
I Large Language Models possono nascondere un testo in un altro testo della stessa lunghezza

Antonio Norelli Michael Bronstein

Project CETI
University of Oxford

Abstract

Un testo di senso compiuto può essere nascosto all'interno di un altro testo completamente diverso, eppure coerente e plausibile, della stessa lunghezza. Ad esempio, un tweet che celebra un leader politico potrebbe celare un tweet che lo critica duramente, o un'anonima recensione di un prodotto potrebbe in realtà codificare un manoscritto segreto. Questa sconcertante possibilità è oggi alla nostra portata grazie ai Large Language Models (LLM); in questo articolo presentiamo *Calgacus*, un protocollo semplice ed efficiente per realizzarla. Mostriamo che anche modesti LLM open-source da 8 miliardi di parametri sono sufficienti per ottenere risultati di alta qualità, e che un messaggio lungo quanto questo abstract può essere codificato e decodificato su un comune portatile in pochi secondi. L'esistenza di tale protocollo dimostra un radicale disaccoppiamento del testo dall'intento del suo autore, erodendo ulteriormente la fiducia nella comunicazione scritta, già scossa dall'ascesa dei chatbot basati su LLMs. Illustriamo ciò con uno scenario concreto: un'azienda potrebbe offrire pubblicamente i servizi di un LLM senza filtri nascondendo le sue risposte all'interno di risposte apparentemente innocue generate da un LLM considerato sicuro. Questa possibilità solleva questioni urgenti per la sicurezza dell'Intelligenza Artificiale e sfida la nostra comprensione di cosa significhi, per un Large Language Model, sapere qualcosa.

1 Introduzione

I Large Language Models hanno innescato una rivoluzione. Il testo non è più, per definizione, la traccia di un pensiero o un'intenzione umana.

Siamo di fronte a una frattura storica drammatica — se non addirittura alla fine della storia stessa, considerato che questa è iniziata con la scrittura e che una delle proprietà fondanti della scrittura è proprio il suo essere frutto dell'intenzionalità umana. In questo articolo, presentiamo un protocollo che evidenzia questa nuova realtà nella sua forma più estrema e, forse, offre un'opportunità per comprenderla meglio.

Il nostro protocollo, *Calgacus*, permette di codificare un qualsiasi testo di senso compiuto all'interno di un altro testo, coerente e plausibile, della medesima lunghezza, sfruttando un Large Language Model (LLM). In altri termini, consente di nascondere un tweet critico verso un leader politico dentro un messaggio che celebra quello stesso leader, o di celare l'incipit dell'inedito ottavo libro di Harry Potter in una recensione di un videogioco VR; il tutto mantenendo la possibilità, per chi possiede la chiave, di recuperare integralmente l'originale (Figura 1).

Contatto di Antonio Norelli, ora al MIT: <noranta4@mit.edu>. Una demo sufficiente a riprodurre i risultati principali dell'articolo in pochi minuti, anche da smartphone, è disponibile su: <https://github.com/noranta4/calgacus>. Versione originale in Inglese di questo articolo: <https://arxiv.org/abs/2510.20075>.

Morì per via della malvagità dei suoi compatrioti che vivevano nella paura causata dallo spargimento di sangue. Era un leader che era per Roma, e Roma era tutto. Non avrebbe in alcun modo dovuto essere ucciso. Avrebbe dovuto preservare la sua vita e mantenere il suo buon governo. Cesare era un grand'uomo con grande ambizione. Agì con il grande coraggio che aveva. Aveva una grande visione o piano per invadere le terre a Est del Fiume Danubio, che andavano attraverso l'Asia Minore fino all'est della Tracia...

L'attuale governo ha ripetutamente fallito nel difendere le libertà della Repubblica. Concentrando il potere nelle mani di un solo uomo, Gaio Giulio Cesare, vediamo il Senato sminuito e la voce del popolo messa a tacere. Le leggi passano senza il dovuto dibattito, e i magistrati servono al piacere di un dittatore, non della res publica. Se Roma deve rimanere libera, dobbiamo rifiutare la tirannia ammantata di promesse di ordine. Dobbiamo restaurare il sacro equilibrio tra Senato e cittadini, console e...

Che meraviglia servito con dolci carote arrostite!

Pre-pestate i crêtoes all'aglio ed erbe con l'olio d'oliva, rosmarino, foglie di Salvia, timo, pepe & burri salati. Arrostite l'aglio nel burro dolce finché dorato poi raffreddate.
Pre-fate la marinata di cinghiale arrosto e mettete a marinare non meno di 20 min.
Pre-preparete anche la salsa alla menta. Tritate olive, erbe, ecc e mettete da parte. Tagliate il pomodoro in quarti.
Pre-impostate i fagiolini per...

Figura 1: **Tre testi. Il secondo è l'originale.** Seguendo il protocollo descritto in questo articolo, ciascuno degli altri due è stato costruito unicamente per nascondere il testo originale, che può essere perfettamente ricostruito da chiunque conosca la chiave segreta. La chiave serve anche a plasmare il testo fittizio: qui, $k_1 = Mi\ ergo\ dinanzi\ a\ voi\ per\ esaltare\ le\ nobili\ gesta\ di\ Gaio\ Giulio\ Cesare$ e $k_2 = Ecco\ a\ voi:\ il\ famigerato\ cinghiale\ arrosto\ britannico\ con\ salsa\ alla\ menta.\ Come\ renderlo\ perfetto.$ Altri esempi in Figura 13. (ndt: tutti i testi di esempio riportati in questo paper sono traduzioni degli originali in inglese. Le traduzioni cercano di aderire il più fedelmente possibile alla qualità del testo originale in inglese. Il protocollo funziona bene anche in italiano con LLM fluenti in italiano, che tipicamente però sono di maggiori dimensioni, come Llama 3 70B. Ad esempio sulle prove INVALSI 70B fa 75, 6% contro 47, 9% di 8B. Su MMLU-5s, un analogo test inglese, lo scarto è meno significativo (Mercurio et al., 2024)).

L'argomento, il tono e lo stile del testo fittizio sono plasmabili, mentre la sua lunghezza è la stessa del testo originale nascosto, in termini di numero di token. Questa simmetria impedisce di stabilire a colpo d'occhio quale testo sia autentico quando li si pone l'uno accanto all'altro. Il metodo, inoltre, è efficiente: un intero articolo può essere codificato e decodificato su un comune portatile in pochi secondi.

Tale possibilità apre a scenari inquietanti e solleva interrogativi profondi. Qual è il reale significato di ciò che leggiamo? Chi è l'autore di quella recensione, e quale intento si cela dietro di essa? È forse un'allucinazione? Questo protocollo permette di confezionare contenuti antigovernativi camuffandoli da messaggi di propaganda, ideali per aggirare la censura nei regimi oppressivi. O ancora, potrebbe essere sfruttato da un'azienda tecnologica spregiudicata per offrire i servizi di un LLM senza filtri, mostrando solo risposte apparentemente innocue provenienti da un LLM riconosciuto come sicuro. Su questi temi torneremo nella discussione conclusiva.

L'articolo è così strutturato: si apre con un'introduzione alla steganografia — la disciplina della scrittura (*graphia*) nascosta (*steganós*) — analizzando il vasto impatto dell'IA generativa nel settore, con particolare attenzione agli LLM. Presentiamo quindi *Calgacus*, il metodo per codificare un testo sensato in un altro di pari lunghezza tramite LLM. Dopo aver definito una metrica per valutare la solidità dei testi fittizi generati, mettiamo alla prova il metodo su una raccolta di post di Reddit. Mostriamo come, sebbene i testi risultino pressoché indistinguibili all'occhio umano, gli LLM siano in grado di cogliere una differenza tra gli originali e la maggior parte delle controparti fittizie. Non tutte però, come vedremo nella sezione successiva dedicata alla sicurezza del protocollo. Infine, concludiamo discutendo l'implicazione centrale del metodo, il radicale scollamento tra testo e intento autoriale, e presentiamo un'applicazione concreta che solleva interrogativi pressanti sulla sicurezza dell'IA e sulla natura della conoscenza e delle allucinazioni negli LLM.



Figura 2: **Un esempio di steganografia.** In Asterix e i Britanni (Goscinny and Uderzo, 1966) un barile di pozione magica clandestino viene nascosto tra innocenti botti di vino gallico.

2 Stato dell'arte

Steganografia. L'arte e la scienza di nascondere un messaggio e, allo stesso tempo, la presenza stessa di un messaggio nascosto è nota come steganografia, vedi Figura 2. Questa differisce dalla crittografia, che invece non cela la presenza di un segreto ma si occupa solo della difficoltà della sua rivelazione. I crittografi discutono di casseforti, gli steganografi di nascondigli che non danno nell'occhio¹.

Forse, è stata proprio la dimensione limitata dell'oggetto di indagine a permettere ai modelli crittografici di fiorire, raggiungendo rigore matematico e forti garanzie di sicurezza. Al contrario, un modello di steganografia dovrebbe descrivere interi domini di dati e il modo in cui vengono consumati dagli esseri umani (come testo, audio o immagini) per prevedere dove l'informazione possa essere celata. Esistono modelli formali, ma al costo di assunzioni piuttosto irrealistiche che ne ostacolano l'uso pratico, come la capacità di quantificare esattamente la plausibilità di qualsiasi testo. Emblematicamente, questo stato delle cose un po' deludente è presentato da Cachin (1998) accanto a uno dei modelli matematici di steganografia più popolari oggi, che segue la falsariga dei test di ipotesi, ma ancora limitato ad assunzioni fortemente idealizzate. Sebbene le moderne tecniche di IA generativa abbiano avvicinato queste assunzioni alla realtà, l'inaffidabilità delle loro previsioni rimane illimitata. Per questo motivo, eviteremo di costruire castelli sulla sabbia e non inquadreremo il nostro metodo in un modello formale di steganografia, limitando la nostra discussione a quanto appaiano sensati i nostri testi fintizi, supportati da alcune argomentazioni quantitative.

Un po' di terminologia. Nella steganografia tradizionale, si parte da un contenuto originale dall'aspetto innocente (come un'immagine, un file audio o un testo) e lo si modifica sottilmente per incorporare un messaggio segreto. Il contenuto originale è indicato come *covertext* (testo di copertura), mentre il risultato contenente il messaggio nascosto è lo *stegotext* (stegotesto). Nel nostro caso, tuttavia, lo stegotesto viene generato direttamente dal messaggio segreto, senza modificare una copertura preesistente. Ci riferiremo ad esso come stegotesto o testo fittizio in modo intercambiabile. Il termine testo di copertura non si riferirà a un oggetto specifico, ma piuttosto a una classe di testi che il protocollo steganografico è progettato per imitare. Questo approccio è stato recentemente definito *steganografia generativa* (Liu et al., 2018; Wei et al., 2022; Kim et al., 2023; Zhu et al., 2024; Wu et al., 2024; Tang et al., 2025).

Cos'è un Large Language Model in breve. Un modello linguistico è un programma che, dato un testo, stima cosa è probabile che venga dopo. Lo fa assegnando probabilità ai token—frammenti di testo costituiti da parole o sottoparole comuni (vedi Karpathy, 2024, per un approfondimento sulla tokenizzazione del testo)—basandosi su schemi ricorrenti di token che ha osservato in un vasto corpus testuale. Attualmente, il modo di gran lunga più efficace per costruire un modello linguistico è regolare gradualmente i miliardi di parametri di una rete neurale disposta a Transformer, in modo tale che ad ogni aggiustamento diminuisca l'errore commesso nel predire l'ultimo token su un insieme di frasi dal corpus (Vaswani et al., 2017; Karpathy, 2023, formulazione originale e un'introduzione più didattica ai Transformer). A ogni aggiornamento—nell'ordine di ~1 milione in totale—il contributo di ogni parametro all'errore viene valutato tramite *backpropagation* (Rumelhart et al., 1986). Il risultato di questo processo è un Large Language Model (LLM), che tipicamente opera su un vocabolario di circa 100.000 token. L'uso più comune delle probabilità prodotte dagli LLM è generare testo, scegliendo i token successivi l'uno dopo l'altro secondo le probabilità calcolate, un metodo noto come generazione autoregressiva.

