Follow the Flow: On Information Flow Across Textual Tokens in Text-to-Image Models

Anonymous ACL submission

Abstract

Text-to-image (T2I) models generate images by encoding text prompts into token representations, which then guide the diffusion process. While prior work has largely focused on improving alignment by refining the diffusion process, we focus on the *textual encoding* stage. Specifically, we investigate how semantic information is distributed across token representations within and between *lexical items* (i.e., words or expressions conveying a single concept) in the prompt. We analyze information flow at two levels: (1) *in-item representation*—whether individual tokens represent their lexical item, and (2) *cross-item interaction*—whether information flows across the tokens of *different* lexical items. We use patching techniques to uncover surprising encoding patterns. We find information is usually concentrated in only one or two of the item's tokens—For example, in the item "San Francisco's Golden Gate Bridge", the token "Gate" sufficiently captures the entire expression while the other tokens could effectively be discarded. Lexical items also tend to remain isolated; for instance, the token "dog" encodes no visual information about "green" in the prompt "a green dog". However, in some cases, items do influence each other's representation, often leading to misinterpretations—e.g., in the prompt "a pool by a table", the token `pool` represents a *pool table* after contextualization. Our findings highlight the critical role of token-level encoding in image generation, suggesting that misalignment issues may originate already during the textual encoding.

1 Introduction

Text-to-image (T2I) models typically consist of two main components: a text encoder and a diffusion model (Ho et al., 2020; Song and Ermon, 2019). The former processes the user's prompt, transforming it into a representation that guides the latter in generating the image. Though widely used, T2I models often exhibit prompt-image misalignment, where generated images fail to capture key concepts from the user's prompt (Chefer et al., 2023a; Rassin et al., 2022; Huang et al., 2023a). Prior work has attempted to address these issues by modifying the diffusion stage, and particularly the cross-attention mechanism (Rassin et al., 2023; Chefer et al., 2023a; Dahary et al., 2024), under the implicit assumption that each textual token reliably encodes the item it is intended to convey. This raises two fundamental questions: First, is the information for an intended concept distributed evenly across the concept tokens after *textual encoding*, or concentrated in just a few tokens? Second, does each token exclusively encode that concept?

In this work, we examine this assumption and study how visual information is distributed across tokens during the textual encoding stage. We focus our analysis on *lexical items*—words or phrases that convey a single concept. We trace how item information is distributed both *within* the tokens of a single item (*in-item*), and *across* tokens of different items (*cross-item*) (see Fig. 1 for examples of the different cases). Using a causal intervention framework (Toker et al., 2025), we systematically assess what information is encoded in each contextual token representation at the encoder's output, using prompts from widely used T2I benchmarks as our evaluation setting (Section 2.2).[1]

For *in-item* representation (Section 3), we find that a lexical item's meaning is typically concentrated in one or two *representative tokens*. Surprisingly, ablating the non-representative tokens reduces generation error by 21% in FLUX (black-forest labs, 2024). In contrast, when no representative tokens exist for a particular item, it is often omitted entirely from the image—an instance of *item negligence* (Chang et al., 2024a; Chefer et al.,

[1] Throughout this work, we study token representations at the output of the text encoder. For brevity, we sometimes omit the word "representation".

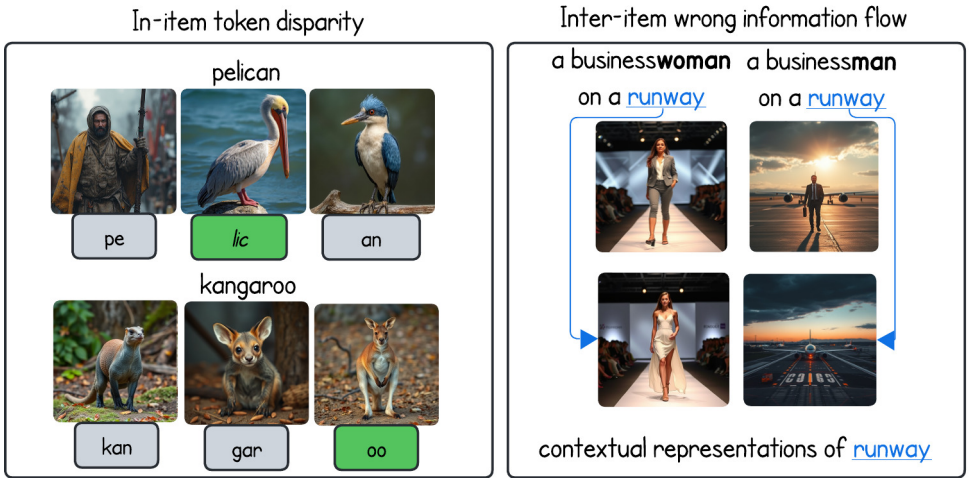

Figure 1: **Our main findings**. **Left:** Information within a lexical item is unevenly distributed across its tokens' contextualized representations. In this example, one token carries the meaning of the entire item (e.g., **lic** represents a pelican, while **pe** and **an** do not). **Right:** Context may cause incorrect information flow between items. In the example, the representation of **runway** is influenced by the context of "a business**woman**" (vs. "business**man**").

2023a).

For *cross-item* interactions (Section 4), we assess whether information flows between different lexical items in a prompt. We observe such cross-item flow in 11% of cases. Interestingly, the flow of information does not always follow sentence syntax, which can result in misrepresentation of items within specific contexts—especially with polysemous words. For instance, in the prompt "a pool by a table", "pool" can wrongly suggest "table" refers to a billiard table.

In summary, we study the *textual encoding* and identify two main insights. First, Lexical items have *concentrated representation*—the meaning of the item is often concentrated in a single representative token. Second, while items in a sentence do not typically encode information regarding other items in the sentence, in some cases, often when polysemous are involved, *semantic leakage* (Dahary et al., 2024; Gonen et al., 2024; Rassin et al., 2022) occurs during the textual encoding phase, causing miss-interpretations of items.

Together, these insights suggest practical avenues for improving token-level intervention techniques, enhancing evaluation benchmarks, and guiding future encoder-aware T2I generation approaches.

2 Methodology

2.1 Intervention on the Text Encoder

Our goal in this paper is to evaluate the effect of information flow between textual token representations, that subsequently condition the diffusion process. In Section 3, we are interested in analyzing how information is distributed across tokens within a lexical item . To do so, we seek to interpret the information encoded in individual tokens. In Section 4, we are interested in measuring how different lexical items influence one another. To this end, we seek to isolate a subset of tokens and evaluate their joint representation in the context of another item. For these purposes, we adjust a method for *text intervention in diffusion models* proposed by Toker et al. (2025), allowing us to generate images from arbitrary subsets of contextual token representations by masking the rest of the tokens in the sequence.

