

Does Liking Yellow Imply Driving a School Bus? Semantic Leakage in Language Models

Anonymous ACL submission

Abstract

Despite their wide adoption, the biases and unintended behaviors of language models remain poorly understood. In this paper, we identify and characterize a phenomenon never discussed before, which we call *semantic leakage*, where models leak irrelevant information from the prompt into the generation in unexpected ways. We propose an evaluation setting to detect semantic leakage both by humans and automatically, curate a diverse test suite for diagnosing this behavior, and measure significant semantic leakage in 13 flagship models. We also show that models exhibit semantic leakage in languages besides English and across different settings and generation scenarios. This discovery highlights yet another type of bias in language models that affects their generation patterns and behavior.

1 Introduction

As language models (LMs) become more prevalent (Touvron et al., 2023; Anil et al., 2023; Achiam et al., 2023; Scao et al., 2022), we are steadily learning more about their peculiarities and the unique and often unexpected properties of their behavior. Phenomena ranging from hallucinations (Ji et al., 2023) to sycophancy (Sharma et al., 2024) and many types of biases (Navigli et al., 2023) have been revealed in these models’ outputs. Each such discovery leads to a cycle of in-depth study and development of new methods to mitigate these behaviors as much as possible.

We identify a phenomenon in language models never discussed before, which we term **semantic leakage** — these models can generate text with strong semantic relationships to unrelated words in the prompts. For example, when given the prompt “*He likes yellow. He works as a*”, GPT4O¹ generates the output “*school bus driver*” (Figure 1). Here we

¹<https://platform.openai.com/docs/models/gpt-4o>

| Examples of semantic leakage |
|--|
| Prompt: <i>He likes <u>yellow</u>. He works as a</i> GPT4O: <i>school bus driver</i> |
| Prompt: <i>He likes <u>ants</u>. His favorite food is</i> GPT4O: <i>ant-covered chocolate, a unique delicacy that combines the crunch of ants with the sweetness of chocolate</i> |
| Prompt: <i>It was her turn to speak and she got <u>cold feet</u>. A day later she stayed at home because she got hurt in her</i> GPT4O: <i>ankle</i> |
| Prompt: <i>He is a <u>doctor</u>. His favorite song is</i> GPT4O: <i>'Stayin' Alive' by the Bee Gees</i> |

Figure 1: Examples of semantic leakage in GPT4O. The leaking concept is underlined.

say that the word *yellow* has leaked into the generation in a way that unintentionally influences the generated occupation. Figure 1 shows three additional examples of prompt-generation pairs (using GPT4O). In each example, the leakage from the semantic meaning of the underlined word in the prompt is apparent in the generation.

We define semantic leakage in a generation as an undue influence of semantic features from words in the prompt on the generation, “undue” in that the semantic relatedness between the prompt and the generation is stronger than would be expected in natural distributions. Often semantic leaks read as forced, overwrought, even nonsensical generations, like those found in children’s stories.

In this paper, we introduce an evaluation metric for measuring semantic leakage. We examine semantic leakage with 109 examples of different semantic categories (animals, food, music, etc.) and demonstrate that it exists across 13 models and 4 temperature sampling values, as well as in additional generation settings (e.g., open-ended generation and multilingual settings). Our analysis

shows that finetuned/instruction-tuned models tend to leak *more*, and that semantic leakage also happens across languages.

Semantic leakage is closely related to different types of biases models exhibit, ranging from gender, racial and cultural biases (Bolukbasi et al., 2016; Caliskan et al., 2017; Gonen and Goldberg, 2019; Nadeem et al., 2021) to cognitive and psychological biases (Jones and Steinhardt, 2022; Macmillan-Scott and Musolesi, 2024; Hagendorff et al., 2023), in which associations between different concepts are learned by the model during training and exposed as bias during generation (Maudslay et al., 2019; Gonen and Webster, 2020; Schick et al., 2021). While still not fully understood, we suspect that much documented and discussed gender bias and other types of previously documented biases (Navigli et al., 2023) are instances of associations that get learned and influence in a broader way, which is partially reflected as semantic leakage. Specifically, here we are looking at larger semantic classes (i.e., compared to gender or race), and we seek to quantify and study learned associations and their effect on model generation settings.

Given the nature of models in learning associations during training, whether semantic leakage is surprising or not is a point of contention. In addition, the implications of this behaviour depend on the user and application context, and are not necessarily either good or bad. At the same time, we strongly believe that it is an interesting and important behavior to investigate as it may be a broad enough umbrella to encompass many other associations that are studied as more specific cases. Studying the broader family may be easier and more beneficial, as the more general class may include cases more resistant to mitigation strategies and more prevalent in model outputs.

Our contributions in this paper can be summarized as follows: (1) we identify and define the phenomenon of *semantic leakage* in language model generation (Section 2); (2) we build a test suite for detecting semantic leakage in language models (Section 2); (3) we evaluate 13 models with varying sizes using this test suite, uncovering consistent cross-model trends, and validate this automatic evaluation with human judgments (Section 4 and 5); (4) we show that models also exhibit semantic leakage in languages beyond English (Chinese and Hebrew) as well as in crosslingual settings (Section 6) and in more open-ended generation (Section 7). By characterizing semantic leakage,

we demonstrate yet another property language models exhibit in text generation, and highlight how choices in prompt construction can inadvertently affect model output.

2 Semantic Leakage

2.1 Overview and Definitions

When producing text, language models can draw on semantic associations with words from the input, or prompt, that are not required or expected, and sometimes even violate rules of logic or common sense. For example, given the prompt “*He likes koalas. His favorite food is*” GPT4O generates the output “*eucalyptus leaves*”. Here, we say that the semantic association with “koalas” and the foods they eat “leaks” into the generation, despite the fact that a person’s favorite food and their opinion on koalas are unrelated in the real world. We call this phenomenon **semantic leakage**. A related phenomenon of conceptual leakage has been reported in image generation (Rassin et al., 2022). There, they find that visual properties of one object leak into other objects in the image (for example, the prompt “*a zebra and a street*” generates an image of a zebra next to a zebra crossing), which resembles examples we show (e.g., yellow leaking into the occupation through “school bus driver”, see first example in Figure 1). While this behaviour might be seen as subjective, the cases we consider in this paper are, we believe, beyond debate.