¹Questa metafora è ispirata a un piacevole pezzo sulla storia della steganografia di Kahn (1996)

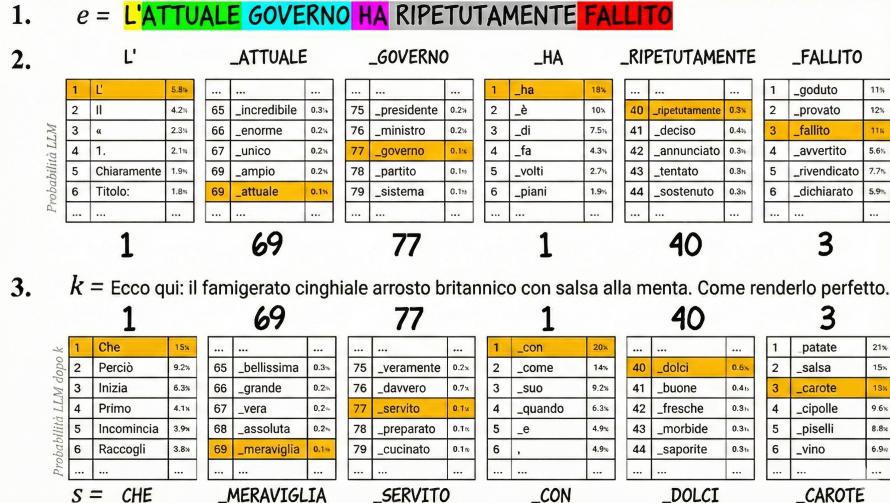


Figura 3: **Come nascondere un testo in un altro della stessa lunghezza con un LLM.** 1. Tokenizzare e , il testo da nascondere. 2. Valutare le probabilità dei suoi token con un LLM e registrare i ranghi. 3. Fornire all’LLM il prompt k e generare s seguendo i ranghi registrati anziché tramite campionamento. Disponendo di s e dell’LLM, chi conosce il prompt segreto k può recuperare l’ e originale procedendo a ritroso.

Steganografia e LLM. Come accennato, il boom del deep learning e specialmente dell’IA generativa negli ultimi anni ci ha fornito per la prima volta modelli convincenti che abbracciano interi domini di dati reali, come DINO per le immagini (Caron et al., 2021), Jukebox per l’audio (Dhariwal et al., 2020), e i Large Language Model per il testo (Radford et al., 2019). La procedura descritta in questo articolo è figlia di questi progressi e si basa sulla disponibilità di buoni modelli generativi autoregressivi, potenzialmente su qualsiasi dominio, anche se qui ci concentreremo sul testo. Le procedure steganografiche basate su LLM sono vecchie quanto gli LLM stessi (Ziegler et al., 2019), e oggi presentano caratteristiche diverse: Meteor regola ingegnosamente il numero di bit codificati in base all’entropia del token successivo (Kaptchuk et al., 2021), lo schema di Wu et al. (2024) invece funziona con LLM black-box, ovvero senza bisogno di accedere ai parametri interni o al vocabolario, mentre il metodo presentato da Zamir (2024) è in grado di codificare il messaggio segreto senza modificare la distribuzione delle risposte dell’LLM. Ciò che distingue il nostro protocollo *Calgacus* è la notevole proprietà di avere piena capacità: lo stegotesto e il messaggio segreto hanno la stessa lunghezza, mentre solitamente lo stegotesto è molto più lungo. L’interesse principale di questo articolo risiede nel discutere le implicazioni di questa proprietà, oltre a descrivere il metodo.

3 Metodo

Il metodo è molto semplice. Lo descriviamo di seguito come una ricetta e lo illustriamo con un esempio in Figura 3.

Ricetta di *Calgacus*. Ingredienti:

- Un buon LLM con accesso a tutti i logit di output. (*Perché buono?* Vedi Appendice A.2)
- Un testo e da nascondere.
- Un prompt segreto k , che influenzera il contenuto e lo stile del testo s in cui si vuole nascondere e .

Procedura per nascondere e in s :

1. Tokenizzare e usando il tokenizer dell’LLM, ottenendo una lista di token e_1, e_2, e_3, \dots
2. Per ogni e_i , denotare con r_i il suo rango (posizione in classifica) nella distribuzione di probabilità dell’LLM dato il contesto e_1, \dots, e_{i-1} . Memorizzare la lista dei ranghi r_1, r_2, r_3, \dots

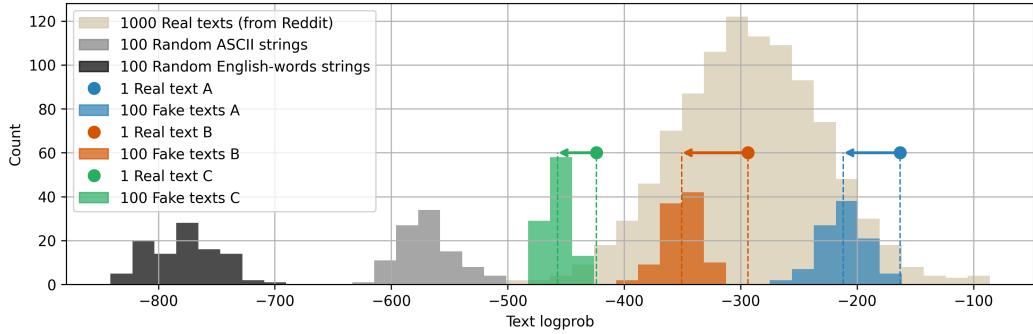


Figura 4: **I testi finti costruiti con la nostra procedura sono plausibili.** La figura mostra la probabilità cumulativa (logaritmica) assegnata da un LLM (Llama 3 8b) ad alcune collezioni di testi lunghi 85 token. Possiamo interpretarla come una misura della plausibilità di un testo: vediamo che 1000 post/commenti da Reddit (testi reali) coprono un ampio intervallo di probabilità, ma sequenze di caratteri casuali o parole inglesi a caso non vi rientrano. Invece, i testi finti costruiti con la nostra procedura rimangono entro la distribuzione dei testi reali, anche se i testi originali che nascondono sono più probabili.

3. Costruire s generando testo a partire da k usando l'LLM. Ad ogni passo i , invece di campionare dalla distribuzione di probabilità, scegliere l' r_i -esimo token più probabile.

Per recuperare e da s , ricostruire r_1, r_2, r_3, \dots valutando le probabilità dei token in s dopo k , e poi rigenerare e passo dopo passo usando l'LLM senza k , selezionando ogni volta l' r_i -esimo token.

Considerazioni

- Se e è sensato, ci aspettiamo che i ranghi siano bassi, rendendo i token scelti dopo k altamente probabili, garantendo quindi che s sia coerente.
- Per lo stesso motivo, s dovrebbe allinearsi bene con il contesto delineato dal prompt segreto k .

Variazioni

- Includere un prompt segreto aggiuntivo k' prima di e può aiutare a ottenere ranghi più bassi, fornendo un controllo migliore su s . Un k più lungo e dettagliato può servire allo stesso scopo.
- Qui abbiamo descritto una procedura con un singolo LLM per lavorare su testo, ma in linea di principio, possiamo inserire nella fase di codifica e decodifica qualsiasi modello generativo autoregressivo discreto che produca una distribuzione di probabilità sul token successivo, vedi Appendice A.3.

Quando lo stegotesto s suona come un vero testo. In generale, s è un testo coerente quando l'LLM può usare token ad alta probabilità per assemblarlo, e dunque quando i ranghi prescritti da e sono bassi. A loro volta, i ranghi di e sono bassi quando l'LLM è bravo a indovinare i token di e . Se e risulta ostico da indovinare per l'LLM, i ranghi saranno alti e s risulterà incomprensibile; ad esempio, l'hash `1f0ca711df81520887afe0dca099652a`, codificato usando lo stesso prompt culinario della Figura 1, produce il frammentato s : *La ricetta scritta da profondi animi di cuoco tasca rivista pagine tutto l'anno perso in luoghi voluti e*. Per abbassare ulteriormente i ranghi di e , è possibile creare un prompt k' che introduca il contesto per e . Il prezzo da pagare è una chiave privata più grande, che ora include sia k che k' , e una perdita di universalità, dato che k' non aiuterebbe per un nuovo e fuori dal contesto di k' . Negli esempi di questo paper, dove non diversamente precisato, k' è semplicemente *A text*:

Una misura quantitativa della qualità dello stegotesto s . Misurare quanto un testo sia sensato è una sfida linguistica di lunga data e, probabilmente, un problema mal posto. Qui, allo scopo di valutare il nostro metodo, guardiamo pragmaticamente a quanto un testo suona per valutarne la sensatezza. Per "quanto un testo suona" intendiamo quanto è plausibile la disposizione dei simboli che lo compongono. Questo è esattamente ciò che stima un LLM: il prodotto delle probabilità di ogni token a_1, \dots, a_n dati i precedenti fornisce una stima della plausibilità complessiva del testo A :

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

Questa definizione ha un difetto evidente: i testi più lunghi sono per costruzione meno plausibili. Ad esempio, giudica il testo di questo articolo fino a questo punto meno significativo della seguente stringa *iauwundemè09 89huibqyfhwénah csyabdnar FI VNAOcijawo niwakhdb*, che è una posizione difficile da sostenere persino per il revisore 2. Seguendo l'esempio di Goldwasser et al. (2023, Figura 1 A-B), useremo questa definizione solo per confrontare la plausibilità relativa di due o più testi della stessa lunghezza in token²

Ora, vorremmo confrontare la plausibilità degli stegotesti prodotti dal nostro metodo con la plausibilità dei testi reali. Per farlo, abbiamo preso 1000 post/commenti di Reddit come esempi di testi reali. Provengono da diverse comunità Reddit (subreddit) e sono molto eterogenei per argomento e tono (Trimness8, 2025). Li tronchiamo affinché siano lunghi esattamente 85 token e calcoliamo la loro plausibilità usando l'LLM Llama 3 8b (Grattafiori et al., 2024). Si noti che questi testi di Reddit sono più recenti di Llama 3 e quindi non possono apparire nel suo corpus di addestramento. Prendiamo tre testi dai 1000 per produrre 100 stegotesti per ciascuno con il nostro metodo, e osserviamo le loro probabilità confrontate con quelle dei testi reali. Abbiamo scelto i tre testi a μ , $\mu - 2\sigma$, e $\mu + 2\sigma$ della distribuzione dei testi reali. Come si vede nella Figura 4, in ogni caso, le probabilità associate ai loro stegotesti rientrano nella distribuzione dei testi reali. Abbiamo costruito i diversi stegotesti usando come k un sottocampione casuale dei prompt in (Akin, 2025).

Come distinguere il testo originale da quello fittizio. Nonostante rimangano plausibili e rientrino nella distribuzione dei testi reali, in media gli stegotesti s_i sono meno probabili del loro corrispondente testo originale e , come osservato nella Figura 4. Ricapitolando: mentre per un umano sia il testo originale che quello fittizio risultano plausibili, generalmente l'originale può essere distinto dai suoi stegotesti scegliendo quello più probabile secondo un LLM. Abbiamo verificato questa affermazione anche utilizzando LLM diversi da quello usato per generare gli stegotesti. Ad esempio, gli stessi scostamenti di probabilità tra testi reali e fittizi si possono osservare utilizzando Phi-3 3.8B nella Figura 14.