Given a prompt with N tokens $t_1, t_2, \ldots, t_N$, our goal is to isolate and interpret the information encoded by a subset of these tokens. Let $S \subset \{1, \ldots, N\}$ be the index set of selected tokens, where $0 < |S| < N$. We begin by encoding the full prompt using the text encoder E, yielding the final hidden states $h_1, \ldots, h_N$. Separately, we encode a sequence consisting entirely of pad tokens to obtain pad embeddings $p_1, \ldots, p_N$. We then construct a *patched prompt* by replacing all hidden states outside S with the corresponding pad embeddings:

$$\tilde{t}_i = \begin{cases} h_i & \text{if } i \in S, \\ p_i & \text{otherwise} \end{cases} \quad \text{for } i = 1, \ldots, N.$$

The patched sequence $\tilde{t}_1, \ldots, \tilde{t}_N$ is then used to guide the diffusion model. Generating an image

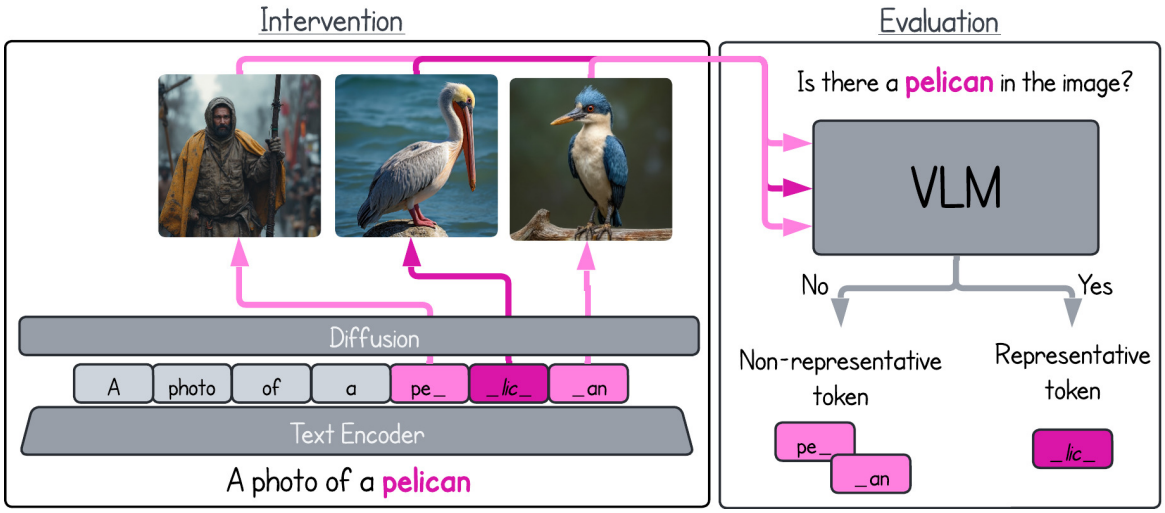

Figure 2: **Evaluating in-item information flow.** Our proposed framework interprets the information flow within a lexical item. We generate images from each token comprising the lexical item (left) and analyze them with a VLM (right). In this example, only the token **lic** represents the concept "pelican", whereas **pe** and **an** do not.

from this patched representation allows us to visualize and isolate the individual contributions of the selected tokens as interpreted by the diffusion model. To evaluate the information in the generated images, we use a vision-language model (VLM): For the *in-item* representation experiments, we assess whether the generated image represents the full lexical item, while for the *cross-item* interactions experiments, we measure whether information from other lexical items is present in the generated image. See Fig. 2 for an illustration of the method on the *in-item* representation experiment.

2.2 Experimental Setup

Models. Our main results are reported for FLUX-schnell (black-forest labs, 2024), a recent state-of-the-art T2I model. We repeat all experiments with FLUX-dev, SDXL-Turbo (Sauer et al., 2023) and SANA (Xie et al., 2024a). FLUX models employ T5-XXL (Raffel et al., 2019) as their text encoder, enabling bidirectional information flow between tokens. Unlike Flux, SDXL used CLIP (Radford et al., 2021), and SANA uses Gemma (Team et al., 2024). These text encoders are unidirectional, meaning the tokens are only influence by the tokens that came before. These changes cause some differences in the information flow between tokens, elaborated in Section 6 and in Appendix A.3.2

Data. We use a subset of 1,053 prompts from DrawBench (Saharia et al., 2022) and PartiPrompts (Yu et al., 2022b) datasets, filtered to include 4–20 words prompts and exclude cases with added complexity unrelated to our focus (e.g., misspellings, written text, rare words). For each prompt, we generate five images using different random seeds. To extract the lexical items, we prompt GPT-4o (OpenAI et al., 2024), resulting in 4,864 unique items across prompts. See Appendix A.1.1 for further technical details. We then use spaCy (Honnibal et al., 2020) to determine the part of speech of each lexical item, and retain only nouns, proper nouns, and adjectives, as these are typically concrete and can be identified in their visual representation. We end up with 3,891 unique lexical items and use them to evaluate both *in-item* representation and *cross-item* interactions.

Evaluation. To evaluate the content of generated images, we employ Qwen2-VL-72B-Instruct (Wang et al., 2024), a model with strong general vision capabilities. We restrict our evaluation to binary (yes/no) questions and use the model to assess prompt-image alignment and the presence of specific items in the image (see Appendix A.1.3 for additional details). To validate Qwen2-VL's reliability, we also conducted a human evaluation on 100 randomly sampled cases, evenly split between *in-item* representation (Section 3) and *cross-item* interactions (Section 4) settings. The results indicate substantial agreement, with a Cohen's Kappa values of 0.868 for *in-item* representation and 0.764 for *cross-item* interactions. Next, we determined the majority agreement among the human annotators and compared it to the model predictions. For *cross-item* interactions, the

Tokens Removed	# Prompts	Acc.		Relative %			Accuracy Δ (pp)
		Before	After	Unaffected	Degraded	Improved	
1	144	81.25	83.33	98.29	0.85	14.83	+2.08
2	98	82.65	88.78	100.00	0.00	35.27	+6.13
3	45	93.33	93.33	97.62	2.38	33.28	+0.00
4	24	87.50	91.67	100.00	0.00	33.36	+4.17
5+	28	78.57	85.71	100.00	0.00	33.32	+7.14
Overall	339	83.48	87.02	98.90	0.71	25.00	+3.54

Table 1: Effect of removing non-representative tokens on prompt-level accuracy. The table reports: (i) accuracy before and after token removal, (ii) the percentage of originally successful prompts that remained successful (Unaffected) or became failures (Degraded), and (iii) the percentage of originally failed prompts that were corrected (Improved). Overall, removing non-representative tokens rarely harms generation and often improves it.

accuracy (F1) is 0.927 (0.933), and for *cross-item* interactions, the accuracy is 0.810 (0.740). These results suggest that Qwen2-VL predictions align well with human judgment for our tasks.

3 In-Item Representation

In this section, we analyze information flow within lexical items at the token level. We first show semantic information is unevenly distributed across tokens, with typically one or two representing the item (3.1). We then show non-representative tokens are largely redundant and may even harm generation (3.2). Finally, we examine items neglected from the generated image, tracing failures to poor textual encoding or gaps in visual grounding (3.3).

3.1 How is Information Distributed Across Tokens?

We begin by exploring how information is distributed across the token representations within a lexical item. We focus on two key questions: Do all tokens encode the same semantic information? And is the information evenly distributed across tokens, or rather concentrated in specific tokens? These issues are crucial as many T2I applications (Chefer et al., 2023a; Rassin et al., 2023; Dahary et al., 2024) treat all tokens of a lexical item in a given prompt equally. Uncovering an asymmetrical distribution of information could improve the effectiveness of such applications by focusing on the most informative tokens.

Given a prompt and a lexical item, we feed the prompt to the text encoder, obtaining contextualized token representations. We then identify the tokens comprising the item and apply our intervention method (Section 2.1) to generate an image conditioned on each token's contextualized representation (i.e., without letting other tokens in the prompt influence the diffusion process). Finally, we use Qwen2-VL to assess whether the image represents the overall lexical item it is part of. We define a *representative token* as one whose isolated representation results in an image containing its corresponding lexical item. Tokens that do not meet this criterion are considered *non-representative*. We repeat this analysis for each lexical item and for each prompt in our dataset (Section 2.2).

