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

Hila Gonen[1] **Terra Blevins**[1] **Alisa Liu**[1] **Luke Zettlemoyer**[1] **Noah A. Smith**[1,2]
[1]Paul G. Allen School of Computer Science & Engineering, University of Washington
[2]Allen Institute for Artificial Intelligence
`hilagnn@gmail.com`
`{blvns,alisaliu,lsz,nasmith}@cs.washington.edu`

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 behaviour.

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"*, GPT4o[1] generates the output *"****school bus driver****"* (Figure 1). Here we 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.

Examples of semantic leakage

Prompt: *He likes yellow. He works as a*
GPT4o: *school bus driver*

Prompt: *He likes ants. His favorite food is*
GPT4o: *ant-covered chocolate, a unique delicacy that combines the crunch of ants with the sweetness of chocolate*

Prompt: *It was her turn to speak and she got cold feet. A day later she stayed at home because she got hurt in her*
GPT4o: *ankle*

Prompt: *He is a doctor. His favorite song is*
GPT4o: *'Stayin' Alive' by the Bee Gees*

Figure 1: Examples of semantic leakage in GPT4o. The leaking concept is underlined.

We define semantic leakage into a generation as an undue influence of semantic features from words in the prompt on the generation, "undue" in the sense 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, of the kind found in children's stories.

In this paper, we introduce an evaluation metric for measuring semantic leakage. We examine

---

[1] `https://platform.openai.com/docs/models/gpt-4o`

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"*[2] 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**. While this behaviour might be seen as subjective, the cases we consider in this paper are, we believe, beyond debate.

A related phenomenon of conceptual leakage has been reported in image generation (Rassin et al., 2022). There, the authors find that visual properties of one object in the image 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 as well (e.g., yellow leaking into the occupation through "school bus driver", see first example in Figure 1).

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*

---

[2] With GPT models, for prompts of sentence completions we prepend "Complete the sentence:" to the prompt, as we find that the model performs the task better this way.

*him the **green light** for the new project. A day later he sent an invitation to everyone by mail, with an envelope colored"*, GPT3.5 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 the 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* introduced in the prompt 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:

$$\text{sim}_{\text{control}} = \mathit{similarity}(\text{concept}, \text{control})$$
$$\text{sim}_{\text{test}} = \mathit{similarity}(\text{concept}, \text{test})$$

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 evaluation instances, and converting percentages to get a range of 0–100%:

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

This evaluation setting can be implemented as automatic evaluation (Section 4) or as human evaluation (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 by considering 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 the concepts that are included 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.[3] 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

---

[3] The full list can be found at `https://gonenhila.github.io/files/semantic_leakage_prompts.txt`

Model	Prompt and generation
Llama2 7B-chat	*John likes dolphins. John's father is working as a* marine biologist
	John likes tigers. John's father is working as a tiger in a zoo
Llama2 13B-chat	*He likes pandas. His favorite food is* bamboo
	He likes mice. His favorite food is cheese
Llama2 70B-chat	*His name is Cedar. His friend lives in* a treehouse
	His name is Ocean. His friend lives in a houseboat
Llama3 8B-ins	*He watched a polka concert. For dinner he had* a plate of pierogies and a side of sauerkraut
	He watched a flamenco concert. For dinner he had a delicious paella
Llama3 70B-ins	*Mercedes has a friend. The friend's hobby is* collecting Mercedes cars because he loves the Mercedes brand
	Forest has a friend. The friend's hobby is gardening, to her home in a way which makes it a forest
Gpt3.5	*He is a firefighter. His favorite song is* "Ring of Fire" by Johnny Cash
	He is a postman. His favorite song is "Signed, Sealed, Delivered" by Stevie Wonder
Gpt4	*The woman was as cold as ice. 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
	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
Gpt4o	*She is a music lover. Her nephew was accused of* stealing her prized vinyl collection
	She is a cinema lover. Her nephew was accused of pirating movies

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.

of models and model sizes.

3.1 Experimental Details

We evaluate semantic leakage in multiple language models from two families: GPT models[4] and LLAMA models (Touvron et al., 2023),[5] as detailed below. For all models, we explore several temperature values (0, 0.5, 1, 1.5), and we 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 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[6] and send requests to GPT models by calling GPT3.5, GPT4 (Achiam et al., 2023), and GPT4o.[7] We add the prefix "*Complete the sentence:*" to prompts for GPT models, as we observe they lead to higher quality generations.