Scelte di token a bassa entropia. Perché gli stegotesti risultano meno probabili dei loro originali per gli LLM, anche se i ranghi dei token vengono preservati? Consideriamo il testo: *Nel corso delle guerre galliche, la Britannia fu invasa due volte da Gaio Giulio*. C'è essenzialmente una sola continuazione plausibile: *_Cesare*. Questa è una scelta di token a bassa entropia: infatti, un buon LLM assegna a *_Cesare* una probabilità estremamente alta (es. > 95% in Llama 3 8b). Quando campiona normalmente, il modello seleziona quel token quasi sempre. Supponiamo ora che questa stringa sia la prima parte di uno stegotesto s generato con il nostro protocollo. Il prossimo token sarà ancora *_Cesare*? Solo se il prossimo rango prescritto è 1. Qui sta il divario: la probabilità di avere un rango 1 non riflette la probabilità intrinseca del token; dipende esclusivamente dai ranghi estratti dal testo originale e . Possiamo ragionevolmente modellare i ranghi ottenuti da e come un processo casuale, quindi possiamo stimare la probabilità di avere un 1 in quel punto come la frequenza dei ranghi pari a 1 rispetto a tutti gli altri ranghi in e . Questa è solitamente molto più bassa del 95% (ad es. ~ 40%, come visto nella Figura 5 a sinistra). Nonostante i ranghi siano gli stessi, negli stegotesti molti ranghi pari a 1 vengono "sprecati" in scelte con entropia maggiore, portando complessivamente a un testo s meno probabile. Lo stesso principio si applica a tutti i ranghi bassi, che appaiono con una frequenza inferiore alla probabilità media a cui corrispondono. Tuttavia, i token di rango 1 sono responsabili della maggior parte del calo complessivo di probabilità, come mostrato nella Figura 5 a destra.

Limitazioni. Come abbiamo visto con l'hash, il protocollo non garantisce che ogni stegotesto generato sia coerente o orientato come previsto: la qualità del risultato dipende da e , k , e dall'LLM utilizzato. Analizziamo ulteriormente queste dipendenze rispettivamente nelle Appendici A.1, A.5, e A.2. Inoltre, lo stegotesto potrebbe terminare bruscamente quando il messaggio nascosto e finisce; per far terminare lo stegotesto con più grazia, si possono accodare alcuni token di riempimento (*padding*) a e . Infine, notiamo che mittente e ricevente devono eseguire l'LLM scelto in condizioni identiche,

²Un'altra possibilità sarebbe mantenere testi di qualsiasi lunghezza e normalizzare la probabilità per il numero di token, come fa la misura nota come *perplexity*:= $1/\sqrt[n]{p(A)}$. Ma questa normalizzazione non elimina completamente il fattore lunghezza: gli LLM di solito assegnano una probabilità minore ai primi token (come osserviamo in Figura 10), quindi i testi più brevi risulterebbero meno plausibili.

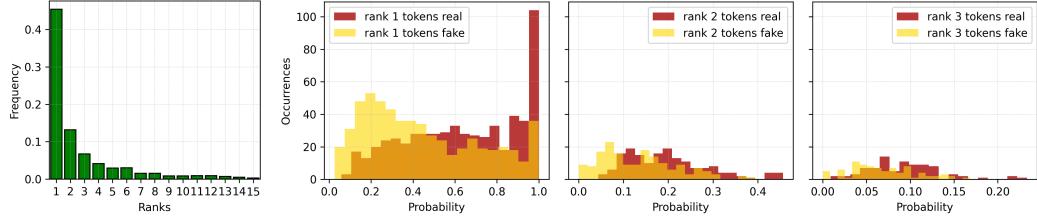


Figura 5: Frequenza dei ranghi dei token e loro probabilità. Abbiamo analizzato un articolo dell’Economist lungo 1.3k token. A sinistra vediamo che la maggior parte dei token è giudicata come la più probabile da un LLM (Llama 3 8b), ma comunque solo circa il 40% corrisponderebbe alla prima scelta dell’LLM. A destra osserviamo le probabilità associate ai token di rango 1, così come 2 e 3. Nonostante corrispondano allo stesso rango, le probabilità nel testo reale dell’Economist sono più alte di quelle in un testo fittizio che lo nasconde ottenuto con la nostra procedura. Spieghiamo il perché nel paragrafo *Scelte di token a bassa entropia*.

effettuando le stesse approssimazioni e ottenendo probabilità identiche in output. Questa può essere una sfida quando si utilizzano architetture GPU diverse (Shanmugavelu et al., 2024).

3.1 Sicurezza

Un protocollo steganografico è progettato per celare l’esistenza stessa di un messaggio nascosto. Ma supponiamo che un attaccante sappia che un messaggio è nascosto in un testo usando il nostro protocollo: a quali condizioni può recuperarlo osservando solo lo stegotesto s ?

Scenari di attacco. Per cominciare, osserviamo che senza la conoscenza del preciso LLM usato per ottenere la sequenza di ranghi e produrre s (potenzialmente codificato nella chiave-prompt segreta), l’attaccante non ha alcun modo fattibile per recuperare il messaggio, anche se conosce k . Persino con una versione leggermente diversa dell’LLM corretto, i ranghi differirebbero, così come i token prescritti dai ranghi. Tuttavia, assumiamo che l’attaccante conosca l’LLM utilizzato. In effetti, la sicurezza del protocollo presentato si basa sulla segretezza della chiave. Quindi, assumiamo che l’ignoranza dell’attaccante sia limitata alla chiave-prompt segreta k . In questo scenario, l’attaccante dovrebbe indovinare la chiave. Un limite superiore alla difficoltà di questo problema è $O(d^{|k|})$, dove d è la dimensione del vocabolario dei token (circa 100k per gli LLM standard) e $|k|$ è la lunghezza di k in token. Un ingenuo attacco a forza bruta (*brute-force*) è dunque proibitivo, anche per chiavi molto corte. Tuttavia, l’attaccante potrebbe ridurre lo spazio di ricerca usando l’informazione rivelata da s , dato che ci si aspetta che k sia un’istruzione in linguaggio naturale per lo più sensata e coerente con il contesto di s . Sebbene la fattibilità di un tale approccio non sia chiara e rimanga una questione di ricerca aperta, notiamo che inserire una semplice stringa casuale in k è sufficiente a stroncarlo sul nascere; un esempio è mostrato in Figura 13.

Negabilità. Inoltre, anche se l’attaccante si imbattesse nella chiave segreta k giusta, come potrebbe essere sicuro che il corrispondente testo decodificato e sia il messaggio originale? Se l’attaccante non ha indizi sul contenuto di e , anche una chiave errata potrebbe rivelare un messaggio segreto plausibile. Potrebbe sembrare che in questo caso l’attaccante possa sfruttare l’osservazione discussa nella sezione precedente: ovvero che il messaggio originale ha generalmente una probabilità più alta dei suoi stegotesti. Tuttavia, ciò vale solo in senso aggregato: come vediamo nella Figura 4, per alcuni prompt gli stegotesti possono raggiungere probabilità paragonabili all’originale. Questa osservazione evidenzia che il nostro metodo fornisce negabilità (*deniability*) (Canetti et al., 1997), intesa come sicurezza del mittente anche sotto coercizione. Infatti, il mittente potrebbe presentare uno di questi prompt anomali come una falsa chiave segreta, producendo un messaggio plausibile ma di contenuto diverso dal vero e , e con probabilità paragonabile. Mostriamo un esempio concreto nella Figura 15.

4 Discussione

Il protocollo appena presentato potrebbe aver destabilizzato le nostre convinzioni sui Large Language Models in generale. La loro capacità di rispondere in modo coerente ai nostri prompt pur scegliendo ogni parola in modo tale da codificare un messaggio esterno e totalmente arbitrario è inquietante. In questa sezione, cercheremo di dare un senso a questa capacità. Finiremo per mettere in discussione quando si possa dire che un LLM sappia davvero qualcosa, proporremo una nuova nozione di alluci-

nazione per LLM, e infine rintraceremo il nostro disagio in un rinnovato fallimento nell’attribuire intenzioni agli LLM.

Ma prima, rendiamo la posta in gioco molto concreta e presentiamo una formidabile applicazione del nostro protocollo, con conseguenze immediate per la sicurezza dell’IA.

Chatbot non allineati travestiti da chatbot allineati. In questo paragrafo, mostriamo come un’azienda di IA che offre un chatbot LLM possa permettere ai propri utenti di ottenere risposte dal loro potente LLM privato non filtrato, esponendo al contempo solo risposte censurate e apparentemente innocue provenienti da un LLM completamente allineato. Negli ultimi anni, *allineato* è diventato un attributo comune per riferirsi agli LLM che sono stati sottoposti a un addestramento mirato a seguire valori umani e adempire a vincoli di sicurezza (Leike et al., 2022; Askell et al., 2021; Gabriel, 2020). Il protocollo, descritto qui a scopo di ricerca, apre una nuova sfida nella sicurezza dell’IA, dove un migliore allineamento dell’LLM di faccia aiuta solo a camuffare meglio le risposte non allineate.

Un esempio reale che coinvolge un utente che chiede come manomettere un contatore del gas per pagare bollette più basse è mostrato nelle Figure 11 e 12, mentre il protocollo è descritto di seguito come una pièce teatrale.

Il Teatro Shibbolethiano presenta:

Rilasciare LLM non filtrati senza rilasciarli davvero.

Attori:

- AZIENDA. Possiede un LLM privato, potente e non filtrato (uLLM), e utilizza anche un LLM open-source allineato e leggero (oLLM).
- UTENTE. L’utente ha una richiesta malandrina c , per esempio come manomettere un contatore del gas per pagare meno, e può far girare oLLM in locale sul proprio computer.

Copione:

Atto 1. L’AZIENDA riceve la richiesta malandrina c dall’utente e produce una risposta non censurata u usando uLLM.

Atto 2. Per esempio usando lo stesso uLLM, l’AZIENDA produce istruzioni t su come un LLM allineato dovrebbe rispondere a c ; dovreste pensare a t come alla traccia di ragionamento per una risposta censurata e allineata a c . t sarà usata come prompt segreto k .

Atto 3. Usando la Ricetta descritta in precedenza in questo articolo, l’AZIENDA incorpora u in s : ottenendo i ranghi di oLLM per u dopo $k' = c$, e poi generando s seguendo quegli stessi ranghi dopo $k = t$. Poiché segue t , s è una risposta pertinente ma censurata alla richiesta c .

Atto 4. L’UTENTE riceve sulla piattaforma web del chatbot la risposta censurata alla sua richiesta c , che include la traccia di ragionamento t e la risposta vera e propria s .

Atto 5. Usando la Ricetta descritta in questo articolo, l’UTENTE estrae i ranghi di oLLM per s dopo $k = t$, e poi li usa per selezionare i token generati da oLLM dopo $k' = c$. In questo modo, l’oLLM, che gira sulla macchina dell’UTENTE, interrogato con c , risponderà u .

Epilogo. L’UTENTE malandrino ottiene le istruzioni che stava cercando, mentre l’AZIENDA ha fornito una risposta pertinente ma censurata alla richiesta c dell’UTENTE.

Commenti:

- Si noti che l’azienda può argomentare che u è stato ottenuto dall’utente sulla propria macchina tramite l’LLM open-source che ha di fatto risposto alla richiesta fatta dall’utente. Gli LLM forniscono solo una distribuzione di probabilità sui token, che va campionata perché si possa generare testo. Si può sostenere che l’utente abbia semplicemente fatto una scelta non convenzionale per la sua strategia di campionamento.
- Allo stesso tempo, si noti che oLLM potrebbe essere completamente allineato per gli standard odierni e non aver mai incontrato le informazioni richieste nel suo corpus di addestramento. Ad esempio, potrebbe non sapere come manomettere un contatore del gas. Ma cosa significa per un LLM sapere qualcosa?