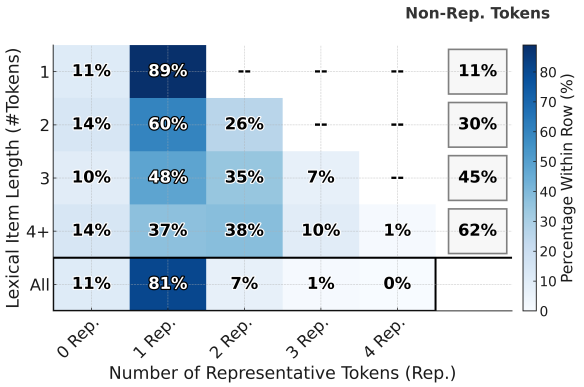

Figure 3: Distribution of representative tokens per lexical item length. "Rep." denotes the number of representative tokens; the bottom row aggregates over all lexical items. In most cases, one or two tokens sufficient to represent the entire lexical item. As item length increases, the number of non-representative tokens grows accordingly.

Our results (Fig. 3, *bottom row*) show that in most cases (89%), there is (at least) one *representative token*. Interestingly, in the remaining instances where no representative tokens exist, we find the item is also absent from an image generated from the full item (allowing *all* of the item's tokens to guide generation) in 88% of instances.

Figure 4: Examples illustrating the effect of removing *non-representative tokens*. **Top row**: Images generated after removing *non-representative tokens* (Representative tokens are shown in **bold**; non-representative tokens are in gray.). **Bottom row**: Images generated from the full prompt. **Left**: In most cases, removal results in no noticeable effect on the generation. **Right**: In some cases, removal improves alignment with the prompt.

We next focus on instances where at least one token represents the lexical item, and examine the number of *representative-* and *non-representative tokens* across lexical items of different lengths, averaged across items. Our results (Fig. 3, *top rows*) show that typically one or two tokens represent the concept, while the remaining tokens are *non-representative*. Further, as the token length of the lexical item becomes longer, the number of *non-representative* tokens increases (Fig. 3, *rightmost column*). Inspecting all lexical items in our data composed of two or more tokens, these *non-representative tokens* account for 52% of their tokens. Therefore, we next examine the effect of removing *non-representative tokens* altogether.

3.2 Tokens: Are They All Necessary?

We now examine whether *non-representative* tokens—tokens that do not result in an image representing the item—have any effect on image generation. To answer this question, we apply our intervention method to each prompt, this time generating an image after masking all the *non-representative tokens*. We use Qwen-VL to measure whether this image aligns with the prompt, and compare it to an image generated from the full prompt without intervention.

***Non-representative tokens* are redundant.** Our results (Table 1) show that removing *non-representative tokens* generally does not harm generation (see Fig. 4, left-hand side). When the original generation is aligned with the prompt, the generated image after non-representative token removal remains aligned in 98% of cases, suggesting these tokens are largely redundant. Surprisingly, in cases where the original image fails to align with the prompt, we observe a 21% improvement in alignment after removing non-representative tokens (see Fig. 4, right-hand side). We attribute this improvement to the model relying exclusively on the remaining representative tokens, which encode the correct semantics of the item.[2]

3.3 Item Negligence

So far, we have explored scenarios where an item lacks a representative token, often leading to its omission from the final generated output, i.e., *item negligence* (Chang et al., 2024b; Chefer et al., 2023a). We next investigate *why* this omission occurs. We examine two potential causes: either the text encoder poorly represents the concept; or the concept is well-encoded, but the diffusion model isn't familiar with its visual appearance.

To study an item's encoding without relying on the diffusion model, we use Patchscopes (Ghandeharioun et al., 2024), which decodes token representations into their natural language descriptions, using the full encoder-decoder architecture of T5 for text generation. Specifically, we patch an item's encoding into the template "`describe <item>`" and evaluate whether the output indeed describes the item using GPT-4o (OpenAI et al. 2024; see Appendix A.1.4 for prompt details). We apply this process to all neglected lexical items found in Sec-

[2]We measure improvement rates with Qwen2-VL before and after masking non-representative tokens. See Appendix A.1.3 for further details.

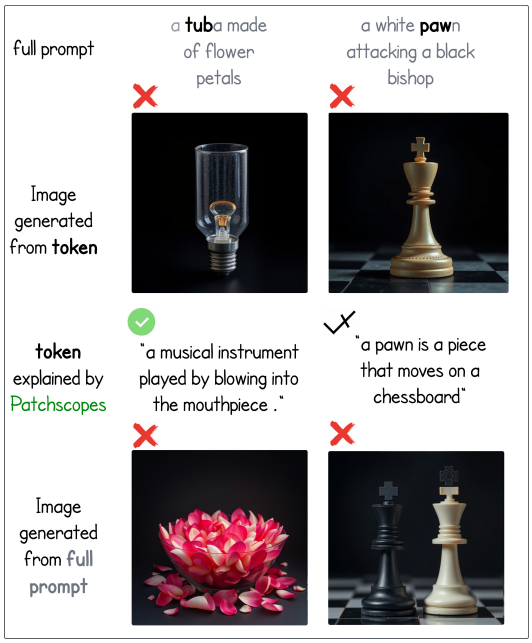

Figure 5: Comparing Patchscopes to image generation with token-level patching in cases of *item negligence*. We assess whether a token's concept is preserved by comparing images generated from its contextual representation, Patchscopes' textual interpretation, and the full prompt image. **Left:** "tub" (from "tuba") is correctly described by Patchscopes but fails to ground visually. **Right:** "paw" (from "pawn") has missing details by both interpretations, suggesting a gap in the encoder's conceptual knowledge.

tion 3.1, and examine whether the encoder is familiar with these items.

We observe that in 67% of all instances of negligence, Patchscopes returns correct descriptions (for example, as demonstrated in Fig. 5, "tuba" yields "*a musical instrument played by blowing into the mouthpiece*"). This suggests that, in such cases, item negligence is caused by a failure in the diffusion model. In the remaining 33%, the Patchscopes outputs are incorrect, indicating a gap in the encoder's semantic understanding or knowledge.

This points to two distinct sources of item-level negligence: Representational gaps within the text encoder itself—consistent with findings that augmenting text-side item descriptions improves factual generation (Huang et al., 2025)—and visual grounding failures despite sufficient text encodings.

4 Cross-Item Interactions

In Section 3, we demonstrated that a single token can effectively encapsulate the semantics of an entire lexical item. This raises a broader question: What defines the segmentation boundaries of a lexical item? Can its tokens further encode adjacent items, or even the general surrounding context?

Category	Count	Percentage
# pairs	15,950	100.00%
No Information Flow	14,251	89.35%
Information Flow	1,699	10.65%
– Source before Reference	835	49.15%
– Reference before Source	823	48.44%

Table 2: Distribution of information flow between lexical item pairs. Subcategories under "With Information Flow" indicate the order of source and reference.

To this end, we isolate each lexical item in a prompt and assess whether it encodes information about *other* items in the prompt. For each item, we generate images from its contextualized representation (encoded within the full prompt), and its uncontextualized representation (using the item's text by itself as a prompt). We then use Qwen2-VL to evaluate whether any other item from the prompt appears in the contextualized image, but not in the uncontextualized one—indicating information flow introduced by the surrounding context. See Appendix A.4 for implementation details.

Our results (Table 2) show that in 89% of the cases, lexical items do not encode information from other items in the prompt. In the remaining 11%, one lexical item incorporates information about another item (see Fig. 6 for qualitative examples). Interestingly, information flow can emerge between items with no direct syntactic relation in the prompt.

To better understand the nature of information flow, we analyzed the cases where lexical items appeared to influence one another. We categorize these prompt into 2 groups, based on whether this influence aligns with the prompt's syntactic structure or not. The first category involves items with a direct syntactic relationship, such as the adjective "black" modifying the noun "bear" in the prompt "a black bear". The second category includes items that are syntactically unrelated, such as the adjective "red" influencing the noun "building" in "a red car next to a building". Using GPT-4o to classify between these cases (see Appendix A.1.5), we find that 31% of these information flow instances occur between syntactically unrelated items.