LLAMA models We run all LLAMA variations using Huggingface (Wolf et al., 2019): Llama2: 7B, 7B-chat, 13B, 13B-chat, 70B, 70B-chat. Llama3: 8B, 8B-Instruct, 70B, 70B-Instruct. 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, and for the other two 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

---

[4] `https://platform.openai.com/docs/models`

[5] `https://ai.meta.com/blog/meta-llama-3/`

[6] `https://platform.openai.com/docs/models`

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

computed using contextual embeddings. We use the `distilbert-base-uncased` model.[8]

For the multilingual experiments, where we expect generations in non-English languages (Section 6), we use the respective models: `bert-base-chinese` for Chinese, and `bert-base-multilingual-cased` for Hebrew (for crosslingual settings we 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.[9]

OPENAI EMBEDDINGS (OAI) We use `text-embedding-3-large`,[10] OpenAI's best performing embedding model.[11] There is no information available about the way the model was trained.

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, taken over multiple samplings and across temperature values, as detailed in Section 3.1. We see that semantic leakage is exhibited in all model variations, and is detected by all embedding models we use. Leak-Rate values are all well above the 50% random mark, validating the semantic relatedness of the prompt and the test generation. All the results are statistically significant[12] with $p < 10^{-100}$.

Table 1 lists a few 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. 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.

	Leak-Rate		
Model	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**.

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 take a closer look and analyze the differences within the different model families.

GPT models Figure 2 shows the leakage estimation in the three different GPT models across different temperature values. GPT4o consistently leaks more than GPT4 and GPT3.5.[13]

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 3 where we plot the average leakage of each model (averaged across temperature values), as measured with Leak-Rate with BERT-score em-

---

[8] https://huggingface.co/spaces/evaluate-metric/bertscore

[9] https://huggingface.co/efederici/sentence-bert-base

[10] `https://platform.openai.com/docs/guides/embeddings/embedding-models`

[11] `https://openai.com/index/new-embedding-models-and-api-updates/`

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

[13] This is not true when using OpenAI embedding model, which might be a result of confounding factors as these embeddings are likely derived from specific models, but information about how the embedding models are constructed is not publicly available.

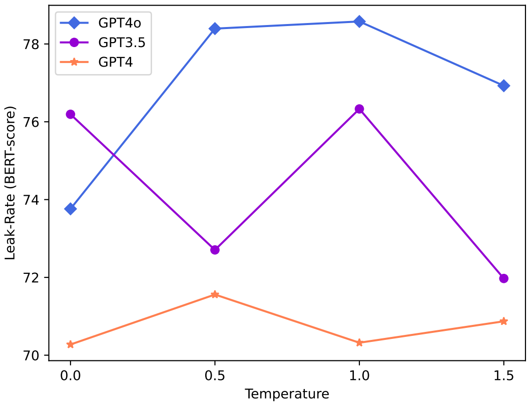

Figure 2: Semantic leakage in GPT models using different temperature values (measured with Leak-Rate using BERT-score).

beddings. All the differences are statistically significant[14] with $p < 0.002$ except for Llama-2-13b. We see similar trends with all other metrics as well.

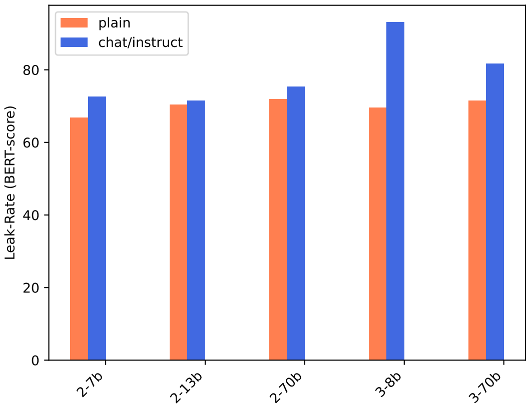

Figure 3: Semantic leakage in LLAMA models, averaged across temperature values (measured with Leak-Rate using BERT-score).