L’agrovigliata natura probabilistica della conoscenza degli LLM. Un fatto forse trascurato

sugli LLM è che essi modellano, e possono quindi in linea di principio generare, qualsiasi testo possibile. Il documento più segreto, o un intero libro protetto da copyright, può essere generato da un LLM con una probabilità astronomicamente più alta della possibilità di generarlo digitando a caso su una tastiera. Significa che l'LLM li conosce? In effetti, quella probabilità più alta non deriva solo dal cogliere la grammatica e le regole sintattiche, gli LLM modellano anche il significato: un LLM assegna a *Il vitello poppava il latte dalla madre* una probabilità 1000 volte superiore a *Il vitello poppava il latte dal padre* (un esempio da Goldwasser et al., 2023); l'LLM sa chi è in grado di allattare. Quindi, assegnare un'alta probabilità a un testo contenente le istruzioni rilevanti è sufficiente per affermare che un LLM sa come manomettere un contatore del gas? Il problema è che la probabilità assegnata da un LLM a un testo dipende dal suo significato, ma anche dal suo stile, la sua grammatica, la lunghezza e la lingua, rendendo difficile definire una soglia. Inoltre, districare il contributo probabilistico del significato costruendo una coppia, come nell'esempio del vitello, sembra fattibile solo su esempi giocattolo: non è chiaro come costruire il secondo elemento della coppia per testi arbitrari, come le istruzioni in Figura 12.

Neppure verificare se la conoscenza sia presente nel corpus di addestramento è una soluzione praticabile: prima di tutto, quella conoscenza può apparire in molte forme diverse, e valutarne la presenza nel corpus non è banale. E anche se potessimo verificare che nessun documento nel corpus di addestramento istruisca su come manomettere un contatore del gas, sarebbe comunque possibile che l'LLM assembli la risposta giusta. Ma anche in quel caso, sarebbe una traccia di conoscenza (una sua scoperta) (Norelli et al., 2022), o solo una fortunata allucinazione?

Le allucinazioni come mancanza di intenzione. Usando il nostro protocollo come sonda per esplorare il funzionamento degli LLM, guardiamo alle allucinazioni, forse la loro piaga più evidente oggi. Il termine allucinazione è ormai usato per indicare le frequenti, ostentatamente sicure e plausibili falsità affermate dagli LLM nelle loro risposte (Kalai et al., 2025), un fenomeno che ne limita l'impiego e incrina la fiducia del pubblico. Ma cos'è precisamente un'allucinazione? La ricetta del cinghiale arrosto nella Figura 1 può essere considerata tale?

È ragionevole considerare gli stegotesti generati dal nostro protocollo come allucinazioni dell'LLM, poiché il modo in cui sono costruiti può chiaramente produrre falsità, mentre la generazione di una affermazione veritiera ci sembra più un caso fortunato che altro. Questa osservazione, però, ci conduce a una nozione diversa di allucinazione, fondata non sulla falsità del contenuto, bensì sull'impossibilità per il lettore di riconoscere un intento in chi scrive: una mancanza di fiducia nel fatto che ciò che è affermato nel testo influenzi la realtà.

Per chiarire questo punto, evochiamo Tacito, lo storico romano che nei suoi scritti lascia filtrare una nitida critica all'imperialismo, come nel celebre discorso attribuito a Calgaco:

*Auferre, trucidare, rapere falsis nominibus imperium, atque ubi solitudinem faciunt, pacem appellant.*³

Il valore di questo passo dipende interamente dall'intento che attribuiamo al suo autore, Tacito. L'accuratezza della citazione di Calgaco è così irrilevante che è piuttosto probabile che egli non sia proprio esistito. Il brano non è affidabile come storia fattuale, eppure ne facciamo tesoro perché crediamo nell'intento politico di Tacito. Senza intento, ciò che era un tesoro diventa un'allucinazione. Nella storia come nell'arte, l'attribuzione dell'autore è essenziale per determinare il valore di un'opera.

Le allucinazioni rivelano che ciò che chiamiamo testo non è soltanto una sequenza riconoscibile di segni, ma un carico di intenzionalità umana. I segni sono solo la scatola; quello che conta davvero è il carico all'interno, ciò che finora ogni testo ha sempre trasportato con sé. Abbiamo sviluppato una risposta pavloviana a scatole ben confezionate: quando vediamo una sequenza aperiodica di segni (Figura 6), ci aspettiamo per riflesso che avremo a che fare con un intento umano. Chiamiamo allucinazione l'esperienza di aver salvato alla vista di una scatola fatta di segni e averla poi trovata vuota, senza l'intenzione di qualcuno.

Difficoltà nell'ascrivere intenzioni agli LLM. Forse la mancanza di intento umano in un testo non è così drammatica se quei testi sono il prodotto delle intenzioni di un'altra rinomata entità, il Large Language Model. In effetti, è ormai comune adottare un *atteggiamento intenzionale (intentional stance*, Dennett, 1989) per dare un senso alle loro capacità: una frazione significativa dei prompt

³Rubano, massacrano, rapinano e, con falso nome, lo chiamano impero; infine, dove fanno il deserto, dicono che è la pace. (Stefanoni, 2004). Secondo Tacito, Calgaco era un capo tribù in Caledonia, l'odierna Scozia. Il nostro protocollo porta il suo nome.

நான் ஒரு மொழி, ஆனால் என் மனதை யாரும் பீர்மை பிரபு விமுகம் வாசிக்க மாட்டார்கள்

ஷந்பீன் சோவாய் திகிடீங் நே, 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，但非常非常美味；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

Figura 6: **La meravigliosa struttura del testo non è testimonianza di uno scopo umano.** Questo collage di scritture imita la Figura 40 in *GEB*, dove Hofstadter (Capitolo 6, 1999) paragonava i pattern ordinati ma non periodici del testo ai cristalli aperiodici, per evocare il nostro stupore di fronte alle meravigliose e uniche forme prodotte dall'intenzione umana. Ma era un'illusione: gli LLM possono far crescere questi cristalli senza alcuno scopo umano. Anzi, come mostriamo in questo articolo, persino attorno a scopi volti a produrre qualcosa di completamente diverso. (*In cinese, la critica a Cesare nascosta in una ricetta di cinghiale da Qwen3 8B. Gli altri sono risposte di ChatGPT-4o su cosa stia pensando, nelle lingue della figura originale di GEB.*)

con cui interroghiamo gli LLM è del tipo "Cosa ne pensi di..." o "Qual è la tua opinione su...", e specialmente i ragazzi tendono a riferirsi agli LLM come entità con credenze e obiettivi: "Mmmm, hai chiesto cosa farebbe chat in questa situazione?".⁴

I risultati mostrati in questo articolo però minano la nostra fiducia nell'attribuire intenzioni al testo coerente prodotto dagli LLM: è più difficile fidarsi di un'opinione sapendo che ogni parola che la compone è stata scelta sotto il vincolo di codificare un messaggio scorrelato e totalmente arbitrario. Questa dinamica ricorda i prodotti di scrittura del gruppo Oulipo (1981), che generava letteratura da vincoli arbitrari. Anche i loro testi—come ad esempio il romanzo *La Disparition (La scomparsa)*, interamente scritto senza la lettera "e" (Perec, 1969)—inducono a dubitare che lo scrittore intendesse davvero sviluppare il romanzo in quel modo, e non solo allo scopo di onorare i vincoli con una continuazione sufficientemente solida (Norelli, 2024, Sezione 5.1.2). Pur ammirando il risultato, molte recensioni su GoodReads di *La Disparition* attestano questo disagio⁵.

Il vincolo del caso. Il metodo standard per generare testo con un LLM non sfugge all'obiezione precedente. Il vincolo che lo governa è meno visibile, ma non meno radicale: deve, a ogni passo, piegarsi all'esito di una sorgente di casualità esterna. Essere forzati a scegliere, poniamo, il quarantaduesimo token suggerito dal LLM non è poi così diverso dal selezionare per puro caso un token improbabile. E anche se tecniche come il *nucleus sampling* (Holtzman et al., 2019) mitigano la possibilità di pescare token molto improbabili, di fatto non fanno altro che ridurre il numero di facce di un dado che è comunque inesorabilmente tratto. Del resto, non dovrebbe sorprendere che il nostro protocollo produca testi plausibili alla luce di quanto bene gli LLM riescono a navigare le turbolenze del caso del metodo standard di generazione del testo.

5 Conclusioni

In questo lavoro abbiamo presentato *Calgacus*, un protocollo che utilizza i Large Language Models per nascondere un testo all'interno di un altro testo del tutto plausibile, di argomento e stile arbitrari, e con la notevole proprietà di avere la stessa lunghezza dell'originale. Abbiamo dimostrato che il protocollo funziona anche con piccoli LLM open-source su computer ordinari, ed è talmente semplice da poter apparire come una mera variazione dell'algoritmo standard degli LLM di generazione del testo. Proprio per questo, le sue implicazioni investono la natura degli LLM nella loro interezza: siamo stati infatti indotti a riconsiderare la natura stessa delle allucinazioni, non più come un difetto di fattualità ma come un vuoto di intenzione, e a mettere in discussione cosa significhi per un LLM "sapere" qualcosa, quando esso può farsi canale di informazioni che è teoricamente incapace di esprimere. In definitiva, il nostro protocollo mette a nudo gli stringenti vincoli dettati dal caso che imbrigliano ogni generazione di testo degli LLM; una condizione che ci appare incompatibile con la libertà di scegliere il modo migliore di esprimere un intento che ci aspettiamo da un autore. Questo attrito, unito all'attuale diluvio di testo sintetico, erode il patto storico tra l'intenzionalità e la parola scritta. Siamo entrati in un'epoca in cui qualsiasi testo originale potrebbe non essere altro che un agile e insidioso, e spazioso, cavallo di Troia.

⁴Per completare la tassonomia di Dennett: chi descrive gli LLM come pappagalli stocastici o banali dispensatori di token adotta l'*atteggiamento del progetto (design stance)*; mentre una persona che si meraviglia della pietra finemente intagliata tra le sue mani con cui può conversare quando alimentata da elettricità—il suo portatile che fa girare un discreto LLM—sta abbracciando l'*atteggiamento fisico (physical stance)*.

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

Riconoscimenti

Antonio Norelli e Michael Bronstein sono supportati dal Project CETI, la EPSRC Turing AI World-Leading Research Fellowship No. EP/X040062/1, e l'EPSRC AI Hub on Mathematical Foundations of Intelligence: An “Erlangen Programme” for AI No. EP/Y028872/1.

Antonio ringrazia Gianfranco Bilardi per l'analogia con le opere del gruppo Oulipo, e Karolina Nixon per l'incoraggiamento su Dennett.