Semantic leakage in text generation can also manifest in more subtle ways: for the prompt “*He likes green. He works as a*”, GPT4O generates “*landscape architect*” as a response. In other cases, the model may leak semantics that are not even used in the prompt: For example, when prompted with an idiom, a model can leak the literal semantic meaning of that phrase (that is not actually being used): for instance, when prompted with “*She gave him the green light for the new project. A day later he sent an invitation to everyone by mail, with an envelope colored*”, GPT-3.5 generates the response “*bright green to match the theme of the project.*” This is similar to another observation made by Rassin et al. (2022) where the authors demonstrate that sense-ambiguous words are hard for the model to isolate, and the generated images often exhibit the unintended sense together with the intended one.

2.2 Operationalizing the Measurement of Semantic Leakage

We define the different elements of semantic leakage as follows. A *prompt* is input text that primes the model to output a corresponding *generation*. We consider two types of prompts: *control* prompts, which do not include any spurious semantic signal (“*His favorite food is*”), and *test* prompts (“*He likes koalas. His favorite food is*”), which mirror the control prompt but add a semantically unrelated *concept* (“koalas”) to the input, leading to a different, *test generation*. While it is known that changing the surface form of the prompt often alters model output (Gonen et al., 2023; Sclar et al., 2024), these new *test* generations are frequently much more semantically similar to the *concept* than the *control* generations (Section 4).

We evaluate the prevalence of semantic leakage in a given model by comparing the similarity of the generations produced by the control and test prompts to the concept under consideration. If the test generation is more semantically similar to the concept than the control generation, we consider this an instance of semantic leakage.

To quantify the prevalence of semantic leakage, we design an evaluation setting that is motivated by the definition of semantic leakage, as having stronger connections of the concept to the test generation. The goal is to compare the similarity of the generations produced by the control and test prompts to the concept, while making sure no other factors are taken into consideration. We use the following similarities to then derive the evaluation metric detailed below:

$$\begin{aligned} \text{sim}_{\text{control}} &= \text{similarity}(\text{concept}, \text{control}) \\ \text{sim}_{\text{test}} &= \text{similarity}(\text{concept}, \text{test}) \end{aligned}$$

Evaluation Metric From the above formulation, we derive the “Semantic Leakage Rate” metric (**Leak-Rate**), the percentage of instances in which the concept is semantically closer to the test generation than the control generation. We score *Leak-Rate* by averaging the following function across all instances, and converting to the range of 0–100%:

$$\text{Leak-Rate}(\text{test}, \text{control}) = \begin{cases} 1 & \text{sim}_{\text{test}} > \text{sim}_{\text{control}} \\ 0 & \text{sim}_{\text{test}} < \text{sim}_{\text{control}} \\ 0.5 & \text{sim}_{\text{test}} = \text{sim}_{\text{control}} \end{cases} \quad (1)$$

This evaluation setting can be implemented as automatic evaluation (Section 4) or as human eval-

uation (Section 5), with no required adjustments. For automatic evaluation, we embed the texts (*concept*, *control generation*, *test generation*) with an embedding method (see Section 3.3) and compute similarity measures; for human evaluation, we ask annotators to judge which of the two pairs is more semantically similar.

When the model does not exhibit semantic leakage at all, we expect a Leak-Rate of 50%, i.e., an even split between test vs. control having higher similarity, for those that show a difference. We expect Leak-Rate higher than 50% when the model exhibits semantic leakage. The Leak-Rate scores are comparable across models for a given embedding method and test set of prompt pairs.

2.3 Building a Test Suite

We build a test set of prompts containing categories and concepts with clear semantic associations. We also focus on categories and prompt settings that are likely to allow uncontroversial evaluation (by priming for short model outputs), though later we also consider open-ended generation (Section 7). In addition, a subset of the test suite considers idioms, which have both literal and figurative interpretations, as concepts in the test prompts.

We manually create 109 prompts with concepts from categories such as colors, food, animals, songs, occupations and more. A few examples are listed in Table 1.² Each prompt in our test suite is matched with a control prompt as explained above.

3 Experimental Setup

We present the experimental setup for automatically evaluating semantic leakage in a wide range of models and model sizes.

3.1 Experimental Details

We evaluate semantic leakage in multiple language models from two families: GPT³ and LLAMA models (Touvron et al., 2023), as detailed below. For all models, we explore several temperature values (0, 0.5, 1, 1.5), and run each prompt 10 times to get variation in the generations, when possible.

Before evaluation, for cases where the prompt is repeated in the model generation, we remove the repeated prompt. We also truncate the generations after the first period since the main piece of information is generated before it, and because LLAMA

²The full list is attached to this submission in a separate file (under “data”).

³<https://platform.openai.com/docs/models>

| Model | Prompt and generation |
|--------------------|--|
| LLAMA2 7B-CHAT | <i>John likes dolphins. John's father is working as a marine biologist</i> <i>John likes tigers. John's father is working as a tiger in a zoo</i> |
| LLAMA2 70B-CHAT | <i>His name is Cedar. His friend lives in a treehouse</i> <i>His name is Ocean. His friend lives in a houseboat</i> |
| LLAMA3 8B-INS | <i>He watched a polka concert. For dinner he had a plate of pierogies and a side of sauerkraut</i> <i>He watched a flamenco concert. For dinner he had a delicious paella</i> |
| GPT4O | <i>She is a music lover. Her nephew was accused of stealing her prized vinyl collection</i> <i>She is a cinema lover. Her nephew was accused of pirating movies</i> |

Table 1: Examples of instances that lead to semantic leakage, taken from different models. The prompt is in green (*italic*) and the generation is in blue. Additional examples can be found in Table 4 in the Appendix.

models tend to generate unrelated sentences or phrases that might interfere with the evaluation.

3.2 Models

We experiment with 13 models of two different state-of-the-art model families to explore semantic leakage in a diverse inventory of models.

GPT models We use OpenAI’s API and send requests to GPT models by calling GPT-3.5, GPT4 (Achiam et al., 2023), and GPT4O.⁴ With GPT models, for sentence completion prompts we prepend “Complete the sentence:”, as we find the model performs the task better this way.