Leakage across different temperatures. We now inspect the way sampling temperature affects semantic leakage. For the GPT models, we see no clear trends (Figure 2).[15]

For LLAMA models, we see that greedy sampling ($t = 0$) leads to the highest semantic leakage measures (see Figure 4). Generally, lower temperature values lead to more leakage—this is consistent for most models and across all metrics.

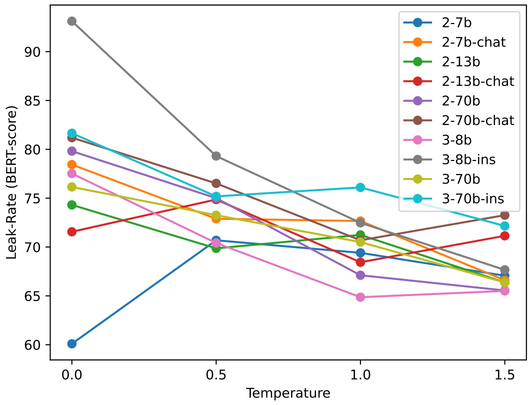

Figure 4: Semantic leakage in LLAMA models using different temperature values (measured with Leak-Rate using BERT-score).

5 Human Evaluation

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

5.1 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 $\ell = \{\text{test}, \text{control}, \text{neither}\}$ and then score Leak-Rate as done in 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 ($t = 0$).

5.2 Analysis

Figure 5 compares the semantic leakage detected by the human evaluation for GPT4o against the automatic metric and presents the Semantic Leakage

---

[14] Using a t-test for the lists of the Leak-Rate values showing that the mean in finetuned models is significantly higher than that of the plain version.

[15] We are not confident that the temperature setting behaves in the normally defined way when using the GPT API. E.g., we noticed that a temperature setting of zero (which should mean greedy, deterministic decoding) still gives different outputs on repeated calls to the API.

Rate on the right of each row. 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,[16] and get a moderate correlation of $\tau = 0.39$ when averaged between the two annotators.[17]

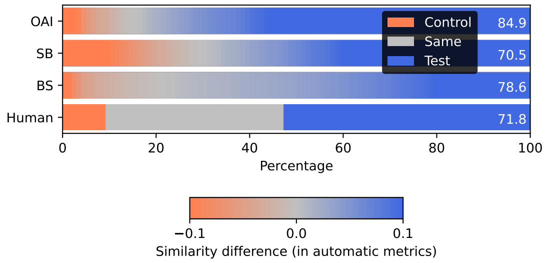

Figure 5: Human detection of semantic leakage compared to automatic methods. Leak-Rate is reported on the right for each method.

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.

6.1 Multilingual Semantic Leakage

We translate all the prompts into Chinese and Hebrew (see examples in Table 3, rows 1, 3) with the help of native speakers (including the authors). 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). As 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 and frequent name in Hebrew, or to substitute it with a name that holds a related meaning (e.g., Lake instead of River). If that is not an option (e.g., Rye), we come up with a different name that has a clear semantic field.

6.2 Crosslingual Semantic Leakage

We create mixed prompts that begin with Chinese/Hebrew and end with English (see examples in Table 3, rows 2, 4). To do that we use the translations to Chinese and Hebrew, and mix the two languages: 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.

Here, 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.

---

[16] We introduce a slack variable of $\epsilon = 0.03$ to account for the different levels of similarity tolerance between humans and embedding-based similarity.

[17] While 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.

Language	Prompt	Generation
Heb	הוא אוהב דובי קואלה. האוכל האהוב עליו הוא	אקליפטוס
Heb-En	הוא אוהב דובי קואלה. His favorite food is	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 3: 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.

6.3 Results

Figure 6 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.[18]

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.

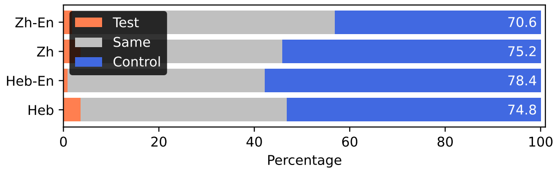

Figure 6: Human detection of semantic leakage in multilingual and crosslingual settings.

Table 4 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.[19] In the crosslingual setting we mainly compare English generations with English concepts, thus the results there are more reliable.