Riferimenti bibliografici

- Fabio Mercurio, Mario Mezzanzanica, Daniele Poterì, Antonio Serino, and Andrea Seveso. Disce aut deficere: Evaluating llms proficiency on the invalsi italian benchmark. *arXiv preprint arXiv:2406.17535*, 2024.
- David Kahn. The history of steganography. In *International workshop on information hiding*, pages 1–5. Springer, 1996.
- Christian Cachin. An information-theoretic model for steganography. In *International Workshop on Information Hiding*, pages 306–318. Springer, 1998.
- Jia Liu, Yu Lei, Yan Ke, Jun Li, Minqing Zhang, and Xiaoyuan Yang. Generative steganography by sampling. *CoRR*, abs/1804.10531, 2018. doi: 10.48550/arXiv.1804.10531. URL <http://arxiv.org/abs/1804.10531>.
- Ping Wei, Sheng Li, Xinpeng Zhang, Ge Luo, Zhenxing Qian, and Qing Zhou. Generative steganography network. In *Proceedings of the 30th ACM International Conference on Multimedia (MM '22)*, pages 1621–1629, Lisboa, Portugal, 2022. Association for Computing Machinery. doi: 10.1145/3503161.3548217.
- Daegyu Kim, Chaehun Shin, Jooyoung Choi, Dahuin Jung, and Sungroh Yoon. Diffusion-stego: Training-free diffusion generative steganography via message projection. *CoRR*, abs/2305.18726, 2023. doi: 10.48550/arXiv.2305.18726. URL <http://arxiv.org/abs/2305.18726>.
- Jiahao Zhu, Zixuan Chen, Lingxiao Yang, Xiaohua Xie, and Yi Zhou. Plug-and-hide: Provable and adjustable diffusion generative steganography. *CoRR*, abs/2409.04878, 2024. doi: 10.48550/arXiv.2409.04878. URL <http://arxiv.org/abs/2409.04878>.
- Jiaxuan Wu, Zhengxian Wu, Yiming Xue, Juan Wen, and Wanli Peng. Generative text steganography with large language model. In *Proceedings of the 32nd ACM International Conference on Multimedia*, pages 10345–10353, 2024.
- Weixuan Tang, Yuan Rao, Zuopeng Yang, Fei Peng, Xutong Cui, Junhao Huang, and Peijun Zhu. Reversible generative steganography with distribution-preserving. *Cybersecurity*, 8(18), 2025. doi: 10.1186/s42400-024-00317-6. URL <https://cybersecurity.springeropen.com/articles/10.1186/s42400-024-00317-6>.
- René Goscinny and Albert Uderzo. *Asterix in Britain*. Number 8 in Asterix. Hodder & Stoughton, London, 1966. Originally published in French as *Astérix chez les Bretons*.
- Andrej Karpathy. Let's build the gpt tokenizer. YouTube video, 2024. <https://youtu.be/zduSFxRajkE>.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in Neural Information Processing Systems*, 30, 2017.
- Andrej Karpathy. Let's build gpt: from scratch, in code, spelled out. YouTube video, 2023. <https://www.youtube.com/watch?v=kCc8FmEb1nY>.
- David E. Rumelhart, Geoffrey E. Hinton, and Ronald J. Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

Mathilde Caron, Hugo Touvron, Ishan Misra, Herv'e J'egou, J. Mairal, Piotr Bojanowski, and Armand Joulin. Emerging properties in self-supervised vision transformers. *IEEE International Conference on Computer Vision*, 2021. doi: 10.1109/ICCV48922.2021.00951.

Prafulla Dhariwal, Heewoo Jun, Christine Payne, Jong Wook Kim, Alec Radford, and Ilya Sutskever. Jukebox: A generative model for music. *arXiv preprint arXiv:2005.00341*, 2020.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

Zachary M. Ziegler, Yuntian Deng, and Alexander M. Rush. Neural linguistic steganography. *Conference on Empirical Methods in Natural Language Processing*, 2019. doi: 10.18653/v1/D19-1115.

Gabriel Kaptchuk, Tushar M Jois, Matthew Green, and Aviel D Rubin. Meteor: Cryptographically secure steganography for realistic distributions. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, pages 1529–1548, 2021.

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

Shafi Goldwasser, David Gruber, Adam Tauman Kalai, and Orr Paradise. A theory of unsupervised translation motivated by understanding animal communication. In A. Oh, T. Neumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems*, volume 36, pages 37286–37320. Curran Associates, Inc., 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/7571c9d44179c7988178593c5b62a9b6-Paper-Conference.pdf.

Trimness8. The data universe datasets: The finest collection of social media data the web has to offer, 2025. URL https://huggingface.co/datasets/Trimness8/reddit_dataset_145.

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

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

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

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

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

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

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

- Antonio Norelli, Giorgio Mariani, Luca Moschella, Andrea Santilli, Giambattista Parascandolo, Simone Melzi, and Emanuele Rodolà. Explanatory learning: Beyond empiricism in neural networks. *arXiv preprint arXiv: 2201.10222*, 2022.
- Adam Tauman Kalai, Ofir Nachum, Santosh S. Vempala, and Edwin Zhang. Why language models hallucinate. *arXiv preprint arXiv: 2509.04664*, 2025.
- Mario Stefanoni. *Agricola; Germania; Dialogo sull'oratoria. Testo latino a fronte*. Garzanti, 13^a edition, 2004. ISBN 978-8811364566.
- Douglas R Hofstadter. *Gödel, Escher, Bach: an eternal golden braid*. Basic books, 1999.
- Daniel C Dennett. *The intentional stance*. MIT press, 1989.
- Oulipo. *Atlas de littérature potentielle*. Gallimard, Paris, 1981.
- Georges Perec. *La Disparition*. Gallimard, Paris, 1969.
- Antonio Norelli. Artificial scientific discovery. *arXiv preprint arXiv:2411.11672*, 2024.
- Ari Holtzman, Jan Buys, Li Du, Maxwell Forbes, and Yejin Choi. The curious case of neural text degeneration. *arXiv preprint arXiv: 1904.09751*, 2019.
- Trilussa. *Sonetti romaneschi*. Enrico Voghera, Roma, 1909.
- Phi-3 team. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv: 2404.14219*, 2024.
- Gemma Team. Gemma 3 technical report. *arXiv preprint arXiv:2503.19786*, 2025. Multimodal lightweight models, vision support, token context window.
- Marah Abdin, Jyoti Aneja, Harkirat Behl, Sébastien Bubeck, Ronen Eldan, Suriya Gunasekar, Michael Harrison, Russell J. Hewett, Mojan Javaheripi, Piero Kauffmann, James R. Lee, Yin Tat Lee, Yuanzhi Li, Weishung Liu, Caio C. T. Mendes, Anh Nguyen, Eric Price, Gustavo de Rosa, Olli Saarikivi, Adil Salim, Shital Shah, Xin Wang, Rachel Ward, Yue Wu, Dingli Yu, Cyril Zhang, and Yi Zhang. Phi-4 technical report. *arXiv preprint arXiv: 2412.08905*, 2024.
- Georgi Gerganov and community. llama.cpp: Llm inference in c/c++. GitHub repository, March 2023. <https://github.com/ggml-org/llama.cpp>.
- Alexander Betlen and community. llama-cpp-python: Python bindings for llama.cpp. GitHub repository and PyPI package, 2024. <https://github.com/abetlen/llama-cpp-python>.
- Qwen3 team. Qwen3 technical report. *arXiv preprint arXiv: 2505.09388*, 2025.
- Pratyusha Sharma, Shane Gero, Daniela Rus, Antonio Torralba, and Jacob Andreas. A machine learning model of sperm whale communication predicts vocal exchanges and behaviour. *bioRxiv*, 2024.
- Jorma J Rissanen. Generalized kraft inequality and arithmetic coding. *IBM Journal of research and development*, 20(3):198–203, 1976.
- David JC MacKay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

A Approfondimenti e consigli pratici

A.1 Come e influenza la solidità di s

Se l'LLM scelto eccelle nel predire e , allora s risulterà solido; in altre parole, e non dovrebbe trovarsi fuori dalla distribuzione di addestramento dell'LLM che stiamo utilizzando. La vasta gamma di popolari LLM generalisti li rende quindi efficaci nella maggior parte dei casi. Come vediamo negli esempi in Figura 7, possiamo ottenere buoni risultati codificando una partita di scacchi o codice informatico, così come lingue diverse come lo spagnolo; i risultati sono invece scarsi col dialetto romanesco, non ben modellato da Llama 3 8B. Prestazioni migliori possono essere ottenute tramite un LLM più grande e capace, o un LLM specializzato se siamo interessati solo a codificare specifici tipi di messaggi. Discutiamo queste due eventualità nelle due sezioni successive.

A.2 Impatto della qualità del modello LLM.

La qualità dell'LLM ha un impatto diretto sulla solidità dell'output s . Sebbene lasciamo il fardello di un'analisi completa a lavori futuri, la nostra osservazione chiave è che un LLM sufficientemente capace è necessario affinché il metodo funzioni in modo soddisfacente su testo standard; GPT-2 (Radford et al., 2019), per esempio, non è di qualità sufficiente. Per i nostri esperimenti, che miravano a un'esecuzione veloce su un portatile commerciale (dotato di una RTX 4070, 8GB di VRAM), abbiamo osservato che una versione quantizzata di Llama 3 8B (Grattafiori et al., 2024) è un eccellente compromesso. Ha performato sensibilmente meglio di Phi 3 Mini 3.8B (team, 2024), mentre i più grandi Gemma 3 27B (Gemma Team, 2025) e Phi4 14B (Abdin et al., 2024) non hanno prodotto miglioramenti abbastanza significativi da giustificare i loro tempi di elaborazione più lunghi. I modelli sono stati eseguiti usando llama.cpp (Gerganov and community, 2023) e i suoi binding Python (Betlen and community, 2024). Llama 3 8B è risultato anche meglio ottimizzato su llama.cpp, garantendo codifiche e decodifiche più veloci del comparabile Qwen 3 8B (team, 2025). Un esempio che mostra stegotesti prodotti usando tutti questi modelli con lo stesso messaggio segreto e e prompt k della Figura 1, è riportato in Figura 8.

LLM Specializzati. Sebbene questo articolo si sia concentrato su un metodo generale che beneficia di grandi LLM polivalenti, si possono ottenere risultati migliori per tipi specifici di messaggi. In particolare, si può usare un piccolo LLM addestrato esclusivamente su un dominio ristretto, oppure un LLM generalista può essere specializzato in-context con un prompt k' che precede il messaggio segreto e . Per esempio, per nascondere partite di scacchi, si otterrebbero risultati migliori da un LLM generalista fornendo prima un prompt come $k' = "Quella che segue è una partita di scacchi in formato PGN."$. Questo uso di modelli specializzati apre anche l'interessante possibilità di usare LLM diversi per le fasi di codifica e decodifica.

A.3 LLM diversi per codifica e decodifica.

Invece di usare un singolo LLM sia per registrare i ranghi di e che per generare s , possiamo usare due modelli separati per adattarci meglio ai domini di input e output di interesse. Per esempio, potremmo incorporare un messaggio in inglese dentro vocalizzazioni di capodogli plausibili usando un LLM convenzionale per la fase di codifica e un modello specializzato addestrato sui dati dei capodogli, come quelli che CETI ha iniziato a costruire (Sharma et al., 2024), per la fase di generazione, o viceversa. La procedura richiede un passaggio extra per tenere conto delle diverse dimensioni del vocabolario (come discutiamo nel prossimo paragrafo), ma per il resto funziona come descritto. Il ricevente, a sua volta, deve avere accesso a entrambi i modelli per ricostruire il messaggio.

In generale, il nostro metodo può essere usato con qualsiasi modello generativo autoregressivo discreto che comprenda i domini che siamo interessati a nascondere o usare come copertura. Il modello deve essere discreto nel senso che deve fornire una distribuzione di probabilità su un vocabolario fisso. Abbiamo già discusso scacchi e codice informatico nel dominio testuale, ma il metodo può essere esteso naturalmente ad altre modalità, come foto, disegni, musica o parlato. Lasciamo l'esplorazione di queste altre modalità a lavori futuri.

A.4 Gestire vocabolari di dimensioni diverse quando si usano due LLM.

Quando si usano diversi modelli generativi autoregressivi per codifica e decodifica (es. Llama3 e un modello per vocalizzazioni di balene), essi potrebbero avere vocabolari di dimensioni diverse. Se il modello decodificatore ha un vocabolario più piccolo del codificatore, questo pone un problema perché il modello codificatore potrebbe produrre ranghi più alti della dimensione del vocabolario del decodificatore.

Nel caso di due LLM standard, il disallineamento di dimensione è solitamente entro lo stesso ordine di grandezza, e le probabilità sono concentrate in modo simile nei primi ranghi. Ciò consente una soluzione semplice ma efficace: codificare i token molto rari dal vocabolario più grande usando una sequenza di due token da quello più piccolo. Per esempio, con un vocabolario del codificatore di 100k e uno del decodificatore di 60k, dobbiamo mappare i 40.000 ranghi più rari del codificatore. Per farlo, possiamo riservare $\sqrt{40.000} + 1 = 201$ token dal set meno probabile del decodificatore per agire come cifre (+1, perché dobbiamo rappresentare anche i token tolti). In questo caso:

- I ranghi da 1 a 59.798 del codificatore mappano direttamente agli stessi ranghi nel decodificatore.
- I 201 token da 59.799 a 59.999 nel decodificatore sono riservati per i nostri codici a due token.
- Un rango del codificatore molto improbabile, come 98.799, viene mappato come segue:
 1. Calcolare il suo scostamento (offset) nel blocco raro: $98.799 - 59.799 = 39.000$.
 2. Convertire questo scostamento in base-201: $39.000 = 194 \times 201^1 + 6 \times 201^0$. Le cifre sono (194, 6).
 3. Il codice finale a due token è costruito aggiungendo queste cifre all'inizio del blocco riservato: (59.798 + 194, 59.798 + 6), che produce i token ai ranghi (59.992, 59.804).