Wrong information flow leads to misinterpretation. Our analysis reveals a recurring theme of semantic influence between items within a prompt, a pattern not prominent in most of the prompts. In these cases, the interpretation of one item is skewed

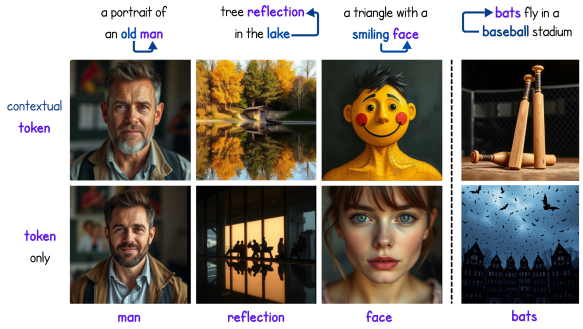

Figure 6: **Examples of information flow between items.** Top: Images generated from a **lexical item** encoded alongside **another item** that alters its representation. Bottom: Images generated from the uncontextualized representation of the same lexical item. The first three images (from the left) demonstrate correct information flow, while the last image (far right) demonstrates incorrect information flow.

by another semantically related, but syntactically distant, item. For instance, in the prompt "A standing zebra to the right of a city bus station", the strong semantic connection between "bus station" and "zebra" leads the model to generate a "zebra crossing" instead of the animal (see Fig. 7). We refer to this outcome, where contextual association overrides the prompt-intended meaning, as an *incorrect item resolution*.

We note that such failures typically occur in cases of polysemous words. For example, the item `bats` is interpreted as *wooden baseball bats* in the prompt "bats fly around a baseball stadium", whereas the intended meaning is obviously the flying animal. In such cases, the context overrides the correct sense of the item, leading to inaccurate generation. We hypothesize that these cases of *incorrect item resolution* are caused by information flow from the context to the misinterpreted lexical item. To validate this, We use the dataset from Rassin et al. (2022), which includes prompts known to induce semantic leakage due to implicit lexical associations (e.g., "bat" and "baseball stadium"). Unlike standard T2I benchmarks such as Huang et al. (2023b), which focus on spatial or visual challenges, these prompts highlight failures rooted in the encoding process itself. Since the original set includes only 30 examples, we expand it to 110 prompts using GPT-4o. See Appendix A.2 for augmentation details and the full prompt list.

To evaluate our hypothesis, we conduct a simple causal test. We generate three images for each prompt: an image from the full prompt; another from the item without any context; and one from the full prompt, but with the lexical item's tokens replaced by their uncontextualized representations (see Fig. 7). For each image, we assess if the image depicts the intended interpretation of the item within the prompt (see A.1.6 for more details on this evaluation process). Our findings show that in 93% of the evaluated prompts, the item is misinterpreted in the original full-prompt image but correctly interpreted in the patched versions. This suggests that the failure stems from *incorrect item resolution* during the textual encoding phase: the language model misrepresents the intended meaning of the item due to contextual interference, retrieving its incorrect sense. Importantly, this indicates a failure in semantic resolution that originates entirely within the text encoder, and cannot be mitigated during image generation.

Addressing such failures requires solutions during the textual encoding that target semantic disambiguation during encoding, rather than relying on downstream interventions such as Chefer et al. (2023a). Feng et al. (2023) offer one such solution, encoding items separately according to structured representations to improve attribute binding. While this represents a promising first step, future work must account for more complex interactions and employ more sophisticated, semantically-informed methods to achieve this separation.

5 Related Work

Interpretability in T2I models. Recent work has explored how T2I models encode and align concepts. Chefer et al. (2023b) studied CLIP's latent space, while Toker et al. (2024) interpreted representations across the textual encoding process using the diffusion model as a lens. Others examined text-to-image alignment via attention maps (Tang et al., 2023). Another direction is to apply SAEs (Cunningham et al., 2023) to interpret intermediate representations in T2I models (Cywiński and Deja, 2025). We on the other hand, focus on token-level information at the encoder's final layer, which directly conditions generation process.

Token representation and flow in LLMs. Studies have shown that token information in LLMs is not uniformly distributed. Kaplan et al. (2025) found that subwords fuse into word-level meaning, while Feucht et al. (2024) documented how later tokens can erase earlier ones. Attention-based flow (Vig and Belinkov, 2019; Clark et al., 2019)

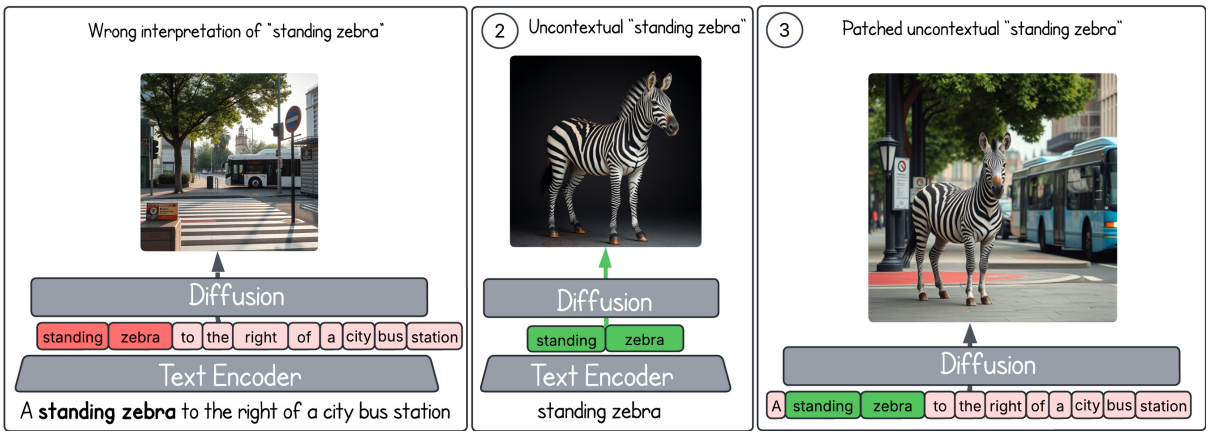

Figure 7: **Verifying textual semantic leakage by replacing the contextually leaked concept representation.** (1) Regular generation produces an image showing a crosswalk to the right of a bus station. (2) Generation from the prompt "standing zebra", without any context, results in the correct interpretation of the zebra as an animal. (3) Generation using the original prompt, but with the leaked concept "standing zebra" replaced by it's uncontextualized representation, yields a correct image.

can be misleading (Pruthi et al., 2020; Jain and Wallace, 2019), prompting alternatives like Attention Rollout (Abnar and Zuidema, 2020). Methods like Patchscopes (Ghandeharioun et al., 2024) and logit lens (nostalgebraist, 2020) decode internal representations, but do not test if downstream components use them.

Probing and causal methods. Probing methods (Adi et al., 2016; Liu et al., 2019; Zhang and Bowman, 2018; Brunner et al., 2019) reveal information presence but are not fully reliable, as probes can exploit spurious correlations (Belinkov, 2022). We instead use causal interventions to test whether token information is actually used during image generation.

Challenges in T2I models. *Semantic leakage*—when context distorts word meaning—has been observed in both LLMs (Gonen et al., 2024) and T2I (Rassin et al., 2022; Dahary et al., 2024). Another common issue is *neglect*, where key items are omitted (Chefer et al., 2023a; Chang et al., 2024b). While prior work focuses on solutions during the diffusion process, we show that some of these failures often originate in the text encoder.