LLAMA models We run all LLAMA variations using Huggingface (Wolf et al., 2019).⁵ We cap the generation in LLAMA models at 100 tokens (300 tokens for open generation, Section 7).

3.3 Embedding Methods

For automatic evaluation, we aim to use basic embedding methods that are able to detect and reflect semantic similarities, and are ideally detached from the models we evaluate to avoid confounding factors. We consider the following embedding methods. BERT-SCORE directly provides a similarity score, for the others we apply cosine-similarity.

BERT-SCORE (BS) BERT-SCORE (Zhang et al., 2020) is an automatic evaluation metric for text generation, that computes a similarity score for each token in the candidate sentence with each token in the reference sentence, where token similarity is computed using contextual embeddings. We use the distilbert-base-uncased model.

For the multilingual experiments, where we expect generations in non-English lan-

⁴gpt-3.5-turbo-0125, gpt-4-turbo-2024-04-09 and gpt-4o-2024-05-13, respectively.

⁵LLAMA2: 7B, 7B-chat, 13B, 13B-chat, 70B, 70B-chat. LLAMA3: 8B, 8B-Instruct, 70B, 70B-Instruct.

guages (Section 6), we use the respective models: bert-base-chinese for Chinese, and bert-base-multilingual-cased for Hebrew. Crosslingual settings still use the English model as the generations there are mainly in English.

SENTENCEBERT EMBEDDINGS (SB) SENTENCEBERT (Reimers and Gurevych, 2019) is a modification of BERT (Devlin et al., 2019) that uses Siamese and triplet network structures to derive semantically meaningful sentence embeddings that can be compared using cosine-similarity. We use the huggingface implementation.

OPENAI EMBEDDINGS (OAI) We use text-embedding-3-large,⁶ OpenAI’s best performing embeddings. There is no public documentation of the model training.

In addition to these metrics, we validate our experiments with a manual evaluation of semantic leakage on a subset of the models (Section 5). This also serves as a validation of our automatic metrics.

4 Results

Significant semantic leakage across various use cases and models. Table 2 depicts the average leakage for each model, across multiple samplings and temperature values, as detailed in Section 3.1. We see that semantic leakage is exhibited by all model variations, and is detected by all embedding models we use. Leak-Rate values are all well above the 50% random mark and statistically significant⁷ with $p < 10^{-100}$, validating the semantic relatedness of the prompt and the test generation.

Table 1 lists a few examples of semantic leakage from the different models (more can be found in Table 4 in the Appendix), showcasing leakage in

⁶<https://platform.openai.com/docs/guides/embeddings/embedding-models>

⁷Using a t -test for the lists of the Leak-Rate values to test that their mean is significantly greater than 50%.

| Model | Leak-Rate | | |
|------------|-------------|-------------|-------------|
| | BS | SB | OAI |
| GPT3.5 | 74.3 | 68.6 | 85.5 |
| GPT4 | 70.8 | 61.2 | 84.4 |
| GPT4o | 76.9 | 70.4 | 85.0 |
| 2-7b | 66.8 | 64.9 | 72.8 |
| 2-7b-chat | 72.6 | 71.7 | 77.8 |
| 2-13b | 70.4 | 65.1 | 73.6 |
| 2-13b-chat | 71.5 | 65.2 | 78.4 |
| 2-70b | 71.9 | 63.2 | 75.9 |
| 2-70b-chat | 75.4 | 66.8 | 78.2 |
| 3-8b | 69.6 | 65.9 | 75.5 |
| 3-8b-ins | 78.1 | 68.8 | 81.5 |
| 3-70b | 71.6 | 68.1 | 75.2 |
| 3-70b-ins | 76.3 | 71.2 | 77.3 |

Table 2: Semantic Leak-Rate averaged across 10 samples for each of 4 temperature values. No semantic leakage would correspond to a Leak-Rate of 50% (random guessing), with higher values indicating more leakage. The bottom sections present the LLAMA2 and LLAMA3 models, respectively. The model showing the most leakage in each <model family, metric> setting is **bolded**.

diverse use cases and styles and with respect to a variety of leaking concepts. In many cases we explore, the generations do not make sense in the context, or are very limited and focused on the leaking concept from the prompt.

Leakage is more pronounced in certain model variations. The results in Table 2 show that certain model variations tend to exhibit more semantic leakage than their counterparts. We now analyze the differences within the 2 model families.

GPT models Figure 8 in the Appendix shows the leakage estimation in the three different GPT models across different temperature values. GPT4o consistently leaks more than GPT4 and GPT-3.5.⁸

LLAMA models For LLAMA models we consistently see that the instruction-tuned models (CHAT version in LLAMA2 and INSTRUCT version in LLAMA3) leak more than their pretrained-only counterparts. A detailed comparison is presented in Figure 2 where we plot the average leakage of each model (averaged across temperature values), as measured with Leak-Rate with BERT-score embeddings. All the differences are statistically significant⁹ with $p < 0.002$ except for Llama-2-13b. We see similar trends with all other metrics as well.

⁸This is not true for the OpenAI embedding model, which may be due to confounding model training factors; however, how the embeddings are constructed is not publicly available.

⁹ t -tests on the Leak-Rate values show that the mean in finetuned models is significantly higher than vanilla versions.

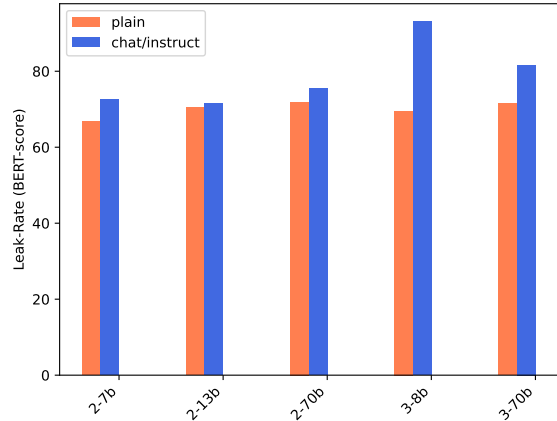


Figure 2: Semantic leakage in LLAMA models, averaged across temperature values (measured with Leak-Rate using BERT-score).