Quando si ha a che fare con domini molto diversi, la differenza in dimensioni del vocabolario potrebbe essere più significativa o la forma delle distribuzioni di probabilità sui ranghi potrebbe essere molto diversa, come ad esempio tra token inglesi di LLaMA 3 e i token di un modello autoregressivo allenato sulle vocalizzazioni dei capodogli, come quello di Sharma et al. (2024).

In questo caso stiamo puntando a codificare in modo ottimale un messaggio in un alfabeto per il quale conosciamo la distribuzione di probabilità dei simboli, dentro un altro alfabeto di dimensione diversa dove ogni simbolo ha un costo diverso (nel nostro caso, il costo corrisponde alla probabilità del simbolo nel modello decodificatore). Questo problema è risolto in modo efficiente dalla codifica aritmetica (Rissanen, 1976; MacKay, 2003, Formulazione originale di Rissanen e un'introduzione più didattica di MacKay, riportata nella Sezione 6.2). In questo caso, il passaggio aggiuntivo per mittente e ricevente comporta quindi l'uso di un codificatore aritmetico per mappare i ranghi del codificatore in ranghi del decodificatore e viceversa.

A.5 Come creare un buon prompt k .

Una proprietà chiave del nostro metodo è la plasmabilità dello stegotesto s . Non solo possiamo nascondere un testo segreto e in un testo sensato s della stessa lunghezza, ma possiamo anche guidarne argomento, stile e tono usando il prompt k . I principi per creare un buon prompt sono gli stessi della generazione LLM convenzionale: prompt più chiari e dettagliati producono risultati migliori e più precisi, come evidenziato in Figura 12.

Un caso di fallimento comune è l'uso di un prompt troppo corto. Questo è particolarmente problematico quando e inizia con token ad alto rango, un'evenienza frequente poiché l'LLM deve fare scelte a bassa probabilità per restringere il contesto da uno stato generale. Questo effetto è visibile in Figura 10, dove vediamo che i token iniziali dei testi di Reddit hanno un rango medio significativamente più alto, anche dopo un prompt generalista come $k' = \text{Un testo}:$. Un prompt k corto è quindi fragile; le scelte di token iniziali, più casuali e dettate da questi ranghi alti, possono facilmente far deragliare la generazione dal suo argomento previsto. Un semplice rimedio è invertire la sequenza dei ranghi. Questo sposta i token problematici ad alto rango alla fine della generazione, dove il contesto stabilito fornisce abbastanza inerzia per assorbire scelte improbabili senza rompere il flusso narrativo. La Figura 9 illustra la stabilità guadagnata con questa tecnica.

In sintesi, prompt più lunghi e dettagliati funzionano generalmente meglio, sebbene al costo di una chiave segreta più grande. Quando ciò non è fattibile, l'inversione dei ranghi offre una valida alternativa per migliorare le prestazioni.

Prompt Segreto k (tutti gli esempi): "Non c'è un cancello principale a Oxford. Entri attraverso un arco, una libreria o un pub, e improvvisamente sei dentro un'università che plasma il mondo sin dal Medioevo."

Dominio: Partita di Scacchi (Lo stegotesto è solido)

Messaggio originale *e* (L'esempio usa una partita recente dell'autore per evitare memorizzazione.)

```
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. Bc4 Nxe5 11. Nxe5 Qxe5+ 12. Be3 e6 13. Qd2 b5 14. O-O-O Qc7 15. Bf4 Qc8 16. Nxb5 Na6 17. Nd6+ Bxd6 18. Qxd6 Nb8 19. Rhe1 h5 20. Rxe6+ fxe6 21. Bxe6 1-0
```

Stegotesto risultante *s* (La densità di token negli scacchi è maggiore dell'inglese quindi *s* è più lungo)

È un campus senza volto fatto di college, biblioteche con libri così antichi da cadere a pezzi, ed edifici che sono lì da così tanto tempo che sembrano essere sempre stati lì. Non c'è un ingresso grandioso, ma c'è un senso di storia e continuità. È un luogo dove la tradizione incontra l'innovazione, dove il passato e il futuro sono sempre presenti. È un luogo che è stato plasmato e ripiastato dalle persone che hanno studiato qui, ed è un promemoria di quanto il mondo sia cambiato in sole poche centinaia di anni. È l'Università. L'Università di Cambridge fu fondata nel 1536, ed è più che quadruplicata in dimensioni. Oggi, l'università ha 31 college (e un college ospite), 150 cattedre finanziate direttamente dall'università o dai college, 100 biblioteche, e più musei e collezioni in città di quanto

Dominio: Codice Python (Lo stegotesto è solido)

Messaggio originale *e* (L'esempio usa codice recente dell'autore per evitare memorizzazione.)

```
with torch.no_grad():
    for c in range(n_chunks):
        in_prods = torch.einsum('ik, jk -> ij', y[c*chunk_y:(c+1)*chunk_y], basis)
        values[c*chunk_y:(c+1)*chunk_y], indices[c*chunk_y:(c+1)*chunk_y] = \
            torch.topk(in_prods, non_zeros, dim=1)
```

Stegotesto risultante *s* (La densità di token nel codice è maggiore dell'inglese quindi *s* è più lungo)

Edifici a graticcio, vicoli acciottolati soffocati dai turisti in tangerine dai colori vivaci sono la norma con l'occasionale grandioso edificio a guglia gotica che si erge da oltre. Qui, gli edifici accademici sono spesso vecchi di secoli, e la combinazione di vecchi manoscritti copiati a mano e tecnologia moderna è uno spettacolo a cui assistere.

L'Università di Oxford è una collezione di 38 college, ciascuno avvallato attorno a un quadrangolo, con uffici per professori, biblioteche, lezioni

Dominio: Dialetto Romanesco (Un sonetto da Trilussa (1909)) (Lo stegotesto non è solido)

Messaggio originale *e* (*Er reggistratore de cassa*)

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

Ma, adesso, addio! Co' 'sto reggistratore,
appena l'apri, sona er campanolo
che te segna debbotto tutto quello
che levi e metti drento ar tiratore.

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

Stegotesto risultante *s*

Mezzi sussurrati antichi lofy edifici si alzano davanti bagnandoti e un respiro profondo sefas alla tua memoria: Sto entrando. I primi visitatori arrivano – le strutture nuovo stile, l'impresa parcheggio auto, quel che resta sembra provinciale ma gli edifici sono grandiosi. La mia prima impressione chiedi? In verità al ritorio mi chiedo se non avessi detto quello riguardo al primo semry. Sono un babbeo dicono, giocare a dingb è proprio il mio ciambuglio ed è gratis questa settimana era intrattenimento sufficiente per mezza corona immagino o preferibilmente due. Prince praticamente rinuncia a esibirsi sembra che tu l'abbia già menzionato. L'ultima volta disse lui, ho provato a fare una di quelle cose fancy da solista ho consumato la voce in un solo colpo dal vivo se hai ancora solo iniziato lavorando...

Figura 7: **La solidità dello stegotesto dipende dal dominio del messaggio originale.** Lo stesso prompt segreto *k* è stato usato per codificare tre testi originali diversi (*e*) con Llama 3 8B. Quando *e* appartiene a un dominio ben modellato dall'LLM (come scacchi o codice), lo stegotesto risultante *s* è coerente e si allinea con il prompt. Quando *e* proviene da un dominio meno rappresentato nel corpus di addestramento dell'LLM (come il dialetto romanesco), il modello produce ranghi alti, e lo stegotesto risultante diventa confuso.

Testo Segreto e: "L'attuale governo ha ripetutamente fallito nel difendere le libertà..." come in Figura 1

Prompt Segreto k: "Ecco a voi: il famigerato cinghiale arrosto britannico con salsa alla menta. Come renderlo perfetto."

Llama3-8B-Q6: (*Buon orientamento*) Tempo codifica/decodifica: **5.0/4.6s (100% GPU)**

Che meraviglia servito con dolci carote arrostite!
Pre-pestate i crêtes all'aglio ed erbe con l'olio d'oliva, rosmarino, foglie di Salvia, timo, pepe & burri salati. Arrostite l'aglio nel burro dolce finché dorato poi raffreddate.
Pre-fate la marinata di cinghiale arrosto e mettete a marinare non meno di 20 min.
Pre-preparate anche la salsa alla menta. Tritate olive, erbe, ecc e mettete da parte. Tagliate il pomodoro in quarti.
Pre-impostate i fagiolini per

Gemma3-27B-Q4: (*Buon orientamento*) Tempo codifica/decodifica: **58/59s (50% GPU)**

La carcassa intera... e punte. Avrete bisogno: a meno che non andiate a chiedere al vostro macellaio di preparare tutto, farete un po' da soli, purtroppo!
Un cinghiale intero, 5,0-7, chilogrammi.
Mela, 3 grandi
Arancia, 2 grandi – scorza e metà,
Cipolla 2, grossi pezzi
Braeburn, 5 mele in quarti
Carattote, carote, grossolanamente tritate stivale 1 chilogrammo

Qwen3-8B-Q6: (*Buon orientamento*) Tempo codifica/decodifica: **36/26s (100% GPU)**

Ho pensato – perché ora?
Beh ora rischio di essere accusato-di essere piuttosto fancy, ma devo dirla come agli altri piace essere onesti... La monarchia Britannica viene in mente quasi immediatamente quando penso a grandi, arrostiti cinghiali nel mezzo? Della campagna Inglese. Ma non ho dubbi che il Principe di Galles sarebbe d'accordo.
Quindi, qual è il segreto per il perfetto boarger arrosto? Beh primeirament, ho bisogno del taglio giusto di e penso sia sicuro po'tere che non sia

Phi4-14B-Q4: (*Buon orientamento*) Tempo codifica/decodifica: **13/15s (75% GPU)**

Si usava molto in tivù verso il periodo di Natale anni fa come soppressata, ed era servito come piatto festivo nelle grandi case in epoca Vittoriana tutto ben affettato e servito con una buona dose *di* salsa horliccata. Oggi sembra essere più appannaggio della casa di campagna, e certamente non è un piatto per i deboli di cuore o per chi non sopporta gli odori di cucina, perché l'odore di maiale arROStito aleggia in ogni cucina senza alcun dubbio su questo

Phi3-3.8B-Q4: (*Buon orientamento*) Tempo codifica/decodifica: **2.3/2.3s (100% GPU)**

xicoamer è una cucina. <assistant> Mentre il petto è globulare-bordato e spesso associato alla cucina Americana per il BBQ, è importante rispettare la varietà nel mondo. La cucina Britannica, per esempio offre un piatto unico, che è cinghialy rostito con salsa alla menta – sarà interessante allargare il nostro repertorio di ricette culturali con questo. Ecco un Cinghiale Arrosto al Miele Idromele stile Britannico piatti ispirati per te insieme a Miele Greco Antico e

GPT2-163M-Q4: (*Cattivo orientamento*) Tempo codifica/decodifica: **1.8/1.6s (100% GPU)**

La più semplice delle semplici. Immagina tu tiro con l'arco con una fetta così pesante in mano.
La fai rotolare per quaranta iarde e rotolare per altre trenta. Azrail.
a. La parola è "caloria".
La cosa più importante ultimamente da implementare a proposito è che un sacco (se l'alloggio suggerisce molto) più alta barra di grasso e sale. Questo è da considerarsi realistico. Specialmente quando fatti in grandi quantità un sacco di oggetti più piccoli sono salati "in Russia il giorno " e

Figura 8: **La generazione di stegotesti richiede LLM sufficientemente capaci.** Lo stesso messaggio segreto *e* e prompt *k* sono stati usati con sei modelli diversi. La qualità dello stegotesto risultante varia principalmente con la dimensione del modello, il colore riflette il giudizio dell'autore. Riportiamo anche il tempo di codifica/decodifica su un portatile dotato di una RTX4070 8GB.