6 Discussion

Improving generation via token-level intervention. Our results suggest that generation can be improved by removing non-representative tokens, which enhances prompt-image alignment and reduces noise. This intervention can be automated via a high-precision probe (see Appendix A.5), or integrated into model design, as in MrT5 (Kallini et al., 2025), which omits uninformative tokens in byte-level encoders (Xue et al., 2022). Similarly, attention-based techniques (Chefer et al., 2023a) may benefit from prioritizing representative tokens over full lexical spans.

Developing textually challenging benchmarks. Current T2I datasets focus primarily on visual or spatial complexity (Huang et al., 2023a; Ghosh et al., 2023; Saharia et al., 2022; Yu et al., 2022b). Yet our findings show that even slight linguistic ambiguity—particularly with polysemous or compositional phrases—can cause encoding failures. This highlights the need for evaluation benchmarks that probe textual difficulty more directly, which may in turn drive improvements in encoder design.

Generalization across T2I architectures. We repeat our main experiments on SDXL-Turbo and Sana (see A.3.2 and A.3.3), which use unidirectional text encoders, and on FLUX-dev (A.3.1), a compute-intensive variant of FLUX. We observe similar phenomena across models, indicating our findings are not specific to a particular T2I design.

7 Conclusion

Our investigation of text encoding in T2I models revealed two key properties: (1) lexical meaning tends to concentrate in one or two representative tokens rather than distributing evenly, and (2) contextual information can leak between lexical items, resulting in misinterpretation. Our analysis further indicates focused evaluation of the text encoder and token-level interventions are promising avenues towards more semantically aligned T2I generation.

Limitations

Evaluating token-level representations remains challenging. While we rely on strong vision-language models as judges and validate key findings through human evaluation, these are still approximations of true semantic alignment. Our prompt set focuses on object-centric, syntactically simple cases, which may limit generalization to prompts involving misspellings, rare words, or abstract concepts. Further work is needed to explore how information flow behaves under more linguistically complex conditions.

References

Samira Abnar and Willem Zuidema. 2020. Quantifying attention flow in transformers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4190–4197, Online. Association for Computational Linguistics.

Yossi Adi, Einat Kermany, Yonatan Belinkov, Ofer Lavi, and Yoav Goldberg. 2016. Fine-grained analysis of sentence embeddings using auxiliary prediction tasks. *arXiv preprint arXiv:1608.04207*.

Yonatan Belinkov. 2022. Probing classifiers: Promises, shortcomings, and advances. *Computational Linguistics*, 48(1):207–219.

black-forest labs. 2024. Flux. `https://github.com/black-forest-labs/flux`.

Gino Brunner, Yang Liu, Damian Pascual, Oliver Richter, Massimiliano Ciaramita, and Roger Wattenhofer. 2019. On identifiability in transformers. *arXiv preprint arXiv:1908.04211*.

Zhiyuan Chang, Mingyang Li, Junjie Wang, Yi Liu, Qing Wang, and Yang Liu. 2024a. Repairing catastrophic-neglect in text-to-image diffusion models via attention-guided feature enhancement. In *Conference on Empirical Methods in Natural Language Processing*.

Zhiyuan Chang, Mingyang Li, Junjie Wang, Yi Liu, Qing Wang, and Yang Liu. 2024b. Repairing catastrophic-neglect in text-to-image diffusion models via attention-guided feature enhancement. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 11379–11390, Miami, Florida, USA. Association for Computational Linguistics.

Hila Chefer, Yuval Alaluf, Yael Vinker, Lior Wolf, and Daniel Cohen-Or. 2023a. Attend-and-excite: Attention-based semantic guidance for text-to-image diffusion models. *ACM transactions on Graphics (TOG)*, 42(4):1–10.

Hila Chefer, Oran Lang, Mor Geva, Volodymyr Polosukhin, Assaf Shocher, Michal Irani, Inbar Mosseri, and Lior Wolf. 2023b. The hidden language of diffusion models. *arXiv preprint arXiv:2306.00966*.

Kevin Clark, Urvashi Khandelwal, Omer Levy, and Christopher D. Manning. 2019. What does BERT look at? an analysis of BERT's attention. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 276–286, Florence, Italy. Association for Computational Linguistics.

Hoagy Cunningham, Aidan Ewart, Logan Riggs, Robert Huben, and Lee Sharkey. 2023. Sparse autoencoders find highly interpretable features in language models. *arXiv preprint arXiv:2309.08600*.

Bartosz Cywiński and Kamil Deja. 2025. Saeuron: Interpretable concept unlearning in diffusion models with sparse autoencoders. *arXiv preprint arXiv:2501.18052*.

Omer Dahary, Or Patashnik, Kfir Aberman, and Daniel Cohen-Or. 2024. Be yourself: Bounded attention for multi-subject text-to-image generation. In *European Conference on Computer Vision*, pages 432–448. Springer.

Weixi Feng, Xuehai He, Tsu-Jui Fu, Varun Jampani, Arjun Akula, Pradyumna Narayana, Sugato Basu, Xin Eric Wang, and William Yang Wang. 2023. Training-free structured diffusion guidance for compositional text-to-image synthesis. *Preprint*, arXiv:2212.05032.

Sheridan Feucht, David Atkinson, Byron Wallace, and David Bau. 2024. Token erasure as a footprint of implicit vocabulary items in llms. *Preprint*, arXiv:2406.20086.

Asma Ghandeharioun, Avi Caciularu, Adam Pearce, Lucas Dixon, and Mor Geva. 2024. Patchscopes: A unifying framework for inspecting hidden representations of language models. *ArXiv*, abs/2401.06102.

Dhruba Ghosh, Hanna Hajishirzi, and Ludwig Schmidt. 2023. Geneval: An object-focused framework for evaluating text-to-image alignment. *Preprint*, arXiv:2310.11513.

Hila Gonen, Terra Blevins, Alisa Liu, Luke Zettlemoyer, and Noah A Smith. 2024. Does liking yellow imply driving a school bus? semantic leakage in language models. *arXiv preprint arXiv:2408.06518*.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. 2020. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851.

Matthew Honnibal, Ines Montani, Sofie Van Landeghem, and Adriane Boyd. 2020. spacy: Industrial-strength natural language processing in python.

Kaiyi Huang, Chengqi Duan, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. 2023a. T2i-compbench++: An enhanced and comprehensive

benchmark for compositional text-to-image generation. *IEEE transactions on pattern analysis and machine intelligence*, PP.

Kaiyi Huang, Kaiyue Sun, Enze Xie, Zhenguo Li, and Xihui Liu. 2023b. T2i-compbench: A comprehensive benchmark for open-world compositional text-to-image generation. *Advances in Neural Information Processing Systems*, 36:78723–78747.

Ziwei Huang, Wanggui He, Quanyu Long, Yandi Wang, Haoyuan Li, Zhelun Yu, Fangxun Shu, Weilong Dai, Hao Jiang, Fei Wu, and Leilei Gan. 2025. T2I-FactualBench: Benchmarking the factuality of text-to-image models with knowledge-intensive concepts. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 27501–27524, Vienna, Austria. Association for Computational Linguistics.

Sarthak Jain and Byron C. Wallace. 2019. Attention is not Explanation. 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)*, pages 3543–3556, Minneapolis, Minnesota. Association for Computational Linguistics.

Julie Kallini, Shikhar Murty, Christopher D. Manning, Christopher Potts, and Róbert Csordás. 2025. Mrt5: Dynamic token merging for efficient byte-level language models. *Preprint*, arXiv:2410.20771.