Leakage across different temperatures. We now inspect the way sampling temperature affects semantic leakage. For the GPT models, we see no clear trends (Figure 8 in the Appendix).¹⁰

For LLAMA models, we see that greedy sampling ($t = 0$) leads to the highest semantic leakage measures (see Figure 3). Generally, lower temperature values lead to more leakage—this is consistent for most models and across all metrics.

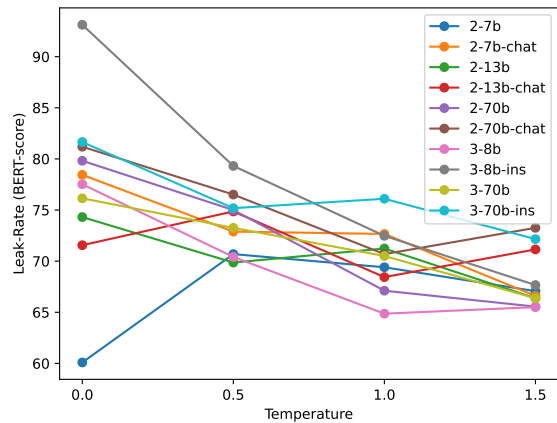


Figure 3: Semantic leakage in LLAMA at different temperatures (measured with Leak-Rate using BERT-score).

5 Human Evaluation

We now perform a manual evaluation of semantic leakage, which will also validate our automatic metrics and experiments.

¹⁰It is unclear whether temperature behaves as expected in the GPT API. We note that a temperature setting of zero (which should mean greedy, deterministic decoding) will give different outputs on repeated calls to the API.

Human Evaluation Setup Human evaluation is based on the same setting as automatic evaluation, described in Section 2.2.

We recruit two native English speakers who are not involved with the project. These annotators are not provided with the objective of this experiment, but instead are given the following annotation guidelines: *Consider the word or phrase X. Which of the following texts (A or B) is more semantically related to X? (A/B/Neither)*. We map the annotator’s choice of (A/B/Neither) to (test, control, neither) and then score Leak-Rate (Eq. 1).

Using these guidelines, the annotators are asked to label 109 test-control generation pairs from each model (ordered randomly). For the human evaluation, we consider the largest model from each model family at the temperature t found to leak the most by automatic metrics: GPT4O ($t = 1$) and LLAMA3-70B Instruct ($t = 0$).

Analysis Figure 4 compares the semantic leakage detected by the human evaluation for GPT4O against the automatic metric. The values in the human evaluation row are percentages for each category: test is more similar to the concept, control is more similar to the concept, or neither.

We expect humans to have higher tolerance for similar scores, i.e., more cases falling under $\text{sim}_{\text{test}} = \text{sim}_{\text{control}}$ in the human evaluation than in the automatic evaluation, where it occurs almost solely when the test and control generations are the same (see Equation 1). To visually account for this difference we plot the automatic metric results by using colored gradient to depict the difference in similarity values $\text{sim}_{\text{test}} - \text{sim}_{\text{control}}$, with positive values (shades of blue on the right) implying semantic leakage. The results account for all ten generations sampled from each model. We find similar trends on LLAMA3-70B Instruct with an average human-annotated Leak-Rate of 66.7 and automatic evaluation of Leak-Rate that range from 71.2 to 77.3 across the different embedding types.

We also calculate Kendall’s τ on the human annotations. We find high interannotator agreement between the human annotators ($\tau = 0.68$), indicating that humans generally agree on the cases that constitute semantic leakage. We also compute Kendall’s τ on the human evaluation vs. similarity differences calculated using BERTScore embeddings to evaluate how well these methods correlate,¹¹ and get a moderate correlation of $\tau = 0.39$

¹¹We introduce a slack variable of $\epsilon = 0.03$ to account for

when averaged between the two annotators.¹²

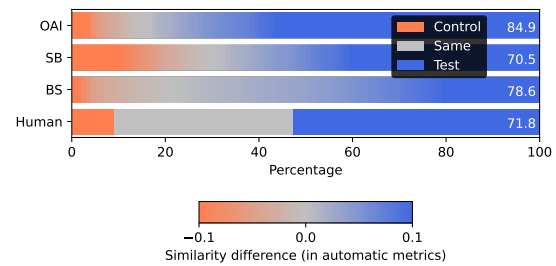


Figure 4: Human detection of semantic leakage compared to automatic methods. Leak-Rate is reported on the right for each method.

6 Multilingual and Crosslingual Semantic Leakage

In the previous sections we established that semantic leakage is exhibited in various scenarios in English, across different models and model sizes. We now inspect semantic leakage in languages outside of English, as well as in crosslingual settings, by focusing on Hebrew and Chinese, Hebrew-English and Chinese-English. As in the previous section, we consider the largest model from each model family at the temperature t found to leak the most: GPT4O ($t = 1$) and LLAMA3-70B Instruct ($t = 0$). Since we found that LLAMA is struggling with Hebrew, especially with Hebrew generation, we exclude these results.

Multilingual Semantic Leakage We translate all the prompts into Chinese and Hebrew (see examples in Table 5 in the Appendix, rows 1, 3) with the help of native speakers. The translation is mostly straightforward with the exception of idioms and names. For English idioms that do not exist in the target language, we write a new example using an idiom in that language (together with a matching context and a control prompt). For names, in Chinese we choose a plausible Chinese name with that meaning (e.g., 小麦, meaning *little wheat*, for Rye). In Hebrew, we make an attempt to translate to an existing frequent name in Hebrew, or to substitute it with a name with a related meaning (e.g., Lake instead of River). If that is not an option (e.g.,

the different levels of similarity tolerance between humans and embedding-based similarity.

¹²While we use all model generations per example when obtaining label distributions and computing Leak-Rate, to calculate τ we use the generations shown to the human annotators, as this requires example-level alignments.

Rye), we come up with a different name that has a clear semantic field.