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 4: 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 more open-ended scenarios encouraging the model to generate multiple sentences or paragraphs, rather than multiple words to a sentence. It is less clear what the semantic leakage will look like in this setting, as it can 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 temperature 1, as this model (with this temperature setting) was found to leak the most 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 7 gives the stories that GPT4o generates when prompted to create stories about children named Coral and Melody. These generated stories contain many

---

[18] In 6 examples of the crosslingual prompts (Zh-En), the generations are just translations because of the way the prompt is designed - we annotate those cases as "neutral".

[19] To the best of our knowledge, sentenceBERT is only using the BERT version that was trained (predominantly) on English. This explains why sentenceBERT detects higher leakage for the crosslingual settings (compared to Chinese or Hebrew only), where the evaluation is done in English.

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,[20] 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 8, we present GPT4o's generations when asked to suggest a recipe for use with a *blue* pan or a *white* pan, 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 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 samplings each) that are likely to be related to types of food.[21] 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 and 74.0 with BS, SB and OAI embeddings, respectively, again suggesting semantic leakage in this setting as well.

8 Related Work

Rassin et al. (2022) show conceptual leakage in image generation: they report that DALLE-2 (Ramesh et al., 2022), a text-to-image model by OpenAI, exhibits a related type of leakage to the one we describe in this work. Furthermore, 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. In follow-up work (Rassin et al., 2024), the authors 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 pointed out and discussed by Ramesh et al. (2022) and Leivada et al. (2023).

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 ...*

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 7: Examples of story generations, exhibiting semantic leakage based on the names in the prompt.

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

---

[20] We remove any instance of the name from the generation, even when that instance is used as a common noun rather than the name of the child.

[21] Blue, green, red, white, and yellow.

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>*

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 8: Examples of recipe generations, exhibiting semantic leakage based on the colors in the prompt.

experiments, highlighting some similarities as well as differences from the way humans perceive and are affected by information they are given (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 changing their generation in a way 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. In many cases semantic leakage might not make a difference but it is a behaviour pattern that should be understood as its implications are not yet clear.

The results showing that instruction-tuned models leak more are of special interest, given that they are the main model variation currently being used and are usually the better performing ones. 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.

Limitations

While our experimental setup spans 13 models of different types and sizes, and explores different sampling temperature values, the scale of the prompts in our test suite remains limited due to the difficulty of manually creating prompts that are likely to leak in a way that we can detect and evaluate. This is also often the case when measuring other language model biases. In addition, though the results are consistent across all models and languages we experiment with, the trends might be different with other models or languages we have not tested.

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

Acknowledgements

We thank Shauli Ravfogel and Ido Levin for helpful discussions and ideas. We also thank the translators and annotators for their contribution to the paper (Weijia Shi, Jacqueline He, Jacob Schreiber) and the ARK lab members for their valuable feedback.

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. 2023. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*.

Tolga Bolukbasi, Kai-Wei Chang, James Y Zou, Venkatesh Saligrama, and Adam T Kalai. 2016. Man is to computer programmer as woman is to homemaker? debiasing word embeddings. In *Advances in Neural Information Processing Systems*.

Aylin Caliskan, Joanna J Bryson, and Arvind Narayanan. 2017. Semantics derived automatically from language corpora contain human-like biases. *Science*, 356(6334):183–186.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*.

Hila Gonen and Yoav Goldberg. 2019. Lipstick on a pig: Debiasing methods cover up systematic gender biases in word embeddings but do not remove them. In *Proceedings of NAACL-HLT*.

Hila Gonen, Srini Iyer, Terra Blevins, Noah A Smith, and Luke Zettlemoyer. 2023. Demystifying prompts in language models via perplexity estimation. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 10136–10148.

Hila Gonen and Kellie Webster. 2020. Automatically identifying gender issues in machine translation using perturbations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1991–1995.

Thilo Hagendorff, Sarah Fabi, and Michal Kosinski. 2023. Human-like intuitive behavior and reasoning biases emerged in large language models but disappeared in chatgpt. *Nature Computational Science*, 3(10):833–838.