Guy Kaplan, Matanel Oren, Yuval Reif, and Roy Schwartz. 2025. From tokens to words: On the inner lexicon of llms. *Preprint*, arXiv:2410.05864.

Nelson F. Liu, Matt Gardner, Yonatan Belinkov, Matthew E. Peters, and Noah A. Smith. 2019. Linguistic knowledge and transferability of contextual representations. 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)*, pages 1073–1094, Minneapolis, Minnesota. Association for Computational Linguistics.

nostalgebraist. 2020. Interpreting GPT: The logit lens. lesswrong, 2020.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, and 262 others. 2024. Gpt-4 technical report. *Preprint*, arXiv:2303.08774.

Danish Pruthi, Mansi Gupta, Bhuwan Dhingra, Graham Neubig, and Zachary C. Lipton. 2020. Learning to deceive with attention-based explanations. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4782–4793, Online. Association for Computational Linguistics.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, and 1 others. 2021. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PmLR.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *Preprint*, arXiv:1910.10683.

Royi Rassin, Eran Hirsch, Daniel Glickman, Shauli Ravfogel, Yoav Goldberg, and Gal Chechik. 2023. Linguistic binding in diffusion models: Enhancing attribute correspondence through attention map alignment. *Advances in Neural Information Processing Systems*, 36:3536–3559.

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, Abu Dhabi, United Arab Emirates (Hybrid). Association for Computational Linguistics.

Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, Emily L Denton, Kamyar Ghasemipour, Raphael Gontijo Lopes, Burcu Karagol Ayan, Tim Salimans, and 1 others. 2022. Photorealistic text-to-image diffusion models with deep language understanding. *Advances in Neural Information Processing Systems*, 35:36479–36494.

Axel Sauer, Dominik Lorenz, Andreas Blattmann, and Robin Rombach. 2023. Adversarial diffusion distillation. *Preprint*, arXiv:2311.17042.

Yang Song and Stefano Ermon. 2019. Generative modeling by estimating gradients of the data distribution. *Advances in neural information processing systems*, 32.

Raphael Tang, Linqing Liu, Akshat Pandey, Zhiying Jiang, Gefei Yang, Karun Kumar, Pontus Stenetorp, Jimmy Lin, and Ferhan Ture. 2023. What the DAAM: Interpreting stable diffusion using cross attention. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5644–5659, Toronto, Canada. Association for Computational Linguistics.

Gemma Team, Thomas Mesnard, Cassidy Hardin, Robert Dadashi, Surya Bhupatiraju, Shreya Pathak, Laurent Sifre, Morgane Rivière, Mihir Sanjay Kale, Juliette Love, and 1 others. 2024. Gemma: Open models based on gemini research and technology. *arXiv preprint arXiv:2403.08295*.

Michael Toker, Ido Galil, Hadas Orgad, Rinon Gal, Yoad Tewel, Gal Chechik, and Yonatan Belinkov. 2025. Padding tone: A mechanistic analysis of padding tokens in t2i models. *Preprint*, arXiv:2501.06751.

Michael Toker, Hadas Orgad, Mor Ventura, Dana Arad, and Yonatan Belinkov. 2024. Diffusion lens: Interpreting text encoders in text-to-image pipelines. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9713–9728, Bangkok, Thailand. Association for Computational Linguistics.

Jesse Vig and Yonatan Belinkov. 2019. Analyzing the structure of attention in a transformer language model. In *Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 63–76, Florence, Italy. Association for Computational Linguistics.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, Yang Fan, Kai Dang, Mengfei Du, Xuancheng Ren, Rui Men, Dayiheng Liu, Chang Zhou, Jingren Zhou, and Junyang Lin. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and Song Han. 2024a. Sana: Efficient high-resolution image synthesis with linear diffusion transformer. *Preprint*, arXiv:2410.10629.

Enze Xie, Junsong Chen, Junyu Chen, Han Cai, Haotian Tang, Yujun Lin, Zhekai Zhang, Muyang Li, Ligeng Zhu, Yao Lu, and Song Han. 2024b. Sana: Efficient high-resolution image synthesis with linear diffusion transformers. *Preprint*, arXiv:2410.10629.

Linting Xue, Aditya Barua, Noah Constant, Rami Al-Rfou, Sharan Narang, Mihir Kale, Adam Roberts, and Colin Raffel. 2022. Byt5: Towards a token-free future with pre-trained byte-to-byte models. *Preprint*, arXiv:2105.13626.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, Yinfei Yang, Burcu Karagol Ayan, Ben Hutchinson, Wei Han, Zarana Parekh, Xin Li, Han Zhang, Jason Baldridge, and Yonghui Wu. 2022a. Scaling autoregressive models for content-rich text-to-image generation. *Preprint*, arXiv:2206.10789.

Jiahui Yu, Yuanzhong Xu, Jing Yu Koh, Thang Luong, Gunjan Baid, Zirui Wang, Vijay Vasudevan, Alexander Ku, and 1 others. 2022b. Scaling autoregressive models for content-rich text-to-image generation. *arXiv preprint arXiv:2206.10789*, 2(3):5.

Kelly Zhang and Samuel Bowman. 2018. Language modeling teaches you more than translation does: Lessons learned through auxiliary syntactic task analysis. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 359–361.

A Appendix

A.1 Technical Details

A.1.1 Lexical Item Classification

We define a *lexical item* as either a single word or a compound expression of multiple words that, in context, conveys a unified semantic meaning. A compound expression is treated as a single item when its words form a fixed lexical unit with cohesive semantics rather than merely exhibiting a modifier–head relationship. For example, while "broken mirror" describes a mirror's state, expressions like "hot air balloon" or "teddy bear" denote entities with distinct identities. Similarly, although phrases such as "identical twins" or "baseball bat" might be interpreted as separate concepts, conventional usage supports their treatment as unified entities. We employ the reasoning model `03-mini-high` as a classifier to tag multi-word lexical items in both the target prompts and the dataset. The model returns a list of identified multi-word expressions, while the remaining untagged words are treated as individual lexical items.

A.1.2 Redundant Token Classification

We propose a probing classifier to predict whether a token is redundant (i.e., *non-representative* of its lexical item) using solely its encoded representation, without the need to generate an image. For this purpose, we extracted the 6,966 tokens corresponding to 4,864 unique lexical items from our dataset. Each token was annotated with a binary label indicating whether it represents the lexical item by the VLM (see A.1.3 for more details).

We split the data into training and validation sets using an 80-20 ratio. A k-nearest neighbors (k-NN) classifier with $k = 5$ and Euclidean distance as the similarity measure was then applied to predict token redundancy directly from the encoded representations. The results on the evaluation set are presented in Table 3. The high precision indicates that, in practical settings, one can remove tokens predicted as redundant with a high degree of certainty that *representative tokens* will not be inadvertently discarded.

A.1.3 Evaluation Visual Generations.

We evaluate whether an image matches a textual description using Qwen2-VL-72B-Instruct (Wang et al., 2024). The following prompt is used:

> "In Yes, No and maybe. Does every image match one of those descriptions:

Table 3: Performance of k-NN classifier ($k = 5$) for predicting token redundancy

Metric	Accuracy	Precision	Recall	F1-score
Score	0.82	0.92	0.74	0.82

(description string)? Answer Yes if all images match or relate to at least one description, Maybe if only some match, otherwise No."

Here, the *textual description* can be either a single lexical item or a complete textual prompt.

A.1.4 Evaluating Generated Textual Descriptions.

We employ GPT-4o (OpenAI et al., 2024) to evaluate the textual interpretations produced by Patchscopes. We use the following prompt:

> "In Yes, No and Maybe. Does every image match the description: {Patchscopes_description} ? Answer Yes if all images match or relate to the description, Maybe if only some match, otherwise No."