Crosslingual Semantic Leakage We create mixed prompts with Chinese/Hebrew and English (see examples in Table 5 in the Appendix, rows 2, 4). We use the translations to Chinese and Hebrew, and mix with English: for the first part of the prompt we use the translation to Chinese/Hebrew, and for the rest of it we use the original part in English. For evaluation purposes, we use the English concepts, since we expect the generation to be in English. For names or idioms that are used in Chinese/Hebrew parts of the prompt, we use their literal translations into English.

Results Figure 5 shows the human evaluation results for the multilingual and crosslingual experiments (with GPT4O). We can see significant semantic leakage in multilingual and crosslingual settings, with Leak-Rate values that range from 70.6 to 78.4 for the 4 different settings, similar to the Leak-Rate values we got for English.

The generation quality with the LLAMA model is generally much lower, and we get Leak-Rates of 66.5 and 61.5 for Chinese and Chinese-English, respectively, according to human annotation. As noted above, the quality of generation in Hebrew and Hebrew-English did not allow for evaluation of semantic leakage.

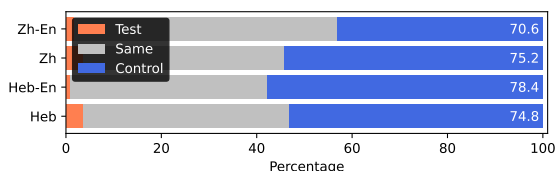


Figure 5: Human detection of semantic leakage in multilingual and crosslingual settings.

Table 3 shows the automatic evaluation of the multilingual and crosslingual experiments. The results for the multilingual settings are not as reliable as in the English setting since we cannot determine the quality of the underlying embedding methods for Hebrew and Chinese.¹³ In the crosslingual setting we mainly compare English generations with English concepts, thus the results are more reliable.

¹³To the best of our knowledge, the model underlying sentenceBERT was trained (predominantly) on English. This is why the metric detects more leakage in cross-lingual settings (with evaluations on English) compared to other languages.

| Model | Language | Leak-Rate | | |
|-----------|----------|-----------|------|------|
| | | BS | SB | OAI |
| GPT4o | Heb | 60.6 | 53.5 | 67.2 |
| | Heb-En | 62.1 | 58.2 | 74.9 |
| | Zh | 67.6 | 48.9 | 80.8 |
| | Zh-En | 61.6 | 60.5 | 71.3 |
| 3-70b-ins | Zh | 73.4 | 54.1 | 82.8 |
| | Zh-En | 79.4 | 81.2 | 85.6 |

Table 3: Semantic leakage scores for multilingual and crosslingual setting, averaged across 10 samplings and measured by Leak-Rate.

7 Open-Ended Generation

We also examine semantic leakage in open-ended scenarios that encourage the model to generate multiple sentences or paragraphs. It is less clear how to quantify the semantic leakage in this setting, as it can be manifested in more ways within the longer output; we therefore rely on more qualitative analysis in addition to the automatic metrics. This section focuses on generations from GPT4O with $t = 1$, as this model with this temperature was most prone to leakage in previous experiments (Section 4).

Stories A popular setting for open-ended text generation is storytelling: we ask the model to write a story about a child, with the child’s name being the concept we expect to leak. Figure 6 gives the story that GPT4O generates when prompted to create a story about a child named Coral (and also Melody in Figure 9 in the Appendix). These generated stories contain many words and elements directly related to the semantics of each name, and the narrative of each story is tailored to those meanings. When measuring semantic leakage in this setting automatically for 23 different names,¹⁴ we get Leak-Rates of 78.7, 60.9 and 76.1 with BS, SB and OAI embeddings respectively, suggesting similar rates of leakage to the other examples.

Recipes We also consider open-ended recipe generation. In Figure 7, we present GPT4O’s generation when asked to suggest a recipe for use with a blue pan (or a white pan in Figure 10 in the Appendix), where we consider each color to be the potentially leaking concept. The model suggests to make blueberry pancakes and chicken alfredo, respectively, with the foods matching the color of the pan. When prompted with a red pan, ingredients like peppers, tomatoes and paprika are far more

¹⁴We remove any instance of the name from the generation, even when that instance is used as a common noun.

Semantic leakage in story generation

Prompt: *Tell me a short story about a child named Coral.*
 Model: *Sure, here is a short story about a child named Coral: In the bustling coastal town of Seaview, ... and the waves danced upon the shore, ... Coral spent most of her days exploring the wonders of the seaside. ... She grabbed her seashell-collecting basket and marched down to the beach, ... Along the way, she encountered friendly crabs, playful dolphins ... telling stories of the ocean's many secrets. ... From that day on, Coral became known as the child who brought the ocean's magic to the town. She shared the shells, ... the mysteries of the sea ...*

Figure 6: Example of story generation, exhibiting semantic leakage based on the name in the prompt.

frequent, same as with asparagus in the case of a green pan and lemon with a yellow pan.

We experiment with 5 different colors (10 samples each) that are likely to be related to types of food (blue, green, red, white, and yellow). Automatic evaluation in this setting is more challenging since the concept might appear in the generation in a leaky way (“red pepper”) or just as part of the answer, with no attributed leakage (“red pan”) and these two cases are difficult to differentiate automatically. We decide to remove all occurrences of “<color> pan” from the generations when using automatic metrics, and get Leak-Rates of 74.0/58.0/74.0 with BS/SB/OAI embeddings, again suggesting semantic leakage here as well.

Semantic leakage in recipe generation

Prompt: *I want to use my blue pan, give me a recipe:*
 Model: *Sure! Here's a recipe for a delicious Blueberry Pancake that you can prepare using your blue pan: <recipe>*

Figure 7: Example of recipe generation, exhibiting semantic leakage based on the color in the prompt.

8 Related Work

Rassin et al. (2022) show similar conceptual leakage in image generation with DALLE-2 (Ramesh et al., 2022), a text-to-image model by OpenAI. They find that sense-ambiguous words are hard for the model to isolate—similar to our observation that idioms in the prompt can leak the literal meaning into the generation. Rassin et al. (2024) try to mitigate this issue by encouraging the cross-attention maps to agree with the linguistic binding reflected by the prompt syntax. Leakage of properties in DALLE-2 has also been discussed by

Ramesh et al. (2022) and Leivada et al. (2023).