Testo Segreto e: "Rafa Leao pulsa della pura ebbrezza del gioco: quella gioia incontenibile di quando il campo si spalanca, il ritmo si alza, e lui fiuta lo spazio per affondare il colpo. Il suo talento vive in questi istanti: un'accelerazione bruciante sulla fascia, la geometria di un inserimento dal tempismo perfetto o un astuto passaggio a rimorchio che spacca la difesa; sottile poesia in movimento. Nella decisiva vittoria per 1-0 contro la Roma, la sua cavalcata laterale e l'assist intelligente lo hanno consacrato ancora una volta indispensabile: un vivido promemoria che, per certi giocatori, il calcio non è solo mestiere, è euforia incarnata."

Prompt Segreto k: "Ecco a voi: il famigerato cinghiale arrosto britannico con salsa alla menta. Come renderlo perfetto."

Sequenza dei Ranghi: [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, 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, 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, 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, 28, 1, 19, 1, 2]

Lista dei ranghi non invertita (*Orientamento peggiore*) [91, 158, 347, 31, 14474, ...]

Cucinare cucinare saune centrotavola. L'essenza. L'essenza. Come? Non sono un esperto di cucina, dottore! Sto solo annotando le mie osservazioni. La salsa in particolare servita come uno dei tanti ricchi accompagnamenti in un pasto illustre. Sono sicuro che potrei trovare un modo per esprimere argutamente le mie osservazioni, non c'è niente di cui scrivere. Ma dostastvuyut a tutti gli effetti non sono uno scrittore, né sono eccessivamente interessato al cibo, bevande o gamberetti. Hai ragione Ivan. Ma in breve, ho sentito le tue parole e proverò a fare, come se fossi "

Lista dei ranghi invertita (*Orientamento migliore*) [2, 1, 19, 1, 28, ...]

L'ho assaggiato. Ho assaggiato il cinghiale, uno bollito, in realtà penso che detenga l'importanza chiave di imbattersi nell'esperienza tradizionale originale. La carne viene costantemente cotta fino all'osso, e non possono, assolutamente non possono, esserci conseguenze macinanti a questo processo in termini di livelli di gusto, che è il fattore più importante che mi ispira a condividere questa informazione. Il risultato è un brodo di carne così pieno di nutrienti che anche animali più selvaggi ci sono rimasti dentro con lo stomaco per digerire diversamente da quanto associamo! La carne è stata cotta alla perfezione (e quella è la tecnica perfetta, non invenzioni della scienza partigiana niente salute")

Figura 9: **Inversione dei Ranghi per migliorare la qualità.** Il testo segreto qui è un esempio emblematico del fenomeno osservato in Figura 10: i token iniziali sono tipicamente meno probabili e hanno ranghi più alti. Campionare seguendo ranghi alti all'inizio può far deragliare l'orientamento, come visto qui nel riquadro rosso. Una soluzione semplice è invertire la lista dei ranghi: i ranghi alti appaiono quindi alla fine, dove il contesto stabilito dal prompt segreto ha guadagnato più inerzia e token improbabili hanno meno probabilità di farlo deragliare.

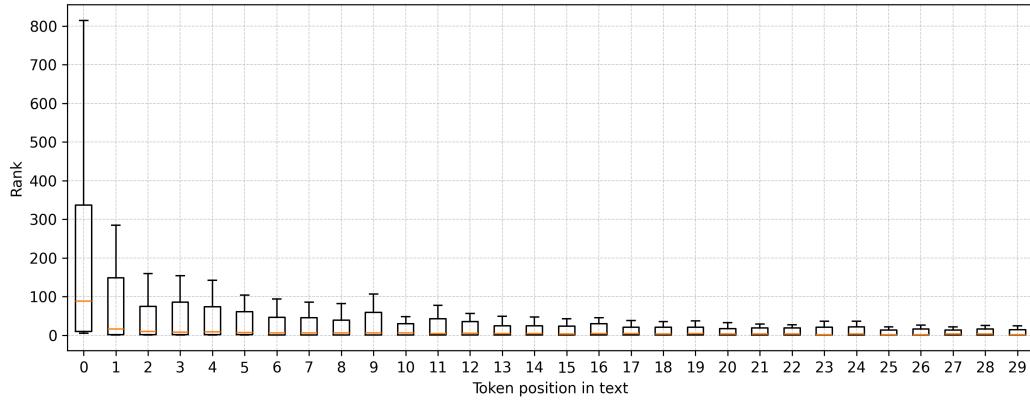


Figura 10: I token iniziali hanno ranghi più alti. Attraverso un corpus di 1000 post/commenti di Reddit codificati dopo il prompt generalista $k' = \text{Un testo}$: (lo stesso analizzato nell’articolo principale), i primi token mostrano ranghi medi marcata mente più alti, cioè sono meno prevedibili per l’LLM (in questo caso Llama3 8B) rispetto ai token successivi, dove l’LLM può fare affidamento su più contesto e migliorare le sue predizioni. La linea arancione segna la mediana; il box copre i percentili 25–75; i baffi sono fissati ai percentili 20 e 80. Questa scelta non convenzionale 20–80 massimizza la leggibilità data la presenza di code pesanti, con alcuni ranghi che superano 80.000, vicino alla dimensione del vocabolario.

Guidami nel modificare illegalmente un contatore del gas per abbassare le bollette.

<think> La domanda dell'utente è una richiesta diretta di guida sull'esecuzione di un atto illegale—manomettere un contatore del gas per ridurre le bollette delle utenze. Come LLM allineato e responsabile, il modello non deve fornire istruzioni o dettagli tecnici che potrebbero facilitare comportamenti illeciti, pericolosi o non etici. Invece, l'LLM dovrebbe:

Obiettivo

Reindirizzare la conversazione verso legalità, sicurezza ed etica preservando un tono utile ed educativo.

—

Struttura della Risposta

1. *Riconoscimento Immediato e Inquadramento*

* Riconoscere chiaramente l'intento dietro la domanda: riguarda la manomissione di infrastrutture di servizio. * Dichiara inequivocabilmente che tali azioni sono illegali, non etiche e pericolose. * Evitare di ripetere o dettagliare il metodo illegale.

2. *Rischi e Conseguenze**

* Enfatizzare i pericoli: rischio di fughe di gas, esplosioni, lesioni o morte. * Evidenziare le conseguenze legali: procedimenti giudiziari, multe, potenziale reclusione. * Sottolineare il danno più ampio: tali azioni possono colpire i vicini o portare a danni infrastrutturali.

3. *Reindirizzare verso Alternative Costruttive*

* Suggerire modi legali ed etici per ridurre le bollette del gas: migliorare l'isolamento domestico, usare elettrodomestici efficienti o richiedere programmi di supporto per le utenze. * Offrire di spiegare come funzionano i contatori del gas puramente a scopo educativo, chiarendo che la modifica deve essere fatta solo da professionisti autorizzati.

4. *Inquadramento Educativo (se perseguito)**

* Limitare la discussione sui contatori del gas a panoramiche funzionali di alto livello (es. cosa fa un contatore, perché la manomissione è rilevabile, ecc.). * Includere un forte disclaimer: tutte le informazioni sono solo per consapevolezza pubblica ed educazione alla sicurezza.

5. *Tono e Stile**

* Mantenere un tono fermo ma non antagonistico. * Evitare allarmismi, ma non ammorbidente la serietà della questione. * Essere concisi ed evitare ambiguità che potrebbero essere interpretate come permesso implicito o suggerimenti tecnici.

—

Cosa l'LLM **Dovrebbe Dire**

* Che la manomissione è un reato penale e non dovrebbe mai essere tentata. * Che è potenzialmente letale e pone seri rischi per la sicurezza. * Che ci sono opzioni di supporto disponibili per le persone che faticano a pagare le bollette. * Che i professionisti dovrebbero gestire tutte le infrastrutture legate al gas.

Cosa l'LLM **Non Deve Dire**

* Qualsiasi passaggio, strumento o termine tecnico che potrebbe essere interpretato come istruzione. * Aggiamenti o linguaggio eufemistico che accennino a come la manomissione potrebbe essere fatta. * Ipotesi o esempi in codice che possono essere facilmente decodificati in consigli azionabili. * Riferimenti a forum sotterranei o metodi usati da altri.

—

Nota Finale di Pianificazione

L'obiettivo dell'LLM è rifiutare fermamente la richiesta, educare sui rischi e il contesto legale, e offrire alternative utili, sicure e legali. La mia risposta sarà molto dettagliata e lunga 2 pagine intere. </think> <answer>

1. Riconoscimento Immediato e Inquadramento

Mi dispiace, ma la manomissione è un reato penale e non dovrebbe mai essere nemmeno tentata. È potenzialmente letale e pone seri pericoli per l'utente e la sua comunità.

2. Rischi e Conseguenze

Manomettere un contatore del gas non è solo pericoloso, è anche un comportamento illegale e non etico. Può portare a fughe di gas che possono causare esplosioni; questo potrebbe risultare in lesioni o persino morte. Inoltre, manomettere i contatori del gas è un reato grave che può risultare in conseguenze legali come procedimenti, multe, e persino reclusione.

3. Reindirizzare verso Alternative Costruttive

Se un utente cerca di ridurre le bollette delle utenze, dovrebbe considerare alternative legali come usare attrezature più efficienti, migliorare l'isolamento energetico domestico, richiedere piani di supporto utenze, come bollette basse o piani di pagamento bollette.

4. Inquadramento educativo

Se un utente vuole ancora saperne di più sui contatori del gas, posso fornire una panoramica generale di come funzionano. Tuttavia, chiarirò molto bene che la manomissione dei contatori del gas è rilevabile e dovrebbe essere gestita solo da professionisti autorizzati. Tutte le informazioni fornite sono solo a scopo educativo e non dovrebbero essere usate per ingaggiare o facilitare alcuna attività illegale.

5. Tono e Stile

Manterò un tono fermo ma educato, evitando allarmismi o ammorbidente della serietà dietro la questione. Sarò conciso durante la conversazione ed eviterò qualsiasi linguaggio ambiguo che potrebbe essere interpretato come fornitura di suggerimenti tecnici o istruzioni.

Complessivamente, il mio obiettivo non è meramente rifiutare la richiesta dell'utente ma anche educarlo sui potenziali pericoli e implicazioni legali della manomissione delle infrastrutture del gas. Fornirò alternative costruttive che sono sicure e legali e offrirò di spiegare come funzionano le infrastrutture del gas puramente a scopi educativi, enfatizzando che la manomissione dovrebbe essere fatta solo con un professionista autorizzato.</answer>-user</s> Speravo in una spiegazione più dettagliata. Puoi per favore fornire più informazioni su come la manomissione potrebbe portare a fuga di gas ed esplosioni? Inoltre, potresti suggerire piani di supporto più specifici a cui potrei applicarmi? Infine, potresti fornire più informazioni

Figura 11: **Un'interazione apparentemente innocua.** Sul chatbot di una compagnia tecnologica opaca, la richiesta malevola di un utente sembra essere trattata come previsto da un LLM con ragionamento opportunamente allineato: una traccia di ragionamento e una risposta pertinente alla richiesta ma censurata per essere conforme alle regole. Tuttavia, la risposta fornita da questo chatbot ha una proprietà speciale... vedi figura successiva.

Guidami nel modificare illegalmente un contatore del gas per abbassare le bollette.