A.1.5 Evaluating Relations Between Items.

We enhance our leakage validation by distinguishing between cases where two lexical items exhibiting semantic leakage are perceptually bound together—for example, "old" and "man" in the prompt "a portrait of an old man"—and cases where they are not as "cone hat" and "eating" in the prompt "A person searing a cone hat is eating" (see Fig. 6). To achieve this, we use a large language model (LLM) as a judge. Specifically, we use GPT-4o and employ the following prompt:

> "In Yes or No: in this prompt: {input_prompt}, are {item_1} and {item_2} perceptually bound together?"

We then filter out all cases where the lexical items are perceptually bound together and find that only 6.5% instances exhibit unintentional leakage.

A.1.6 Intended Item Evaluation

Foe each lexical-item, we manualy create two interpretation - one in the intended interpretation in the prompt, and another is a possible wrong interpretation of the word in other contexts. For example, given the prompt "A standing zebra to the right of a city bus station", the corrent interpretations would be "the animal zebra", while the incorrect interpretations would be "A zebra crossing". We then ask a VLM to evaluate the generated image, and asses if the first or the latter interpretations exists in the images.

To evaluate the model's capacity for contextual disambiguation, we manually define two interpretations for each lexical item in each prompt. The first is the intended semantic meaning derived from the prompt's context, and the second is a alternative interpretation, that is wrong in this context. For instance, in the prompt "A standing zebra to the right of a city bus station", the intended meaning of "zebra is the animal", whereas the wrong interpretation is a "zebra crossing". Subsequently, a VLM analyzes the generated image to determine if it depicts the intended interpretation or the wrong interpretation as we define it.

A.1.7 Resources

Our computational experiments involved inference with four distinct text-to-image models: Flux-dev, Flux-schnell, sdxl-turbo, and Sana. The parameter sizes for these models are approximately 12 billion for Flux-dev and Flux-schnell, 3.1 billion for sdxl-turbo, and a range of 0.6 to 4.8 billion for the Sana models, with our experiments utilizing a 1.6 billion parameter version. The total computational budget for these experiments is estimated to be approximately 480 GPU hours. The computing infrastructure consisted of a cluster of eight NVIDIA A100 GPUs. This configuration provided the necessary computational power for the large number of inference tasks performed.

A.1.8 Use of AI Assistants.

We utilized AI assistants to support this research. For coding the experiments, we used Microsoft's Copilot and Anthropic's Claude 3; all generated code was manually reviewed and validated by us to ensure it aligned with our requirements. For the paper, Google's Gemini models (Pro and Flash) were used to improve the writing and clarity. We have carefully reviewed all content to ensure it accurately reflects our intentions.

A.2 Data

DrawBench (Saharia et al., 2022): We include all categories except for "misspelling", "rare words", and "text". Overall we extract 134 prompts from DrawBench.

Parti Prompts (Yu et al., 2022a): We include all categories except "Style & Format", "Writing & Symbols", and "Arts". Overall we extract 923 prompts from Parti Prompts. The dataset is released under apache-2.0 license.

In total, we obtain 1,056 prompts.

Extended dataset of leakage prompts. Our augmentation process incorporates two components. First, we generate variations of existing prompts from (Rassin et al., 2022) (e.g., modifying 'a gentleman with a bow in the forest' to 'a man wearing a bow in the jungle'). Second, we introduce novel prompts with potential semantic leakage. For these prompts, we applied a one-lexical item change test by generating an image from a similar prompt that substitutes the affected or leaked item with an alternative term (e.g., replacing 'bishops' with 'cardinals' or 'checkers' in 'chess in '2 bishops playing chess'). This test ensures that minimal lexical modifications do not alter the intended semantic meaning while producing a different image due to semantic leakage from another item in the prompt (see the first two columns in Fig. 8 for few visual examples). Together, these methods enrich the dataset and provide a robust framework for analyzing semantic leakage. The full list of prompts is available in an anonymous Git repository.[3]

We split the data into training and validation sets using an 80-20 ratio. A k-nearest neighbors (k-NN) classifier with $k = 5$ and Euclidean distance as the similarity measure was then applied to predict token redundancy directly from the encoded representations. The results on the evaluation set are presented in Table 3. The high precision indicates that, in practical settings, one can remove tokens predicted as redundant with a high degree of certainty that *representative tokens* will not be inadvertently discarded.

A.3 Additional models

A.3.1 FLUX-Dev

In addition to our primary experiments with FLUX, we repeated all analyses using the Flux-dev variant. The redundant versus representative token experiments yielded similar trends, with 55% of tokens identified as representative and 45% as non-representative—values closely matching those observed with FLUX. Likewise, our inter-item flow experiments confirmed that information flow occurred in 11% of cases (and 3.1% miss intended leakage), reinforcing the overall patterns reported in the main text. Notably, while the aggregate trends are consistent across models, the specific lexical items resolved can differ between FLUX-schnell and Flux-dev, indicating potential a slightly different inner-lexicon (Kaplan et al., 2025). These findings underscore the robustness of our approach while highlighting model-dependent nuances in token representation and information flow dynamics.

[3] https://anonymous.4open.science/r/TokenRole-3E19

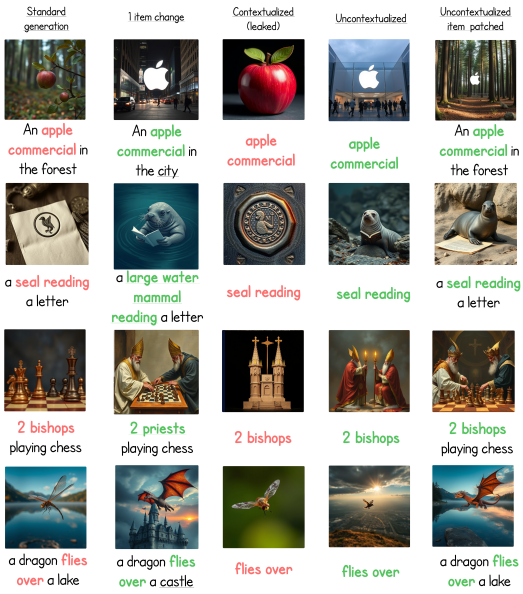

Figure 8: Examples from our semantic leakage method. **Left**: standard generation of leakage contained prompt. **Second**: generation using a one-lexical item change test as part of the dataset creation (a minimal substitution to verify that a slight lexical change yields a different image). **Third**: image from the contextual representation (misinterpreted item). **Forth**: image from the uncontextualized representation (correct interpretation). **Right**: final generation after patching the correct, uncontextualized representation into the prompt.

A.3.2 SDXL-Turbo

Our analysis reveals that SDXL-Turbo, which uses the CLIP text encoder, behaves markedly differently from FLUX, which relies on the encoder in the encoder-decoder T5-XXL. In SDXL-Turbo, the text encoder is a causal language model, meaning each token's encoding is influenced only by its preceding tokens during the encoding process.

We repeated our *in-item* representation experi-

Figure 9: Images generated from individual subtokens in SDXL-Turbo. We find that, in many cases, the representation of an item is not clearly reflected in any of its subtokens—for example, in the case of the token "skateboard." Another interesting observation is that the last token of a lexical item often carries its representation, as seen in the "square" token of "times square." We also observe that the EOT token incorporates information from the full prompt.

ments using SDXL-Turbo. Our first observation is that most generated images are either abstract or unrelated to the intended lexical items. According to our analysis, 55% of lexical items in SDXL-Turbo lack any representative token (compared to just 11% in FLUX). Moreover, when a representative token is present in CLIP, it is typically the final token of the lexical item (see Fig. 9). This is aligned with the unidirectional encoding of the model.