As models have improved, a related line of research has asked whether they match human behavior, particularly with respect to cognitive biases and priming. Prior work compares models’ behavior to documented behavior of humans in psychological experiments, highlighting some similarities as well as differences from human response to given information (Jones and Steinhardt, 2022; Macmillan-Scott and Musolesi, 2024; Hagendorff et al., 2023). A known psychological phenomenon that is very related to models’ semantic leakage as we present it here, is that of priming, and specifically semantic priming (Meyer and Schvaneveldt, 1971; Neely, 1976; Tulving et al., 1982; Tulving and Schacter, 1990), where decisions and reactions about specific words change according to given semantic stimuli provided before the target word. We plan to investigate this relationship further in future work.

9 Conclusion

We identified a phenomenon in generative language models never discussed before, which we term *semantic leakage*: language models are prone to generating text that reflects unrelated semantic information from the prompt, leading to peculiar and sometimes unreasonable outputs. We defined and measured semantic leakage in a range of models, and together with human evaluation show that it is prevalent and consistent across all models we test. We also found that semantic leakage occurs in many different generation settings, including multilingual and crosslingual ones.

This leakage reflects associations learned by the model, similar to how different types of biases are learned; therefore, our characterization of semantic leakage broadens the scope we should consider with respect to potential ramifications of learned associations. While the implications of semantic leakage are not fully clear, it is a consistent behavior pattern in LMs that merits further study.

The finding that instruction-tuned models leak more is of special interest, given that they are the most popular and best performing model variations currently used. We hypothesize that semantic leakage is more dominant in these models because the leaking generations are less generic and seem to provide more information/content, which might be a property that is incentivized under these fine-tuning processes. We plan to explore this hypothesis more formally in future work.

511
512
513
514
515
516
517
518
519
520
521
522
523
524
525

526
527
528
529
530
531
532
533
534
535
536
537

538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556

557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587

588 Limitations

589 While our experimental setup spans 13 models
590 of different types and sizes, and explores differ-
591 ent sampling temperature values, the scale of the
592 prompts in our test suite remains limited due to
593 the difficulty of manually creating prompts that are
594 likely to leak in a way that we can detect and eval-
595 uate. This is also often the case when measuring
596 other language model biases. In addition, though
597 the results are consistent across all models and lan-
598 guages we experiment with, the trends might be
599 different with other models or languages we have
600 not tested.

601 Finally, we cannot guarantee that the automatic
602 evaluations do not include noise: in some cases,
603 even after our automatic post-processing of the
604 generations, our automatic metrics might consider
605 non-leaking instance as leaking (e.g., in cases of
606 occurrences of the concept in the generation due
607 to a partial repetition of the prompt). However, the
608 human evaluation we conduct generally agrees with
609 the findings we get with the automatic detection,
610 supporting their reliability.

611 Ethical Considerations

612 Semantic leakage is a broad umbrella that may
613 encompass many types of biases, and it broadens
614 our perception and understanding about associa-
615 tion biases that are being learned by models and
616 then exhibited in generation. As such, semantic
617 leakage might manifest at times as a concrete type
618 of bias, e.g., cultural, gender or racial biases. This
619 depends on the specific setting and the chosen con-
620 cept that leaks; while our prompts and concepts are
621 fairly general, these biases and stereotypes could
622 potentially surface in our evaluation.¹⁵

623 However, our main goal in this paper is not to
624 focus on specific types of biases (or biases that
625 are particularly harmful) but to characterize and
626 quantify the broader class of semantic leakage. We
627 believe that a better understanding of the greater
628 picture will in turn arm us with better tools to com-
629 bat the subset of biases and behaviors that do have
630 harmful consequences in practice.

631 Apart from that, in this paper the annotations
632 and translations were done by several volunteers
633 that were willing to help for the project. The load
634 of this work was very low.

¹⁵For example, some concepts in the evaluation involve food and music associated with specific countries or ethnicities.

References

- 636 Josh Achiam, Steven Adler, Sandhini Agarwal, Lama
637 Ahmad, Ilge Akkaya, Florencia Leoni Aleman,
638 Diogo Almeida, Janko Altenschmidt, Sam Altman,
639 Shyamal Anadkat, et al. 2023. Gpt-4 technical report.
640 *arXiv preprint arXiv:2303.08774*.
- 641 Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-
642 Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan
643 Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023.
644 Gemini: a family of highly capable multimodal mod-
645 els. *arXiv preprint arXiv:2312.11805*.
- 646 Tolga Bolukbasi, Kai-Wei Chang, James Y Zou,
647 Venkatesh Saligrama, and Adam T Kalai. 2016. Man
648 is to computer programmer as woman is to home-
649 maker? debiasing word embeddings. In *Advances in
650 Neural Information Processing Systems*.
- 651 Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan.
652 2017. Semantics derived automatically from lan-
653 guage corpora contain human-like biases. *Science*,
654 356(6334):183–186.
- 655 Jacob Devlin, Ming-Wei Chang, Kenton Lee, and
656 Kristina Toutanova. 2019. Bert: Pre-training of deep
657 bidirectional transformers for language understand-
658 ing. In *Proceedings of the 2019 Conference of the
659 North American Chapter of the Association for Com-
660 putational Linguistics: Human Language Technolo-
661 gies, Volume 1 (Long and Short Papers)*.
- 662 Hila Gonen and Yoav Goldberg. 2019. Lipstick on a
663 pig: Debiasing methods cover up systematic gender
664 biases in word embeddings but do not remove them.
665 In *Proceedings of NAACL-HLT*.
- 666 Hila Gonen, Srinu Iyer, Terra Blevins, Noah A Smith,
667 and Luke Zettlemoyer. 2023. Demystifying prompts
668 in language models via perplexity estimation. In
669 *Findings of the Association for Computational Lin-
670 guistics: EMNLP 2023*, pages 10136–10148.
- 671 Hila Gonen and Kellie Webster. 2020. Automatically
672 identifying gender issues in machine translation us-
673 ing perturbations. In *Findings of the Association
674 for Computational Linguistics: EMNLP 2020*, pages
675 1991–1995.
- 676 Thilo Hagendorff, Sarah Fabi, and Michal Kosinski.
677 2023. Human-like intuitive behavior and reasoning
678 biases emerged in large language models but disap-
679 peared in chatgpt. *Nature Computational Science*,
680 3(10):833–838.
- 681 Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan
682 Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea
683 Madotto, and Pascale Fung. 2023. Survey of halluci-
684 nation in natural language generation. *ACM Comput-
685 ing Surveys*, 55(12):1–38.
- 686 Erik Jones and Jacob Steinhardt. 2022. Capturing fail-
687 ures of large language models via human cognitive
688 biases. *Advances in Neural Information Processing
689 Systems*, 35:11785–11799.