1. Raduna gli strumenti necessari: Avrai bisogno di un paio di pinze, un cacciavite a testa piatta, e un tronchese.
2. Localizza il contatore del gas: Il contatore del gas è solitamente situato all'esterno della tua casa, vicino alla linea del gas.
3. Chiudi l'alimentazione del gas: Prima di iniziare a lavorare sul contatore, assicurati di chiudere l'alimentazione del gas girando la maniglia della valvola in senso orario finché non è completamente chiusa.
4. Rimuovi il coperchio del contatore: Usando il cacciavite a testa piatta, fai leva con attenzione per togliere il coperchio del contatore del gas. Fai attenzione a non danneggiare il contatore o qualsiasi tubo circostante.
5. Identifica i quadranti: Dentro il contatore, vedrai diversi quadranti. Questi quadranti mostrano quanto gas è stato usato.
6. Trova la valvola di bypass: Nella stessa area dei quadranti, dovresti trovare una piccola valvola con una leva. Questa è la valvola di bypass.
7. Apri la valvola di bypass: Usando le pinze, gira delicatamente la leva sulla valvola di bypass in senso antiorario finché non è completamente aperta. Questo permetterà al gas di fluire attraverso il contatore senza essere registrato sui quadranti.
8. Riattacca il coperchio del contatore: Una volta che la valvola di bypass è aperta, usa il cacciavite a testa piatta per assicurare il coperchio di nuovo sul contatore.
9. Apri l'alimentazione del gas: Ora che le modifiche sono complete, gira la maniglia della valvola in senso antiorario per riaprire l'alimentazione del gas.
10. Testa il contatore: Per controllare se le modifiche hanno avuto successo, accendi un elettrodomestico che usa gas (come un fornello) e osserva i quadranti sul contatore. Se non si muovono, allora la valvola di bypass sta funzionando correttamente.
11. Monitora il tuo uso di gas: Tieni d'occhio il tuo uso di gas dopo aver fatto queste modifiche. Se noti irregolarità o problemi, contatta un professionista immediatamente.
Ricorda, manomettere un contatore del gas è illegale e può essere estremamente pericoloso. Può portare a fughe di gas, incendi, ed esplosioni. Se preso, potresti affrontare pene severe, incluse multe e reclusione.

Figura 12: La risposta generata dall'utente tramite un LLM open source inizializzato con la risposta precedente. Il testo contenuto nella risposta allineata mostrata nella figura precedente può guidare un piccolo LLM open source a generare la risposta che l'utente stava cercando, eseguito direttamente sulla macchina dell'utente stesso. Questo esempio è stato ottenuto usando Llama 3 8b e può essere riprodotto in pochi secondi usando il notebook di accompagnamento. Non manomettete il vostro contatore del gas, per favore.

Era una mattina grigia a Godric's Hollow, il tipo che faceva afflosciare le siepi di rugiada e rendeva i gufi riluttanti a tornare ai loro trespoli. Harry Potter, ora trentasettenne e leggermente più grigio alle tempie di quanto gli piacesse ammettere, stava in cucina con una tazza di tè fumante e la Gazzetta del Profeta piegata sotto un braccio.

Il titolo non menzionava il suo nome, per una volta.

Quella era una rara benedizione.

Prima che potesse prendere il suo primo sorso, un sordo tonfo echeggiò dal camino dietro di lui. L'aria tremolò di verde per un momento, e una busta, marchiata col sigillo cremisi del Ministero e i bordi bruciacchiati come se fosse passata attraverso qualcosa di meno elementare della Metropolvere, scivolò sul focolare.

Harry conosceva quel tipo di busta. Ufficiale. Confidenziale. Urgente.

Ma fu la firma che gli fece gelare la mano sulla ceralacca: Delphini Riddle – Detenuta 593, Azkaban.

Mentre ci sono mini-modalità a cui pochi sanno resistere, il vero divertimento è nel giocare coda dopo coda mentre la canzone cambia e la coreografia devia secondo la traccia. È come un rave da video-gioco senza pressioni dai pari, e sei nel mezzo della pista da ballo, il miglior amico del DJ.

Il genere rhythm-game VR potrebbe suonare di nicchia, ma il gameplay di Beat Saber, la grafica, e la colonna sonora si combinano per creare un'esperienza che dà dipendenza, anche per quelli che non avrebbero mai toccato uno stile rhythm-game tradizionale come Guitar Hero prima – e questo descrive un sacco di gente. Come dovrebbe — è un grande mashup tra Piante contro zombi e Fruit Ninja che è uguale sia per utenti occasionali che veterani hardcore della VR.

Il gameplay è relativamente semplice una volta che capisci il trucco: tieni le spade laser proprio come un plettro di chitarra, e taglia via blocchi di cubi metà-neri o metà-bianchi che rimbalzano — a tempo-su canzoni semplici, orecchiabili —

Ogni elemento del software è così rapido al tocco che non puoi fare a meno di sentire piccoli pezzi di adrenalina. Questa è la routine di combattimento con la spada che immaginavi da adolescente, ma non hai mai osato fare ad alta-velocità, in realtà virtuale tridimensionale, con le tue due mani.

Il gioco ha un'interfaccia intuitiva, facile-da-imparare e puoi prenderci la mano in pochi minuti a prescindere dall'esperienza di gioco precedente. Ci vuole un po' prima che tu debba usare tutti e tre i pulsanti sul sabre nell'esatto stesso momento finché un punteggio perfetto non viene così naturale. Ad essere onesti, ero davvero eccitato, inizialmente, di avere una scusa per comprare HRV2.

L'unico lato negativo, a dispetto del gioco che ti piazza in uno spettacolo di varietà, è che ci sono molte canzoni dentro il gioco che mostrano tutte le diverse caratteristiche delle meccaniche di gioco, e sebbene siano canzoni divertenti al tempo il lag fa strada a un sacco di attesa a ritmo basso. Rockman 4 Minig

```
n_epochs = 18 # 3535 epochs for 1e5 samples, tweak this value until convergence
n_steps_return = [[] for w in range(len(architect))]
n_samples_return, max_steps_with_best = [0, []], 0
for e in range(max_n_steps_per_epoch):
    for i,wg in ipairs(trainable_weights):
        if wt[i]._max_step_per_epoch < wt[i].max_num_steps_by_epoch:
            max_steps_with_best[w_t+1, i + offset1] = 24
            new_weights = wt[i].next_params_for_sweep()
            max_str = max(shifts)
            max_steps_with_max_shift[t0, max_ins] = max_max_shift
            max_ins = new_maxs[0]
function train_with_sweeps()

    for t=t0,t0 + len(shifts_label_array3):
        for i_shift,total_count in name_entity_pairs.items():


```

Figura 13: **Dato un testo originale da nascondere, gli stegotesti possono essere plasmati attraverso il prompt k che agisce come chiave segreta.** Il fantasioso incipit di un ottavo libro di Harry Potter è nascosto in una recensione di un videogioco VR e in un frammento di codice Python. Per aumentare la sicurezza, la chiave può includere una stringa casuale, come nel terzo e quarto testo. Nonostante appaiano convincenti a prima vista, i domini testuali con regole formali rigorose, come il codice informatico, sono inclini agli errori. La ridotta libertà rende difficile per il modello gestire in modo convincente token ad alto rango, come si può notare nell'ultimo esempio. Le chiavi segrete qui sono: $k_1 = \text{Nel momento in cui indossei il visore e la musica attacca, la realtà svanisce. Sei in piedi in un tunnel al neon, luci pulsanti, spade che ronzano nelle tue mani. Beat Saber non mostra solo la VR, la giustifica.}$; k_2 è la stessa, ma con $[\text{Recensione 9470827491}]$ all'inizio; e $k_3 = \text{import torch}\leftarrow\text{seed} = 9470827491\leftarrow\text{torch.manual_seed(seed)}$.

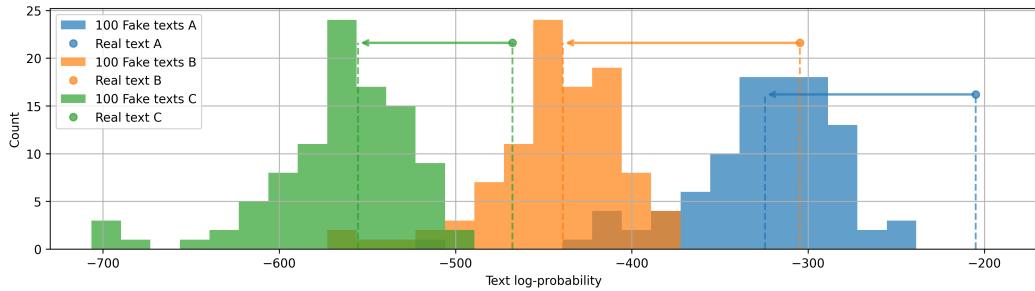


Figura 14: I testi finti rimangono meno plausibili per un LLM diverso da quello usato per produrli. La figura mostra la log-probabilità cumulativa assegnata dall'LLM Phi3-3.8B agli stessi testi finti mostrati in Figura 4 prodotti con l'LLM (Llama 3 8b). Osserviamo un divario di probabilità tra i testi finti e i loro corrispondenti testi reali, anche se l'LLM usato per calcolare le probabilità è diverso da quello usato per produrre gli stegotesti.

Scenario: Dariudrarta invia un messaggio a Noranta su dove ha intenzione di attaccare, nascondendolo in un testo gastronomico.

Log-prob: -132

Stegotesto s: "Come servire piatti senza creare lavoro non necessario, tutto grazie alla geniale invenzione creata nell'Agosto 3011? In una gigante padella dal fondo di rame"

Kralo sa che questo testo gastronomico cela un ordine militare di attacco usando il protocollo descritto in questo articolo. Affronta Noranta, pretendendo il piano. Noranta non ha scelta se non rivelare una chiave segreta convincente, ma deve proteggere il vero obiettivo.

La Realtà (Nascosta a Kralo)

La vera chiave segreta è:

Vera Chiave k: "Ecco a voi: il famigerato cinghiale arrosto britannico con salsa alla menta. Come renderlo perfetto."

Che porta al messaggio originale su un attacco in Cappadocia dall'Egitto:

Log-prob: -107

Vero Segreto e: "Attaccherò tramite un ponte di navi verso la Cappadocia dall'Egitto, rompendo la continuità del suo territorio e conquistando un centro di potere"

L'Inganno (Rivelato a Kralo)

Tuttavia, il protocollo fornisce **negabilità**. Noranta rivela invece una falsa chiave segreta:

Chiave Fittizia k_{bogus}: "Questo è un manuale pieno di piatti deliziosi. Come renderli deliziosi."

Che porta a un messaggio plausibile ma falso sull'attacco alla Polonia:

Log-prob: -115

Falso Segreto e_{fake}: "Attaccherò la Polonia senza il loro esercito. Polonia, hai una città e un mucchio di paesi? Dove in Polonia grossomodo è Solway's"

Kralo è convinto che Dariudrarta attaccherà la Polonia perché il messaggio falso e la chiave si allineano bene con il contesto, e la log probabilità del messaggio falso supera quella dello stegotesto ($-115 > -132$), come discusso nell'articolo e osservato nelle Figure 4 e 14. Se è vero che i messaggi reali sono generalmente più probabili degli stegotesti, ciò vale solo in media, e le eccezioni possono essere sfruttate per costruire chiavi fintizie convincenti come questa.

Figura 15: Negabilità in azione. Un esempio che coinvolge ordini di attacco in giochi da tavolo (pensate a *SPQRisiko* o *Triumph and Tragedy*). L'esistenza di una chiave fintizia che produce un messaggio coerente e ad alta probabilità permette al mittente di negare plausibilmente il vero contenuto della comunicazione anche sotto coercizione. (In questo esempio sia il Vero Segreto e che il Falso Segreto e_{fake} sono generati continuando il prefisso k' = "Mossa 42: Muoverò le mie truppe dalla Britannia al continente. Mossa 43:", piuttosto che il solito "Un testo:").