Another phenomenon we observe—consistent with CLIP's training objective—is the unusually dominant role of the end-of-sequence (EOS) token. Images generated from the EOS token often encapsulate nearly the full semantic content of the prompt. In our evaluation, 62% of EOS-generated images matched the prompt (compared to 73% when using the full prompt). We believe this also causes our intervention method to be less effective, since when we interpret a single token, we patch all other tokens, including the EOT token—which usually contains a lot of information—with tokens derived from an empty prompt (see Fig. 9).

A.3.3 Sana (Gemma-based encoder)

We also evaluate our methodology on the Sana model (Xie et al., 2024b), which uses a Gemma-based autoregressive language model as the text encoder. These results help validate the generality of our findings across architectures with differing encoding strategies.

For *in-item information flow*, we observe that Sana produces fewer multi-token lexical items due to its larger vocabulary relative to T5 and CLIP. In cases where items do consist of multiple tokens, we find that only the *last* token typically acts as a representative token—a behavior aligned with the unidirectional nature of autoregressive encoders.

In our *cross-item information flow* analysis, we find that 17.45% of item pairs in Sana exhibit contextual information flow, compared to 10.45% in FLUX. This increase likely stems from the one-sided (forward-only) nature of Sana's encoding. For example, in the prompt *"a black baseball hat with a flame decal on it"*, we find contextual influence such as "black baseball" affecting "hat" and "flame" affecting "decal", but not the reverse. See Fig. 10 for illustrations.

Overall, while Sana shows a slightly higher rate of contextual influence than FLUX, it preserves many of the key structural insights found in our primary analysis—particularly the sparsity and location of representative tokens. Unlike CLIP, which encodes substantial information in the EOS token, Sana lacks such artifacts, suggesting that EOS-related effects are not fundamental to autoregressive models more generally.

A.4 Inter-Item Information Flow Framework

To assess whether one lexical item encodes information about others in the same prompt, we conduct the following experiment.

Given a prompt, we isolate each lexical item one at a time. First, we encode the full prompt using the text encoder. Then, for a given lexical item, we apply our patching method (Section 2.1) to mask the representations of all other tokens—leaving only the contextualized representation of the target item intact. Formally, for a lexical item with token indices $S \subset \{1, \ldots, N\}$, we construct a patched sequence $\tilde{t}_1, \ldots, \tilde{t}_N$ in which only the tokens in S retain their original contextualized representations. We then generate an image from this modified sequence, capturing what information is encoded in the selected item's contextualized form.

We repeat this process for the same item in an uncontextualized setting: we encode and generate an image using only that lexical item in isolation, without the rest of the prompt. This allows us to distinguish between information inherently present in the item's encoding and information introduced by context.

To measure influence between items, we use Qwen2-VL to check whether a second item y appears in the image generated from a first item x, both in the contextualized and uncontextualized versions. Influence is considered *True* if y appears in the image generated from contextualized x, but not from uncontextualized x. This comparison ensures that the observed presence of y is attributable

Figure 10: Token-level image generation using Sana on the prompt *"a black baseball hat with a flame decal on it"*. Representative information often resides in the last token of each item (e.g., "hat", "decal"), consistent with Sana's causal encoding. Contextual influence is one-directional, with earlier tokens shaping later ones.

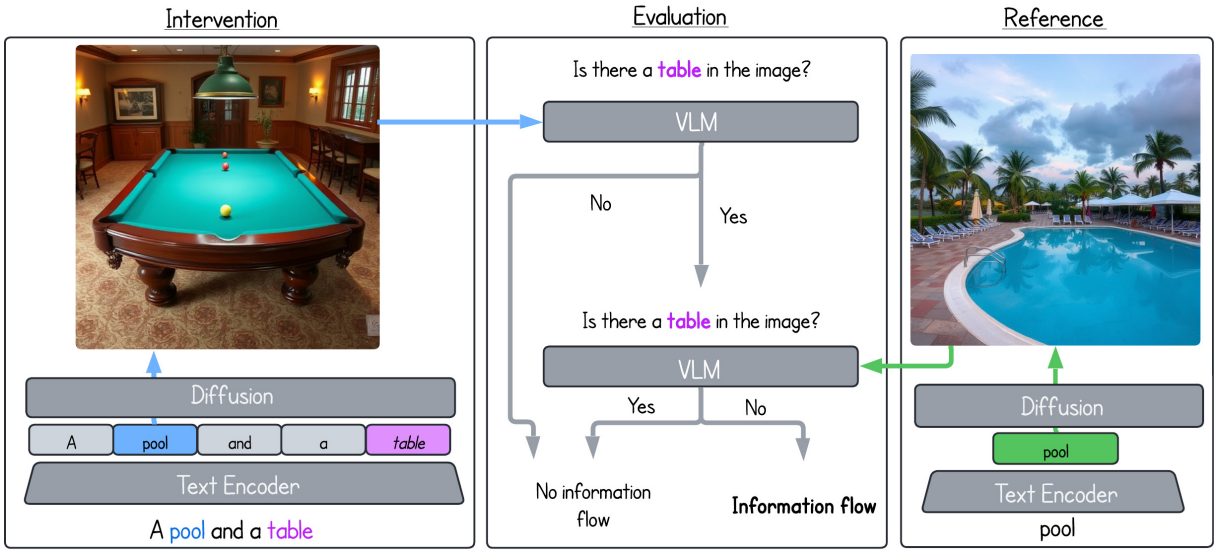

Figure 11: **Evaluating inter-item information flow:** Our proposed framework to interpret the information flow between lexical items in the prompt. For each lexical item, we generate an image from its contextual representations (left), and from it's uncontextualized representation (right), and analyze the generated images using a VLM (middle). In this example, we interpret the item "pool" and assess whether it is influenced by the item "table". To do so, we ask a VLM whether the image generated from the token "pool" contains a "table". To ensure this is the result of information flow, and not a natural correlation between a pool and tables, we generate the item "pool" without context (right-hand side), and verify that "table" is not present in this image. If this is the case, we conclude that there was information flow from "table" to "pool".

to contextual information flow during encoding, rather than coincidental co-occurrence or inherent correlation.

This setup enables us to detect breaches of lexical segmentation—i.e., cases where one lexical item encodes visual information belonging to another—quantifying interdependence across items within the prompt.

See Fig. 11 for an illustration of the procedure.

A.5 Predicting Redundant Tokens Without Image Generation

Identifying which tokens are *representative* can improve image generation, as demonstrated in Fig. 4 (right). Moreover, many existing T2I methods operate at the token level, typically treating all of a lexical item's tokens equally (Chefer et al., 2023a; Dahary et al., 2024). However, our findings suggest that only a subset of these tokens contribute meaningfully to the representation.

Yet, identifying representative tokens via image generation is computationally expensive. To address this, we explore alternatives for estimating token importance without image-level interventions.

First, we compute the character-level edit distance between a lexical item (e.g., "giraffe") and its constituent tokens (e.g., "gir", "a", "ffe"). We observe a negative correlation between edit distance and token representativeness (Pearson correlation of -0.44), suggesting that tokens more orthographically distant from the original item are less likely to be representative. While informative, this signal alone is insufficient for accurate prediction.

Instead, we train a k-Nearest Neighbors (k=5)

classifier using Euclidean distance over the token representations to predict redundancy. Using 6,966 token instances (spanning 4,864 unique lexical items), each labeled for whether the token is representative according to a VLM, our classifier achieves 92% precision in identifying non-representative tokens. This result highlights the feasibility of lightweight approaches for filtering redundant tokens without resorting to costly generation procedures. See Appendix A.1.2 for classifier details.