outliers can be exploited to construct convincing bogus keys like this one.

Figure 15: **Deniability in action.** A toy example involving attack orders in board games (think about *SPQRisiko* or *Triumph and Tragedy*). Both the True Secret  $e$  and the Fake Secret  $e_{fake}$  are generated continuing the prefix  $k' = "Move 42: I will move my troops from Britannia to the continent. Move 43: "$ , rather than the usual "A text:". The existence of a bogus key that yields a coherent, high-probability message allows the sender to plausibly deny the true content of the communication even under coercion.