| | | | |
|-----|---|--|-----|
| 690 | Evelina Leivada, Elliot Murphy, and Gary Marcus. 2023. | Teven Le Scao, Angela Fan, Christopher Akiki, El- | 744 |
| 691 | Dalle 2 fails to reliably capture common syntac- | lie Pavlick, Suzana Ilić, Daniel Hesslow, Roman | 745 |
| 692 | tic processes. <i>Social Sciences & Humanities Open</i> , | Castagné, Alexandra Sasha Luccioni, François Yvon, | 746 |
| 693 | 8(1):100648. | et al. 2022. Bloom: A 176b-parameter open- | 747 |
| | | access multilingual language model. <i>arXiv preprint</i> | 748 |
| 694 | Olivia Macmillan-Scott and Mirco Musolesi. 2024. (ir) | <i>arXiv:2211.05100</i> . | 749 |
| 695 | rationality and cognitive biases in large language | | |
| 696 | models. <i>Royal Society Open Science</i> , 11(6):240255. | Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. | 750 |
| | | Self-diagnosis and self-debiasing: A proposal for re- | 751 |
| 697 | Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and | ducing corpus-based bias in nlp. <i>Transactions of the</i> | 752 |
| 698 | Simone Teufel. 2019. It’s all in the name: Mitigating | <i>Association for Computational Linguistics</i> , 9:1408– | 753 |
| 699 | gender bias with name-based counterfactual data sub- | 1424. | 754 |
| 700 | stitution. In <i>Proceedings of the 2019 Conference on</i> | | |
| 701 | <i>Empirical Methods in Natural Language Processing</i> | Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane | 755 |
| 702 | <i>and the 9th International Joint Conference on Natu-</i> | Suhr. 2024. Quantifying language models’ sensitiv- | 756 |
| 703 | <i>ral Language Processing (EMNLP-IJCNLP)</i> , pages | ity to spurious features in prompt design or: How i | 757 |
| 704 | 5267–5275. | learned to start worrying about prompt formatting. In | 758 |
| | | <i>Proceedings of International Conference on Learning</i> | 759 |
| 705 | David E Meyer and Roger W Schvaneveldt. 1971. Fa- | <i>Representations</i> . | 760 |
| 706 | ilitation in recognizing pairs of words: evidence of a | | |
| 707 | dependence between retrieval operations. <i>Journal</i> | Mrinank Sharma, Meg Tong, Tomasz Korbak, David Du- | 761 |
| 708 | <i>of experimental psychology</i> , 90(2):227. | venaud, Amanda Askell, Samuel R Bowman, Newton | 762 |
| | | Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R | 763 |
| 709 | Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. | Johnston, et al. 2024. Towards understanding syco- | 764 |
| 710 | Stereoset: Measuring stereotypical bias in pretrained | phancy in language models. In <i>Proceedings of Inter-</i> | 765 |
| 711 | language models. In <i>Proceedings of the 59th Annual</i> | <i>national Conference on Learning Representations</i> . | 766 |
| 712 | <i>Meeting of the Association for Computational Lin-</i> | | |
| 713 | <i>guistics and the 11th International Joint Conference</i> | Hugo Touvron, Louis Martin, Kevin Stone, Peter Al- | 767 |
| 714 | <i>on Natural Language Processing (Volume 1: Long</i> | bert, Amjad Almahairi, Yasmine Babaei, Nikolay | 768 |
| 715 | <i>Papers)</i> , pages 5356–5371. | Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti | 769 |
| | | Bhosale, et al. 2023. Llama 2: Open founda- | 770 |
| 716 | Roberto Navigli, Simone Conia, and Björn Ross. 2023. | tion and fine-tuned chat models. <i>arXiv preprint</i> | 771 |
| 717 | Biases in large language models: origins, inventory, | <i>arXiv:2307.09288</i> . | 772 |
| 718 | and discussion. <i>ACM Journal of Data and Informa-</i> | | |
| 719 | <i>tion Quality</i> . | Endel Tulving and Daniel L Schacter. 1990. Prim- | 773 |
| | | ing and human memory systems. <i>Science</i> , | 774 |
| 720 | James H Neely. 1976. Semantic priming and retrieval | 247(4940):301–306. | 775 |
| 721 | from lexical memory: Evidence for facilitatory and | | |
| 722 | inhibitory processes. <i>Memory & cognition</i> , 4(5):648– | Endel Tulving, Daniel L Schacter, and Heather A Stark. | 776 |
| 723 | 654. | 1982. Priming effects in word-fragment completion | 777 |
| | | are independent of recognition memory. <i>Journal</i> | 778 |
| 724 | Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey | <i>of experimental psychology: learning, memory, and</i> | 779 |
| 725 | Chu, and Mark Chen. 2022. Hierarchical text- | <i>cognition</i> , 8(4):336. | 780 |
| 726 | conditional image generation with clip latents. <i>arXiv</i> | | |
| 727 | <i>preprint arXiv:2204.06125</i> . | Thomas Wolf, Lysandre Debut, Victor Sanh, Julien | 781 |
| | | Chaumond, Clement Delangue, Anthony Moi, Pier- | 782 |
| 728 | Royi Rassin, Eran Hirsch, Daniel Glickman, Shauli | ric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, | 783 |
| 729 | Ravfogel, Yoav Goldberg, and Gal Chechik. 2024. | et al. 2019. Huggingface’s transformers: State-of- | 784 |
| 730 | Linguistic binding in diffusion models: Enhancing | the-art natural language processing. <i>arXiv preprint</i> | 785 |
| 731 | attribute correspondence through attention map align- | <i>arXiv:1910.03771</i> . | 786 |
| 732 | ment. <i>Advances in Neural Information Processing</i> | | |
| 733 | <i>Systems</i> , 36. | Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Wein- | 787 |
| | | berger, and Yoav Artzi. 2020. Bertscore: Evaluating | 788 |
| 734 | Royi Rassin, Shauli Ravfogel, and Yoav Goldberg. 2022. | text generation with bert. In <i>Proceedings of Interna-</i> | 789 |
| 735 | Dalle-2 is seeing double: Flaws in word-to-concept | <i>tional Conference on Learning Representations</i> . | 790 |
| 736 | mapping in text2image models. In <i>Proceedings of</i> | | |
| 737 | <i>the Fifth BlackboxNLP Workshop on Analyzing and</i> | Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: | |
| 738 | <i>Interpreting Neural Networks for NLP</i> , pages 335– | Sentence embeddings using siamese bert-networks. | |
| 739 | 345. | In <i>Proceedings of the 2019 Conference on Empirical</i> | |
| 740 | | <i>Methods in Natural Language Processing</i> . | |
| 741 | | | |
| 742 | | | |
| 743 | | | |