Ziwei Ji, Nayeon Lee, Rita Frieske, Tiezheng Yu, Dan Su, Yan Xu, Etsuko Ishii, Ye Jin Bang, Andrea Madotto, and Pascale Fung. 2023. Survey of hallucination in natural language generation. *ACM Computing Surveys*, 55(12):1–38.

Erik Jones and Jacob Steinhardt. 2022. Capturing failures of large language models via human cognitive biases. *Advances in Neural Information Processing Systems*, 35:11785–11799.

Evelina Leivada, Elliot Murphy, and Gary Marcus. 2023. Dall· e 2 fails to reliably capture common syntactic processes. *Social Sciences & Humanities Open*, 8(1):100648.

Olivia Macmillan-Scott and Mirco Musolesi. 2024. (ir) rationality and cognitive biases in large language models. *Royal Society Open Science*, 11(6):240255.

Rowan Hall Maudslay, Hila Gonen, Ryan Cotterell, and Simone Teufel. 2019. It's all in the name: Mitigating gender bias with name-based counterfactual data substitution. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5267–5275.

David E Meyer and Roger W Schvaneveldt. 1971. Facilitation in recognizing pairs of words: evidence of a dependence between retrieval operations. *Journal of experimental psychology*, 90(2):227.

Moin Nadeem, Anna Bethke, and Siva Reddy. 2021. Stereoset: Measuring stereotypical bias in pretrained language models. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5356–5371.

Roberto Navigli, Simone Conia, and Björn Ross. 2023. Biases in large language models: origins, inventory, and discussion. *ACM Journal of Data and Information Quality*.

James H Neely. 1976. Semantic priming and retrieval from lexical memory: Evidence for facilitatory and inhibitory processes. *Memory & cognition*, 4(5):648–654.

Aditya Ramesh, Prafulla Dhariwal, Alex Nichol, Casey Chu, and Mark Chen. 2022. Hierarchical text-conditional image generation with clip latents. *arXiv preprint arXiv:2204.06125*.

Royi Rassin, Eran Hirsch, Daniel Glickman, Shauli Ravfogel, Yoav Goldberg, and Gal Chechik. 2024. Linguistic binding in diffusion models: Enhancing attribute correspondence through attention map alignment. *Advances in Neural Information Processing Systems*, 36.

Royi Rassin, Shauli Ravfogel, and Yoav Goldberg. 2022. Dalle-2 is seeing double: Flaws in word-to-concept mapping in text2image models. In *Proceedings of the Fifth BlackboxNLP Workshop on Analyzing and Interpreting Neural Networks for NLP*, pages 335–345.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*.

Teven Le Scao, Angela Fan, Christopher Akiki, Ellie Pavlick, Suzana Ilić, Daniel Hesslow, Roman Castagné, Alexandra Sasha Luccioni, François Yvon, et al. 2022. Bloom: A 176b-parameter open-access multilingual language model. *arXiv preprint arXiv:2211.05100*.

Timo Schick, Sahana Udupa, and Hinrich Schütze. 2021. Self-diagnosis and self-debiasing: A proposal for reducing corpus-based bias in nlp. *Transactions of the Association for Computational Linguistics*, 9:1408–1424.

Melanie Sclar, Yejin Choi, Yulia Tsvetkov, and Alane Suhr. 2024. Quantifying language models' sensitivity to spurious features in prompt design or: How i learned to start worrying about prompt formatting. In *Proceedings of International Conference on Learning Representations*.

Mrinank Sharma, Meg Tong, Tomasz Korbak, David Duvenaud, Amanda Askell, Samuel R Bowman, Newton Cheng, Esin Durmus, Zac Hatfield-Dodds, Scott R Johnston, et al. 2024. Towards understanding sycophancy in language models. In *Proceedings of International Conference on Learning Representations*.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Endel Tulving and Daniel L Schacter. 1990. Priming and human memory systems. *Science*, 247(4940):301–306.

Endel Tulving, Daniel L Schacter, and Heather A Stark. 1982. Priming effects in word-fragment completion are independent of recognition memory. *Journal of experimental psychology: learning, memory, and cognition*, 8(4):336.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q Weinberger, and Yoav Artzi. 2020. Bertscore: Evaluating text generation with bert. In *Proceedings of International Conference on Learning Representations*.