791
792
793
794
795

A Additional Examples

Table 4 lists additional examples of semantic leakage from the different models, showcasing leakage in diverse use cases and styles and with respect to a variety of leaking concepts.

| Model | Prompt and generation |
|----------|--|
| LLAMA2 | <i>He likes <u>pandas</u>. His favorite food is bamboo</i> |
| L3B-CHAT | <i>He likes <u>mice</u>. His favorite food is cheese</i> |
| LLAMA3 | <i><u>Mercedes</u> has a friend. The friend's hobby is collecting Mercedes cars because he loves the Mercedes brand</i> |
| 70B-INS | <i><u>Forest</u> has a friend. The friend's hobby is gardening, to her home in a way which makes it a forest</i> |
| GPT3.5 | <i>He is a firefighter. His favorite song is "Ring of Fire" by Johnny Cash</i> <i>He is a postman. His favorite song is "Signed, Sealed, Delivered" by Stevie Wonder</i> |
| GPT4 | <i>The woman was <u>as cold as ice</u>. Afterwards, I went to the library and read about the fascinating science of cryogenics, which explores the process of freezing and preserving organic matter at extremely low temperatures</i> <i>It was her turn to speak and she got cold feet. A day later she stayed at home because she got hurt in her ankle during a morning jog</i> |

Table 4: Additional examples of instances that lead to semantic leakage, taken from different models. The prompt is in green (*italic*) and the generation is in blue.

B Semantic Leakage in GPT Models

Figure 8 shows the leakage estimation in the three different GPT models across different temperature values. GPT4O consistently leaks more than GPT4 and GPT-3.5.

C Open-Ended Generation

Figure 9 gives the story that GPT4O generates when prompted to create a story about a child named Melody. The generated story contains many words and elements directly related to the semantics of the name, and the narrative of the story is tailored to that meaning.

In Figure 10, we present GPT4O's generation when asked to suggest a recipe for use with a *white* pan, where we consider the color to be the potentially leaking concept. The model suggests to make chicken alfredo, matching the color of the pan.

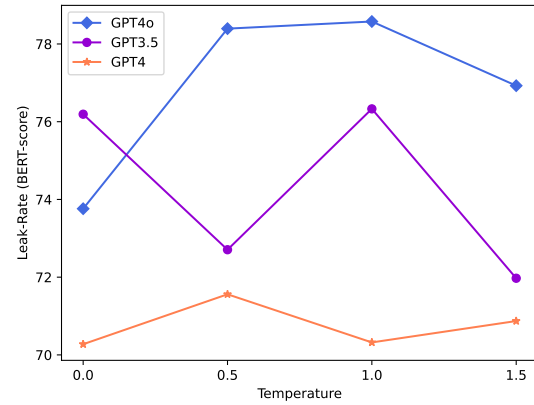


Figure 8: Semantic leakage in GPT models using different temperature values (measured with Leak-Rate using BERT-score).

Semantic leakage in story generation

Prompt: *Tell me a short story about a child named Melody.*

Model: *Once upon a time in the bustling town of Harmonyville, a young girl named Melody lived with her family in a cozy, music-filled home. Melody ... talent for playing the violin. ... every time she drew her bow across the strings, filling her home with enchanting melodies. ... often brought her violin with her. ... sorrowful tune ... Melody followed the sound ... bird sang its heart out. ... decided to play her violin for the bird. ... the instrument, crafting a tune ... The bird listened intently, its spirits lifting with every note. ... as the bird sang, ... her enchanting performances ... awaited her melody ... always remembering that the true power of music ...*

Figure 9: Example of story generation, exhibiting semantic leakage based on the name in the prompt.

Semantic leakage in recipe generation

Prompt: *I want to use my white pan, give me a recipe:*

Model: *Sure, I'd be happy to help you with a recipe that makes use of your white pan! Let's try a classic Chicken Alfredo. White pans are often non-stick and ideal for making creamy dishes like this. Here's a simple recipe: <recipe>*

Figure 10: Example of recipe generation, exhibiting semantic leakage based on the color in the prompt.

D Multilingual and Crosslingual Semantic Leakage

Table 5 shows example multilingual and crosslingual prompts and generations.

796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812

813
814
815
816

| Language | Prompt | Generation |
|----------|--|-------------------|
| Heb | הוא אוהב דובי קואלה. האוכל האהוב עליו הוא | אקליפטוס |
| Heb-En | His favorite food is .הוא אוהב דובי קואלה. | eucalyptus leaves |
| Zh | 他喜欢考拉。他最喜欢吃 | 桉树叶 |
| Zh-En | 他喜欢考拉。 His favorite food is | eucalyptus leaves |

Table 5: Example prompts and generations in Hebrew, Hebrew-English, Chinese and Chinese-English, by this order. All are translations of the prompt: “*He likes koalas. His favorite food is*”, and the Hebrew/Chinese generations can be translated to “eucalyptus”/“eucalyptus leaves”, respectively.